Guia do usuário

AWS OpsWorks

Versão da API 2013-02-18

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS OpsWorks: Guia do usuário

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens comerciais da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestigie a Amazon. Todas as outras marcas comerciais que não são propriedade da Amazon pertencem aos respectivos proprietários, os quais podem ou não ser afiliados, estar conectados ou ser patrocinados pela Amazon.

Table of Contents

O que é AWS OpsWorks? ... 1
AWS OpsWorks Serviços ... 1
AWS OpsWorks para Puppet Enterprise ... 4
Support regional OpsWorks para for Puppet Enterprise ... 5
Fim da vida FAQs ... 6
Como os clientes existentes serão afetados por esse fim da vida útil? ... 6
O que acontecerá aos meus servidores se eu não tomar nenhuma ação? ... 7
Está AWS OpsWorks for Puppet Enterprise aceitando novos clientes? ... 7
O fim da vida afetará tudo Regiões da AWS ao mesmo tempo? ... 7
Para qual nível de suporte técnico está disponível AWS OpsWorks for Puppet Enterprise? ... 7
Sou cliente atual da OpsWorks Puppet Enterprise e preciso iniciar um servidor em uma conta que não estava usando o serviço anteriormente. Eu sou capaz de fazer isso? ... 7
Haverá algum lançamento de novos recursos para AWS OpsWorks for Puppet Enterprise? ... 8
Conceitos básicos ... 8
Pré-requisitos ... 8
Criar um Puppet master ... 13
Concluir configuração ... 25
Adicionar nós para gerenciar ... 30
Faça login no console do Puppet Enterprise ... 33
Opcional: Use CodeCommit ... 37
Crie um Puppet Master em CloudFormation ... 44
Pré-requisitos ... 45
Criar um Puppet Enterprise mestre no AWS CloudFormation ... 45
Atualizar um servidor para usar um domínio personalizado ... 52
Pré-requisitos ... 53
Limitações ... 53
Atualizar um servidor para usar um domínio personalizado ... 54
Consulte também ... 58
Trabalhar com tags ... 58
Como as tags funcionam em AWS OpsWorks for Puppet Enterprise ... 59
Adicionar e gerenciar tags no OpsWorks Puppet Enterprise (Console) ... 61
Adicionar e gerenciar tags no OpsWorks Puppet Enterprise (CLI) ... 63
Consulte também ... 68

Fazer backup e restaurar servidores ... 69
Criar um backup de OpsWorks servidor para Puppet Enterprise ... 69
Restaurar um OpsWorks servidor para Puppet Enterprise ... 73
Manutenção do sistema ... 74
Configurar a manutenção do sistema ... 76
Iniciar a manutenção do sistema sob demanda ... 78
Restaurar configurações e arquivos personalizados após a manutenção ... 79
Adição automática de nós ... 79
Etapa 1: criar um perfil do IAM para usar como o perfil de instância ... 80
Etapa 2: criar instâncias com um script de associação autônoma ... 80
Remover nós ... 82
Consulte também ... 83
Excluir um Puppet master ... 83
Etapa 1: Dissociação dos nós gerenciados ... 84
Etapa 2: Exclusão do servidor ... 84
Consulte também ... 84
Migrar um servidor Puppet para a Amazon EC2 ... 85
Etapa 1: entre em contato com o Puppet para comprar uma licença ... 85
Etapa 2: Obtenha detalhes sobre seu servidor OpsWorks for Puppet Enterprise ... 86
Etapa 3: Faça um backup do seu servidor OpsWorks para o Puppet Enterprise ... 86
Etapa 4: iniciar uma nova EC2 instância ... 87
Etapa 5: instalar o Puppet Enterprise na nova instância EC2 ... 89
Etapa 6: restaurar o backup na nova EC2 instância ... 89
Etapa 7: configurar sua licença do Puppet ... 90
Etapa 8: migrar seus nós ... 90
Etapa 9: Exclua seu servidor OpsWorks for Puppet Enterprise ... 92
Usando AWS CloudTrail ... 93
OpsWorks para obter informações sobre a Puppet Enterprise em CloudTrail ... 94
Compreendendo OpsWorks as entradas do arquivo de log do Puppet Enterprise ... 95
Solução de problemas ... 97
Dicas para solução de problemas em geral ... 97
Solução de problemas de erros específicos ... 98
Ajuda e suporte adicionais ... 103
AWS OpsWorks para Chef Automate ... 104
Suporte regional para AWS OpsWorks for Chef Automate ... 107
Fim da vida FAQs ... 108

Como os usuários existentes serão afetados por esse fim da vida útil? 109
O que acontecerá aos meus servidores se eu não tomar nenhuma ação? 109
Para quais alternativas posso fazer a transição? 109
O serviço ainda está aceitando novos clientes? 109
O fim da vida afetará tudo Regiões da AWS ao mesmo tempo? 110
Qual nível de suporte técnico está disponível? 110
Sou cliente atual do OpsWorks Chef Automate e preciso iniciar um servidor em uma conta que não estava usando o serviço anteriormente. Eu sou capaz de fazer isso? 110
Haverá algum lançamento de recursos importantes no próximo ano? 110
Atualizar para o Chef Automate 2 110
Pré-requisitos para a atualização para o Chef Automate 2 111
Sobre o processo de atualização 111
Atualizar para o Chef Automate 2 (console) 112
Atualizar para o Chef Automate 2 (CLI) 112
Reverta um AWS OpsWorks for Chef Automate servidor para o Chef Automate 1 (CLI) 114
Consulte também 114
Conceitos básicos 115
Pré-requisitos 115
Criar um servidor Chef Automate 118
Concluir a configuração e fazer o upload de livros de receitas 131
Adicionar nós para gerenciar 140
Faça login no painel do Chef Automate 147
Crie um Chef Automate Server em CloudFormation 150
Pré-requisitos 151
Criar um servidor Chef Automate no AWS CloudFormation 152
Atualizar um servidor para usar um domínio personalizado 159
Pré-requisitos 160
Limitações 53
Atualizar um servidor para usar um domínio personalizado 54
Consulte também 58
Regenere o starter kit 165
Regenere o kit AWS OpsWorks for Chef Automate inicial com o AWS CLI 166
Trabalhar com tags 167
Como as tags funcionam em AWS OpsWorks for Chef Automate 168
Adicionar e gerenciar tags no AWS OpsWorks for Chef Automate (console) 169
Adicionar e gerenciar tags em AWS OpsWorks for Chef Automate (CLI) 172

Consulte também ... 177
Fazer backup e restaurar servidores ... 178
Faça backup de um AWS OpsWorks for Chef Automate servidor ... 178
Restaurar um AWS OpsWorks for Chef Automate servidor ... 181
Manutenção do sistema ... 182
Garantir que os nós confiem na Autoridade AWS OpsWorks de Certificação ... 184
Configurar a manutenção do sistema ... 185
Iniciar a manutenção do sistema sob demanda ... 187
Restaurar configurações e arquivos personalizados após a manutenção ... 187
Verificações de conformidade ... 188
Conformidade no Chef Automate 2.0 ... 189
Conformidade no Chef Automate 1.x ... 197
Atualizações feitas na conformidade ... 203
Perfis de conformidade e comunidade personalizados ... 203
Consulte também ... 204
Remover nós ... 204
Related Topics ... 205
Exclusão de um servidor do Chef Automate ... 206
Etapa 1: Dissociação dos nós gerenciados ... 206
Etapa 2: Exclusão do servidor ... 207
Redefinição das credenciais do Chef ... 207
Usando AWS CloudTrail ... 208
AWS OpsWorks for Chef Automate Informações em CloudTrail ... 209
Entendendo as entradas do arquivo de AWS OpsWorks for Chef Automate log ... 210
Solução de problemas ... 212
Dicas para solução de problemas em geral ... 213
Solução de problemas de erros específicos ... 213
Ajuda e suporte adicionais ... 221
Segurança no gerenciamento de AWS OpsWorks configuração (CM) ... 222
Proteção de dados ... 223
Integração com AWS Secrets Manager ... 224
Criptografia de dados ... 225
Criptografia em repouso ... 225
Criptografia em trânsito ... 225
Gerenciamento de chaves ... 226
Gerenciamento de Identidade e Acesso ... 226

Público ... 226
Autenticação com identidades ... 227
Gerenciamento do acesso usando políticas ... 230
Como o AWS OpsWorks CM funciona com o IAM ... 233
Exemplos de políticas baseadas em identidade ... 239
Solução de problemas ... 242
Políticas gerenciadas pela AWS ... 244
Prevenção confusa de delegados entre serviços em AWS OpsWorks CM ... 253
Privacidade do tráfego entre redes ... 257
Registro e Monitoramento ... 257
Validação de conformidade ... 257
Resiliência ... 258
Segurança da infraestrutura ... 259
Análise de configuração e vulnerabilidade ... 260
Práticas recomendadas de segurança ... 260
AWS OpsWorks Pilhas ... 262
Pilhas ... 265
Camadas ... 266
Receitas e LifeCycle eventos ... 266
Instâncias ... 267
Apps ... 268
Personalização de sua pilha ... 269
Gerenciamento de recursos ... 270
Segurança e permissões ... 270
Monitoramento e registro em log ... 270
CLI, SDK e modelos AWS CloudFormation ... 271
Fim da vida FAQs ... 272
Como os clientes existentes serão afetados por esse fim da vida útil? ... 272
Está AWS OpsWorks Stacks aceitando novos clientes? ... 272
Para onde devo migrar minhas pilhas existentes? ... 272
Como posso manter minhas EC2 instâncias existentes da Amazon após o fim da vida útil? ... 273
O fim da vida afetará tudo Regiões da AWS ao mesmo tempo? ... 273
Para qual nível de suporte técnico está disponível AWS OpsWorks Stacks? ... 273
Haverá algum lançamento de novos recursos para AWS OpsWorks Stacks? ... 274
Como migrar seus aplicativos para o Systems Manager Application Manager ... 274

Como o script funciona ... 275
Pré-requisitos ... 275
Limitações ... 276
Conceitos básicos ... 276
Perguntas frequentes ... 292
Solução de problemas ... 304
Usando a AWS OpsWorks Stacks ferramenta Detach in Place ... 306
Como o processo funciona ... 307
Limitações ... 309
Conceitos básicos ... 309
Conceitos básicos ... 319
Suporte regional ... 320
Conceitos básicos: exemplo ... 321
Conceitos básicos: Linux ... 342
Conceitos básicos: Windows ... 374
Conceitos básicos: livros de receitas ... 410
Práticas recomendadas ... 445
Armazenamento do dispositivo raiz ... 445
Otimização do número de servidores ... 448
Gerenciamento de permissões ... 451
Gerenciamento e implementação de aplicativos e guias de procedimentos ... 454
Empacotar dependências do livro de receitas localmente ... 464
Pilhas ... 469
Migre pilhas do -Classic EC2 ... 470
Criar uma nova pilha ... 473
Execução de uma Stack em uma VPC ... 482
Atualizar uma pilha ... 494
Clonar uma pilha ... 495
Executa comandos de pilha ... 497
Usar JSON personalizado ... 500
Excluir uma pilha ... 503
Camadas ... 508
OpsWorks Noções básicas sobre camadas ... 509
Camada Elastic Load Balancing ... 525
Camada de serviços do Amazon RDS ... 530
Camadas de cluster do ECS ... 536

Camadas personalizadas ... 543
Instalações de pacotes por camada ... 544
Instâncias ... 546
Usando AWS OpsWorks instâncias Stacks ... 547
Uso de recursos de cálculo criados fora do AWS OpsWorks Stacks ... 610
Edição da configuração da instância ... 658
Excluindo instâncias do AWS OpsWorks Stacks ... 660
Login com SSH ... 662
Login com RDP ... 665
Apps ... 670
Adição de aplicativos ... 671
Implementação de aplicativos ... 678
Editar aplicativos ... 683
Conectar-se a um banco de dados ... 684
Usar variáveis de ambiente do ... 686
Transmissão de dados para aplicativos ... 688
Utilização de chaves SSH de repositório Git ... 692
Usando domínios predefinidos ... 693
Uso de SSL ... 695
Livros de receitas e receitas ... 703
Repositórios de livro de receitas ... 704
Versões do Chef ... 708
Versões do Ruby ... 728
Instalação de livros de receitas personalizados ... 729
Atualizar livros de receitas personalizadas ... 733
Execução de receitas ... 736
Gerenciamento de recursos ... 744
Como registrar recursos com uma pilha ... 746
Como anexar e mover recursos ... 752
Como desanexar recursos ... 758
Como cancelar registros de recursos ... 760
Tags ... 763
Definição de tags no nível da pilha ... 764
Definição de tags no nível da camada ... 766
Gerenciando tags com o AWS CLI ... 768
Limitações de tag ... 769

Monitoramento ... 770
Usando a Amazon CloudWatch ... 770
Usando AWS CloudTrail ... 783
Usando o Amazon CloudWatch Logs ... 786
Usando Amazon CloudWatch Events ... 791
Segurança e permissões ... 792
Gerenciamento de permissões de usuário ... 794
Permitindo que AWS OpsWorks as pilhas ajam em seu nome ... 820
Prevenção contra representante confuso ... 825
Especificação de permissões para aplicativos executados em instâncias EC2 ... 829
Gerenciamento do acesso por SSH ... 833
Gerenciamento de atualizações de segurança ... 841
Usar grupos de segurança ... 843
Chef 12 Linux ... 846
Visão geral ... 847
A mudança para o Chef 12 ... 848
Sistemas operacionais compatíveis ... 849
Tipos de instâncias compatíveis ... 850
Mais informações ... 850
Movendo para recipientes de dados ... 850
Versões anteriores do Chef ... 852
Chef 11.10 e versões anteriores para Linux ... 853
Usando AWS OpsWorks Stacks com outros serviços da AWS ... 1297
Uso de um armazenamento de dados de back-end ... 1298
ElastiCache Redis ... 1307
Usar um bucket do Amazon S3 ... 1322
Usando AWS CodePipeline com AWS OpsWorks Stacks ... 1336
Usando a CLI do AWS OpsWorks Stacks ... 1399
Criar uma instância do ... 1401
Implantar um aplicativo ... 1405
Listar aplicativos ... 1406
Comandos de lista ... 1407
Listar implantações ... 1408
Listar endereços IP elásticos ... 1410
Listar as instâncias ... 1411
Listar pilhas ... 1412

Listar camadas ... 1414
Executar uma receita ... 1418
Instalar dependências ... 1419
Atualizar uma configuração da pilha ... 1420
Guia de depuração e solução de problemas ... 1421
Depurar receitas ... 1422
Depuração e solução de problemas comuns ... 1440
AWS OpsWorks CLI do Stacks Agent ... 1450
agent_report ... 1453
get_json ... 1453
instance_report ... 1458
list_commands ... 1459
run_command ... 1460
show_log ... 1461
stack_state ... 1462
AWS OpsWorks Referência do Stacks Data Bag ... 1465
Data bag do aplicativo (aws_opsworks_app) ... 1469
Data bag do comando (aws_opsworks_command) ... 1473
Data bag do cluster do Amazon ECS (aws_opsworks_ecs_cluster) ... 1475
Data Bag do Elastic Load Balancing (aws_opsworks_elastic_load_balancer) ... 1476
Data bag da instância (aws_opsworks_instance) ... 1477
Data bag de camada (aws_opsworks_layer) ... 1483
Data bag do Amazon RDS (aws_opsworks_rds_db_instance) ... 1485
Data bag da pilha (aws_opsworks_stack) ... 1486
Data bag do usuário (aws_opsworks_user) ... 1488
OpsWorks Mudanças no agente ... 1490
Versões do agente do Chef 12 ... 1490
Versões do agente do Chef 11.10 ... 1494
Recursos ... 1501
Guias de referência, Ferramentas e Recursos de suporte ... 1501
AWS Kits de desenvolvimento de software ... 1502
Software de código aberto ... 1503
AWS OpsWorks histórico do documento ... 1504
Atualizações anteriores ... 1512
... mdxvii

O que é AWS OpsWorks?

> **Important**
>
> Os AWS OpsWorks serviços chegaram ao fim da vida útil e foram desativados para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks é um serviço de gerenciamento de configuração que ajuda você a configurar e operar aplicativos em uma empresa na nuvem usando o Puppet ou o Chef. AWS OpsWorks Empilha e AWS OpsWorks for Chef Automate permite que você use os livros de receitas e soluções do Chef para gerenciamento de configuração, enquanto que OpsWorks para o Puppet Enterprise permite que você configure um servidor mestre do Puppet Enterprise no. AWS O Puppet oferece um conjunto de ferramentas para impor o estado desejado de sua infraestrutura e automatizar tarefas sob demanda.

AWS OpsWorks Serviços

AWS OpsWorks para Puppet Enterprise

OpsWorks for Puppet Enterprise permite que você crie AWS servidores master Puppet gerenciados. Um servidor master Puppet gerencia nós em sua infraestrutura, armazena fatos sobre esses nós e serve de repositório central para seus módulos do Puppet. Os módulos são unidades reutilizáveis e compartilháveis de código do Puppet que contêm instruções sobre como sua infraestrutura deve ser configurada. Você pode fazer download de módulos da comunidade em Puppet Forge, ou usar o Kit de desenvolvimento do Puppet para criar seus próprios módulos personalizados e, em seguida, gerenciar sua implantação de código com o Gerenciador de códigos do Puppet.

OpsWorks for Puppet Enterprise fornece um Puppet master totalmente gerenciado, um conjunto de ferramentas de automação que permite inspecionar, entregar, operar e preparar seus aplicativos para o futuro, além de acessar uma interface de usuário que permite visualizar informações sobre seus nós e atividades do Puppet. OpsWorks for Puppet Enterprise permite

que você use o Puppet para automatizar a forma como os nós são configurados, implantados e gerenciados, sejam eles EC2 instâncias da Amazon ou dispositivos locais. Um master OpsWorks for Puppet Enterprise fornece automação completa ao lidar com tarefas como configurações de software e sistema operacional, instalações de pacotes, configurações de banco de dados, gerenciamento de mudanças, aplicação de políticas, monitoramento e garantia de qualidade.

Como OpsWorks o Puppet Enterprise gerencia o software Puppet Enterprise, seu servidor pode ser copiado automaticamente no momento que você escolher, está sempre executando a versão mais recente do Puppet compatível com a AWS e sempre tem as atualizações de segurança mais recentes aplicadas. Você pode usar grupos do Amazon EC2 Auto Scaling para associar novos EC2 nós da Amazon ao seu servidor automaticamente.

AWS OpsWorks para Chef Automate

AWS OpsWorks for Chef Automate permite criar AWS servidores Chef gerenciados que incluem recursos premium do Chef Automate e usar o Chef DK e outras ferramentas do Chef para gerenciá-los. Um servidor do Chef gerencia nós em seu ambiente, armazena informações sobre esses nós e serve como um repositório central para seus livros de receitas do Chef. Os livros de receitas contêm receitas executadas pelo agente (`chef-client`) cliente do Chef Infra em cada nó que você gerencia usando o Chef. Você pode usar ferramentas do Chef, como `knife`o Test Kitchen, para gerenciar nós e livros de receitas em um servidor Chef no AWS OpsWorks for Chef Automate serviço.

O Chef Automate é um pacote de software de servidor incluído que fornece fluxo de trabalho automatizado para implantação contínua e verificações de conformidade. AWS OpsWorks for Chef Automate instala e gerencia o Chef Automate, o Chef Infra e o Chef InSpec usando uma única instância do Amazon Elastic Compute Cloud. Com AWS OpsWorks for Chef Automate, você pode usar livros de receitas do Chef personalizados ou criados pela comunidade sem fazer alterações específicas. AWS OpsWorks

Como AWS OpsWorks for Chef Automate gerencia os componentes do Chef Automate em uma única instância, seu servidor pode ser copiado automaticamente no momento que você escolher, está sempre executando a versão secundária mais atual do Chef e sempre tem as atualizações de segurança mais recentes aplicadas. Você pode usar grupos do Amazon EC2 Auto Scaling para associar novos EC2 nós da Amazon ao seu servidor automaticamente.

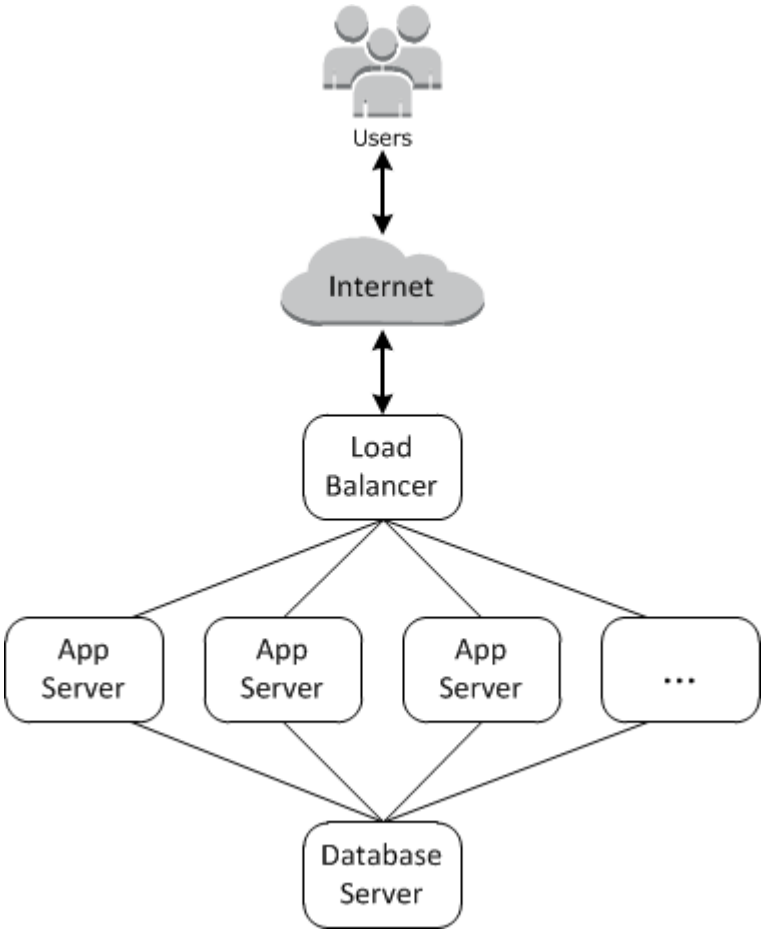
AWS OpsWorks Pilhas

A computação baseada em nuvem geralmente envolve grupos de recursos da AWS, como EC2 instâncias e instâncias do Amazon Relational Database Service (RDS). Por exemplo, um

aplicativo da Web geralmente requer servidores de aplicativos, servidores de banco de dados, load balancers e outros recursos. Este grupo de instâncias é geralmente chamado de pilha.

AWS OpsWorks Stacks, o serviço original, fornece uma maneira simples e flexível de criar e gerenciar pilhas e aplicativos. AWS OpsWorks O Stacks permite que você implante e monitore aplicativos em suas pilhas. Você pode criar pilhas que ajudam a gerenciar recursos na nuvem em grupos especializados denominados layers. Uma camada representa um conjunto de EC2 instâncias que servem a uma finalidade específica, como servir aplicativos ou hospedar um servidor de banco de dados. As layers dependem de receitas do Chef para gerenciar tarefas como instalação de pacotes nas instâncias, implantação de aplicativos e execução de scripts.

Ao contrário AWS OpsWorks for Chef Automate, o AWS OpsWorks Stacks não exige nem cria servidores Chef; o AWS OpsWorks Stacks executa parte do trabalho de um servidor Chef para você. AWS OpsWorks O Stacks monitora a saúde da instância e provisiona novas instâncias para você quando necessário, usando a Autorrecuperação e o Auto Scaling (Ajuste de escala automático). Uma simples pilha de servidor de aplicativo pode ter a aparência do seguinte diagrama.



AWS OpsWorks para Puppet Enterprise

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

OpsWorks for Puppet Enterprise permite que você inicie um master do Puppet Enterprise em minutos e AWS OpsWorks gerencie suas operações, backups, restaurações e atualizações de software. OpsWorks for Puppet Enterprise permite que você se concentre nas principais tarefas de gerenciamento de configuração, em vez de gerenciar um Puppet master. Ao usar o OpsWorks Puppet Enterprise, você pode usar as mesmas configurações para gerenciar sua infraestrutura local e na nuvem, ajudando você a escalar com eficiência suas operações em um ambiente híbrido. O gerenciamento de seu Puppet master é simplificado pelo console do Puppet Enterprise, no AWS Management Console e na AWS CLI.

Um Puppet master gerencia a configuração de nós em seu ambiente, servindo catálogos de configuração para nós específicos para o software `puppet-agent` e serve como um repositório central para os módulos do Puppet. Um Puppet master no OpsWorks Puppet Enterprise é `puppet-agent` implantado em seus nós gerenciados e fornece recursos premium do Puppet Enterprise.

Um master OpsWorks for Puppet Enterprise é executado em uma instância do Amazon Elastic Compute Cloud. OpsWorks Os servidores do Puppet Enterprise estão configurados para executar a versão mais recente do Amazon Linux (Amazon Linux 2) e a versão mais atual do Puppet Enterprise Master, versão 2019.8.5. Para obter mais informações sobre as alterações no Puppet Enterprise 2019.8.5, consulte as Notas de versão do Puppet Enterprise.

Quando novas versões do software do Puppet forem disponibilizadas, a manutenção do sistema é projetada para atualizar a versão do Puppet Enterprise automaticamente no servidor, assim que ela passar nos testes da AWS. A AWS realiza testes abrangentes para verificar se as atualizações do Puppet estão prontas para produção e não atrapalham os ambientes existentes de clientes.

Você pode conectar qualquer computador ou EC2 instância local que esteja executando um sistema operacional compatível e tenha acesso à rede a um OpsWorks mestre do Puppet Enterprise. O software do agente `puppet` é instalado pelo Puppet master nos nós que você deseja gerenciar.

Tópicos

- Support regional OpsWorks para for Puppet Enterprise
- AWS OpsWorks for Puppet Enterprise Fim da vida FAQs
- Introdução ao OpsWorks Puppet Enterprise
- Crie um AWS OpsWorks for Puppet Enterprise mestre usando AWS CloudFormation
- Atualize um OpsWorks para o Puppet Enterprise Server para usar um domínio personalizado
- Trabalhando com tags em AWS OpsWorks for Puppet Enterprise recursos
- Fazer o backup e a restauração de um servidor do OpsWorks para Puppet Enterprise
- Manutenção do sistema OpsWorks para Puppet Enterprise
- Adicionando nós automaticamente OpsWorks para o Puppet Enterprise
- Desassociar um Node de um OpsWorks for Puppet Enterprise Server
- Excluir um OpsWorks para o Puppet Enterprise Server
- Como migrar um servidor OpsWorks for Puppet Enterprise para o Amazon Elastic Compute Cloud (Amazon) EC2
- Registro OpsWorks de chamadas de API do Puppet Enterprise com AWS CloudTrail
- Solução de problemas OpsWorks do Puppet Enterprise

Support regional OpsWorks para for Puppet Enterprise

Os seguintes endpoints regionais oferecem suporte OpsWorks para mestres do Puppet Enterprise. OpsWorks for Puppet Enterprise cria recursos associados aos seus mestres do Puppet, como perfis de instância, usuários e funções de serviço, no mesmo endpoint regional do seu mestre do Puppet. O Puppet master deve estar em uma VPC. Você pode criar uma VPC, usar a que já tem ou optar pela VPC padrão.

- Região Leste dos EUA (Ohio)
- Região Leste dos EUA (Norte da Virgínia)
- Região Oeste dos EUA (Norte da Califórnia)

- Região Oeste dos EUA (Oregon)
- Região Ásia-Pacífico (Tóquio)
- Região Ásia-Pacífico (Singapura)
- Região Ásia-Pacífico (Sydney)
- Região Europa (Frankfurt)
- Região Europa (Irlanda)

AWS OpsWorks for Puppet Enterprise Fim da vida FAQs

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Tópicos

- Como os clientes existentes serão afetados por esse fim da vida útil?
- O que acontecerá aos meus servidores se eu não tomar nenhuma ação?
- Está AWS OpsWorks for Puppet Enterprise aceitando novos clientes?
- O fim da vida afetará tudo Regiões da AWS ao mesmo tempo?
- Para qual nível de suporte técnico está disponível AWS OpsWorks for Puppet Enterprise?
- Sou cliente atual da OpsWorks Puppet Enterprise e preciso iniciar um servidor em uma conta que não estava usando o serviço anteriormente. Eu sou capaz de fazer isso?
- Haverá algum lançamento de novos recursos para AWS OpsWorks for Puppet Enterprise?

Como os clientes existentes serão afetados por esse fim da vida útil?

Os clientes existentes não serão afetados até 31 de março de 2024, a data de fim da vida útil do Puppet OpsWorks Enterprise. Após a data de fim da vida útil, os clientes não poderão mais gerenciar seus servidores usando o OpsWorks console ou a API.

O que acontecerá aos meus servidores se eu não tomar nenhuma ação?

A partir de 31 de março de 2024, você não poderá mais gerenciar seus servidores usando o OpsWorks console ou a API. Nesse momento, deixaremos de realizar quaisquer funções de gerenciamento contínuas de seus servidores, como backups ou manutenção. Para limitar o impacto para os clientes, deixaremos as EC2 instâncias em execução que fazem backup dos servidores do Puppet Enterprise, mas suas licenças não serão mais válidas, pois o uso não é mais coberto (ou cobrado) pelo contrato de serviço do OpsWorks Puppet Enterprise. Se você quiser continuar gerenciando sua infraestrutura com o Puppet Enterprise, consulte Como migrar um servidor OpsWorks for Puppet Enterprise para o Amazon Elastic Compute Cloud (Amazon). EC2

Está AWS OpsWorks for Puppet Enterprise aceitando novos clientes?

Não. AWS OpsWorks for Puppet Enterprise não está mais aceitando novos clientes.

O fim da vida afetará tudo Regiões da AWS ao mesmo tempo?

Sim. A API e o console chegarão ao fim da vida útil e ficarão inutilizáveis a partir de 31 de março de 2024 em todas as regiões. Para obter uma lista de Regiões da AWS onde AWS OpsWorks for Puppet Enterprise está disponível, consulte Lista de serviços AWS regionais.

Para qual nível de suporte técnico está disponível AWS OpsWorks for Puppet Enterprise?

AWS continuarão a fornecer o mesmo nível de suporte AWS OpsWorks for Puppet Enterprise que os clientes têm hoje até a data de fim da vida útil. Se você tiver dúvidas ou preocupações, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Sou cliente atual da OpsWorks Puppet Enterprise e preciso iniciar um servidor em uma conta que não estava usando o serviço anteriormente. Eu sou capaz de fazer isso?

Geralmente não, a menos que haja circunstâncias excepcionais para fazer isso. Se você tiver uma situação especial, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support com os detalhes e a justificativa para isso e analisaremos sua solicitação.

Haverá algum lançamento de novos recursos para AWS OpsWorks for Puppet Enterprise?

Não. Como o serviço está chegando ao fim da vida útil, não lançaremos nenhum atributo novo. No entanto, continuaremos fazendo melhorias na segurança e gerenciando os servidores conforme o esperado até a data de fim da vida útil.

Introdução ao OpsWorks Puppet Enterprise

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

OpsWorks for Puppet Enterprise permite que você execute um servidor Puppet Enterprise em. AWS Você pode provisionar um servidor master Puppet Enterprise em aproximadamente 15 minutos.

A partir de 3 de maio de 2021, OpsWorks o for Puppet Enterprise armazena alguns atributos do servidor Puppet Enterprise em. AWS Secrets Manager Para obter mais informações, consulte Integração com AWS Secrets Manager.

O passo a passo a seguir ajuda você a criar seu primeiro mestre do Puppet no OpsWorks Puppet Enterprise.

Pré-requisitos

Antes de começar, conclua os pré-requisitos a seguir.

Tópicos

- Instalar o Kit de desenvolvimento do Puppet
- Instalar as ferramentas de cliente do Puppet Enterprise
- Configurar um repositório de controle do Git
- Configuração de um VPC

- Configurar um par de EC2 chaves (opcional)
- Pré-requisitos para usar um domínio personalizado (opcional)

Instalar o Kit de desenvolvimento do Puppet

1. No site do Puppet, faça download do Kit de desenvolvimento do Puppet que corresponde ao seu sistema operacional.
2. Instale o Kit de desenvolvimento do Puppet.
3. Adicione o Kit de desenvolvimento do Puppet na variável `PATH` do seu computador local.

   - Em um sistema operacional Linux ou macOS, você pode adicionar o Kit de desenvolvimento do Puppet à sua variável `PATH`, executando o seguinte comando em um shell Bash.

     ```
     echo 'export PATH=/opt/puppetlabs/pdk/bin/pdk:$PATH' >> ~/.bash_profile && source
      ~/.bash_profile
     ```

   - Em um sistema operacional baseado em Windows, você pode adicionar o Puppet Development Kit à sua **`PATH`** variável usando o seguinte comando.NET Framework em uma PowerShell sessão ou na caixa de diálogo Variáveis de Ambiente acessível em Propriedades do Sistema. Talvez seja necessário executar sua PowerShell sessão como administrador para executar o comando a seguir.

     ```
     [Environment]::SetEnvironmentVariable("Path","new path value","Machine")
     ```

Instalar as ferramentas de cliente do Puppet Enterprise

As ferramentas de cliente do Puppet Enterprise (PE) são um conjunto de ferramentas de linha de comando que permitem que você acesse os serviços do Puppet Enterprise da estação de trabalho. As ferramentas podem ser instaladas em diversos sistemas operacionais, e eles também podem ser instalados nos nós que você gerencia usando o Puppet. Para obter informações sobre os sistemas operacionais suportados para as ferramentas e como instalá-los, consulte Installing PE client tools na documentação do Puppet Enterprise.

Configurar um repositório de controle do Git

Antes de iniciar um Puppet master, você deve ter um repositório de controle configurado no Git para gerenciar o armazenamento e a alteração de suas classes e módulos do Puppet. Um URL

para um repositório Git e informações de uma conta SSH ou HTTPS para acessar o repositório são necessários nas etapas para iniciar o servidor Puppet Enterprise master. Para obter mais informações sobre como configurar um repositório de controle que o Puppet Enterprise master usará, consulte Setting up a control repository. Você também pode encontrar instruções de configuração do repositório de controle no readme do repositório de `control-repoamostra` do Puppet em. GitHub A estrutura do repositório de controle se parece com isto.

```
### LICENSE
### Puppetfile
### README.md
### environment.conf
### hieradata
#   ### common.yaml
#   ### nodes
#       ### example-node.yaml
### manifests
#   ### site.pp
### scripts
#   ### code_manager_config_version.rb
#   ### config_version.rb
#   ### config_version.sh
### site
    ### profile
    #   ### manifests
    #       ### base.pp
    #       ### example.pp
    ### role
        ### manifests
            ### database_server.pp
            ### example.pp
            ### webserver.pp
```

Configurando um repositório usando CodeCommit

Você pode criar um novo repositório usando o. CodeCommit Para obter mais informações sobre como usar CodeCommit para criar seu repositório de controle, consulte the section called “Opcional: Use CodeCommit” este guia. Para obter mais informações sobre como começar a usar o Git on CodeCommit, consulte Getting started with AWS. CodeCommit Para autorizar seu OpsWorks servidor do Puppet Enterprise para seu repositório, anexe a `AWSCodeCommitReadOnly` política à sua função de perfil da instância do IAM.

Configuração de um VPC

Seu OpsWorks mestre do For Puppet Enterprise deve operar em uma Amazon Virtual Private Cloud. É possível adicioná-lo a um VPC existente, usar o VPC padrão ou criar um novo VPC para conter o servidor. Para obter informações sobre a Amazon VPC e como criar uma nova VPC, consulte o Guia de conceitos básicos da Amazon VPC.

Se você criar seu próprio VPC ou usar um existente, ele deve ter as configurações ou propriedades a seguir.

- A VPC deve ter no mínimo uma sub-rede.

  Se o seu master OpsWorks for Puppet Enterprise estiver acessível ao público, torne a sub-rede pública e ative a atribuição automática de IP público.
- DNS resolution deve ser habilitada.
- Na sub-rede, ative Auto-assign public IP.

Configurar um par de EC2 chaves (opcional)

Uma conexão SSH não é necessária ou recomendada para o gerenciamento típico do servidor Puppet; você pode usar os AWS CLI comandos AWS Management Console e para realizar várias tarefas de gerenciamento no seu servidor Puppet.

É necessário um par de EC2 chaves para se conectar ao seu servidor usando SSH, caso você perca ou queira alterar a senha de login do console baseado na web do Puppet Enterprise. Você pode usar um par de chaves existente ou criar um novo par de chaves. Para obter mais informações sobre como criar um novo par de EC2 chaves, consulte Amazon EC2 Key Pairs.

Se você não precisar de um par de EC2 chaves, você está pronto para criar um mestre do Puppet Enterprise.

Pré-requisitos para usar um domínio personalizado (opcional)

É possível configurar o servidor Puppet Enterprise mestre em seu próprio domínio especificando um endpoint público em um domínio personalizado para usar como o endpoint do servidor. Quando você usa um domínio personalizado, todos os itens a seguir são necessários, conforme descrito em detalhes nesta seção.

Tópicos

- Configurar um domínio personalizado
- Obter um certificado
- Obter uma chave privada

Configurar um domínio personalizado

Para executar o servidor Puppet Enterprise mestre em seu próprio domínio personalizado, será necessário um endpoint público de um servidor, como `https://aws.my-company.com`. Se você especificar um domínio personalizado, também deverá fornecer um certificado e uma chave privada, conforme descrito nas seções anteriores.

Para acessar o servidor depois de criá-lo, inclua um registro DNS CNAME em seu serviço DNS preferido. Esse registro deve apontar o domínio personalizado para o endpoint (o valor do atributo `Endpoint` do servidor) que é gerado pelo processo de criação do Puppet mestre. Você não poderá acessar o servidor usando o valor de `Endpoint` gerado se o servidor estiver usando um domínio personalizado.

Obter um certificado

Para configurar o Puppet mestre em seu próprio domínio personalizado, será necessário um certificado HTTPS no formato PEM. Ele pode ser um certificado autoassinado único ou uma cadeia de certificados. Ao concluir o fluxo de trabalho Create Puppet Enterprise Master (Criar Puppet Enterprise mestre), se você especificar esse certificado, também deverá fornecer um domínio personalizado e uma chave privada.

Veja a seguir os requisitos para o valor do certificado:

- Você pode fornecer um certificado autoassinado personalizado ou a cadeia de certificados completa.
- O certificado deve ser um certificado X509 válido ou uma cadeia de certificados no formato PEM.
- O certificado deve ser válido no momento do upload. Não é possível usar um certificado antes de seu período de validade começar (a data `NotBefore` do certificado) ou após sua expiração (a data `NotAfter` do certificado).
- O nome comum do certificado ou os nomes alternativos do assunto (SANs), se presentes, devem corresponder ao valor do domínio personalizado.
- O certificado deve corresponder ao valor do campo Custom private key (Chave privada personalizada).

Obter uma chave privada

Para configurar o Puppet mestre em seu próprio domínio personalizado, será necessária uma chave privada no formato PEM para se conectar ao servidor usando HTTPS. A chave privada não deve ser criptografada e não pode ser protegida por senha nem por frase secreta. Se especificar uma chave privada personalizada, também precisará fornecer um domínio personalizado e um certificado.

Criar um Puppet Enterprise master

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

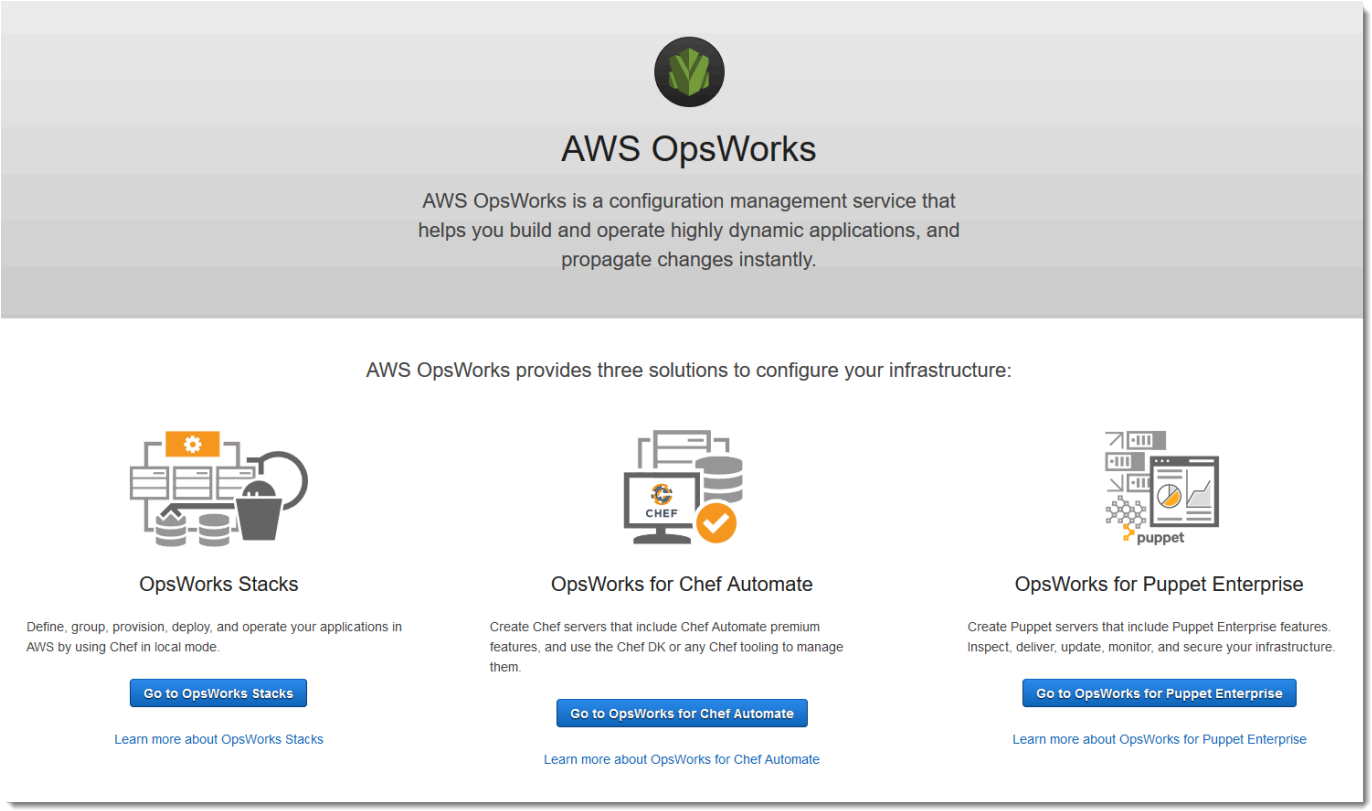
Você pode criar um Puppet master usando o console OpsWorks for Puppet Enterprise ou o. AWS CLI

Tópicos

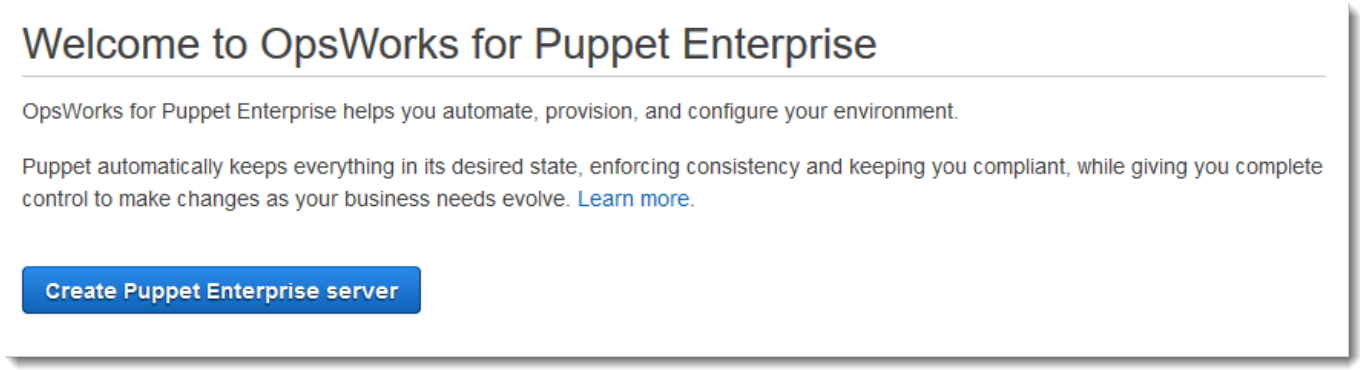
- Crie um Puppet Enterprise Master usando o AWS Management Console
- Crie um Puppet Enterprise Master usando o AWS CLI

Crie um Puppet Enterprise Master usando o AWS Management Console

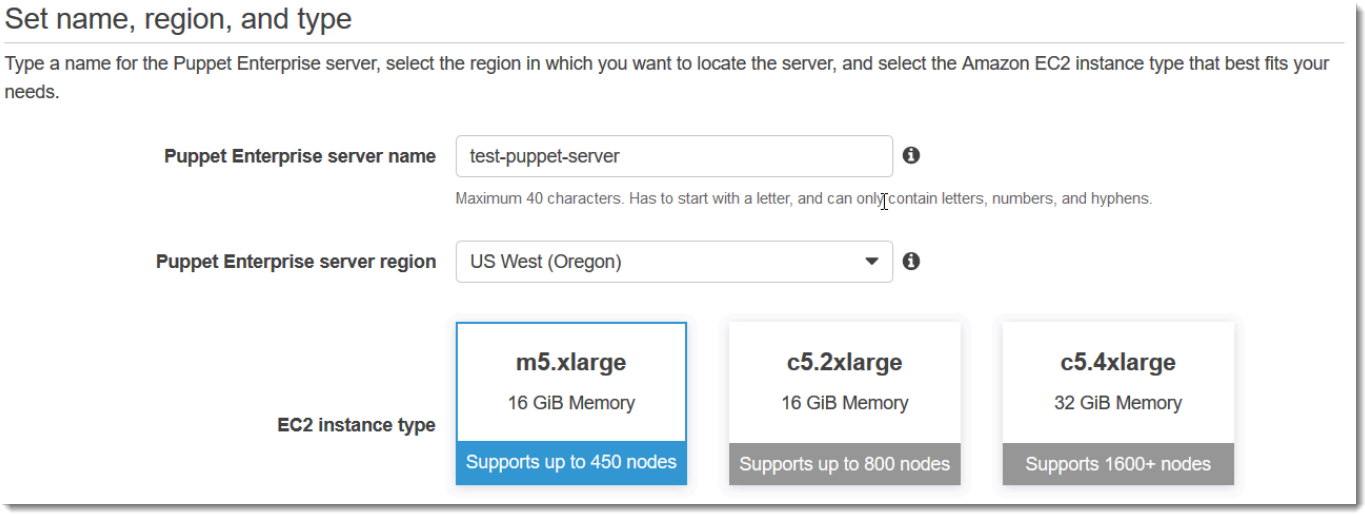
1. Faça login no AWS Management Console e abra o AWS OpsWorks console em https://console.aws.amazon.com/opsworks/.
2. Na página AWS OpsWorks inicial, escolha Ir OpsWorks para o Puppet Enterprise.



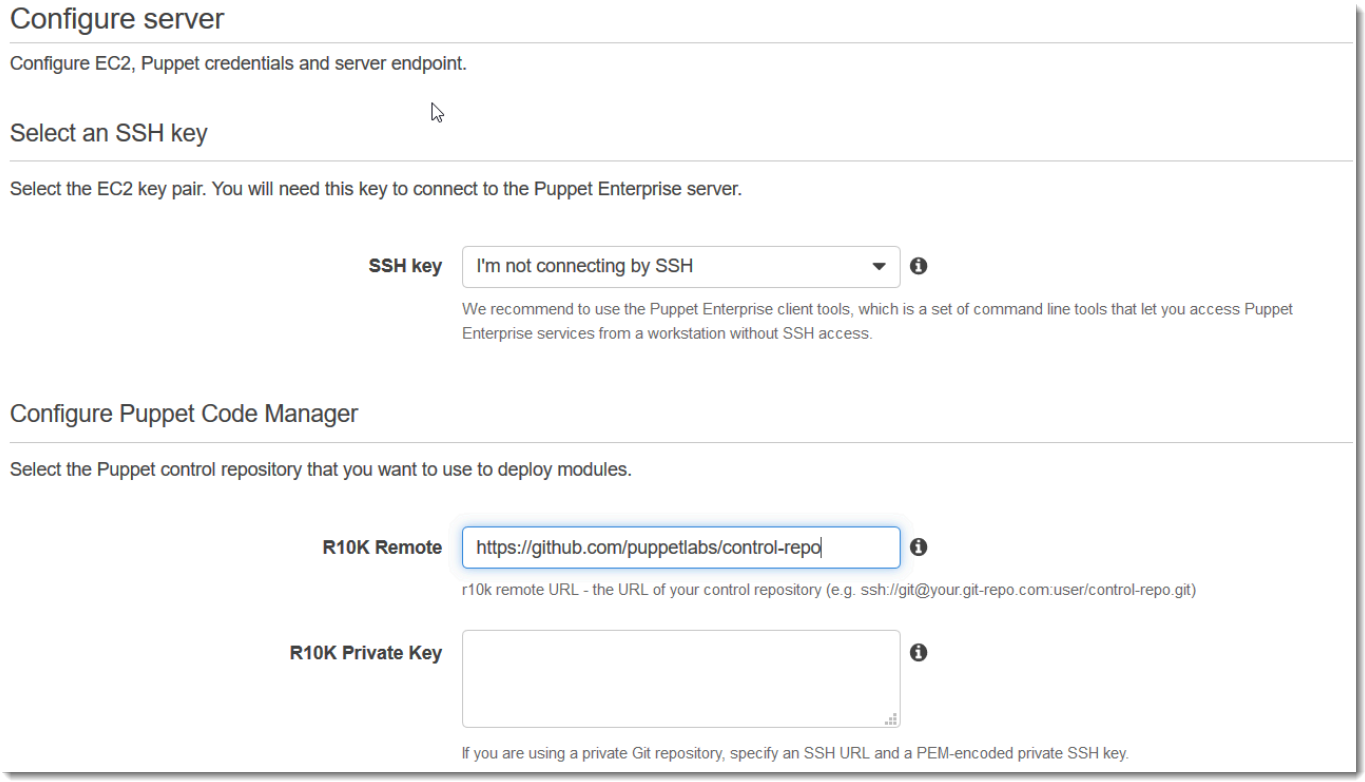
3. Na página inicial do OpsWorks Puppet Enterprise, escolha Criar servidor Puppet Enterprise.



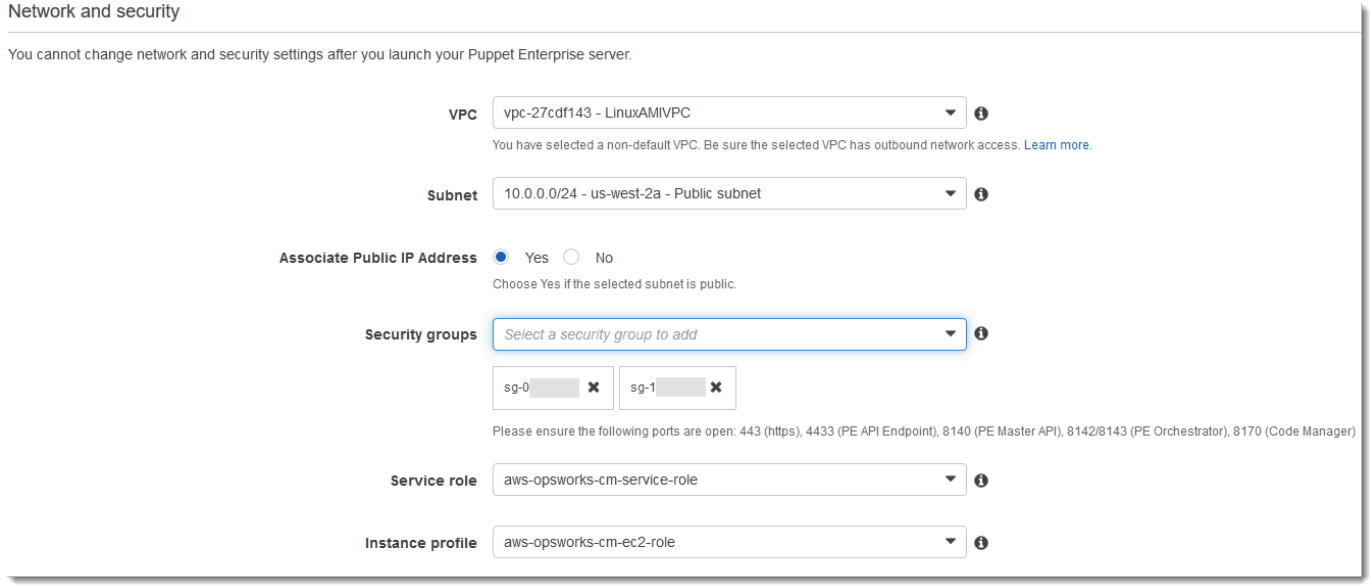
4. Na página Set name, region, and type, especifique um nome para o seu servidor. Os nomes do Puppet master podem ter no máximo 40 caracteres, devem iniciar com uma letra e podem conter somente caracteres alfanuméricos e traços. Selecione uma região compatível e, em seguida, escolha um tipo de instância compatível com o número de nós que deseja gerenciar. É possível alterar o tipo de instância após a criação do servidor, se necessário. Para esta demonstração, estamos criando um tipo de instância m5.large na região Oeste dos EUA (Oregon). Escolha Próximo.



5. Na página Configure server (Configurar servidor), deixe a seleção padrão na lista suspensa SSH key (Chave SSH), a não ser que você deseje especificar um nome de par de chaves. No campo r10k remote (r10k remoto) da área Configure Puppet Code Manager (Configurar gerenciador de códigos do Puppet), especifique um URL de SSH ou HTTPS válido do Git remoto. No campo chave privada r10k, cole a chave privada SSH que AWS OpsWorks pode ser usada para acessar o repositório remoto r10k. Isso é fornecido pelo Git quando você cria um repositório privado, mas não será necessário se você estiver usando a autenticação HTTPS para acessar seu repositório de controle. Escolha Próximo.

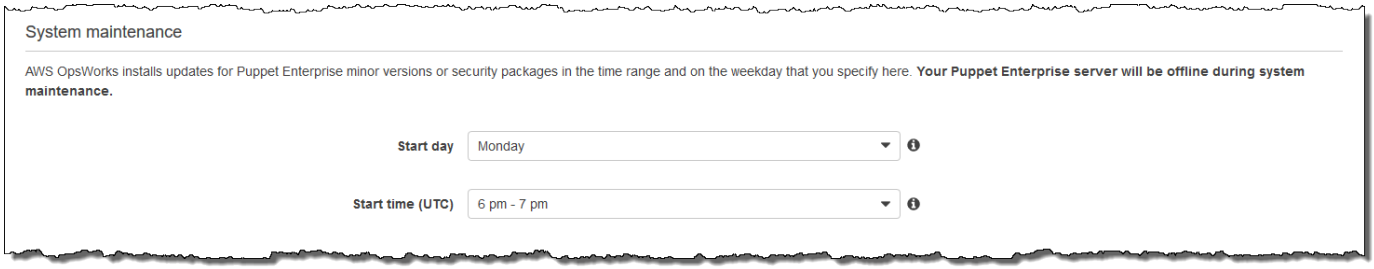


6. Em Specify server endpoint (Especificar endpoint do servidor), deixe o padrão, Use an automatically-generated endpoint (Usar um endpoint gerado automaticamente) e escolha Next (Avançar), a menos que você queira que o servidor esteja em um domínio personalizado. Para configurar um domínio personalizado, vá para a próxima etapa.
7. Para usar um domínio personalizado, em Specify server endpoint (Especificar endpoint do servidor), escolha Use a custom domain (Usar um domínio personalizado) na lista suspensa.
   a. Em Fully qualified domain name (FQDN) (Nome de domínio totalmente qualificado [FQDN]), especifique um FQDN. Você deve ser proprietário do nome de domínio que usará.
   b. Em SSL certificate (Certificado SSL), cole todo o certificado em formato PEM, começando com `-----BEGIN CERTIFICATE-----` e terminando com `-----END CERTIFICATE-----`. O assunto do certificado SSL deve corresponder ao FQDN inserido na etapa anterior. Remova todas as linhas extras antes e depois do certificado.
   c. Em SSL private key (Chave privada SSL), cole toda a chave privada RSA, começando com `-----BEGIN RSA PRIVATE KEY-----` e terminando com `-----END RSA PRIVATE KEY-----`. A chave privada SSL precisa corresponder à chave pública do certificado SSL inserido na etapa anterior. Remova todas as linhas extras antes e depois da chave privada. Escolha Próximo.
8. Na página Definir configurações avançadas, na área Rede e segurança, escolha uma VPC, uma sub-rede e um ou mais grupos de segurança. AWS OpsWorks pode gerar um grupo de segurança, uma função de serviço e um perfil de instância para você, caso ainda não tenha alguns que queira usar. O servidor pode ser membro de vários grupos de segurança. Não é possível alterar as configurações de rede e segurança do servidor Puppet master depois de sair desta página.

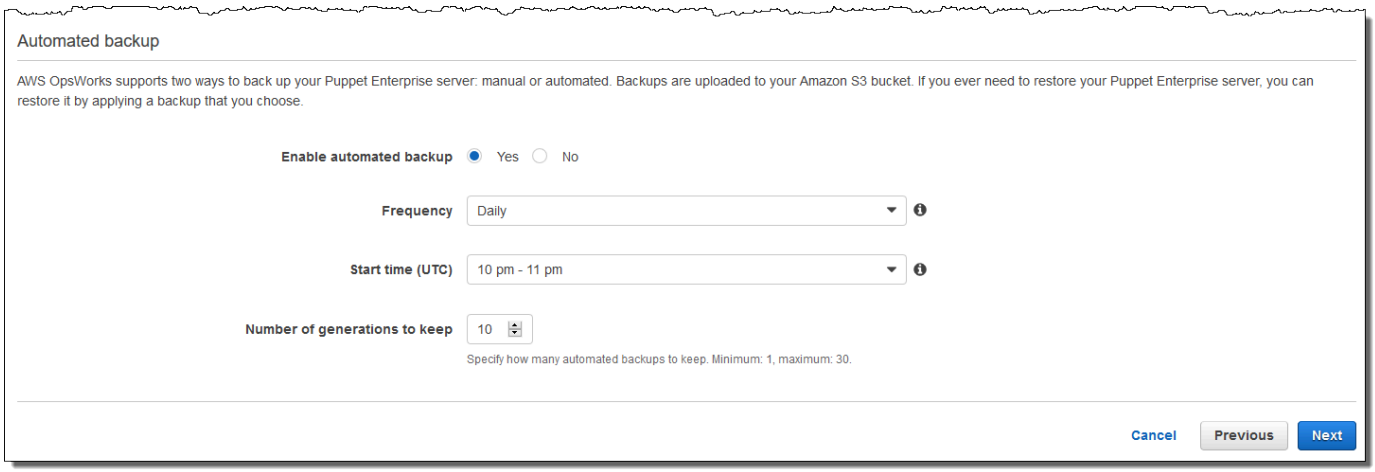


9. Na seção System maintenance, defina o dia e a hora em que você deseja que a manutenção do sistema se inicie. Como é recomendável que o servidor esteja offline durante a manutenção do sistema, escolha um horário de baixa demanda do servidor no horário comercial regular.

   A janela de manutenção é necessária. Você pode alterar o dia e a hora de início posteriormente usando o AWS Management Console AWS CLI, ou APIs o.



10. Configure os backups. Por padrão, os backups automáticos estão ativados. Defina uma preferência de frequência e hora para o início do backup automático e defina o número de gerações de backup a armazenar no Amazon Simple Storage Service. Um máximo de 30 backups podem ser mantidos; quando o máximo é atingido, OpsWorks pois o Puppet Enterprise exclui os backups mais antigos para abrir espaço para novos.



11. (Opcional) Em Tags, adicione tags ao servidor e aos recursos relacionados, como EC2 instância, endereço IP elástico, grupo de segurança, bucket do S3 e backups. Para obter mais informações sobre como marcar um servidor OpsWorks para o Puppet Enterprise, consulte. Trabalhando com tags em AWS OpsWorks for Puppet Enterprise recursos
12. Ao concluir a configuração das definições avançadas, selecione Next.
13. Na página Analisar, examine suas escolhas. Quando você estiver pronto para criar o servidor, escolha Launch.

    Enquanto você espera AWS OpsWorks para criar seu Puppet master, faça o download do Starter Kit Configurar o Puppet master usando o Starter Kit e das credenciais do console Puppet Enterprise. Não espere até que o servidor esteja online para fazer o download desses itens.

    Quando a criação do servidor estiver concluída, seu mestre do Puppet estará disponível na OpsWorks página inicial do Puppet Enterprise, com o status de online. Depois que o servidor estiver online, o console do Puppet Enterprise estará disponível no domínio do servidor, em um URL no seguinte formato: `https://your_server_name-randomID.region.opsworks-cm.io`.

Crie um Puppet Enterprise Master usando o AWS CLI

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se

você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Criar um servidor mestre OpsWorks para o Puppet Enterprise executando AWS CLI comandos é diferente de criar um servidor no console. No console, AWS OpsWorks cria uma função de serviço e um grupo de segurança para você, se você não especificar os existentes que deseja usar. No AWS CLI, AWS OpsWorks pode criar um grupo de segurança para você se você não especificar um, mas ele não cria automaticamente uma função de serviço; você deve fornecer um ARN da função de serviço como parte do seu `create-server` comando. No console, ao AWS OpsWorks criar seu Puppet master, você baixa o kit inicial e as credenciais de login para o console do Puppet Enterprise. Como você não pode fazer isso ao criar um mestre OpsWorks para o Puppet Enterprise usando o AWS CLI, você usa um utilitário de processamento JSON para obter as credenciais de login e o kit inicial a partir dos resultados do `create-server` comando depois que seu novo OpsWorks mestre do Puppet Enterprise estiver online.

Se o seu computador local ainda não estiver executando o AWS CLI, faça o download e instale-o AWS CLI seguindo as instruções de instalação no Guia do usuário da interface de linha de comando da AWS. Esta seção não descreve todos os parâmetros que você pode usar com o comando `create-server`. Para obter mais informações sobre os parâmetros do `create-server`, consulte `create-server` na Referência da AWS CLI .

1. Conclua o Pré-requisitos. Para criar o Puppet master, você precisa de um ID de sub-rede, portanto, você deve ter uma VPC.
2. Crie uma função de serviço e um perfil de instância.
3. Encontre e copie as funções ARNs de serviço em sua conta.

   ```
   aws iam list-roles --path-prefix "/service-role/" --no-paginate
   ```

   Nos resultados do comando `list-roles`, procure pelas entradas de ARN da função de serviço semelhantes às seguintes. Anote a função de serviço ARNs. Você precisa desses valores para criar o Puppet Enterprise mestre.

   ```
   {
       "AssumeRolePolicyDocument": {
           "Version": "2012-10-17",
           "Statement": [
               {
   ```

```
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "ec2.amazonaws.com"
                }
            }
        ]
    },
    "RoleId": "AROZZZZZZZZZZQG6R22HC",
    "CreateDate": "2018-01-05T20:42:20Z",
    "RoleName": "aws-opsworks-cm-ec2-role",
    "Path": "/service-role/",
    "Arn": "arn:aws:iam::000000000000:role/service-role/aws-opsworks-cm-ec2-role"
},
{
    "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "opsworks-cm.amazonaws.com"
                }
            }
        ]
    },
    "RoleId": "AROZZZZZZZZZZZZZZZ6QE",
    "CreateDate": "2018-01-05T20:42:20Z",
    "RoleName": "aws-opsworks-cm-service-role",
    "Path": "/service-role/",
    "Arn": "arn:aws:iam::000000000000:role/service-role/aws-opsworks-cm-service-
role"
}
```

4. Encontre e copie os perfis ARNs da instância em sua conta.

```
aws iam list-instance-profiles --no-paginate
```

Nos resultados do comando `list-instance-profiles`, procure pelas entradas de ARN do perfil de instância semelhantes às seguintes. Anote o perfil da instância ARNs. Você precisa desses valores para criar o Puppet Enterprise mestre.

```
{
    "Path": "/",
    "InstanceProfileName": "aws-opsworks-cm-ec2-role",
    "InstanceProfileId": "EXAMPLEDC6UR3LTUW7VHK",
    "Arn": "arn:aws:iam::123456789012:instance-profile/aws-opsworks-cm-ec2-role",
    "CreateDate": "2017-01-05T20:42:20Z",
    "Roles": [
        {
            "Path": "/service-role/",
            "RoleName": "aws-opsworks-cm-ec2-role",
            "RoleId": "EXAMPLEE4STNUQG6R22HC",
            "Arn": "arn:aws:iam::123456789012:role/service-role/aws-opsworks-cm-
ec2-role",
            "CreateDate": "2017-01-05T20:42:20Z",
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            }
        }
    ]
},
```

5. Crie o master OpsWorks for Puppet Enterprise executando o `create-server` comando.

   - O `--engine` valor é `Puppet`, `--engine-model` é `Monolithic` e `--engine-version` pode ser `2019` ou `2017`.
   - O nome do servidor deve ser exclusivo em sua AWS conta, em cada região. Os nomes dos servidores devem começar com uma letra; letras, números ou hífens (-) são permitidos, até um máximo de 40 caracteres.
   - Use o ARN do perfil de instância e o ARN da função de serviço que você copiou nas etapas 3 e 4.

- Os tipos de instância válidos são `m5.xlarge`, `c5.2xlarge` ou `c5.4xlarge`. Para obter mais informações sobre as especificações desses tipos de instância, consulte Tipos de instância no Guia EC2 do usuário da Amazon.
- O parâmetro `--engine-attributes` é opcional; se você não especificar uma senha de administrador do Puppet, o processo de criação do servidor gerará uma para você. Se você adicionar `--engine-attributes`, especifique uma `PUPPET_ADMIN_PASSWORD`, uma senha de administrador para fazer login na página da web do console do Puppet Enterprise. A senha deve ter entre 8 e 32 caracteres ASCII.
- Um par de chaves SSH é opcional, mas pode ajudá-lo a se conectar ao Puppet master se você precisar redefinir a senha de administrador do console. Para obter mais informações sobre a criação de um par de chaves SSH, consulte Amazon EC2 Key Pairs no Amazon EC2 User Guide.
- Para usar um domínio personalizado, adicione os seguintes parâmetros ao comando. Caso contrário, o processo de criação do Puppet mestre gerará automaticamente um endpoint para você. Todos os três parâmetros são necessários para configurar um domínio personalizado. Para obter informações sobre requisitos adicionais para usar esses parâmetros, consulte CreateServera Referência da API AWS OpsWorks CM.
  - `--custom-domain` – um endpoint público opcional de um servidor, como `https://aws.my-company.com`.
  - `--custom-certificate` – um certificado HTTPS no formato PEM. O valor pode ser um certificado autoassinado único ou uma cadeia de certificados.
  - `--custom-private-key` – uma chave privada no formato PEM para a conexão com o servidor usando HTTPS. A chave privada não deve ser criptografada e não pode ser protegida por senha nem por frase secreta.
- É necessário fazer a manutenção semanal do sistema. Os valores válidos devem sempre ser especificados no seguinte formato: `DDD:HH:MM`. A hora especificada está em formato de Tempo Universal Coordenado (UTC). Se você não especificar um valor para `--preferred-maintenance-window`, o valor padrão será aleatório, um período de uma hora na terça, quarta ou sexta-feira.
- Os valores válidos para `--preferred-backup-window` devem ser especificados em um dos seguintes formatos: `HH:MM` para backups diários ou `DDD:HH:MM` para backups semanais. A hora é especificada em UTC. O valor padrão é uma hora de início aleatória estabelecida diariamente. Para cancelar os backups automáticos, adicione o parâmetro `--disable-automated-backup`.

- Para `--security-group-ids`, insira um ou mais grupos de segurança IDs, separados por um espaço.
- Em `--subnet-ids`, insira um ID de sub-rede.

```
aws opsworks-cm create-server --engine "Puppet" --engine-model "Monolithic"
 --engine-version "2019" --server-name "server_name" --instance-profile-arn
 "instance_profile_ARN" --instance-type "instance_type" --engine-attributes
 '{"PUPPET_ADMIN_PASSWORD":"ASCII_password"}' --key-pair "key_pair_name" --
preferred-maintenance-window "ddd:hh:mm" --preferred-backup-window "ddd:hh:mm"
 --security-group-ids security_group_id1 security_group_id2 --service-role-arn
 "service_role_ARN" --subnet-ids subnet_ID
```

Veja um exemplo a seguir.

```
aws opsworks-cm create-server --engine "Puppet" --engine-model
 "Monolithic" --engine-version "2019" --server-name "puppet-02" --
instance-profile-arn "arn:aws:iam::111122223333:instance-profile/aws-
opsworks-cm-ec2-role" --instance-type "m5.xlarge" --engine-attributes
 '{"PUPPET_ADMIN_PASSWORD":"zZZzDj2DLYXSZFRv1d"}' --key-pair "amazon-test"
 --preferred-maintenance-window "Mon:08:00" --preferred-backup-window
 "Sun:02:00" --security-group-ids sg-b00000001 sg-b0000008 --service-role-arn
 "arn:aws:iam::111122223333:role/service-role/aws-opsworks-cm-service-role" --
subnet-ids subnet-383daa71
```

O exemplo a seguir cria um Puppet mestre que usa um domínio personalizado.

```
aws opsworks-cm create-server \
    --engine "Puppet" \
    --engine-model "Monolithic" \
    --engine-version "2019" \
    --server-name "puppet-02" \
    --instance-profile-arn "arn:aws:iam::111122223333:instance-profile/aws-
opsworks-cm-ec2-role" \
    --instance-type "m5.xlarge" \
    --engine-attributes '{"PUPPET_ADMIN_PASSWORD":"zZZzDj2DLYXSZFRv1d"}' \
    --custom-domain "my-puppet-master.my-corp.com" \
    --custom-certificate "-----BEGIN CERTIFICATE----- EXAMPLEqEXAMPLE== -----END
 CERTIFICATE-----" \
    --custom-private-key "-----BEGIN RSA PRIVATE KEY----- EXAMPLEqEXAMPLE= -----END
 RSA PRIVATE KEY-----" \
```

```
    --key-pair "amazon-test"
    --preferred-maintenance-window "Mon:08:00" \
    --preferred-backup-window "Sun:02:00" \
    --security-group-ids sg-b00000001 sg-b0000008 \
    --service-role-arn "arn:aws:iam::111122223333:role/service-role/aws-opsworks-
cm-service-role" \
    --subnet-ids subnet-383daa71
```

O exemplo a seguir cria um Puppet mestre que adiciona duas tags: `Stage: Production` e `Department: Marketing`. Para obter mais informações sobre como adicionar e gerenciar tags nos OpsWorks servidores do Puppet Enterprise, consulte Trabalhando com tags em AWS OpsWorks for Puppet Enterprise recursos este guia.

```
aws opsworks-cm create-server \
    --engine "Puppet" \
    --engine-model "Monolithic" \
    --engine-version "2019" \
    --server-name "puppet-02" \
    --instance-profile-arn "arn:aws:iam::111122223333:instance-profile/aws-
opsworks-cm-ec2-role" \
    --instance-type "m5.xlarge" \
    --engine-attributes '{"PUPPET_ADMIN_PASSWORD":"zZZzDj2DLYXSZFRv1d"}' \
    --key-pair "amazon-test"
    --preferred-maintenance-window "Mon:08:00" \
    --preferred-backup-window "Sun:02:00" \
    --security-group-ids sg-b00000001 sg-b0000008 \
    --service-role-arn "arn:aws:iam::111122223333:role/service-role/aws-opsworks-
cm-service-role" \
    --subnet-ids subnet-383daa71 \
    --tags [{\"Key\":\"Stage\",\"Value\":\"Production\"},{\"Key\":\"Department\",
\"Value\":\"Marketing\"}]
```

6. OpsWorks para Puppet Enterprise leva cerca de 15 minutos para criar um novo servidor. Não descarte a saída do comando `create-server` ou feche a sessão de shell, pois a saída pode conter informações importantes que não serão exibidas novamente. Para obter as senhas e o starter kit dos resultados do comando `create-server`, vá para a próxima etapa.

   Se você estiver usando um domínio personalizado com o servidor, na saída do comando `create-server`, copie o valor do atributo `Endpoint`. Veja um exemplo a seguir.

```
"Endpoint": "puppet-07-exampleexample.opsworks-cm.us-east-1.amazonaws.com"
```

7. Se você optou OpsWorks por fazer com que o Puppet Enterprise gerasse uma senha para você, você pode extraí-la dos `create-server` resultados em um formato utilizável usando um processador JSON, como jq. Após instalar o jq, você pode executar os comandos a seguir para extrair a senha de administrador do Puppet e o starter kit. Se você não forneceu sua própria senha na etapa 3, salve a senha de administrador extraída em um local seguro e conveniente.

```
#Get the Puppet password:
cat resp.json | jq -r '.Server.EngineAttributes[] | select(.Name ==
 "PUPPET_ADMIN_PASSWORD") | .Value'

#Get the Puppet Starter Kit:
cat resp.json | jq -r '.Server.EngineAttributes[] | select(.Name ==
 "PUPPET_STARTER_KIT") | .Value' | base64 -D > starterkit.zip
```

> **Note**
>
> Não é possível gerar novamente um starter kit do Puppet master no AWS Management Console. Ao criar um Puppet master usando o AWS CLI, execute o `jq` comando anterior para salvar o kit inicial codificado em base64 nos resultados como um arquivo ZIP. `create-server`

8. Se você não estiver usando um domínio personalizado, vá para a próxima etapa. Se você estiver usando um domínio personalizado com o servidor, crie uma entrada CNAME na ferramenta de gerenciamento de DNS da sua empresa para direcionar seu domínio personalizado OpsWorks para o endpoint do Puppet Enterprise que você copiou na etapa 6. Você não poderá acessar nem se conectar a um servidor com um domínio personalizado se não concluir esta etapa.
9. Prossiga para a próxima seção, the section called “Concluir configuração”.

Configurar o Puppet master usando o Starter Kit

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se

você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Enquanto a criação do Puppet master ainda está em andamento, a página Propriedades do servidor é aberta no console do OpsWorks Puppet Enterprise. Na primeira vez que você trabalhar com um novo Puppet master, a página Properties solicitará que você faça download de dois itens necessários. Faça download desses itens antes que o servidor Puppet fique online. Os botões de download não ficam disponíveis depois que um novo servidor está online.

test-puppet-server

Puppet Enterprise dashboard (not yet available) Actions

AWS OpsWorks is creating your Puppet Enterprise server. This takes about 20 minutes.

Creating an Elastic IP address | Launching an EC2 instance | Installing Puppet Enterprise server

Make sure you download the following before your server is online.

1. Sign-in credentials for your Puppet Enterprise dashboard
2. Starter Kit for your Puppet Enterprise server

Download the sign-in credentials for your Puppet Enterprise dashboard.

Show sign-in credentials

Download credentials

AWS OpsWorks does not save these credentials, so it is the last time they are available for viewing and downloading. After your server is online, you can change the password by signing in to its Puppet Enterprise dashboard.

Download the Starter Kit, and follow the documentation to finish the setup when your server is online.

Download Starter Kit

The Starter Kit contains a Readme with examples, and instructions how to install Puppet Enterprise client tools, as well as userdata templates for Windows and Linux.

Server information

More settings

Status	Version	Region	System maintenance	Automated backup
creating	2017.3.0	US West (Oregon)	5 pm - 6 pm UTC, every Tuesday	10 pm - 11 pm UTC, daily

Puppet Enterprise Console

https://test-puppet-server-nxdx8g13l0wi6ug9.us-west-2.opsworks-cm.io

- Credenciais de login para o Puppet master. Você usará essas credenciais para entrar no console do Puppet Enterprise, onde você executa a maior parte do gerenciamento de nós. AWS OpsWorks não salva essas credenciais; essa é a última vez que elas estão disponíveis para visualização e download. Se necessário, será possível alterar a senha fornecida com essas credenciais depois de fazer login.
- Starter Kit. O Starter Kit contém um arquivo README com informações e exemplos que descrevem como concluir a configuração e as credenciais de administrador para o console do Puppet Enterprise. Novas credenciais serão geradas, e as credenciais antigas invalidadas, sempre que você fizer download do Starter Kit.

Pré-requisitos

1. Embora a criação do servidor ainda esteja em andamento, faça download das credenciais de login para o Puppet master e salve-as em um local seguro, mas conveniente.
2. Faça o download do Starter Kit e descompacte o arquivo Starter Kit..zip no diretório do espaço de trabalho. Não compartilhe suas credenciais de login. Se outros usuários forem gerenciar o Puppet master, adicione-os como administradores no console do Puppet Enterprise posteriormente. Para obter mais informações sobre como adicionar usuários no Puppet master, consulte Creating and managing users and user roles na documentação do Puppet Enterprise.

Instalar o certificado do Puppet master

Para trabalhar com o seu Puppet master e adicionar nós para gerenciar, você precisará instalar o certificado dele. Instale-o executando o seguinte AWS CLI comando. Você não pode executar essa tarefa no AWS Management Console.

```
aws --region region opsworks-cm describe-servers --server-name server_name --query
 "Servers[0].EngineAttributes[?Name=='PUPPET_API_CA_CERT'].Value" --output text
 > .config/ssl/cert/ca.pem
```

Gerar um token de curto prazo

Para usar a API do Puppet, você deve criar um token de curto prazo para si mesmo. Esta etapa não é necessária para usar o console do Puppet Enterprise. Gere o token executando o comando a seguir.

A vida útil do token padrão é cinco minutos, mas você pode alterar este padrão.

```
puppet-access login --config-file .config/puppetlabs/client-tools/puppet-access.conf --
lifetime 8h
```

> **Note**
>
> Como a vida útil do token padrão é de cinco minutos, o comando de exemplo anterior adiciona o parâmetro `--lifetime` para estender a vida útil do token para um período maior. Você pode definir a vida útil do token para um período de até 10 anos (`10y`). Para obter mais informações sobre como alterar a vida útil do token padrão, consulte Change the token's default lifetime na documentação do Puppet Enterprise.

Configurar o exemplo do Starter Kit do Apache

Depois de fazer download e descompactar o Starter Kit, você poderá usar a ramificação de exemplo na pasta `control-repo-example` de amostra incluída, para configurar um servidor da web do Apache em seus nós gerenciados.

O Starter Kit inclui duas `control-repo` pastas: `control-repo` e `control-repo-example`. A `control-repo` pasta inclui uma `production` ramificação que não é alterada em relação ao que você veria no repositório do Puppet GitHub . A pasta `control-repo-example` também tem uma ramificação `production` que inclui código de exemplo para configurar um servidor Apache com um site de teste.

1. Envie a ramificação `control-repo-example` `production` para o Git remoto (o `r10k_remote` URL do seu Puppet master). No diretório raiz do Starter Kit, execute o seguinte, *`r10kRemoteUrl`* substituindo-o pelo seu `r10k_remote` URL.

   ```
   cd  control-repo-example
   git remote add origin r10kRemoteUrl
   git push origin production
   ```

   O gerenciador de códigos do Puppet usa as ramificações do Git como ambientes. Por padrão, todos os nós estão no ambiente de produção.

> **Important**
>
> Não envie para uma ramificação `master`. A ramificação `master` é reservada para o Puppet master.

2. Implante o código na ramificação `control-repo-example` para o Puppet master. Isso permite que o Puppet master faça download do código do Puppet no repositório do Git (`r10k_remote`). No diretório raiz do Starter Kit, faça isto.

```
puppet-code deploy --all --wait --config-file .config/puppet-code.conf
```

Para obter mais informações sobre como você pode aplicar o exemplo de configuração do Apache aos nós gerenciados que você cria na Amazon EC2, consulte Etapa 2: criar instâncias com um script de associação autônoma este guia.

Adicionar nós para o Puppet master gerenciar

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Tópicos

- Executar chamadas da API associateNode()
- Considerações sobre a adição de nós locais
- Mais informações

A forma recomendada de adicionar nós é usando a AWS OpsWorks `associateNode()` API. O servidor Puppet Enterprise master hospeda um repositório usado para instalar o software do agente do Puppet nos nós que você deseja gerenciar, sejam esses nós computadores físicos no local ou máquinas virtuais. O software do agente Puppet para alguns sistemas operacionais é instalado no

servidor do OpsWorks Puppet Enterprise como parte do processo de inicialização. A tabela a seguir mostra os agentes do sistema operacional que estão disponíveis em seu servidor OpsWorks para o Puppet Enterprise na inicialização.

Agentes do sistema operacional pré-instalados

Sistema operacional com suporte	Versões
Ubuntu	16.04, 18.04, 20.04
Red Hat Enterprise Linux (RHEL)	6, 7, 8
Windows	Edições de 64 bits de todos os lançamentos do Windows compatíveis com o Puppet

Você pode adicionar `puppet-agent` ao seu servidor para outros sistemas operacionais. Lembre-se de que a manutenção do sistema excluirá agentes adicionados ao seu servidor após a inicialização. Embora a maioria dos nós conectados existentes que já executam o agente excluído continuem a verificar, os nós que executam os sistemas operacionais Debian podem interromper a geração de relatórios. Recomendamos que você instale manualmente `puppet-agent` em nós que estejam executando sistemas operacionais para os quais o software do agente não esteja pré-instalado no seu servidor OpsWorks for Puppet Enterprise. Para obter informações detalhadas sobre como disponibilizar o `puppet-agent` no servidor para nós com outros sistemas operacionais, consulte Installing agents na documentação do Puppet Enterprise.

Para obter informações sobre como associar nós ao seu mestre Puppet automaticamente preenchendo os dados do usuário da EC2 instância, consulte. Adicionando nós automaticamente OpsWorks para o Puppet Enterprise

Executar chamadas da API **`associateNode()`**

Depois de adicionar nós por meio da instalaçãopuppet-agent, os nós enviam solicitações de assinatura de certificado (CSRs) OpsWorks para o servidor do Puppet Enterprise. Você pode ver o CSRs no console do Puppet; para obter mais informações sobre o node CSRs, consulte Gerenciando solicitações de assinatura de certificados na documentação do Puppet Enterprise. Executar o CSRs nó de processos de chamada `associateNode()` da API OpsWorks for Puppet Enterprise e associar o nó ao seu servidor. Veja a seguir um exemplo de como usar essa chamada de API no AWS CLI para associar um único nó. Você precisará da CSR no formato PEM que o nó envia. Isso pode ser obtido no console do Puppet.

```
aws opsworks-cm associate-node --server-name "test-puppet-
server" --node-name "node or instance ID" --engine-attributes
 "Name=PUPPET_NODE_CSR,Value='PEM_formatted_CSR_from_the_node'
```

Para obter mais informações sobre como adicionar nós automaticamente usando `associateNode()`, consulte Adicionando nós automaticamente OpsWorks para o Puppet Enterprise.

Considerações sobre a adição de nós locais

Depois de instalar `puppet-agent` em seus computadores locais ou máquinas virtuais, você pode usar uma das duas maneiras de associar nós locais ao seu OpsWorks mestre do Puppet Enterprise.

- Se um nó oferecer suporte à instalação do SDK da AWS, da AWS CLIou do Ferramentas da AWS para PowerShell, você poderá usar o método recomendado para associar um nó, que é executar uma chamada da API `associateNode()`. O kit inicial que você baixa ao criar um master OpsWorks para o Puppet Enterprise pela primeira vez mostra como atribuir funções aos nós usando tags. Você pode aplicar tags ao mesmo tempo em que associa os nós ao Puppet master especificando fatos confiáveis na CSR. Por exemplo, o repositório de controle de demonstração incluído no kit inicial está configurado para usar a tag para atribuir funções `pp_role` às instâncias da Amazon EC2 . Para obter mais informações sobre como adicionar tags a uma CSR como fatos confiáveis, consulte Solicitações de extensão (dados certificados permanentes) na documentação da plataforma do Puppet.
- Se o nó não puder executar ferramentas AWS de gerenciamento ou desenvolvimento, você ainda poderá registrá-lo com seu OpsWorks mestre do Puppet Enterprise da mesma forma que você o registraria com qualquer mestre não gerenciado do Puppet Enterprise. Conforme mencionado neste tópico, a instalação `puppet-agent` envia uma CSR OpsWorks para o mestre do Puppet Enterprise. Um usuário autorizado do Puppet pode assinar a CSR manualmente ou configurar a assinatura automática CSRs editando o `autosign.conf` arquivo armazenado no Puppet master. Para obter mais informações sobre a configuração de autoassinatura e a edição `autosign.conf`, consulte Configuração de SSL: solicitações de certificado de autoassinatura na documentação da plataforma do Puppet.

Para associar nós locais a um mestre do Puppet e permitir que o mestre do Puppet aceite tudo CSRs, faça o seguinte no console do Puppet Enterprise. O parâmetro que controla o comportamento é `puppet_enterprise::profile::master::allow_unauthenticated_ca`.

> **Important**
>
> Não é recomendável permitir que o Puppet master aceite o autoassinado CSRs ou tudo não CSRs é recomendado por motivos de segurança. Por padrão, permitir a autenticação não autenticada CSRs torna o Puppet master acessível ao mundo. Definir o carregamento de solicitações de certificado como habilitado por padrão pode deixar o Puppet master vulnerável a ataques de negação de serviço (DoS).

1. Faça login no console do Puppet Enterprise.
2. Selecione Configure (Configurar), Classification (Classificação), PE Master e, em seguida, selecione a guia Configuration (Configuração).
3. Na guia Classification (Classificação), localize a classe puppet_enterprise::profile::master.
4. Defina o valor do parâmetro allow_unauthenticated_ca como true (verdadeiro).
5. Salve as alterações. Suas alterações serão aplicadas durante a próxima execução do Puppet. Você pode permitir 30 minutos para as alterações entrarem em vigor (e os nós locais serem adicionados) ou você pode iniciar uma execução do Puppet manualmente na seção Run (Execução) do console do PE.

Mais informações

Visite o site de tutoriais do Learn Puppet para saber mais sobre como usar OpsWorks os servidores do Puppet Enterprise e os recursos do console do Puppet Enterprise.

Faça login no console do Puppet Enterprise

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Depois de ter feito o download das credenciais de login na página Properties do Puppet master e o servidor estiver online, faça login no console do Puppet Enterprise. Nestas demonstrações

detalhadas, instruímos que você especifique seu repositório de controle que contém os módulos e adicionar pelo menos um nó para gerenciar. Isso permite que você consulte informações sobre o agente e os nós no console.

Quando você tenta se conectar à página da web do console do Puppet Enterprise, os avisos de certificado aparecem no seu navegador até que você instale um certificado SSL AWS OpsWorks específico assinado por CA no computador cliente que você está usando para gerenciar seu servidor Puppet. Se você preferir não ver os avisos antes de continuar para a página da web do painel, instale o certificado SSL antes de fazer login.

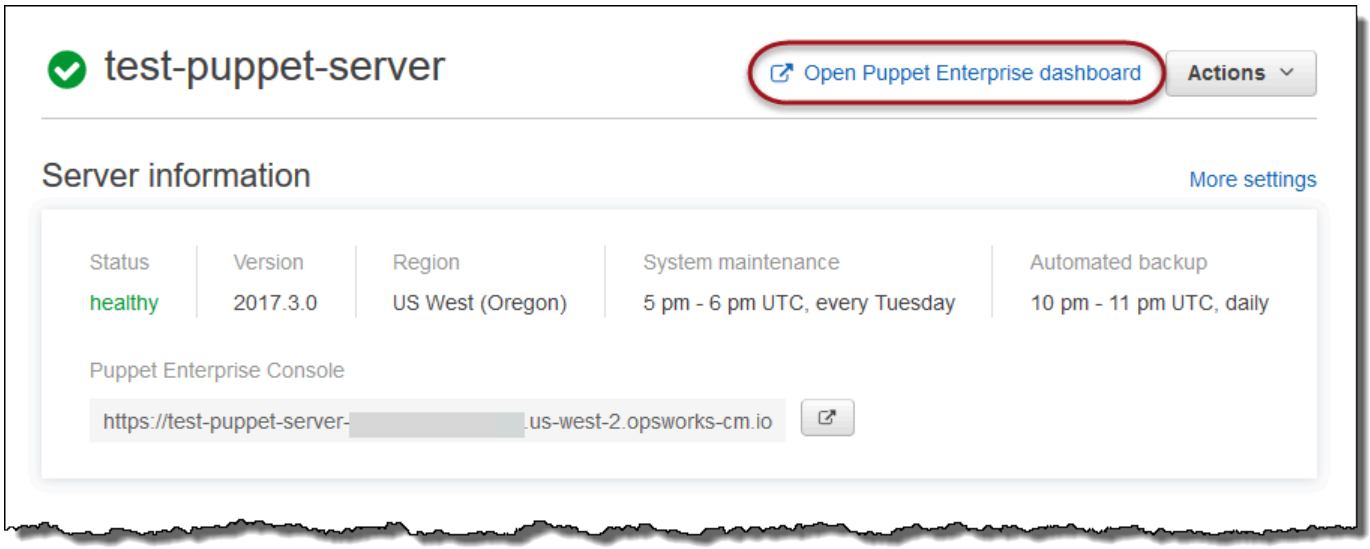
Para instalar o AWS OpsWorks certificado SSL

- Escolha o certificado que corresponde ao seu sistema.

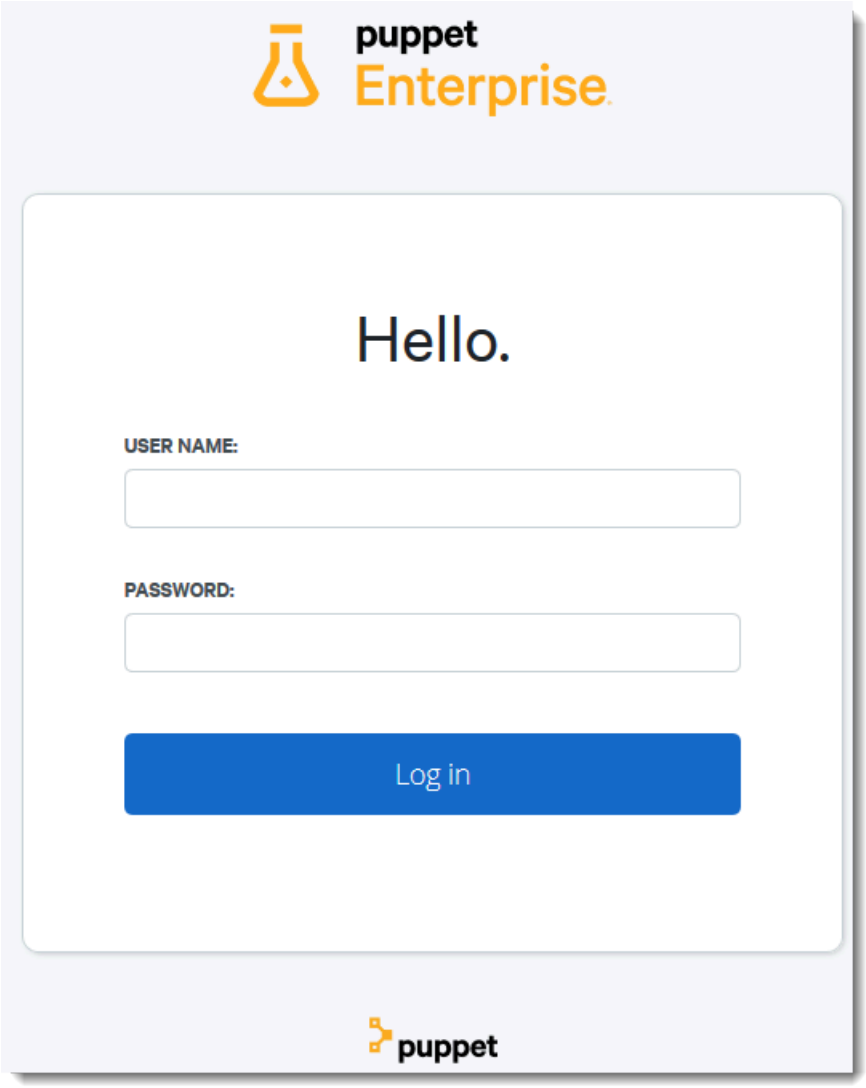
Depois de ter instalado o certificado SSL do lado do cliente, é possível fazer login no console do Puppet Enterprise sem visualizar mensagens de aviso.

Para fazer login no console do Puppet Enterprise

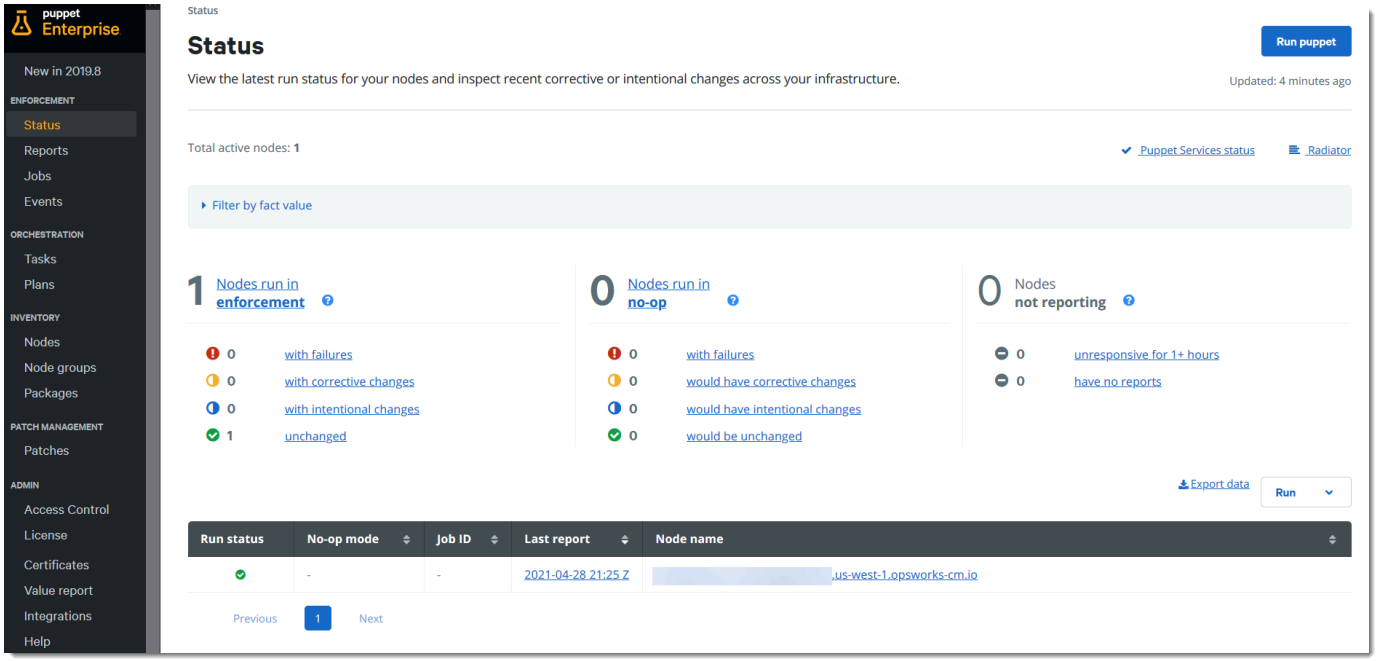
1. Descompacte e abra as credenciais do Puppet Enterprise que você baixou em Pré-requisitos. Você precisará essas credenciais para fazer login.
2. No AWS Management Console, abra a página Propriedades do seu servidor Puppet.
3. No canto superior direito da página Properties, escolha Open Puppet Enterprise console.



4. Faça login usando as credenciais da Etapa 1.



5. No console do Puppet Enterprise, você pode visualizar informações detalhadas sobre os nós que está gerenciando, os eventos e o andamento da execução do módulo, o nível de compatibilidade dos nós e muito mais. Para obter mais informações sobre os recursos do console do Puppet Enterprise e como usá-los, consulte Gerenciamento de nós na documentação do Puppet Enterprise.



Agrupar e classificar os nós

Antes de você especificar a configuração desejada de seus nós aplicando classes a eles, agrupe os nós de acordo com suas funções na sua empresa ou suas características comuns. O agrupamento e a classificação de nós envolve as seguintes tarefas de alto nível. Você pode concluir essas tarefas usando o console do PE. Para obter informações detalhadas sobre como agrupar e classificar seus nós, consulte Grouping and classifying nodes na documentação do Puppet Enterprise.

1. Crie grupos de nós.
2. Adicione nós a grupos manualmente ou automaticamente aplicando as regras que você criar.
3. Atribua classes a grupos de nós.

Redefinir as senhas de administrador e usuário

Para obter informações sobre como alterar a senha que você usa para fazer login no console do Puppet Enterprise, consulte Redefinir a senha de administrador do console na documentação do Puppet Enterprise.

Por padrão, após dez tentativas de login, os usuários serão bloqueadas do console do Puppet. Para obter mais informações sobre como redefinir as senhas de usuários no caso de um bloqueio, consulte Password endpoints na documentação do Puppet Enterprise.

Opcional: Use AWS CodeCommit como um repositório de controle remoto do Puppet r10k

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

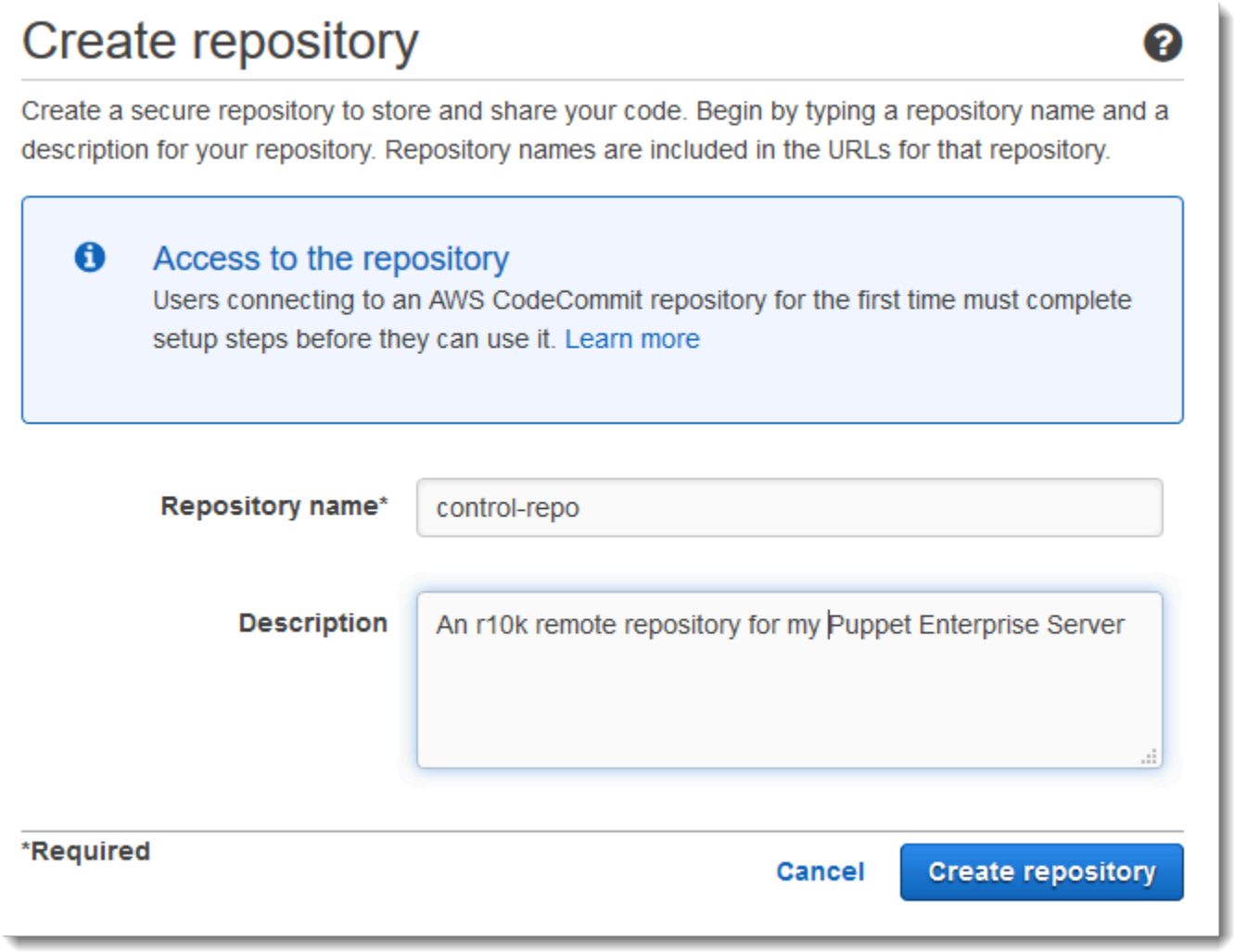
Você pode criar um novo repositório usando AWS CodeCommit e usá-lo como seu repositório de controle remoto r10k. Para concluir as etapas desta seção e trabalhar com um CodeCommit repositório, você precisa de um usuário que tenha as permissões fornecidas pela política AWSCodeCommitReadOnlygerenciada.

Tópicos

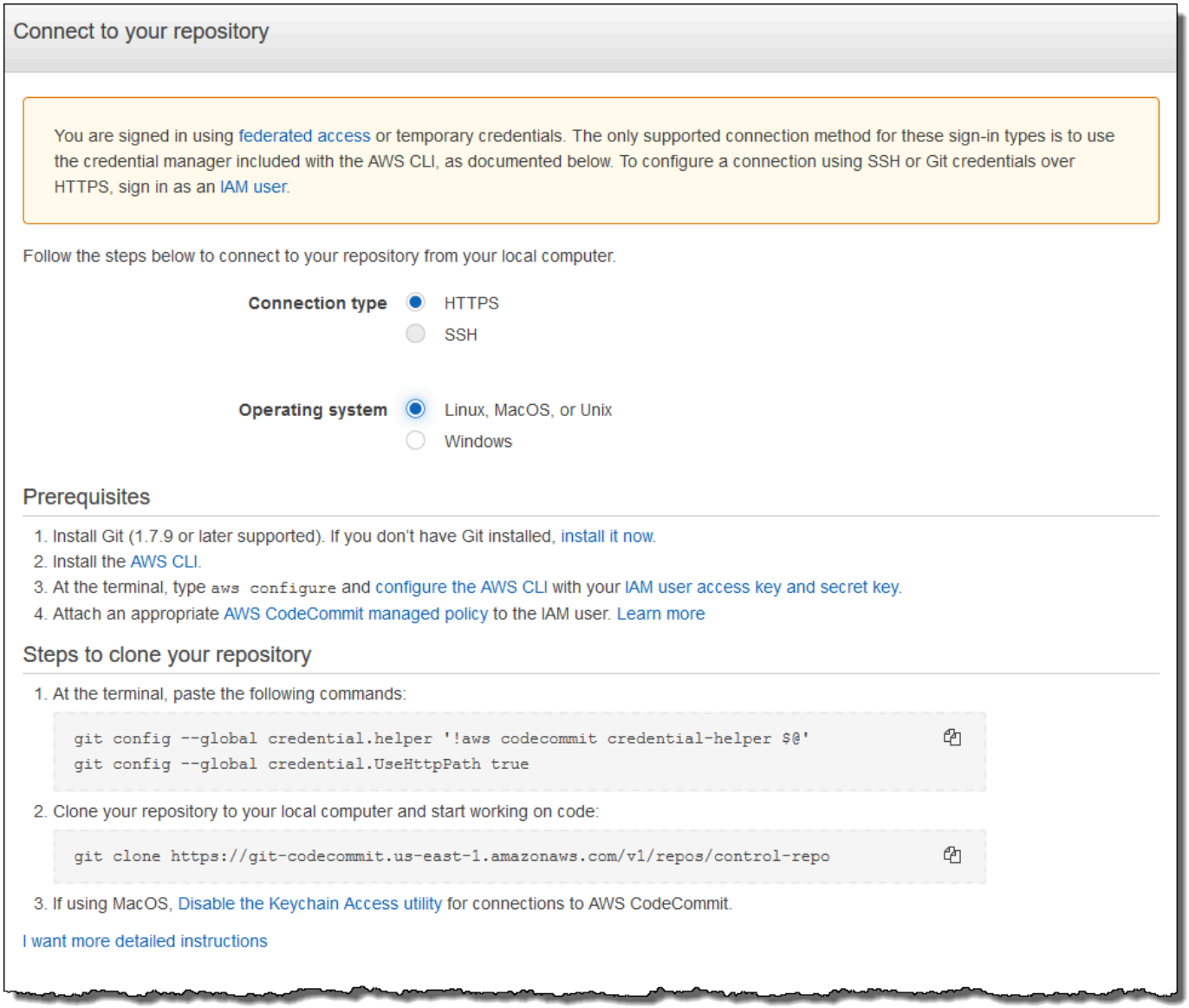
- Etapa 1: Usar CodeCommit como um repositório com um tipo de conexão HTTPS
- Etapa 2: (Opcional) Usar CodeCommit como repositório com um tipo de conexão SSH

Etapa 1: Usar CodeCommit como um repositório com um tipo de conexão HTTPS

1. No CodeCommit console, crie um novo repositório.



2. Escolha Skip para ignorar a configuração de um tópico do Amazon SNS.
3. Na página Code, escolha Connect to your repository.
4. Na página Connect to your repository, escolha HTTPS como o Connection type e selecione seu sistema operacional.



Na área Steps to clone your repository (Etapas para clonar seu repositório), seu `git clone` URL deverá se parecer com o seguinte: `https://git-codecommit.`*`region`*`.amazonaws.com/v1/repos/control-repo`. Copie esse URL para um local conveniente para uso na configuração do servidor Puppet.

5. Feche a página Conectar-se ao seu repositório e retorne à configuração do OpsWorks servidor do Puppet Enterprise.
6. Cole o URL que você copiou na Etapa 4 na caixa de strings r10k remote na página Configure credentials do assistente de configuração do Puppet master. Deixe a caixa r10k private key vazia. Conclua a criação e a inicialização do Puppet master.

7. No console do IAM, anexe a AWSCodeCommitReadOnlypolítica à função de perfil da instância do seu mestre do Puppet. Para obter mais informações sobre como acrescentar uma política de permissões a um perfil do IAM, consulte Adicionar permissões de identidade do IAM (console) no Guia do usuário do IAM.
8. Siga as etapas em Configuração para usuários HTTPS usando credenciais do Git no Guia do AWS CodeCommit usuário para enviar seu `control-repo` conteúdo existente para o novo repositório. CodeCommit
9. Agora, você pode continuar seguindo as instruções no the section called "Concluir configuração", e usar o Starter Kit para implantar o código para o Puppet master. O comando a seguir é um exemplo.

```
puppet-code deploy --all --wait --config-file .config/puppet-code.conf
```

Etapa 2: (Opcional) Usar CodeCommit como repositório com um tipo de conexão SSH

Você pode configurar um repositório de controle remoto AWS CodeCommit r10k para usar a autenticação de key pair SSH. Os pré-requisitos a seguir devem ser concluídos antes de iniciar esse procedimento.

- Você deve ter iniciado seu servidor OpsWorks for Puppet Enterprise com um repositório de controle HTTPS, conforme descrito na seção anterior,. the section called "Etapa 1: Usar CodeCommit como um repositório com um tipo de conexão HTTPS" Isso deve ser concluído primeiro para que você possa fazer upload da configuração necessária para o Puppet master.
- Verifique se você tem um usuário com a política AWSCodeCommitReadOnlygerenciada anexada. Para obter mais informações sobre como criar um usuário, consulte Como criar um usuário do IAM em sua AWS conta no Guia do usuário do IAM.
- Crie e associe uma chave SSH ao usuário do . Siga as instruções para criar um par de public/private chaves *`ssh-keygen`* na Etapa 3: Configurar credenciais no Linux, macOS ou Unix no AWS CodeCommit Guia do Usuário.

1. Em uma AWS CLI sessão, execute o comando a seguir para carregar o conteúdo do arquivo de chave privada no AWS Systems Manager Parameter Store. Seu servidor OpsWorks for Puppet Enterprise consulta esse parâmetro para obter um arquivo de certificado necessário. *`private_key_file`*Substitua pelo caminho para seu arquivo de chave privada SSH.

```
aws ssm put-parameter --name puppet_user_pk --type String --value
 "`cat private_key_file`"
```

2. Adicione permissões do Parameter Store do Systems Manager para o Puppet master.

   a. Abra o console do IAM em https://console.aws.amazon.com/iam/.

   b. No painel de navegação à esquerda, selecione Perfis.

   c. Escolha aws-opsworks-cm-ec2 funções.

   d. Na guia Permissions (Permissões), escolha Attach policies (Anexar políticas).

   e. Na barra Search (Buscar), insira **`AmazonSSMManagedInstanceCore`**.

   f. Nos resultados da pesquisa, escolha Amazon SSMManaged InstanceCore.

   g. Escolha Anexar política.

3. Crie o manifesto do arquivo de configuração. Se você estiver usando o repositório `control-repo-example` fornecido no kit de inicialização, crie os arquivos a seguir nos locais mostrados no repositório de exemplo. Caso contrário, crie-os de acordo com sua própria estrutura do repositório de controle. Substitua o *`IAM_USER_SSH_KEY`* valor pelo ID da chave SSH que você criou nos pré-requisitos desse procedimento.

```
control-repo-example/site/profile/manifests/codecommit.pp
```

```
class profile::codecommit {
  $configfile = @(CONFIGFILE)
      Host git-codecommit.*.amazonaws.com
      User IAM_USER_SSH_KEY
      IdentityFile /etc/puppetlabs/puppetserver/ssh/codecommit.rsa
      StrictHostKeyChecking=no
      | CONFIGFILE

  # Replace REGION with the correct region for your server.
  $command = @(COMMAND)
      aws ssm get-parameters \
      --region REGION \
      --names puppet_user_pk \
      --query "Parameters[0].Value" \
      --output text >| /etc/puppetlabs/puppetserver/ssh/codecommit.rsa
      | COMMAND

  $dirs = [
```

```
            '/opt/puppetlabs/server/data/puppetserver/.ssh',
            '/etc/puppetlabs/puppetserver/ssh',
          ]

  file { $dirs:
    ensure => 'directory',
    group  => 'pe-puppet',
    owner  => 'pe-puppet',
    mode   => '0750',
  }

  file { 'ssh-config':
    path    => '/opt/puppetlabs/server/data/puppetserver/.ssh/config',
    require => File[$dirs],
    content => $configfile,
    group   => 'pe-puppet',
    owner   => 'pe-puppet',
    mode    => '0600',
  }

  exec { 'download-codecommit-certificate':
    command => $command,
    require => File[$dirs],
    creates => '/etc/puppetlabs/puppetserver/ssh/codecommit.rsa',
    path    => '/bin',
    cwd     => '/etc/puppetlabs',
  }

  file { 'private-key-permissions':
    subscribe => Exec['download-codecommit-certificate'],
    path      => '/etc/puppetlabs/puppetserver/ssh/codecommit.rsa',
    group     => 'pe-puppet',
    owner     => 'pe-puppet',
    mode      => '0600',
  }
}
```

4. Envie seu repositório de controle para o. CodeCommit Execute os comandos a seguir para enviar o novo arquivo manifesto para o seu repositório.

```
git add ./site/profile/manifests/codecommit.pp
git commit -m 'Configuring for SSH connection to CodeCommit'
git push origin production
```

5. Implante os arquivos manifesto. Execute os comandos a seguir para implantar a configuração atualizada no seu OpsWorks servidor do Puppet Enterprise. *STARTER_KIT_DIRECTORY*Substitua pelo caminho para seus arquivos de configuração do Puppet.

```
cd STARTER_KIT_DIRECTORY

puppet-access login --config-file .config/puppetlabs/client-tools/puppet-
access.conf

puppet-code deploy --all --wait \
--config-file .config/puppet-code.conf \
--token-file .config/puppetlabs/token
```

6. Atualize OpsWorks a classificação do servidor do Puppet Enterprise. Por padrão, o agente do Puppet é executado em nós (incluindo o principal) a cada 30 minutos. Para evitar espera, você pode executar manualmente o agente no Puppet master. Executar o agente seleciona o novo arquivo manifesto.

   a. Faça login no console do Puppet Enterprise.

   b. Selecione Classification (Classificação).

   c. Expanda PE Infrastructure (Infraestrutura do PE).

   d. Selecione PE Master.

   e. Na guia Configuration (Configuração), insira **`profile::codecommit`** em Add new class (Adicionar nova classe).

      A nova classe, `profile::codecommit`, pode não aparecer imediatamente após a execução de `puppet-code deploy`. Selecione Refresh (Atualizar) nesta página, se ela não for exibida.

   f. Selecione Add class (Adicionar classe) e selecione Commit 1 change (Confirmar 1 alteração).

   g. Execute manualmente o agente Puppet no servidor OpsWorks for Puppet Enterprise. Selecione Nodes (Nós), escolha seu servidor na lista, selecione Run Puppet (Executar Puppet) e selecione Run (Executar).

7. No console do Puppet Enterprise, altere o URL do repositório para usar SSH em vez de HTTPS. A configuração que você executa nessas etapas é salva OpsWorks durante o processo de

backup e restauração do Puppet Enterprise, portanto, você não precisa alterar manualmente a configuração do repositório após as atividades de manutenção.

a. Selecione Classification (Classificação).
b. Expanda PE Infrastructure (Infraestrutura do PE).
c. Selecione PE Master.
d. Na guia Configuration (Configuração), localize a classe `puppet_enterprise::profile::master`.
e. Selecione Edit (Editar) ao lado do parâmetro `r10k_remote`.
f. Substitua o URL de HTTPS pelo URL de SSH para o seu repositório e selecione Commit 1 change (Confirmar 1 alteração).
g. Execute manualmente o agente Puppet no servidor OpsWorks for Puppet Enterprise. Selecione Nodes (Nós), escolha seu servidor na lista, selecione Run Puppet (Executar Puppet) e selecione Run (Executar).

Crie um AWS OpsWorks for Puppet Enterprise mestre usando AWS CloudFormation

> **Important**
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks for Puppet Enterprise permite que você execute um servidor Puppet Enterprise em AWS. Você pode provisionar um servidor master Puppet Enterprise em aproximadamente 15 minutos.

A partir de 3 de maio de 2021, OpsWorks o for Puppet Enterprise armazena alguns atributos do servidor Puppet Enterprise em. AWS Secrets Manager Para obter mais informações, consulte Integração com AWS Secrets Manager.

O passo a passo a seguir ajuda você a criar um mestre do Puppet OpsWorks para o Puppet Enterprise criando uma pilha. AWS CloudFormation

Tópicos

- Pré-requisitos
- Criar um Puppet Enterprise mestre no AWS CloudFormation

Pré-requisitos

Antes de criar um novo Puppet master, crie os recursos externos do OpsWorks Puppet Enterprise que você precisará para acessar e gerenciar seu Puppet master. Para obter mais informações, consulte Pré-requisitos na seção de Conceitos básicos deste guia.

Se você estiver criando um servidor que use um domínio personalizado, precisará de um domínio, um certificado e uma chave privada personalizados. Você deve especificar valores para todos esses três parâmetros em seu AWS CloudFormation modelo. Para obter mais informações sobre os requisitos dos `CustomPrivateKey` parâmetros `CustomDomainCustomCertificate`,, e, consulte CreateServera Referência da API AWS OpsWorks CM.

Consulte a seção OpsWorks -CM da Referência do Modelo AWS CloudFormation do Guia do Usuário para saber mais sobre os valores suportados e necessários no AWS CloudFormation modelo que você usa para criar seu servidor.

Criar um Puppet Enterprise mestre no AWS CloudFormation

Esta seção descreve como usar um AWS CloudFormation modelo para criar uma pilha que cria um servidor mestre OpsWorks para o Puppet Enterprise. Você pode fazer isso usando o AWS CloudFormation console ou AWS CLI o. Um AWS CloudFormation modelo de exemplo está disponível para você usar para criar uma pilha de servidores OpsWorks para o Puppet Enterprise. Certifique-se de atualizar o modelo de exemplo com suas próprias informações de nome do servidor, perfis do IAM, perfil de instância, descrição do servidor, contagem de retenção de backup, opções de manutenção e tags opcionais. Se o servidor usar um domínio personalizado, você deverá especificar valores para os parâmetros `CustomDomain`, `CustomCertificate` e `CustomPrivateKey` no modelo do AWS CloudFormation . Para obter mais informações sobre essas opções, consulte the section called “Crie um Puppet Enterprise Master usando o AWS Management Console” na seção de Conceitos básicos deste guia.

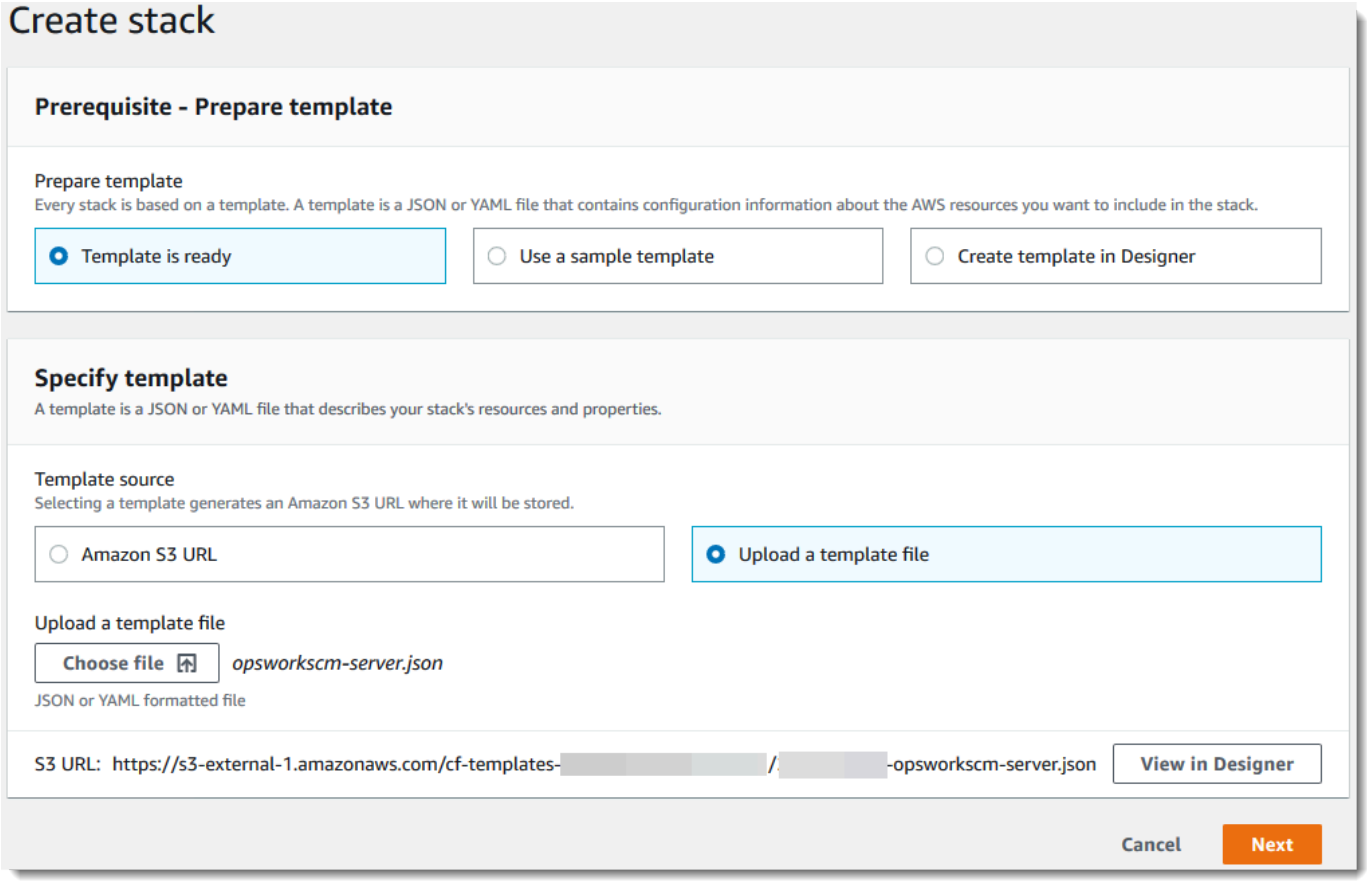
Tópicos

- Crie um Puppet Enterprise Master usando AWS CloudFormation (Console)
- Crie um Puppet Enterprise Master usando AWS CloudFormation (CLI)

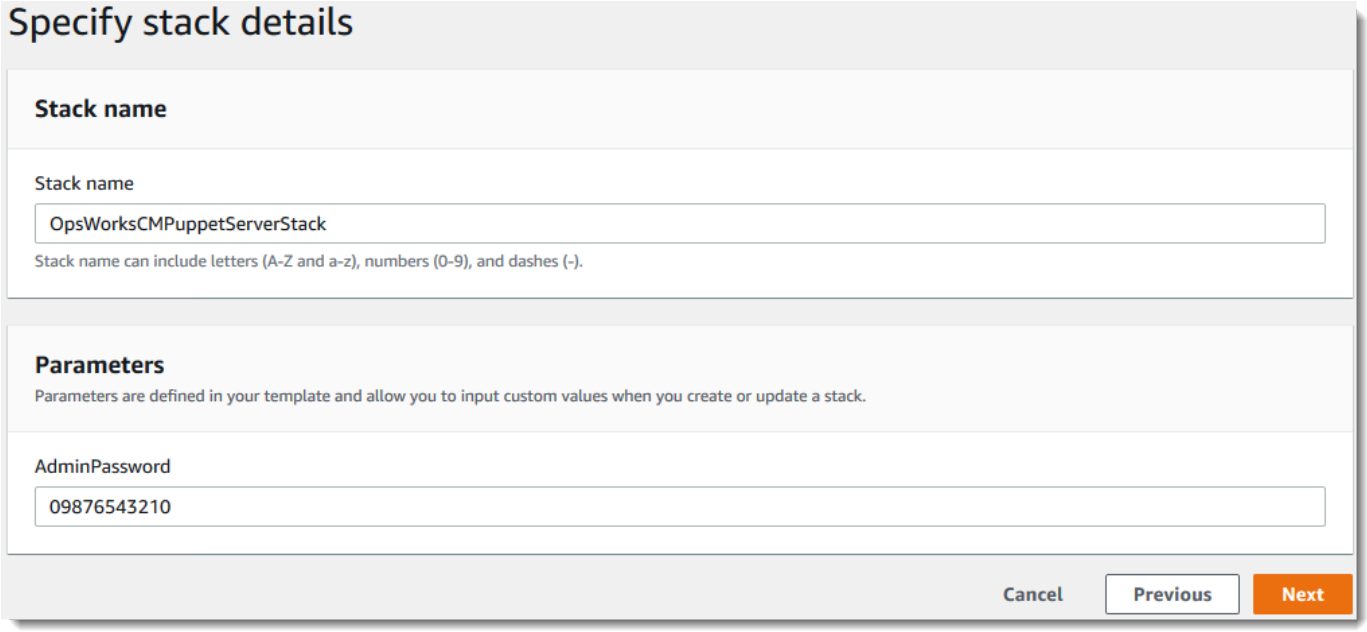
Crie um Puppet Enterprise Master usando AWS CloudFormation (Console)

1. Faça login no AWS Management Console e abra o AWS CloudFormation console em https://console.aws.amazon.com/cloudformation.
2. Na página AWS CloudFormation inicial, escolha Criar pilha.
3. Em Prerequisite - Prepare template (Pré-requisito — Preparar modelo), se você estiver usando o modelo do AWS CloudFormation de exemplo, escolha Template is ready (O modelo está pronto).
4. Em Specify template (Especificar modelo), escolha a origem do seu modelo. Para este passo a passo, escolha Carregar um arquivo de modelo e carregue um AWS CloudFormation modelo que crie um servidor Puppet Enterprise. Procure seu arquivo de modelo e escolha Next (Próximo).

   Um AWS CloudFormation modelo pode estar no formato YAML ou JSON. Um AWS CloudFormation modelo de exemplo está disponível para você usar; certifique-se de substituir os valores de exemplo pelos seus. Você pode usar o designer AWS CloudFormation de modelos para criar um novo modelo ou validar um existente. Para obter mais informações sobre como fazer isso, consulte Visão geral da interface do AWS CloudFormation Designer no Guia do usuário do AWS CloudFormation .



5. Na página Specify stack details (Especificar detalhes da pilha), insira um nome para a pilha. Ele não será o mesmo que o nome do servidor, apenas o nome de uma pilha. Na área Parameters (Parâmetros), insira uma senha de administrador para fazer login na página da web do console do Puppet Enterprise. A senha deve ter entre 8 e 32 caracteres ASCII. Escolha Próximo.



6. Na página Opções, você pode adicionar tags ao servidor que está criando com a pilha e escolher um perfil do IAM para criar recursos, caso ainda não tenha especificado um perfil do IAM a ser usado no modelo. Quando terminar de especificar as opções, selecione Next (Próximo). Para obter mais informações sobre opções avançadas, como acionadores de reversão, consulte Configurando opções de AWS CloudFormation pilha no Guia do usuário.AWS CloudFormation

7. Na página Analisar, examine suas escolhas. Quando estiver pronto para criar a pilha do servidor, selecione Create (Criar).

   Enquanto você espera para criar AWS CloudFormation a pilha, veja o status de criação da pilha. Se a criação da pilha falhar, analise as mensagens de erro mostradas no console para ajudá-lo a resolver os problemas. Para obter mais informações sobre como solucionar problemas de erros em pilhas do AWS CloudFormation , consulte Solução de problemas de erros no Guia do usuário do AWS CloudFormation .

   Quando a criação do servidor estiver concluída, seu mestre do Puppet estará disponível na OpsWorks página inicial do Puppet Enterprise, com o status de online. Depois que o servidor estiver online, o console do Puppet Enterprise estará disponível no domínio do servidor, em um URL no seguinte formato: `https://your_server_name-randomID.region.opsworks-cm.io`.

> **Note**
>
> Se você especificou um domínio, certificado e chave privada personalizados para seu servidor, crie uma entrada CNAME na ferramenta de gerenciamento de DNS da sua empresa que mapeie seu domínio personalizado para o endpoint gerado automaticamente OpsWorks para o servidor pelo Puppet Enterprise. Não é possível gerenciar o servidor ou se conectar ao site de gerenciamento do Puppet Enterprise para o servidor até que você mapeie o endpoint gerado para o valor do domínio personalizado.
>
> Para obter o valor do endpoint gerado, execute o seguinte AWS CLI comando depois que seu servidor estiver on-line:
>
> ```
> aws opsworks describe-servers --server-name server_name
> ```

Crie um Puppet Enterprise Master usando AWS CloudFormation (CLI)

Se o seu computador local ainda não estiver executando o AWS CLI, faça o download e instale-o AWS CLI seguindo as instruções de instalação no Guia do usuário da interface de linha de comando da AWS. Esta seção não descreve todos os parâmetros que você pode usar com o comando create-stack. Para obter mais informações sobre os parâmetros do create-stack, consulte create-stack na Referência da AWS CLI .

1. Certifique-se de preencher o formulário Pré-requisitos para criar um mestre OpsWorks para o Puppet Enterprise.
2. Crie uma função de serviço e um perfil de instância.

   Encontre e copie as funções ARNs de serviço em sua conta.

   ```
   aws iam list-roles --path-prefix "/service-role/" --no-paginate
   ```

   Nos resultados do comando `list-roles`, procure pelas entradas da função de serviço e do perfil de instância que sejam semelhantes aos seguintes. Anote a função de serviço e o perfil ARNs da instância e adicione-os ao AWS CloudFormation modelo que você está usando para criar sua pilha de servidores mestre do Puppet.

```
{
    "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "ec2.amazonaws.com"
                }
            }
        ]
    },
    "RoleId": "AROZZZZZZZZZZQG6R22HC",
    "CreateDate": "2018-01-05T20:42:20Z",
    "RoleName": "aws-opsworks-cm-ec2-role",
    "Path": "/service-role/",
    "Arn": "arn:aws:iam::000000000000:role/service-role/aws-opsworks-cm-ec2-role"
},
{
    "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "opsworks-cm.amazonaws.com"
                }
            }
        ]
    },
    "RoleId": "AROZZZZZZZZZZZZZZ6QE",
    "CreateDate": "2018-01-05T20:42:20Z",
    "RoleName": "aws-opsworks-cm-service-role",
    "Path": "/service-role/",
    "Arn": "arn:aws:iam::000000000000:role/service-role/aws-opsworks-cm-service-
role"
}
```

3. Crie o master OpsWorks for Puppet Enterprise executando o create-stack comando novamente.

- *stack_name*Substitua pelo nome da sua pilha. Esse é o nome da AWS CloudFormation pilha, não do seu mestre de marionetes. O nome do Puppet mestre é o valor de `ServerName` no modelo do AWS CloudFormation .
- *template*Substitua pelo caminho do arquivo de modelo e pela extensão *yaml or json* por `.yaml` ou `.json` conforme apropriado.
- Os valores de `--parameters` correspondem aos EngineAttributesda CreateServerAPI. Para o Puppet, veja a seguir os atributos do mecanismo fornecidos pelo usuário para criar um servidor. Os atributos do mecanismo r10k conectam o Puppet mestre a um repositório de códigos para gerenciar a configuração do ambiente do servidor. Para obter mais informações sobre os atributos do mecanismo r10k, consulte Managing code with r10k (Gerenciar código com r10k) na documentação do Puppet Enterprise.
  - `PUPPET_ADMIN_PASSWORD`, uma senha de administrador para fazer login na página da web do console do Puppet Enterprise. A senha deve usar de 8 a 32 caracteres ASCII e exige pelo menos uma letra maiúscula, uma minúscula, um número e um caractere especial.
  - `PUPPET_R10K_REMOTE`, o URL do repositório de controle (por exemplo, ssh://git@your.git-repo.com:user/control-repo.git). Especificar um r10k remoto abre a porta TCP 8170.
  - `PUPPET_R10K_PRIVATE_KEY`. Se você estiver usando um repositório Git privado, adicione PUPPET_R10K_PRIVATE_KEY para especificar um URL de SSH e uma chave SSH privada codificada por PEM.

```
aws cloudformation create-stack --stack-name stack_name
 --template-body file://template.yaml or json --parameters
 ParameterKey=AdminPassword,ParameterValue="password"
```

Veja um exemplo a seguir.

```
aws cloudformation create-stack --stack-name "OpsWorksCMPuppetServerStack"
 --template-body file://opsworkscm-puppet-server.json --parameters
 ParameterKey=AdminPassword,ParameterValue="09876543210Ab#"
```

O exemplo a seguir especifica os atributos do mecanismo r10k como parâmetros, quando eles não são fornecidos no modelo. AWS CloudFormation Um exemplo de modelo que inclui os atributos do mecanismo r10k, `puppet-server-param-attributes.yaml`, está incluso no exemplo de modelos do AWS CloudFormation.

```
aws cloudformation create-stack --stack-name MyPuppetStack --
template-body file://puppet-server-param-attributes.yaml --parameters
 ParameterKey=AdminPassword,ParameterValue="superSecret1%3"
 ParameterKey=R10KRemote,ParameterValue="https://www.yourRemote.com"
 ParameterKey=R10KKey,ParameterValue="$(cat puppet-r10k.pem)"
```

O exemplo a seguir especifica os atributos do mecanismo r10k e seus valores no modelo do AWS CloudFormation ; o comando precisa apenas indicar o arquivo do modelo. O modelo especificado como o valor de `--template-body`, `puppet-server-in-file-attributes.yaml`, está incluso no exemplo de modelos do AWS CloudFormation.

```
aws cloudformation create-stack --stack-name MyPuppetStack --template-body file://
puppet-server-in-file-attributes.yaml
```

4. (Opcional) Para obter o status da criação da pilha, execute o comando a seguir.

```
aws cloudformation describe-stacks --stack-name stack_name
```

5. Quando a criação da pilha for concluída, vá para a próxima seção, the section called “Concluir configuração”. Se a criação da pilha falhar, analise as mensagens de erro mostradas no console para ajudá-lo a resolver os problemas. Para obter mais informações sobre como solucionar erros em AWS CloudFormation pilhas, consulte Solução de problemas no Guia do AWS CloudFormation usuário.

Atualize um OpsWorks para o Puppet Enterprise Server para usar um domínio personalizado

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como atualizar um servidor existente OpsWorks do Puppet Enterprise para usar um domínio e certificado personalizados usando um backup do servidor para criar um novo servidor. Basicamente, você está copiando um servidor existente OpsWorks para o Puppet Enterprise 2.0 criando um novo servidor a partir de um backup e, em seguida, configurando o novo servidor para usar um domínio, certificado e chave privada personalizados.

Tópicos

- Pré-requisitos
- Limitações
- Atualizar um servidor para usar um domínio personalizado
- Consulte também

Pré-requisitos

A seguir estão os requisitos para atualizar um servidor existente OpsWorks do Puppet Enterprise para usar um domínio e um certificado personalizados.

- O servidor que você deseja atualizar (ou copiar) deve estar executando o Puppet Enterprise 2019.8.5.
- Decida qual backup você deseja usar para criar um novo servidor. É necessário ter pelo menos um backup de segurança disponível do servidor que você quer atualizar. Para obter mais informações sobre backups no OpsWorks Puppet Enterprise, consulteCriar um backup de OpsWorks servidor para Puppet Enterprise.
- Prepare a função de serviço e o perfil da instância ARNs que você usou para criar o servidor existente que é a origem do seu backup.
- Certifique-se de que você esteja executando a versão mais atual da AWS CLI. Para obter mais informações sobre a atualização de suas AWS CLI ferramentas, consulte Instalando o AWS CLI no Guia do usuário da interface de linha de comando da AWS.

Limitações

Quando você atualiza um servidor existente criando um novo servidor a partir de um backup, o novo servidor não pode ser exatamente igual ao existente OpsWorks para o servidor Puppet Enterprise.

- Você só pode concluir esse procedimento usando o AWS CLI ou um dos AWS SDKs. Não é possível criar um novo servidor de um backup utilizando o AWS Management Console.

- O novo servidor não pode usar o mesmo nome do servidor existente em uma conta e em uma região da AWS. O nome deve ser diferente do servidor existente que você usou como a origem do backup.
- Os nós que foram conectados ao servidor existente não são gerenciados pelo novo servidor. Você deve fazer um dos seguintes procedimentos.
  - Associe nós diferentes, porque os nós não podem ser gerenciados por mais de um Puppet mestre.
  - Migre os nós do servidor existente (a origem do backup) para o novo servidor e o novo endpoint de domínio personalizado. Para obter mais informações sobre como migrar nós, consulte a documentação do Puppet Enterprise.

Atualizar um servidor para usar um domínio personalizado

Para atualizar um Puppet mestre existente, faça uma cópia dele executando o comando `create-server`, adicionando parâmetros para especificar um backup, um domínio, um certificado e uma chave privada personalizados.

1. Se você não tiver uma função de serviço ou um perfil de instância ARNs disponível para especificar em seu `create-server` comando, siga as etapas de 1 a 5 Crie um servidor Chef Automate usando o AWS CLI para criar uma função de serviço e um perfil de instância que você possa usar.
2. Se ainda não tiver feito isso, localize o backup do Puppet mestre existente no qual você deseja basear um novo servidor com um domínio personalizado. Execute o comando a seguir para mostrar informações sobre todos os OpsWorks backups do Puppet Enterprise em sua conta e em uma região. Anote o ID do backup que você deseja usar.

   ```
   aws opsworks-cm --region region name describe-backups
   ```

3. Crie o servidor OpsWorks for Puppet Enterprise executando o `create-server` comando.
   - O valor `--engine` é `Puppet`, `--engine-model` é `Monolithic`, e `--engine-version` é `2019` ou `2017`.
   - O nome do servidor deve ser exclusivo em sua AWS conta, em cada região. Os nomes dos servidores devem começar com uma letra; letras, números ou hífens (-) são permitidos, até um máximo de 40 caracteres.

- Use o ARN do perfil de instância e o ARN da função de serviço que você copiou nas etapas 3 e 4.
- Os tipos de instância válidos são `c4.large`, `c4.xlarge` ou `c4.2xlarge`. Para obter mais informações sobre as especificações desses tipos de instância, consulte Tipos de instância no Guia EC2 do usuário da Amazon.
- O parâmetro `--engine-attributes` é opcional; se você não especificar uma senha de administrador do Puppet, o processo de criação do servidor gerará uma para você. Se você adicionar `--engine-attributes`, especifique uma `PUPPET_ADMIN_PASSWORD`, uma senha de administrador para fazer login na página da web do console do Puppet Enterprise. A senha deve ter entre 8 e 32 caracteres ASCII.
- Um par de chaves SSH é opcional, mas pode ajudá-lo a se conectar ao Puppet master se você precisar redefinir a senha de administrador do console. Para obter mais informações sobre a criação de um par de chaves SSH, consulte Amazon EC2 Key Pairs no Amazon EC2 User Guide.
- Para usar um domínio personalizado, adicione os seguintes parâmetros ao comando. Caso contrário, o processo de criação do Puppet mestre gerará automaticamente um endpoint para você. Todos os três parâmetros são necessários para configurar um domínio personalizado. Para obter informações sobre requisitos adicionais para usar esses parâmetros, consulte CreateServera Referência da API AWS OpsWorks CM.
  - `--custom-domain` – um endpoint público opcional de um servidor, como `https://aws.my-company.com`.
  - `--custom-certificate` – um certificado HTTPS no formato PEM. O valor pode ser um certificado autoassinado único ou uma cadeia de certificados.
  - `--custom-private-key` – uma chave privada no formato PEM para a conexão com o servidor usando HTTPS. A chave privada não deve ser criptografada e não pode ser protegida por senha nem por frase secreta.
- É necessário fazer a manutenção semanal do sistema. Os valores válidos devem sempre ser especificados no seguinte formato: `DDD:HH:MM`. A hora especificada está em formato de Tempo Universal Coordenado (UTC). Se você não especificar um valor para `--preferred-maintenance-window`, o valor padrão será aleatório, um período de uma hora na terça, quarta ou sexta-feira.
- Os valores válidos para `--preferred-backup-window` devem ser especificados em um dos seguintes formatos: `HH:MM` para backups diários ou `DDD:HH:MM` para backups semanais. A hora é especificada em UTC. O valor padrão é uma hora de início aleatória estabelecida

diariamente. Para cancelar os backups automáticos, adicione o parâmetro `--disable-automated-backup`.

- Para `--security-group-ids`, insira um ou mais grupos de segurança IDs, separados por um espaço.
- Em `--subnet-ids`, insira um ID de sub-rede.

```
aws opsworks-cm create-server --engine "Puppet" --engine-model "Monolithic"
 --engine-version "2019" --server-name "server_name" --instance-profile-arn
 "instance_profile_ARN" --instance-type "instance_type" --engine-attributes
 '{"PUPPET_ADMIN_PASSWORD":"ASCII_password"}' --key-pair "key_pair_name" --
preferred-maintenance-window "ddd:hh:mm" --preferred-backup-window "ddd:hh:mm"
 --security-group-ids security_group_id1 security_group_id2 --service-role-arn
 "service_role_ARN" --subnet-ids subnet_ID
```

O exemplo a seguir cria um Puppet mestre que usa um domínio personalizado.

```
aws opsworks-cm create-server \
    --engine "Puppet" \
    --engine-model "Monolithic" \
    --engine-version "2019" \
    --server-name "puppet-02" \
    --instance-profile-arn "arn:aws:iam::1019881987024:instance-profile/aws-
opsworks-cm-ec2-role" \
    --instance-type "c4.large" \
    --engine-attributes '{"PUPPET_ADMIN_PASSWORD":"zZZzDj2DLYXSZFRv1d"}' \
    --custom-domain "my-puppet-master.my-corp.com" \
    --custom-certificate "-----BEGIN CERTIFICATE----- EXAMPLEqEXAMPLE== -----END
 CERTIFICATE-----" \
    --custom-private-key "-----BEGIN RSA PRIVATE KEY----- EXAMPLEqEXAMPLE= -----END
 RSA PRIVATE KEY-----" \
    --key-pair "amazon-test"
    --preferred-maintenance-window "Mon:08:00" \
    --preferred-backup-window "Sun:02:00" \
    --security-group-ids sg-b00000001 sg-b0000008 \
    --service-role-arn "arn:aws:iam::044726508045:role/service-role/aws-opsworks-
cm-service-role" \
    --subnet-ids subnet-383daa71
```

4. OpsWorks para Puppet Enterprise leva cerca de 15 minutos para criar um novo servidor. Na saída do comando `create-server`, copie o valor do atributo `Endpoint`. Veja um exemplo a seguir.

```
"Endpoint": "puppet-2019-exampleexample.opsworks-cm.us-east-1.amazonaws.com"
```

   Não descarte a saída do comando `create-server` ou feche a sessão de shell, pois a saída pode conter informações importantes que não serão exibidas novamente. Para obter as senhas e o starter kit dos resultados do comando `create-server`, vá para a próxima etapa.

5. Se você optou OpsWorks por fazer com que o Puppet Enterprise gerasse uma senha para você, você pode extraí-la dos `create-server` resultados em um formato utilizável usando um processador JSON, como jq. Após instalar o jq, você pode executar os comandos a seguir para extrair a senha de administrador do Puppet e o starter kit. Se você não forneceu sua própria senha na etapa 3, salve a senha de administrador extraída em um local seguro e conveniente.

```
#Get the Puppet password:
cat resp.json | jq -r '.Server.EngineAttributes[] | select(.Name ==
 "PUPPET_ADMIN_PASSWORD") | .Value'

#Get the Puppet Starter Kit:
cat resp.json | jq -r '.Server.EngineAttributes[] | select(.Name ==
 "PUPPET_STARTER_KIT") | .Value' | base64 -D > starterkit.zip
```

   > **Note**
   >
   > Não é possível gerar novamente um starter kit do Puppet master no AWS Management Console. Ao criar um Puppet master usando o AWS CLI, execute o `jq` comando anterior para salvar o kit inicial codificado em base64 nos resultados como um arquivo ZIP. `create-server`

6. Opcionalmente, se você não extraiu o kit inicial dos resultados do `create-server` comando, você pode baixar um novo kit inicial na página de propriedades do servidor no console do Puppet OpsWorks Enterprise.

7. Se você não estiver usando um domínio personalizado, vá para a próxima etapa. Se você estiver usando um domínio personalizado com o servidor, crie uma entrada CNAME na ferramenta de gerenciamento de DNS da sua empresa para direcionar seu domínio personalizado OpsWorks para o endpoint do Puppet Enterprise que você copiou na etapa 4.

Você não poderá acessar nem se conectar a um servidor com um domínio personalizado se não concluir esta etapa.

8. Quando o processo de criação do servidor for concluído, vá para Configurar o Puppet master usando o Starter Kit.

Consulte também

- Crie um Puppet Enterprise Master usando o AWS CLI
- Fazer o backup e a restauração de um servidor do OpsWorks para Puppet Enterprise
- CreateServerna referência da API AWS OpsWorks CM
- create-server na AWS CLI Command Reference

Trabalhando com tags em AWS OpsWorks for Puppet Enterprise recursos

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As tags são palavras ou frases que atuam como metadados para identificar e organizar os recursos da AWS. No OpsWorks Puppet Enterprise, um recurso pode ter até 50 tags aplicadas pelo usuário. Cada tag consiste em uma chave e em um valor opcional. Você pode aplicar tags aos seguintes recursos no OpsWorks Puppet Enterprise:

- OpsWorks para servidores Puppet Enterprise
- Backups de OpsWorks para servidores Puppet Enterprise

As tags nos AWS recursos podem ajudar você a monitorar custos, controlar o acesso aos recursos, agrupar recursos para automatizar tarefas ou organizar recursos por finalidade ou estágio do ciclo de

vida. Para obter mais informações sobre os benefícios das tags, consulte Estratégias de marcação da AWS no AWS Answers e Usar tags de alocação de custos no Guia do usuário do Gerenciamento de Faturamento e Custos da AWS .

Para usar tags para controlar o acesso aos OpsWorks servidores ou backups do Puppet Enterprise, crie ou edite declarações de política no AWS Identity and Access Management (IAM). Para obter mais informações, consulte Controlar o acesso aos recursos da AWS usando tags de recurso no Guia do usuário do AWS Identity and Access Management .

Quando você aplica tags a um OpsWorks mestre do Puppet Enterprise, as tags também são aplicadas aos backups do mestre, ao bucket do Amazon S3 que armazena os backups, à instância EC2 Amazon do mestre, aos segredos do mestre que estão armazenados AWS Secrets Manager e ao endereço IP elástico usado pelo mestre. As tags não são propagadas para a AWS CloudFormation pilha AWS OpsWorks usada para criar seu Puppet master.

Tópicos

- Como as tags funcionam em AWS OpsWorks for Puppet Enterprise
- Adicionar e gerenciar tags no OpsWorks Puppet Enterprise (Console)
- Adicionar e gerenciar tags no OpsWorks Puppet Enterprise (CLI)
- Consulte também

Como as tags funcionam em AWS OpsWorks for Puppet Enterprise

Nesta versão, é possível adicionar e gerenciar tags usando a API do AWS OpsWorks CM ou o AWS Management Console. AWS OpsWorks O CM também tenta adicionar tags que você adiciona a um servidor aos AWS recursos associados ao servidor, incluindo a EC2 instância, os segredos no Secrets Manager, o endereço IP elástico, o grupo de segurança, o bucket do S3 e os backups.

A tabela a seguir fornece uma visão geral de como você adiciona e gerencia tags no OpsWorks Puppet Enterprise.

Ação	O que usar
Adicione tags a um novo OpsWorks servidor do Puppet Enterprise ou a um backup que você está criando manualmente.	• Escolha Create Puppet Enterprise server (Criar servidor Puppet Enterprise) e adicione tags na página Configure advanced settings (Definir configurações avançadas).

Ação	O que usar
	• Escolha Create backup (Criar backup) na página Backups de um servidor existente e adicione tags na página Create a backup of your Puppet Enterprise server (Criar um backup do servidor Puppet Enterprise). • Adicione um parâmetro `Tags` aos comandos `CreateServer` ou `CreateBackup`.
Visualize tags em um recurso.	• Na página de detalhes do servidor, escolha Tags no painel de navegação. • Na página Backups do servidor, selecione um backup e escolha Edit backup (Editar backup). • Execute o comando `ListTagsForResource`.
Adicione tags a um servidor existente OpsWorks do Puppet Enterprise ou a um backup, independentemente de o backup ter sido criado manual ou automaticamente.	• Na página de detalhes do servidor, escolha Tags no painel de navegação e escolha Edit (Editar). • Na página Backups do servidor, selecione um backup e escolha Edit backup (Editar backup). • Execute o comando `TagResource`.
Exclua tags de um recurso.	• Na página de detalhes do servidor, escolha Tags no painel de navegação e escolha Edit (Editar). Escolha X ao lado das tags que deseja excluir. • Na página Backups do servidor, selecione um backup e escolha Edit backup (Editar backup). Escolha X ao lado das tags que deseja excluir. • Execute o comando `UntagResource`.

As respostas `DescribeServers` e `DescribeBackups` não incluem informações de tag. Para mostrar tags, use a API `ListTagsForResource`.

Adicionar e gerenciar tags no OpsWorks Puppet Enterprise (Console)

Os procedimentos nesta seção são realizados na AWS Management Console.

Se você adicionar tags, uma chave de tag não poderá ficar vazia. A chave pode ter, no máximo, 127 caracteres e pode conter somente letras, números ou separadores Unicode, ou os seguintes caracteres especiais: `+ - = . _ : / @`. Um valor de tag é opcional. É possível adicionar uma tag que tenha uma chave, mas nenhum valor. O valor pode ter, no máximo, 255 caracteres e conter somente letras, números ou separadores Unicode, ou os seguintes caracteres especiais: `+ - = . _ : / @`

Tópicos

- Adicionar tags a um novo OpsWorks para o Puppet Enterprise Server (console)
- Adicionar tags a um novo backup (console)
- Adicionar ou visualizar tags em um servidor existente (console)
- Adicionar ou visualizar tags em um backup existente (console)
- Excluir tags de um servidor (console)
- Excluir tags de um backup (console)

Adicionar tags a um novo OpsWorks para o Puppet Enterprise Server (console)

1. Certifique-se de preencher todos os pré-requisitos para criar um mestre para o Puppet OpsWorks Enterprise.
2. Siga as etapas de 1 a 8 em Crie um Puppet Enterprise Master usando o AWS Management Console.
3. Depois que especificar as configurações de backup automatizado, adicione as tags na área Tags da página Configure advanced settings (Definir configurações avançadas). É possível adicionar um máximo de 50 tags. Quando terminar de adicionar tags, escolha Próximo.
4. Vá para a etapa 11 de Crie um Puppet Enterprise Master usando o AWS Management Console e revise as configurações escolhidas para o novo servidor.

Adicionar tags a um novo backup (console)

1. Na página inicial do OpsWorks Puppet Enterprise, escolha um Puppet master existente.
2. Na página de detalhes do servidor, escolha Backups no painel de navegação.
3. Na página Backups, escolha Create backup (Criar backup).
4. Adicione as tags. Escolha Create (Criar) quando terminar de adicionar as tags.

Adicionar ou visualizar tags em um servidor existente (console)

1. Na página inicial do OpsWorks Puppet Enterprise, escolha um Puppet master existente para abrir sua página de detalhes.
2. Escolha Tags no painel de navegação ou, na parte inferior da página de detalhes, escolha View all tags (Visualizar todas as tags).
3. Na página Tags, escolha Edit (Editar).
4. Adicione ou edite as tags no servidor. Escolha Salvar quando terminar.

> **Note**
>
> Lembre-se de que alterar as tags no seu Puppet master também altera as tags nos recursos associados ao servidor, como a EC2 instância, o endereço IP elástico, o grupo de segurança, o bucket do S3 e os backups.

Adicionar ou visualizar tags em um backup existente (console)

1. Na página inicial do OpsWorks Puppet Enterprise, escolha um Puppet master existente para abrir sua página de detalhes.
2. Escolha Backups no painel de navegação ou, na área Recent backups (Backups recentes) da página de detalhes, escolha View all backups (Visualizar todos os backups).
3. Na página Backups, escolha um backup para gerenciar e, depois, escolha Edit backup (Editar backup).
4. Adicione ou edite as tags no backup. Escolha Update (Atualizar) quando terminar.

Excluir tags de um servidor (console)

1. Na página inicial do OpsWorks Puppet Enterprise, escolha um Puppet master existente para abrir sua página de detalhes.
2. Escolha Tags no painel de navegação ou, na parte inferior da página de detalhes, escolha View all tags (Visualizar todas as tags).
3. Na página Tags, escolha Edit (Editar).
4. Escolha X ao lado de uma tag para excluí-la. Escolha Salvar quando terminar.

> **Note**
>
> Lembre-se de que alterar as tags no seu Puppet master também altera as tags nos recursos associados ao servidor, como a EC2 instância, o endereço IP elástico, o grupo de segurança, o bucket do S3 e os backups.

Excluir tags de um backup (console)

1. Na página inicial do OpsWorks Puppet Enterprise, escolha um Puppet master existente para abrir sua página de detalhes.
2. Escolha Backups no painel de navegação ou, na área Recent backups (Backups recentes) da página de detalhes, escolha View all backups (Visualizar todos os backups).
3. Na página Backups, escolha um backup para gerenciar e, depois, escolha Edit backup (Editar backup).
4. Escolha X ao lado de uma tag para excluí-la. Escolha Update (Atualizar) quando terminar.

Adicionar e gerenciar tags no OpsWorks Puppet Enterprise (CLI)

Os procedimentos nesta seção são realizados na AWS CLI. Verifique se você está executando a versão mais recente do AWS CLI antes de começar a trabalhar com tags. Para obter mais informações sobre como instalar ou atualizar o AWS CLI, consulte Instalando o AWS CLI no Guia AWS Command Line Interface do Usuário.

Se você adicionar tags, uma chave de tag não poderá ficar vazia. A chave pode ter, no máximo, 127 caracteres e pode conter somente letras, números ou separadores Unicode, ou os seguintes caracteres especiais: `+ - = . _ : / @`. Um valor de tag é opcional. É possível adicionar uma

tag que tenha uma chave, mas nenhum valor. O valor pode ter, no máximo, 255 caracteres e conter somente letras, números ou separadores Unicode, ou os seguintes caracteres especiais: `+ - = . _ : / @`

Tópicos

- Adicionar tags a um novo OpsWorks para o Puppet Enterprise Server (CLI)
- Adicionar tags a um novo backup (CLI)
- Adicionar tags a backups ou servidores existentes (CLI)
- Listar tags de recurso (CLI)
- Excluir tags de um recurso (CLI)

Adicionar tags a um novo OpsWorks para o Puppet Enterprise Server (CLI)

Você pode usar o AWS CLI para adicionar tags ao criar um servidor OpsWorks para o Puppet Enterprise. Este procedimento não descreve na íntegra como criar um servidor. Para obter informações detalhadas sobre como criar um servidor OpsWorks para o Puppet Enterprise usando o AWS CLI, consulte Crie um Puppet Enterprise Master usando o AWS CLI este guia. É possível adicionar até 50 tags a um servidor.

1. Certifique-se de preencher todos os pré-requisitos para criar um servidor para o Puppet OpsWorks Enterprise.
2. Conclua as etapas 1 a 4 de Crie um Puppet Enterprise Master usando o AWS CLI.
3. Na etapa 5, ao executar o comando `create-server`, adicione o parâmetro `--tags` ao comando, conforme mostrado no exemplo a seguir.

   ```
   aws opsworks-cm create-server ... --tags Key=Key1,Value=Value1
    Key=Key2,Value=Value2
   ```

   Veja a seguir um exemplo que mostra somente a parte de tags do comando `create-server`.

   ```
   aws opsworks-cm create-server ... --tags Key=Stage,Value=Production
    Key=Department,Value=Marketing
   ```

4. Conclua as etapas restantes em Crie um Puppet Enterprise Master usando o AWS CLI. Para verificar se as tags foram adicionadas ao novo servidor, siga as etapas em Listar tags de recurso (CLI) neste tópico.

Adicionar tags a um novo backup (CLI)

Você pode usar o AWS CLI para adicionar tags ao criar um novo backup manual de um servidor OpsWorks para o Puppet Enterprise. Este procedimento não descreve na íntegra como criar um backup manual. Para obter informações detalhadas sobre como criar um backup manual, consulte “Para realizar um backup manual no AWS CLI" emCriar um backup de OpsWorks servidor para Puppet Enterprise. É possível adicionar até 50 tags a um backup. Se um servidor tiver tags, novos backups serão marcados automaticamente com as tags do servidor.

Por padrão, quando você cria um novo servidor OpsWorks para o Puppet Enterprise, os backups automatizados são habilitados. É possível adicionar tags a um backup automatizado executando o comando `tag-resource`, descrito em Adicionar tags a backups ou servidores existentes (CLI) neste tópico.

- Para adicionar tags a um backup manual à medida que cria o backup, execute o comando a seguir. Somente a parte de tags do comando é exibida. Para obter um exemplo do comando `create-backup` completo, consulte "Como executar um backup manual na AWS CLI" em Criar um backup de OpsWorks servidor para Puppet Enterprise.

  ```
  aws opsworks-cm create-backup ... --tags Key=Key1,Value=Value1
   Key=Key2,Value=Value2
  ```

  O exemplo a seguir mostra somente a parte de tags do comando `create-backup`.

  ```
  aws opsworks-cm create-backup ... --tags Key=Stage,Value=Production
   Key=Department,Value=Marketing
  ```

Adicionar tags a backups ou servidores existentes (CLI)

Você pode executar o `tag-resource` comando para adicionar tags aos servidores ou backups existentes OpsWorks do Puppet Enterprise (independentemente de os backups terem sido criados automática ou manualmente). Especifique o número de recurso da Amazon (ARN) de um recurso de destino para adicionar tags a ele.

1. Para obter o ARN do recurso ao qual deseja aplicar tags:

   - Para um servidor, execute `describe-servers --server-name` *`server_name`*. Os resultados do comando mostram o ARN do servidor.

- Para um backup, execute `describe-backups --backup-id` *`backup_ID`*. Os resultados do comando mostram o ARN do backup. Você também pode executar `describe-backups --server-name` *`server_name`* para mostrar informações sobre todos os backups de um servidor específico OpsWorks do Puppet Enterprise.

O exemplo a seguir mostra somente o `ServerArn` nos resultados de um comando `describe-servers --server-name opsworks-cm-test`. O valor `ServerArn` é adicionado a um comando `tag-resource` para adicionar tags ao servidor.

```
{
    "Servers": [
        {
            ...
            "ServerArn": "arn:aws:opsworks-cm:us-west-2:123456789012:server/
opsworks-cm-test/EXAMPLEd-66b0-4196-8274-d1a2bEXAMPLE"
        }
    ]
}
```

2. Execute o comando `tag-resource` com o ARN retornado na etapa 1.

```
aws opsworks-cm tag-resource --resource-arn "server_or_backup_ARN" --tags
 Key=Key1,Value=Value1 Key=Key2,Value=Value2
```

Veja um exemplo a seguir.

```
aws opsworks-cm tag-resource --resource-arn "arn:aws:opsworks-cm:us-
west-2:123456789012:server/opsworks-cm-test/EXAMPLEd-66b0-4196-8274-d1a2bEXAMPLE"
 --tags Key=Stage,Value=Production Key=Department,Value=Marketing
```

3. Para verificar se as tags foram adicionadas com êxito, vá para o próximo procedimento, Listar tags de recurso (CLI).

Listar tags de recurso (CLI)

Você pode executar o `list-tags-for-resource` comando para mostrar as tags anexadas aos OpsWorks servidores ou backups do Puppet Enterprise. Especifique o ARN de um recurso de destino para exibir suas tags.

1. Para obter o ARN do recurso para o qual deseja listar tags:

   - Para um servidor, execute `describe-servers --server-name` *`server_name`*. Os resultados do comando mostram o ARN do servidor.
   - Para um backup, execute `describe-backups --backup-id` *`backup_ID`*. Os resultados do comando mostram o ARN do backup. Você também pode executar `describe-backups --server-name` *`server_name`* para mostrar informações sobre todos os backups de um servidor específico OpsWorks do Puppet Enterprise.

2. Execute o comando `list-tags-for-resource` com o ARN retornado na etapa 1.

```
aws opsworks-cm list-tags-for-resource --resource-arn "server_or_backup_ARN"
```

   Veja um exemplo a seguir.

```
aws opsworks-cm tag-resource --resource-arn "arn:aws:opsworks-cm:us-
west-2:123456789012:server/opsworks-cm-test/EXAMPLEd-66b0-4196-8274-d1a2bEXAMPLE"
```

   Se houver tags no recurso, o comando retornará resultados como os seguintes.

```
{
    "Tags": [
        {
            "Key": "Stage",
            "Value": "Production"
        },
        {
            "Key": "Department",
            "Value": "Marketing"
        }
    ]
}
```

Excluir tags de um recurso (CLI)

Você pode executar o `untag-resource` comando para excluir tags dos OpsWorks servidores ou backups do Puppet Enterprise. Se o recurso for excluído, as tags no recurso também serão excluídas. Especifique o número de recurso da Amazon (ARN) de um recurso de destino para remover as tags dele.

1. Para obter o ARN do recurso do qual deseja remover tags:
   - Para um servidor, execute `describe-servers --server-name server_name`. Os resultados do comando mostram o ARN do servidor.
   - Para um backup, execute `describe-backups --backup-id backup_ID`. Os resultados do comando mostram o ARN do backup. Você também pode executar `describe-backups --server-name server_name` para mostrar informações sobre todos os backups de um servidor específico OpsWorks do Puppet Enterprise.
2. Execute o comando `untag-resource` com o ARN retornado na etapa 1. Especifique somente as tags que deseja excluir.

   ```
   aws opsworks-cm untag-resource --resource-arn "server_or_backup_ARN" --tags
    Key=Key1,Value=Value1 Key=Key2,Value=Value2
   ```

   Neste exemplo, o comando `untag-resource` remove somente a tag com uma chave de `Stage` e um valor de `Production`.

   ```
   aws opsworks-cm untag-resource --resource-arn "arn:aws:opsworks-cm:us-
   west-2:123456789012:server/opsworks-cm-test/EXAMPLEd-66b0-4196-8274-d1a2bEXAMPLE"
    --tags Key=Stage,Value=Production
   ```
3. Para verificar se as tags foram excluídas com êxito, siga as etapas em Listar tags de recurso (CLI) neste tópico.

Consulte também

- Crie um Puppet Enterprise Master usando o AWS CLI
- Criar um backup de OpsWorks servidor para Puppet Enterprise
- Estratégias de marcação da AWS
- Controlando o acesso aos AWS recursos usando tags de recursos no Guia AWS Identity and Access Management do usuário
- Usar tags de alocação de custos no Guia do usuário do Gerenciamento de Faturamento e Custos da AWS .
- CreateBackup na Referência de API do AWS OpsWorks CM
- CreateServer na Referência de API do AWS OpsWorks CM
- TagResource na Referência de API do AWS OpsWorks CM

- ListTagsForResource na Referência de API do AWS OpsWorks CM
- UntagResource na Referência de API do AWS OpsWorks CM

Fazer o backup e a restauração de um servidor do OpsWorks para Puppet Enterprise

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como fazer backup e restaurar um servidor OpsWorks para Puppet Enterprise.

Tópicos

- Criar um backup de OpsWorks servidor para Puppet Enterprise
- Restaurar um servidor do OpsWorks para Puppet Enterprise a partir de um backup

Criar um backup de OpsWorks servidor para Puppet Enterprise

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode definir um backup recorrente diário ou semanal OpsWorks de um servidor do Ouppet Enterprise e fazer com que o serviço armazene os backups no Amazon Simple Storage Service (Amazon S3) em seu nome. Como alternativa, você pode fazer backups manuais sob demanda.

Como os backups são armazenados no Amazon S3, há a cobrança de taxas adicionais. É possível definir um período de retenção de backups de até 30 gerações. Você pode enviar uma solicitação de serviço para alterar esse limite usando os canais de AWS suporte. O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Você pode adicionar tags aos backups de um master OpsWorks para Puppet Enterprise. Se você adicionou tags a um master OpsWorks para Puppet Enterprise master, os backups automatizados do Puppet master herdarão essas tags. Para obter mais informações sobre como adicionar e gerenciar tags em backups, consulte Trabalhando com tags em AWS OpsWorks for Puppet Enterprise recursos neste guia.

Tópicos

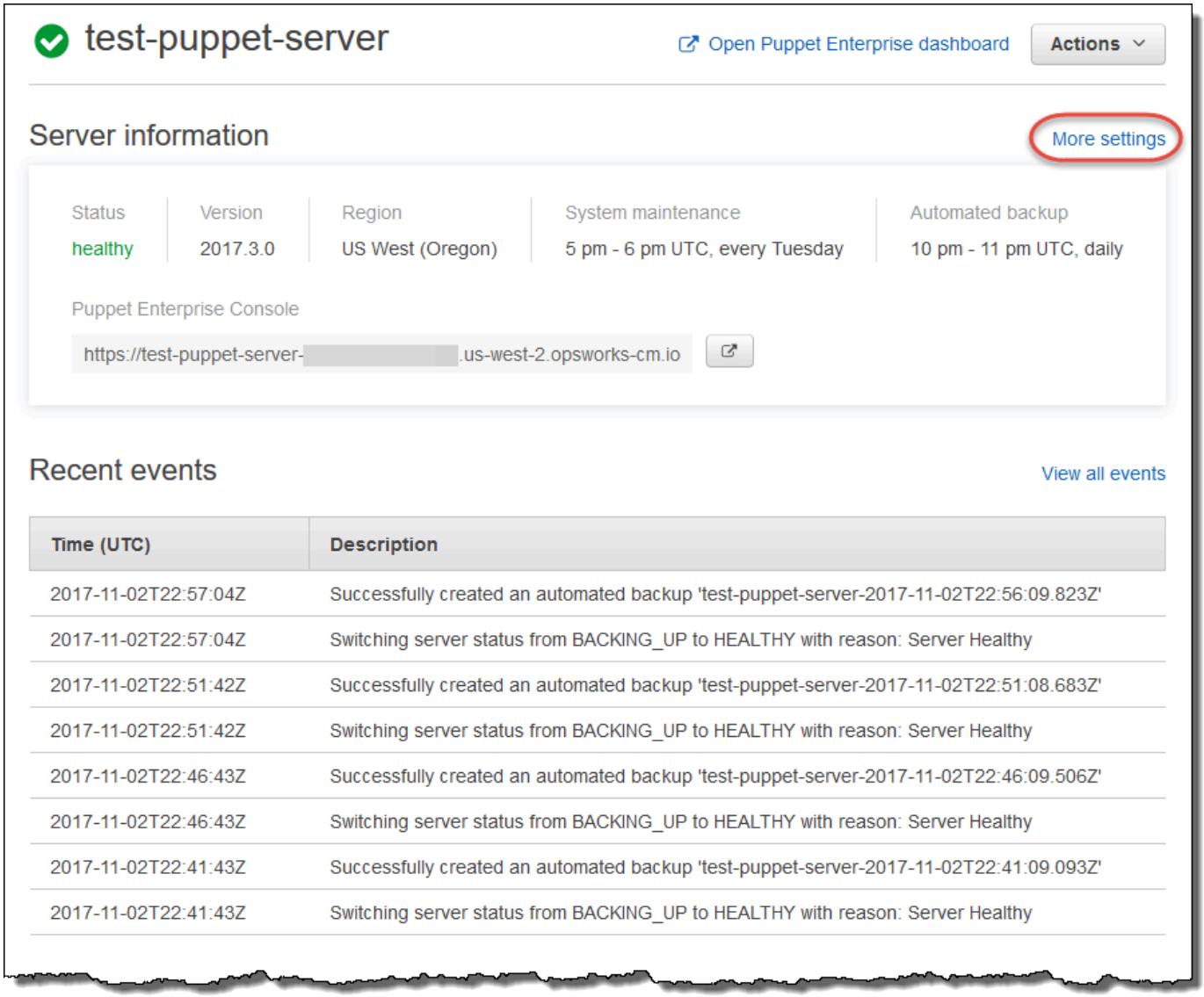
- Backups automatizados
- Backups manuais
- Excluir backups

Backups automatizados

Quando você configura o servidor OpsWorks para Puppet Enterprise é possível escolher backups manuais ou automáticos. OpsWorks para Puppet Enterprise inicia os backups automáticos na hora e no dia que você escolher na seção Backup automático da página de configurações Configurar opções avançadas do assistente de configuração. Depois que o servidor estiver online, você poderá alterar as configurações de backup executando as seguintes etapas na página de propriedades do servidor.

Para alterar configurações de backup automático

1. Na página de propriedades do servidor, escolha More settings.



2. Para desativar backups automatizados, escolha No para a opção Enable automated backups. Salve as alterações; não é necessário ir para a próxima etapa.
3. Na seção Automated Backup, altere a frequência, o horário de início ou as gerações que serão mantidas. Salve as alterações.

Backups manuais

Você pode iniciar um backup manual a qualquer momento no AWS Management Console, ou executando o comando AWS CLI create-backup. Os backups manuais não estão incluídos no máximo de 30 gerações de backups automatizados que são armazenados. Um máximo de 10 backups manuais são armazenados e devem ser excluídos manualmente do Amazon S3.

Para realizar um backup manual no AWS Management Console

1. Na página Puppet Enterprise servers, selecione o servidor do qual você deseja fazer backup.
2. Na página de propriedades do servidor, no painel de navegação esquerdo, selecione Backups.
3. Escolha Create backup.
4. O backup manual é finalizado quando a página mostra uma marca de seleção verde na coluna Status do backup.

Para realizar um backup manual no AWS CLI

Você pode adicionar tags ao criar um backup manual de um servidor OpsWorks para Puppet Enterprise. Para obter mais informações sobre como adicionar tags ao criar um backup manual, consulte Adicionar tags a um novo backup (CLI).

- Para iniciar um backup manual, execute o AWS CLI comando a seguir.

```
aws opsworks-cm --region region name create-backup --server-name "Puppet server
 name" --description "optional descriptive string"
```

Excluir backups

A exclusão de um backup permanentemente o exclui do bucket do S3 em que os backups estão armazenados.

Para excluir um backup no AWS Management Console

1. Na página Puppet Enterprise servers, selecione o servidor do qual você deseja fazer backup.
2. Na página de propriedades do servidor, no painel de navegação esquerdo, selecione Backups.
3. Escolha o backup que você deseja excluir e, em seguida, selecione Delete backup. Você pode selecionar somente um backup por vez.
4. Quando você for solicitado a confirmar a exclusão, marque a caixa de seleção Delete the backup, which is stored in an S3 bucket e, em seguida, escolha Yes, Delete.

Para excluir um backup no AWS CLI

- Para excluir um backup, execute o AWS CLI comando a seguir, substituindo o valor de `--backup-id` pelo ID do backup que você deseja excluir. O backup IDs

está no formato*ServerName-yyyyMMddHHmmssSSS*. Por exemplo, .**puppet-server-20171218132604388**

```
aws opsworks-cm --region region name delete-backup --backup-id ServerName-yyyyMMddHHmmssSSS
```

Restaurar um servidor do OpsWorks para Puppet Enterprise a partir de um backup

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Após navegar pelos backups disponíveis, você poderá escolher facilmente um ponto no tempo para restaurar o servidor OpsWorks para Puppet Enterprise. Backups de servidor contêm dados de software persistentes de gerenciamento de configuração, como módulos, classes, associações de nó, informações do banco de dados (incluindo relatórios, fatos etc.). Realizar uma restauração in-loco de um servidor (ou seja, restaurar o servidor do servidor do servidor do servidor do Ouppet Enterprise existente OpsWorks para uma nova EC2 instância) registrará novamente os nós que foram registrados no momento do backup usado para restaurar o servidor e alternará o tráfego para a nova instância se a restauração for bem-sucedida, e o estado do servidor do servidor do servidor do servidor do Ouppet Enterprise restaurado OpsWorks para Puppet Enterprise será. `Healthy` Restaurar para um servidor OpsWorks para Puppet Enterprise recém-criado não mantém conexões de nó. A restauração de um servidor não atualiza a versão do software do Puppet;, ela aplica as mesmas versões do Puppet e dados de gerenciamento de configuração que estão disponíveis no backup que você escolher.

A restauração de um servidor normalmente leva mais tempo do que a criação de um novo servidor; o tempo depende do tamanho do backup escolhido. Após a conclusão da restauração, a EC2 instância antiga permanece no `Stopped` estado `Running` ou, mas apenas temporariamente. Ela será encerrada por fim.

Nesta versão, você pode usar a AWS CLI para restaurar um Puppet master OpsWorks para Puppet Enterprise.

> **Note**
>
> Você também pode executar o comando restore-server para alterar o tipo de instância atual ou para restaurar ou definir a chave SSH se ela for perdida ou comprometida.

Para restaurar um servidor a partir de um backup

1. No AWS CLI, execute o comando a seguir para retornar uma lista dos backups disponíveis e seus IDs. Anote o ID do backup que você deseja usar. O backup IDs está no formato*myServerName-yyyyMMddHHmmssSSS*.

   ```
   aws opsworks-cm --region region name describe-backups
   ```

2. Execute o seguinte comando:

   ```
   aws opsworks-cm --region region name restore-server --backup-id "myServerName-
   yyyyMMddHHmmssSSS" --instance-type "Type of instance" --key-pair "name of your EC2
    key pair" --server-name "name of Puppet master"
   ```

   Veja um exemplo a seguir.

   ```
   aws opsworks-cm --region us-west-2 restore-server --backup-id
    "MyPuppetServer-20161120122143125" --server-name "MyPuppetServer"
   ```

3. Aguarde até que a restauração esteja concluída.

Manutenção do sistema OpsWorks para Puppet Enterprise

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se

você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A manutenção obrigatória do sistema garante que as últimas versões AWS testadas do Puppet Server, incluindo atualizações de segurança, estejam sempre em execução em um OpsWorks servidor do Puppet Enterprise. É necessário fazer a manutenção do sistema pelo menos uma vez por semana. Ao usar o AWS CLI, você pode configurar a manutenção automática diária, se desejar. Você também pode usar o AWS CLI para realizar a manutenção do sistema sob demanda, além da manutenção programada do sistema.

Quando novas versões do software do Puppet forem disponibilizadas, a manutenção do sistema é projetada para atualizar a versão do servidor Puppet automaticamente no servidor, assim que ela passar nos testes da AWS. A AWS realiza testes extensivos para verificar se as atualizações do Puppet estão prontas para produção e não interrompem os ambientes existentes dos clientes, portanto, pode haver atrasos entre os lançamentos do software Puppet e sua disponibilidade para aplicação e os existentes nos servidores do Puppet Enterprise. OpsWorks Para atualizar versões de software do Puppet disponíveis sob demanda, consulte Iniciar a manutenção do sistema sob demanda neste tópico.

A manutenção do sistema inicia uma nova instância a partir de um backup que é executado como parte do processo de manutenção, o que ajuda a reduzir o risco de EC2 instâncias Amazon degradadas ou danificadas que passam por manutenção periódica.

> **Important**
>
> A manutenção do sistema exclui todos os arquivos ou configurações personalizadas que você adicionou ao servidor do OpsWorks Puppet Enterprise. Para obter mais informações sobre como reparar a configuração ou a perda de arquivos, consulte Restaurar configurações e arquivos personalizados após a manutenção neste tópico.

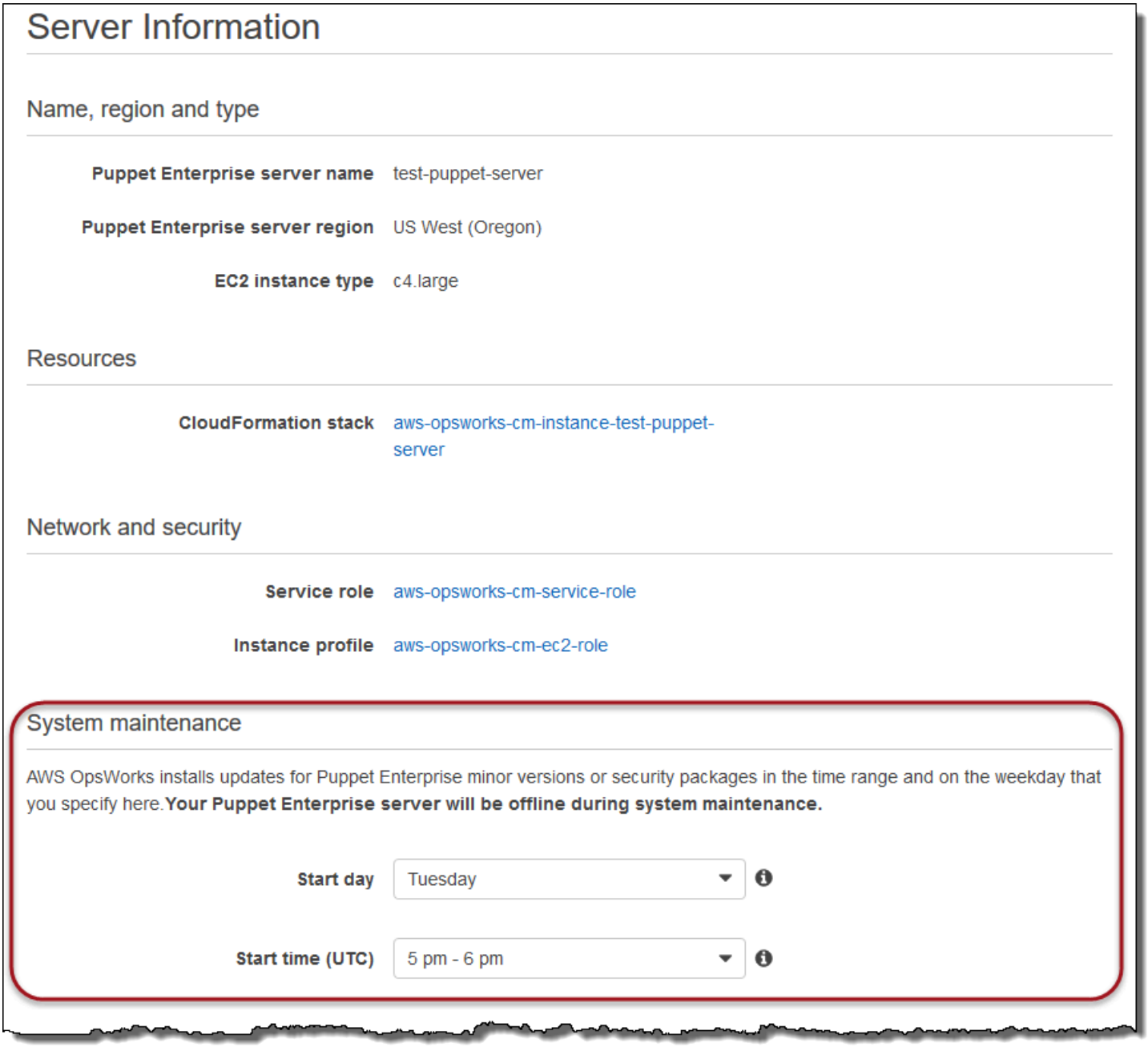
Tópicos

- Configurar a manutenção do sistema
- Iniciar a manutenção do sistema sob demanda
- Restaurar configurações e arquivos personalizados após a manutenção

Configurar a manutenção do sistema

Ao criar um novo servidor OpsWorks para o Puppet Enterprise, você pode configurar um dia e hora da semana, no Horário Universal Coordenado (UTC), para iniciar a manutenção do sistema. A manutenção começa na hora que você especificar. Como é recomendável que o servidor esteja offline durante a manutenção do sistema, escolha um horário de baixa demanda do servidor no horário comercial regular. Enquanto a manutenção estiver em andamento, o status do servidor será `UNDER_MAINTENANCE`.

Você também pode alterar as configurações de manutenção do sistema em um servidor existente OpsWorks do Puppet Enterprise, alterando as configurações na área Manutenção do sistema da página Configurações do seu servidor, conforme mostrado na captura de tela a seguir.



Na seção System maintenance, defina o dia e a hora em que você deseja que a manutenção do sistema se inicie.

Configurando a manutenção do sistema usando o AWS CLI

Você também pode configurar a hora do início automático da manutenção do sistema usando a AWS CLI. O AWS CLI permite que você configure a manutenção automática diária, se desejar, omitindo o prefixo de três caracteres do dia da semana.

Em um comando `create-server`, adicione o parâmetro `--preferred-maintenance-window`, depois de especificar os requisitos para a criação da instância do servidor (como o tipo de instância e o Nome de região da Amazon (ARN) do perfil da instância e da função de serviço). No exemplo de `create-server` a seguir, `--preferred-maintenance-window` está definido como `Mon:08:00`. Isso significa que você configurou a manutenção para começar toda segunda-feira às 8:00 da manhã. UTC.

```
aws opsworks-cm create-server --engine "Puppet" --engine-model "Monolithic"
 --engine-version "2017" --server-name "puppet-06" --instance-profile-arn
 "arn:aws:iam::1119001987000:instance-profile/aws-opsworks-cm-ec2-role"
 --instance-type "c4.large" --key-pair "amazon-test" --service-role-arn
 "arn:aws:iam::044726508045:role/aws-opsworks-cm-service-role" --preferred-maintenance-
window "Mon:08:00"
```

Em um comando `update-server`, você pode atualizar o valor `--preferred-maintenance-window` sozinho, se desejar. No exemplo a seguir, a janela de manutenção está configurada para toda sexta-feira às 18h15. UTC.

```
aws opsworks-cm update-server --server-name "puppet-06" --preferred-maintenance-window
 "Fri:18:15"
```

Para alterar o horário de início da janela de manutenção para todos os dias às 18h15, UTC, omita o prefixo de dia da semana de três caracteres, como mostrado no exemplo a seguir.

```
aws opsworks-cm update-server --server-name "puppet-06" --preferred-maintenance-window
 "18:15"
```

Para obter mais informações sobre como definir a janela preferencial de manutenção do sistema usando o AWS CLI, consulte create-server e update-server.

Iniciar a manutenção do sistema sob demanda

Para iniciar a manutenção do sistema sob demanda, fora da manutenção automática semanal ou diária configurada, execute o AWS CLI comando a seguir. Não é possível iniciar a manutenção sob demanda no AWS Management Console.

```
aws opsworks-cm start-maintenance --server-name server_name
```

Para obter mais informações sobre esse comando, consulte start-maintenance.

Restaurar configurações e arquivos personalizados após a manutenção

A manutenção do sistema pode excluir ou alterar arquivos ou configurações personalizados que você adicionou ao seu servidor OpsWorks para o Puppet Enterprise.

Se, após uma execução de manutenção, faltarem arquivos ou configurações do Puppet master que você adicionou usando `RunCommand` ou SSH, você pode usar uma Amazon Machine Image (AMI) para iniciar uma nova instância da Amazon EC2 . AMIs estão disponíveis e são criados a partir da configuração de pré-manutenção de um servidor.

A nova instância estará no mesmo estado em que o Puppet master estava antes da manutenção e deve incluir os arquivos e configurações que estão faltando.

> **Important**
>
> Não é possível usar a nova instância para restaurar seu servidor, pois ela não pode ser executada como Puppet master. Você pode usar a instância apenas para recuperar seus arquivos e configurações.

Para iniciar uma EC2 instância a partir de uma AMI, no EC2 console da Amazon, abra o assistente Launch, escolha My e AMIs, em seguida, escolha a AMI que tem o nome do seu servidor. Siga as etapas do EC2 assistente da Amazon como você faria para iniciar qualquer outra instância.

Adicionando nós automaticamente OpsWorks para o Puppet Enterprise

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico descreve como adicionar automaticamente nós do Amazon Elastic Compute Cloud (Amazon EC2) ao seu servidor OpsWorks for Puppet Enterprise. Em Adicionar nós para o Puppet

master gerenciar, foi ensinado como usar o comando `associate-node` para adicionar um nó de cada vez no servidor Puppet Enterprise. O código neste tópico mostra como adicionar nós automaticamente usando o método desassistido. O método recomendado de associação autônoma (ou automática) de novos nós é configurar os dados do EC2 usuário da Amazon. Por padrão, um servidor OpsWorks for Puppet Enterprise já `puppet-agent`está disponível para os sistemas operacionais de nós Ubuntu, Amazon Linux e RHEL.

Para obter informações sobre como desassociar um nó, consulte Desassociar um Node de um OpsWorks for Puppet Enterprise Server este guia e `disassociate-node`a documentação da API OpsWorks for Puppet Enterprise.

Etapa 1: criar um perfil do IAM para usar como o perfil de instância

Crie uma função AWS Identity and Access Management (IAM) para usar como seu perfil de EC2 instância e anexe a política a seguir à função do IAM. Essa política permite que a `opsworks-cm` API se comunique com a EC2 instância durante o registro do nó. Para obter mais informações sobre perfis de instância, consulte Como usar perfis de instância na EC2 documentação da Amazon. Para obter informações sobre como criar uma função do IAM, consulte Como criar uma função do IAM no console na EC2 documentação da Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "opsworks-cm:AssociateNode",
                "opsworks-cm:DescribeNodeAssociationStatus",
                "opsworks-cm:DescribeServers",
                "ec2:DescribeTags"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Etapa 2: criar instâncias com um script de associação autônoma

Para criar EC2 instâncias, você pode copiar o script de dados do usuário incluído no Starter Kit para a `userdata` seção de instruções de EC2 instância, configurações de lançamento em grupo do

Amazon EC2 Auto Scaling ou um modelo. AWS CloudFormation O script é compatível somente com EC2 instâncias que executam os sistemas operacionais Ubuntu e Amazon Linux. Para obter mais informações sobre como adicionar scripts aos dados do usuário, consulte Executando comandos na sua instância Linux no lançamento na EC2 documentação da Amazon. A maneira mais fácil de criar um novo nó é usar o assistente de execução de EC2 instâncias da Amazon. Esta descrição usa a configuração do módulo de exemplo do servidor da web Apache descrito em Introdução ao OpsWorks Puppet Enterprise.

1. O script de dados do usuário no Starter Kit executa o comando da `opsworks-cm` API `associate-node` para associar um novo nó ao Puppet master. Nesta versão, ele também instala a versão atual do AWS CLI no nó para você, caso ainda não esteja executando a maioria das up-to-date versões. Salve esse script em um local conveniente como `userdata.sh`.

   Por padrão, o nome do novo nó registrado é o ID da instância.
2. Siga o procedimento em Iniciando uma instância na EC2 documentação, com modificações aqui. No assistente de execução da EC2 instância, escolha uma Amazon Linux AMI.
3. Na página Configure Instance Details, selecione myPuppetinstanceprofile, a função que você criou em Etapa 1: criar um perfil do IAM para usar como o perfil de instância, como a função do IAM.
4. Na área Details, faça upload do script `userdata.sh` que você criou na Etapa 1.
5. Nenhuma alteração é necessária na página Add Storage. Vá para Add Tags.

   Ao aplicar tags à sua EC2 instância, você pode personalizar o comportamento do`userdata.sh`. Neste exemplo, aplique a função `apache_webserver` ao seu nó adicionando a seguinte tag: **`pp_role`** com o valor **`apache_webserver`**.

   A configuração do valor `pp_role` no nó define os valores de dados que são permanentemente armazenados no certificado do agente do nó, permitindo a classificação confiável do nó. Para obter mais informações, consulte Solicitações de extensão (dados certificados permanentes) na documentação da plataforma do Puppet.
6. Na página Configurar grupo de segurança, selecione Adicionar regra e selecione o tipo HTTP para abrir a porta 8080 para o servidor web Apache neste exemplo.
7. Escolha Review and Launch e, em seguida, selecione Launch. Quando o novo nó é iniciado, ele aplica a configuração do Apache do módulo de amostra que você configurou em Configurar o exemplo do Starter Kit do Apache.

8. Quando você abrir a página da web vinculada ao DNS público do seu novo nó, deverá ver um site hospedado pelo servidor da web Apache gerenciado pelo Puppet.

Desassociar um Node de um OpsWorks for Puppet Enterprise Server

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como desassociar ou remover um nó gerenciado do gerenciamento por um servidor OpsWorks for Puppet Enterprise. Essa operação é executada na linha de comando ou no console do Puppet Enterprise; você não pode desassociar nós no console de gerenciamento do OpsWorks Puppet Enterprise. Atualmente, a API OpsWorks for Puppet Enterprise não permite a remoção em lote de vários nós. O comando nesta seção dissociará um nó por vez.

Recomendamos que você desassocie os nós de um servidor Puppet master antes de excluir o servidor, de forma que os nós continuem a operar sem tentar se reconectar ao servidor. Para fazer isso, execute o `disassociate-node` AWS CLI comando. Para remover completamente um nó do PE, você deve desassociar o nó e revogar seu certificado, para que o nó não tente continuamente verificar com o Puppet master. Você também deve desinstalar a `puppet-agent` de nós quando não desejar mais gerenciá-los usando o Puppet principal.

Como desassociar nós

1. No AWS CLI, execute o comando a seguir para dissociar os nós. *`Node_name`*é o nome do nó que você deseja desassociar; para EC2 instâncias da Amazon, esse é o ID da instância. *`Server_name`*é o nome do Puppet master do qual você deseja desassociar o nó. Ambos os parâmetros são necessários. O parâmetro `--region` não será necessário, a menos que você queira desassociar um nó de um Puppet master que não esteja na região padrão.

```
aws opsworks-cm --region Region_name disassociate-node --node-name Node_name --
server-name Server_name
```

O comando a seguir é um exemplo.

```
aws opsworks-cm --region us-west-2 disassociate-node --node-name
 i-0010zzz00d66zzz90 --server-name opsworkstest
```

2. Aguarde pela mensagem de resposta indicando que a dissociação foi concluída.

Para obter mais informações sobre como excluir um OpsWorks servidor do Puppet Enterprise, consulteExcluir um OpsWorks para o Puppet Enterprise Server.

Consulte também

- Remover nós na documentação do Puppet Enterprise

Excluir um OpsWorks para o Puppet Enterprise Server

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como excluir um servidor OpsWorks for Puppet Enterprise. A exclusão de um servidor também exclui seus eventos, logs e quaisquer módulos que foram armazenados no servidor. Os recursos de suporte (instância do Amazon Elastic Compute Cloud, volume do Amazon Elastic Block Store etc.) também são excluídos, junto com todos os backups automatizados.

Embora a exclusão de um servidor não exclua os nós, eles não serão mais gerenciados pelo servidor excluído e tentarão continuamente a se reconectar. Por esse motivo, recomendamos que você desassocie os nós gerenciados antes de excluir um Puppet master. Nesta versão, você pode dissociar os nós executando um AWS CLI comando.

Etapa 1: Dissociação dos nós gerenciados

Desassocie os nós do Puppet master antes de excluir o servidor, de forma que os nós continuem a operar sem tentar a reconexão com o servidor. Para fazer isso, execute o `disassociate-node` AWS CLI comando.

Como desassociar nós

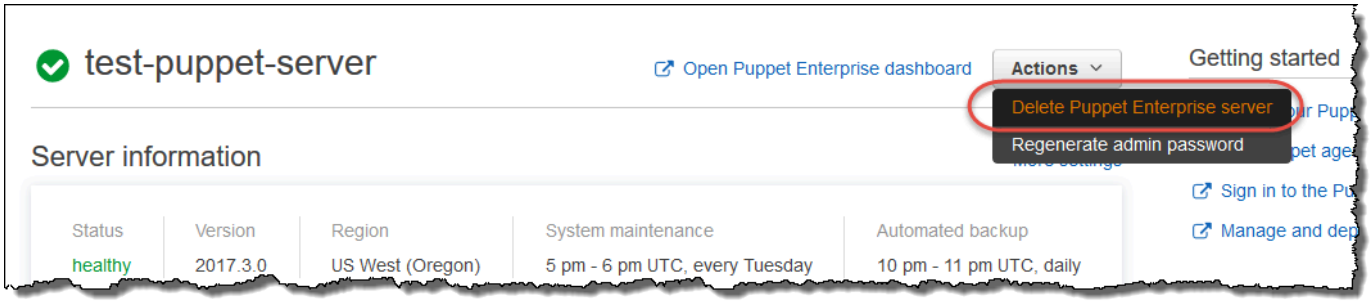
1. No AWS CLI, execute o comando a seguir para dissociar os nós. *`Server_name`*é o nome do Puppet master do qual você deseja desassociar o nó. O valor de `--node-name` pode ser um ID da instância.

```
aws opsworks-cm --region Region_name disassociate-node --node-name Node_name --
server-name Server_name
```

2. Aguarde pela mensagem de resposta indicando que a dissociação foi concluída.

Etapa 2: Exclusão do servidor

1. No bloco do servidor no painel, expanda o menu Actions.



2. Escolha Delete Puppet Enterprise server.
3. Quando você for solicitado a confirmar a exclusão, preencha a caixa de seleção para excluir funções e recursos associados e, em seguida, selecione Yes, Delete.

Consulte também

- Desassociar um Node de um OpsWorks for Puppet Enterprise Server

Como migrar um servidor OpsWorks for Puppet Enterprise para o Amazon Elastic Compute Cloud (Amazon) EC2

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As instruções abaixo descrevem como migrar os servidores existentes do Puppet Enterprise para a Amazon EC2, caso você queira continuar usando o Puppet Enterprise para suas necessidades de gerenciamento de configuração fora da. OpsWorks

Tópicos

- Etapa 1: entre em contato com o Puppet para comprar uma licença
- Etapa 2: Obtenha detalhes sobre seu servidor OpsWorks for Puppet Enterprise
- Etapa 3: Faça um backup do seu servidor OpsWorks para o Puppet Enterprise
- Etapa 4: iniciar uma nova EC2 instância
- Etapa 5: instalar o Puppet Enterprise na nova instância EC2
- Etapa 6: restaurar o backup na nova EC2 instância
- Etapa 7: configurar sua licença do Puppet
- Etapa 8: migrar seus nós
- Etapa 9: Exclua seu servidor OpsWorks for Puppet Enterprise

Etapa 1: entre em contato com o Puppet para comprar uma licença

Quando você migra seus servidores para EC2, a nova instância não vem com uma licença do Puppet. Para comprar uma chave de licença, siga as instruções no site do Puppet.

Etapa 2: Obtenha detalhes sobre seu servidor OpsWorks for Puppet Enterprise

Encontre e salve os valores do seu servidor OpsWorks para o Puppet Enterprise.

1. Faça login no AWS Management Console e abra o console do Amazon S3 em. https://console.aws.amazon.com/s3/

   Copie o nome do bucket Amazon S3 existente para o seu servidor OpsWorks para o Puppet Enterprise. O nome do bucket tem o formato: `aws-opsworks-cm-`*`server-name`*`-`*`random-string`*

2. Execute o aws opsworks-cm describe-servers comando para obter a configuração do seu servidor OpsWorks para o Puppet Enterprise.

   ```
   aws opsworks-cm describe-servers \
      --server-name server-name \
      --region region
   ```

   Armazene os valores para `InstanceType`, `KeyPair`, `SubnetIds`,`SecurityGroupIds`, `InstanceProfileArn`, e `Endpoint` da resposta.

3. Use SSH para se conectar ao servidor existente OpsWorks do Puppet Enterprise. Você pode usar o Gerenciador de Sessões no EC2 console em vez do SSH.

   Execute o seguinte comando:

   ```
   rpm -qa | grep opsworks-cm-puppet-enterprise | cut -d '-' -f 5
   ```

   A resposta fornece a versão do Puppet Enterprise (por exemplo, 2019.8.10). Armazene esse valor.

Você usará o SSH ou o Gerenciador de sessões na próxima etapa.

Etapa 3: Faça um backup do seu servidor OpsWorks para o Puppet Enterprise

1. Execute os comandos a seguir para fazer um backup local.

```
mkdir /tmp/puppet-backup/
sudo /opt/puppetlabs/bin/puppet-backup create --dir=/tmp/puppet-backup/
```

2. Execute o comando a seguir para armazenar o nome do backup.

```
ls /tmp/puppet-backup/
PUPPET_BACKUP=$(ls /tmp/puppet-backup/)
```

3. Execute o comando a seguir para fazer upload do seu backup em um bucket S3. *S3-Bucket*Substitua pelo valor da etapa 1 emEtapa 2: Obtenha detalhes sobre seu servidor OpsWorks for Puppet Enterprise.

```
aws s3 cp /tmp/puppet-backup/PUPPET_BACKUP s3://S3_Bucket/tmp/puppet-backup/
```

   Armazene os valores `PUPPET_BACKUP` e `S3_BUCKET`. Você importará esses valores para a nova EC2 instância.

Você pode sair da sessão SSH ou do Gerenciador de sessões.

Etapa 4: iniciar uma nova EC2 instância

Execute uma nova EC2 instância a partir do EC2 console https://console.aws.amazon.com/ec2/usando a mesma configuração do servidor do OpsWorks Puppet Enterprise.

Nome do parâmetro	Valor
SO	Amazon Linux 2
Tipo de instância	O valor `InstanceType` da etapa 2 do Etapa 2: Obtenha detalhes sobre seu servidor OpsWorks for Puppet Enterprise.
Key pair name (Nome do par de chaves)	O valor `KeyPair` da etapa 2 do Etapa 2: Obtenha detalhes sobre seu servidor OpsWorks for Puppet Enterprise.
VPC	O VPC da etapa 2 `SubnetIds` do Etapa 2: Obtenha detalhes sobre seu servidor OpsWorks for Puppet Enterprise.

Nome do parâmetro	Valor
Sub-rede	O `SubnetIds` da etapa 2 do Etapa 2: Obtenha detalhes sobre seu servidor OpsWorks for Puppet Enterprise.
Selecione um grupo de segurança existente -> Grupos de segurança comuns	O `SecurityGroupIds` da etapa 2 do Etapa 2: Obtenha detalhes sobre seu servidor OpsWorks for Puppet Enterprise.
Armazenamento	Pelo menos 120 GB.
Perfil de instância do IAM	O `InstanceProfileArn` da etapa 2 do Etapa 2: Obtenha detalhes sobre seu servidor OpsWorks for Puppet Enterprise.

Se você quiser criar e anexar um IP elástico à nova instância, copie o ID da instância da nova instância e conclua as etapas em (Opcional) Etapa 4.1: criar e anexar um IP elástico.

(Opcional) Etapa 4.1: criar e anexar um IP elástico

Com um endereço IP elástico, é possível mascarar a falha de uma instância ou software remapeando rapidamente o endereço para outra instância na conta.

Como criar e associar um endereço IP elástico

1. Faça login no AWS Management Console e abra o EC2 console da Amazon em https://console.aws.amazon.com/ec2/.
2. Escolha Elastic IPs.
3. Escolha Alocar endereço IP elástico.
4. Na página Alocar endereço IP elástico, escolha Alocar. Isso cria um IPv4 endereço público.
5. Copie o IPv4 endereço alocado.
6. Em Ações, escolha Associar endereços IP elásticos.
7. Em Instância, insira o ID da instância nova.
8. Selecione Associar .

Etapa 5: instalar o Puppet Enterprise na nova instância EC2

Use SSH para se conectar à nova EC2 instância. Você pode usar o Gerenciador de Sessões no EC2 console em vez do SSH.

```
# switch to sudo user
sudo -i

# Setup environment variables
PUPPET_ENTERPRISE_VERSION=Puppet Enterprise version from step 2.3
hostname Public IPv4 DNS or Custom Domain if available

# Install Puppet Enterprise
curl -JLO https://pm.puppetlabs.com/puppet-enterprise/$PUPPET_ENTERPRISE_VERSION/
puppet-enterprise-$PUPPET_ENTERPRISE_VERSION-el-7-x86_64.tar.gz
tar -xf puppet-enterprise-$PUPPET_ENTERPRISE_VERSION-el-7-x86_64.tar.gz

./puppet-enterprise-$PUPPET_ENTERPRISE_VERSION-el-7-x86_64/puppet-enterprise-installer
```

Você pode manter sua sessão SSH ou Gerenciador de sessão abertos para a próxima etapa.

Etapa 6: restaurar o backup na nova EC2 instância

```
# Setup environment variables
S3_BUCKET=S3 bucket name from step 2.1
PUPPET_BACKUP=Puppet backup file name from step 3.2

# download backup
aws s3 cp s3://$S3_BUCKET/tmp/puppet-backup/$PUPPET_BACKUP

# Prepare Puppet Enterprise backup to remove OpsWorks metadata
mkdir output
tar -xf $PUPPET_BACKUP -C output/
cd output/
rm -f opt/puppetlabs/facter/facts.d/opsworks.json
tar -cf ../$PUPPET_BACKUP *
cd ..
rm -rf output/

# Restore from backup
PATH=$PATH:/opt/puppetlabs/puppet/bin/
puppet-backup restore $PUPPET_BACKUP
```

```
puppet agent -t
```

Você pode acessar o console do Puppet para a EC2 instância restaurada em https://*Public IPv4 of the instance*. Você pode encontrar o IPv4 DNS público na página de detalhes da instância no EC2 console. As credenciais de login são as mesmas que você usa para acessar seu OpsWorks servidor do Puppet Enterprise.

Você pode manter sua sessão SSH ou Gerenciador de sessão abertos para a próxima etapa.

Etapa 7: configurar sua licença do Puppet

Siga as etapas no site do Puppet para configurar sua licença.

Você pode manter sua sessão SSH ou Gerenciador de sessão abertos para a próxima etapa.

Etapa 8: migrar seus nós

Há dois tipos de domínios suportados pelos servidores do OpsWorks Puppet Enterprise:

- BYODC (traga seu próprio domínio e certificado)
- OpsWorks endpoint

Etapa 8.1: para BYODC (traga seu próprio domínio e certificado)

Para esses nós, tudo o que você precisa fazer é apontar o domínio personalizado em seu provedor de DNS para o IPv4 DNS público ou IPv4 endereço público da nova EC2 instância.

Etapa 8.2: Para endpoint OpsWorks

Para um OpsWorks endpoint, a documentação do Puppet recomenda desinstalar o agente Puppet no nó e depois instalar o agente Puppet usando o servidor Puppet Enterprise recém-restaurado.

> **Note**
>
> Embora o Puppet não tenha um procedimento automatizado para mover um nó de agente, há alguns módulos que os membros da comunidade do Puppet publicaram no site do Puppet Forge para realizar a migração automatizada de nós. Esses módulos incluem o módulo `pe_migrate` e um segundo módulo de migração de um autor diferente. Os módulos no

site do Puppet Forge não são suportados pelo Puppet ou a OpsWorks menos que estejam explicitamente indicados no módulo Forge. Recomendamos ter cuidado com esses módulos e testá-los antes de usá-los amplamente.

As seções a seguir fornecem as etapas para desinstalar e reinstalar agentes do Puppet em instâncias Linux.

Tópicos

- Etapa 8.2.1: copie o desinstalador do servidor Puppet
- Etapa 8.2.2: baixe o desinstalador e execute-o em um nó
- Etapa 8.2.3: reinstalar o agente Puppet em um nó

Etapa 8.2.1: copie o desinstalador do servidor Puppet

Antes de desinstalar o agente, certifique-se de que o perfil da instância do IAM do nó forneça ReadOnly permissões do S3.

Execute o comando a seguir para copiar o desinstalador do servidor do Puppet para o bucket do S3.

```
aws s3 cp \
   /opt/puppetlabs/bin/puppet-enterprise-uninstaller \
   s3://$S3_BUCKET/tmp/puppet-enterprise-uninstaller
```

Depois de executar o comando, você pode sair da sessão SSH ou do Gerenciador de sessões do servidor Puppet.

Etapa 8.2.2: baixe o desinstalador e execute-o em um nó

Use o SSH para conectar ao nó. Você pode usar o Gerenciador de Sessões no EC2 console em vez do SSH se o nó for uma EC2 instância.

```
sudo -i

S3_BUCKET=aws-opsworks-cm-abcdefg-uuhtyn6messn
aws s3 cp s3://$S3_BUCKET/tmp/puppet-enterprise-uninstaller /opt/puppetlabs/bin/
chmod 700 /opt/puppetlabs/bin/puppet-enterprise-uninstaller
/opt/puppetlabs/bin/puppet-enterprise-uninstaller
```

Você pode manter sua sessão SSH ou Gerenciador de sessão abertos para a próxima etapa.

Etapa 8.2.3: reinstalar o agente Puppet em um nó

Conclua as etapas a seguir para reinstalar o agente do Puppet em um nó.

Tópicos

- Etapa 8.2.3.1: instalar o agente Puppet com a configuração correta
- Etapa 8.2.3.2: aceitar o certificado no console do Puppet
- Etapa 8.2.3.3: verifique o nó no servidor Puppet Enterprise

Etapa 8.2.3.1: instalar o agente Puppet com a configuração correta

Execute o seguinte comando para instalar o agente do Puppet.

```
curl -k https://Public_IPv4_DNS:8140/packages/current/install.bash | bash
```

Você pode manter sua sessão SSH ou Gerenciador de sessão abertos para a etapa 8.2.2.3.

Etapa 8.2.3.2: aceitar o certificado no console do Puppet

1. Acesse o console do servidor Puppet em `https://`*`Public_IPv4_DNS`*.
2. Escolha Certificados e, em seguida, Certificados não assinados.
3. Escolha Aceitar para assinar o certificado do agente do Puppet.

Etapa 8.2.3.3: verifique o nó no servidor Puppet Enterprise

Execute o comando a seguir no nó para verificá-lo no servidor.

```
puppet agent -t
```

O nó agora deve estar visível no console do servidor Puppet.

Etapa 9: Exclua seu servidor OpsWorks for Puppet Enterprise

Você pode usar o OpsWorks console ou AWS CLI excluir seu OpsWorks servidor do Puppet Enterprise.

Para excluir seu servidor usando o OpsWorks console

1. Faça login no AWS Management Console e abra o AWS OpsWorks console em https://console.aws.amazon.com/opsworks/.
2. No painel de navegação, escolha Servidores Puppet Enterprise.
3. Na página Servidores Puppet Enterprise, selecione o servidor que você deseja excluir.
4. Em Ações, selecione Excluir servidor Puppet Enterprise.

Para excluir seu servidor usando o AWS CLI

Execute o seguinte comando:

```
aws opsworks-cm delete-server \
   --server-name server-name \
   --region region
```

Registro OpsWorks de chamadas de API do Puppet Enterprise com AWS CloudTrail

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

OpsWorks for Puppet Enterprise é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço no OpsWorks Puppet Enterprise. CloudTrail captura todas as chamadas de API OpsWorks para o Puppet Enterprise como eventos, incluindo chamadas do console do OpsWorks Puppet Enterprise e de chamadas de código OpsWorks para o Puppet Enterprise. APIs Se você criar uma trilha, poderá habilitar a entrega contínua de CloudTrail eventos para um bucket do Amazon S3, incluindo eventos OpsWorks para o Puppet Enterprise. Se você não configurar uma trilha, ainda poderá ver os eventos mais recentes no CloudTrail console no Histórico de eventos. Usando as informações coletadas por CloudTrail, você

pode determinar a solicitação que foi feita OpsWorks para o Puppet Enterprise, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais sobre isso CloudTrail, consulte o Guia AWS CloudTrail do usuário.

OpsWorks para obter informações sobre a Puppet Enterprise em CloudTrail

CloudTrail é ativado em sua AWS conta quando você cria a conta. Quando a atividade ocorre no OpsWorks Puppet Enterprise, essa atividade é registrada em um CloudTrail evento junto com outros eventos de AWS serviço no histórico de eventos. Você pode visualizar, pesquisar e baixar eventos recentes em sua AWS conta. Para obter mais informações, consulte Visualização de eventos com histórico de CloudTrail eventos.

Para um registro contínuo dos eventos em sua AWS conta, incluindo eventos OpsWorks para o Puppet Enterprise, crie uma trilha. Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Por padrão, ao criar uma trilha no console, ela é aplicada a todas as regiões da . A trilha registra eventos de todas as regiões na AWS partição e entrega os arquivos de log ao bucket do Amazon S3 que você especificar. Além disso, você pode configurar outros AWS serviços para analisar e agir com base nos dados de eventos coletados nos CloudTrail registros. Para obter mais informações, consulte:

- Visão geral da criação de uma trilha
- CloudTrail Serviços e integrações compatíveis
- Configurando notificações do Amazon SNS para CloudTrail
- Recebendo arquivos de CloudTrail log de várias regiões e recebendo arquivos de CloudTrail log de várias contas

Todas OpsWorks as ações do Puppet Enterprise são registradas CloudTrail e documentadas na Referência da API do OpsWorks Puppet Enterprise. Por exemplo, chamadas para as DescribeServersações CreateServerCreateBackup, e geram entradas nos arquivos de CloudTrail log.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário-raiz ou usuário do IAM.
- Se a solicitação foi feita com credenciais de segurança temporárias de uma função ou de um usuário federado.

- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte Elemento userIdentity do CloudTrail .

Compreendendo OpsWorks as entradas do arquivo de log do Puppet Enterprise

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log para um bucket do Amazon S3 que você especificar. CloudTrail os arquivos de log contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros da solicitação e assim por diante. CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas de API, portanto, eles não aparecem em nenhuma ordem específica.

O exemplo a seguir mostra uma entrada de CloudTrail registro para a `CreateServer` ação OpsWorks for Puppet Enterprise.

```
{"eventVersion":"1.05",
"userIdentity":{
    "type":"AssumedRole",
    "principalId":"ID number:OpsWorksCMUser",
    "arn":"arn:aws:sts::831000000000:assumed-role/Admin/OpsWorksCMUser",
    "accountId":"831000000000","accessKeyId":"ID number",
    "sessionContext":{
        "attributes":{
            "mfaAuthenticated":"false",
            "creationDate":"2017-01-05T22:03:47Z"
            },
        "sessionIssuer":{
            "type":"Role",
            "principalId":"ID number",
            "arn":"arn:aws:iam::831000000000:role/Admin",
            "accountId":"831000000000",
            "userName":"Admin"
            }
        }
    },
"eventTime":"2017-01-05T22:18:23Z",
"eventSource":"opsworks-cm.amazonaws.com",
"eventName":"CreateServer",
"awsRegion":"us-west-2",
```

```
"sourceIPAddress":"101.25.190.51",
"userAgent":"console.amazonaws.com",
"requestParameters":{
    "serverName":"test-puppet-server",
    "engineModel":"Single",
    "engine":"Puppet",
    "instanceProfileArn":"arn:aws:iam::831000000000:instance-profile/aws-opsworks-cm-
ec2-role",
    "backupRetentionCount":3,"serviceRoleArn":"arn:aws:iam::831000000000:role/service-
role/aws-opsworks-cm-service-role",
    "engineVersion":"12",
    "preferredMaintenanceWindow":"Fri:21:00",
    "instanceType":"t2.medium",
    "subnetIds":["subnet-1e111f11"],
    "preferredBackupWindow":"Wed:08:00"
    },
"responseElements":{
    "server":{
        "endpoint":"test-puppet-server-xxxx8u4390xo6pd9.us-west-2.opsworks-cm.io",
        "createdAt":"Jan 5, 2017 10:18:22 PM",
        "serviceRoleArn":"arn:aws:iam::831000000000:role/service-role/aws-opsworks-cm-
service-role",
        "preferredBackupWindow":"Wed:08:00",
        "status":"CREATING",
        "subnetIds":["subnet-1e111f11"],
        "engine":"Puppet",
        "instanceType":"t2.medium",
        "serverName":"test-puppet-server",
        "serverArn":"arn:aws:opsworks-cm:us-west-2:831000000000:server/test-puppet-
server/8ezz7f6z-e91f-4z10-89z5-8c6219zzz09f",
        "engineModel":"Single",
        "backupRetentionCount":3,
        "engineAttributes":[
            {"name":"PUPPET_ADMIN_PASSWORD","value":"*** Redacted ***"},
            {"name":"PUPPET_API_CA_CERT","value":"*** Redacted ***"},
            ],
        "engineVersion":"12.11.1",
        "instanceProfileArn":"arn:aws:iam::831000000000:instance-profile/aws-opsworks-
cm-ec2-role",
        "preferredMaintenanceWindow":"Fri:21:00"
        }
    },
"requestID":"de7z64z9-d394-12ug-8081-7zz0386fbcb6",
"eventID":"8z7z18dz-6z90-47bz-87cf-e8346428zzz3",
```

```
"eventType":"AwsApiCall",
"recipientAccountId":"831000000000"
}
```

Solução de problemas OpsWorks do Puppet Enterprise

> **Important**
>
> O AWS OpsWorks for Puppet Enterprise serviço chegou ao fim da vida útil em 31 de março de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico contém alguns problemas comuns OpsWorks do Puppet Enterprise e sugestões de soluções para esses problemas.

Tópicos

- Dicas para solução de problemas em geral
- Solução de problemas de erros específicos
- Ajuda e suporte adicionais

Dicas para solução de problemas em geral

Caso não consiga criar ou trabalhar com um Puppet master, você pode visualizar mensagens de erro ou logs para ajudar a solucionar o problema. As tarefas a seguir descrevem locais em geral para iniciar quando você está solucionando um problema no Puppet master. Para obter mais informações sobre erros específicos e soluções, consulte a seção Solução de problemas de erros específicos deste tópico.

- Use o console OpsWorks for Puppet Enterprise para visualizar mensagens de erro se um Puppet master falhar ao iniciar. Na página de propriedades do Puppet master, as mensagens de erro relacionadas à inicialização e à execução do servidor são mostradas na parte superior da página. Os erros podem vir dos OpsWorks serviços do Puppet Enterprise ou da Amazon EC2 que são usados para criar um Puppet master. AWS CloudFormation Na página de propriedades, você

também pode visualizar eventos que ocorrem em um servidor em execução, que podem conter mensagens de evento de falha.

- Para ajudar a resolver EC2 problemas, conecte-se à instância do seu servidor usando SSH e visualize os registros. EC2 os registros da instância são armazenados no `/var/log/aws/opsworks-cm` diretório. Esses registros capturam as saídas do comando, enquanto o OpsWorks Puppet Enterprise inicia um Puppet master.

Solução de problemas de erros específicos

Tópicos

- O servidor está em um estado de conexão perdida
- Falha na criação do servidor com a mensagem "requested configuration is currently not supported"
- Não é possível criar a EC2 instância Amazon do servidor
- Erro na função de serviço impede a criação do servidor
- Limite de endereço IP elástico excedido
- Falha na associação do nó autônomo
- Falha na manutenção do sistema

O servidor está em um estado de conexão perdida

Problema: o status de um servidor é exibido como Conexão perdida.

Causa: Isso geralmente ocorre quando uma entidade externa AWS OpsWorks faz alterações em um OpsWorks servidor do Puppet Enterprise ou em seus recursos de suporte. AWS OpsWorks não pode se conectar aos servidores do Puppet Enterprise em estados perdidos de conexão para lidar com tarefas de manutenção, como criar backups, aplicar patches do sistema operacional ou atualizar o Puppet. Como resultado, seu servidor pode estar perdendo atualizações importantes, suscetível a problemas de segurança ou não funcionando conforme o esperado.

Solução: tente as etapas a seguir para restaurar a conexão do servidor.

1. Verifique se seu perfil de serviço tem todas as permissões necessárias.

    a. Na página Configurações do seu servidor, em Rede e segurança, escolha o link para o perfil de serviço que o servidor está usando. Isso abrirá o perfil de serviço para visualização no console do IAM.

   b. Na guia Permissões, verifique se `AWSOpsWorksCMServiceRole` está na lista de Políticas de permissões. Se ela não estiver listada, adicione a política `AWSOpsWorksCMServiceRole` gerenciada manualmente ao perfil.

   c. Na guia Relações de confiança, verifique se o perfil de serviço tem uma política de confiança que confia no serviço `opsworks-cm.amazonaws.com` para assumir perfis em seu nome. Para obter mais informações sobre como usar políticas de confiança com funções, consulte Modificação de uma função (console) ou a postagem do blog de AWS segurança, Como usar políticas de confiança com funções do IAM.

2. Verifique se seu perfil de instância tem todas as permissões necessárias.

   a. Na página Configurações do seu servidor, em Rede e segurança, escolha o link para o perfil de instância que o servidor está usando. Isso abrirá o perfil de instância para visualização no console do IAM.

   b. Na guia Permissões, verifique se `AmazonEC2RoleforSSM` e `AWSOpsWorksCMInstanceProfileRole` estão na lista de Políticas de permissões. Se uma ou ambas não estiverem listadas, adicione essas políticas gerenciadas manualmente ao perfil.

   c. Na guia Relações de confiança, verifique se o perfil de serviço tem uma política de confiança que confia no serviço `ec2.amazonaws.com` para assumir perfis em seu nome. Para obter mais informações sobre como usar políticas de confiança com funções, consulte Modificação de uma função (console) ou a postagem do blog de AWS segurança, Como usar políticas de confiança com funções do IAM.

3. No EC2 console da Amazon, verifique se você está na mesma região do servidor do Puppet Enterprise e, em seguida, reinicie a EC2 instância que seu servidor está usando. OpsWorks

   a. Escolha a EC2 instância nomeada `aws-opsworks-cm-instance-`*`server-name`*.

   b. Escolha Estado da instância e Reinicializar instância no menu.

   c. Aguarde até 15 minutos para que seu servidor reinicie e fique totalmente on-line.

4. No console do OpsWorks Puppet Enterprise, na página de detalhes do servidor, verifique se o status do servidor agora está íntegro.

Se o status do servidor ainda for Conexão perdida após a execução das etapas anteriores, tente uma das opções a seguir.

- Substitua o servidor criando um novo e excluindo o original. Se os dados do servidor atual forem importantes para você, restaure o servidor a partir de um backup recente e verifique se os dados estão atualizados antes de excluir o servidor original que não responde.
- Entre em contato com o suporte da AWS.

Falha na criação do servidor com a mensagem "requested configuration is currently not supported"

Problema: você está tentando criar um servidor Puppet Enterprise, mas a criação do servidor falha com uma mensagem de erro semelhante a "A configuração solicitada não é suportada atualmente. Please check the documentation for supported configurations."

Causa: um tipo de instância não compatível pode ter sido especificado para o Puppet master. Se você optar por criar o servidor Puppet em uma VPC que tenha uma locação não padrão, como uma para instâncias dedicadas, todas as instâncias dentro da VPC especificada também devem ser de locação dedicada ou host. Como alguns tipos de instância, como t2, só estão disponíveis com a locação padrão, o tipo de instância do servidor Puppet master talvez não seja compatível com a VPC especificada, e a criação do servidor falha.

Solução: Se você escolher uma VPC que tenha uma locação não padrão, use um tipo de instância m4, que pode dar suporte à locação dedicada.

Não é possível criar a EC2 instância Amazon do servidor

Problema: a criação do servidor falhou com uma mensagem de erro semelhante à seguinte: “Falha na criação dos seguintes recursos: [EC2Instância]. Failed to receive 1 resource signal(s) within the specified duration."

Causa: Isso provavelmente ocorre porque a EC2 instância não tem acesso à rede.

Solução: certifique-se de que a instância tenha acesso de saída à Internet e que o agente AWS de serviço seja capaz de emitir comandos. Certifique-se de que a VPC (uma VPC com uma única sub-rede pública) tenha DNS resolution ativado e que a sub-rede tenha a configuração Auto-assign Public IP ativada.

Erro na função de serviço impede a criação do servidor

Problema: A criação do servidor falha com uma mensagem de erro que diz: “Não autorizado a realizar sts:”AssumeRole.

Causa: Isso pode ocorrer quando a função de serviço que você está usando não tem permissões adequadas para criar um novo servidor.

Solução: Abra o OpsWorks console do Puppet Enterprise; use o console para gerar uma nova função de serviço e uma função de perfil de instância. Se você preferir usar sua própria função de serviço, anexe a AWSOps política de CMServiceFunção de Trabalho à função. Verifique se opsworks-cm.amazonaws.com está listado entre os serviços nas Relações de confiança do perfil. Verifique se a função de serviço associada ao Puppet master tem a política gerenciada do AWSOpsWorks CMService Role anexada.

Limite de endereço IP elástico excedido

Problema: a criação do servidor falha com uma mensagem de erro que diz: “Falha na criação dos seguintes recursos: [EIP, EC2 Instância]. Resource creation cancelled, the maximum number of addresses has been reached."

Causa: Isso ocorre quando a conta usou o número máximo de endereços Elastic IP (EIP – IP elástico). O limite de endereços EIP padrão é cinco.

Solução: você pode liberar endereços EIP existentes ou excluir aqueles que sua conta não está usando ativamente, ou pode entrar em contato com o Suporte AWS ao Cliente para aumentar o limite de endereços EIP associados à sua conta.

Falha na associação do nó autônomo

Problema: a associação autônoma ou automática de novos EC2 nós da Amazon está falhando. Os nós que deveriam ter sido adicionados ao Puppet master não aparecem no painel do Puppet Enterprise.

Causa: isso pode ocorrer quando você não tem uma função do IAM configurada como um perfil de instância que permita que as chamadas de `opsworks-cm` API se comuniquem com novas EC2 instâncias.

Solução: anexe uma política ao seu perfil de EC2 instância que permita que as chamadas de `DescribeNodeAssociationStatus` API `AssociateNode` e as chamadas de API funcionem EC2, conforme descrito emAdicionando nós automaticamente OpsWorks para o Puppet Enterprise.

Falha na manutenção do sistema

AWS OpsWorks CM realiza manutenção semanal do sistema para garantir que as últimas versões AWS testadas do Puppet Server, incluindo atualizações de segurança, estejam sempre em execução

em um servidor OpsWorks para Puppet Enterprise. Se, por algum motivo, a manutenção do sistema falhar, AWS OpsWorks CM notifica você sobre a falha. Para obter mais informações sobre manutenção do sistema, consulte Manutenção do sistema OpsWorks para Puppet Enterprise.

Esta seção descreve os possíveis motivos da falha e sugere soluções.

Tópicos

- Erro no perfil de serviço ou no perfil de instância impede a manutenção do sistema

Erro no perfil de serviço ou no perfil de instância impede a manutenção do sistema

Problema: A manutenção do sistema falha com uma mensagem de erro que diz: “Não autorizado a executar sts: AssumeRole “ou uma mensagem de erro semelhante sobre permissões.

Causa: isso pode ocorrer quando o perfil de serviço ou o perfil de instância que você está usando não têm permissões adequadas para realizar a manutenção do sistema no servidor.

Solução: verifique se seu perfil de serviço e perfil de instância têm todas as permissões necessárias.

1. Verifique se seu perfil de serviço tem todas as permissões necessárias.
   a. Na página Configurações do seu servidor, em Rede e segurança, escolha o link para o perfil de serviço que o servidor está usando. Isso abrirá o perfil de serviço para visualização no console do IAM.
   b. Na guia Permissões, verifique se `AWSOpsWorksCMServiceRole` está anexado ao perfil de serviço. Se `AWSOpsWorksCMServiceRole` não estiver listada, adicione essa política ao perfil.
   c. Verifique se opsworks-cm.amazonaws.com está listado entre os serviços nas Relações de confiança do perfil. Para obter mais informações sobre como usar políticas de confiança com funções, consulte Modificação de uma função (console) ou a postagem do blog de AWS segurança, Como usar políticas de confiança com funções do IAM.
2. Verifique se seu perfil de instância tem todas as permissões necessárias.
   a. Na página Configurações do seu servidor, em Rede e segurança, escolha o link para o perfil de instância que o servidor está usando. Isso abrirá o perfil de instância para visualização no console do IAM.
   b. Na guia Permissões, verifique se `AmazonEC2RoleforSSM` e `AWSOpsWorksCMInstanceProfileRole` estão na lista de Políticas de permissões. Se

uma ou ambas não estiverem listadas, adicione essas políticas gerenciadas manualmente ao perfil.

c. Na guia Relações de confiança, verifique se o perfil de serviço tem uma política de confiança que confia no serviço `ec2.amazonaws.com` para assumir perfis em seu nome. Para obter mais informações sobre como usar políticas de confiança com funções, consulte Modificação de uma função (console) ou a postagem do blog de AWS segurança, Como usar políticas de confiança com funções do IAM.

Ajuda e suporte adicionais

Caso você não veja o problema específico descrito neste tópico ou tenha tentado as sugestões neste tópico e ainda esteja enfrentando problemas, visite os fóruns do AWS OpsWorks.

Você também pode acessar o AWS Support Center. O AWS Support Center é o hub para criar e gerenciar casos de AWS suporte. O AWS Support Center também inclui links para outros recursos úteis, como fóruns, informações técnicas FAQs, status de integridade do serviço AWS Trusted Advisor e.

AWS OpsWorks para Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks for Chef Automate permite que você execute um servidor Chef Automate na AWS. Você pode provisionar um servidor Chef em minutos e deixar AWS OpsWorks for Chef Automate lidar com suas operações, backups, restaurações e atualizações de software. AWS OpsWorks for Chef Automate libera você para se concentrar nas principais tarefas de gerenciamento de configuração, em vez de gerenciar um servidor Chef.

Um servidor Chef Automate gerencia a configuração dos nós em seu ambiente instruindo `chef-client`quais receitas do Chef devem ser executadas nos nós, armazena informações sobre os nós e serve como um repositório central para seus livros de receitas do Chef. AWS OpsWorks for Chef Automate fornece servidores Chef que incluem recursos premium do Chef Automate: Chef Infra e Chef. InSpec

Um AWS OpsWorks for Chef Automate servidor é executado em uma instância do Amazon Elastic Compute Cloud. AWS OpsWorks for Chef Automate os servidores estão configurados para executar a versão mais recente do Amazon Linux (Amazon Linux 2). Para obter informações sobre o que foi alterado nesta versão do Chef Automate, consulte as Notas de versão do Chef Automate. A tabela a seguir descreve os componentes do Chef que estão instalados em um AWS OpsWorks for Chef Automate servidor.

Nome do componente	Descrição	Versão instalada no AWS OpsWorks for Chef Automate servidor
Chef Automate	O Chef Automate é um pacote de software de servidor empresarial que	2,0

Nome do componente	Descrição	Versão instalada no AWS OpsWorks for Chef Automate servidor
	oferece um fluxo de trabalho automatizado para implantação contínua e informações sobre nós gerenciados em um console de gerenciamento baseado na Web. O Chef Automate oferece automação de infraestrutura incluindo o Chef Infra, informações e fiscalização de segurança e conformidade, incluindo o Chef InSpec, e implantação automatizada, incluindo o Chef Habitat. Para obter mais informações sobre o Chef Automate, consulte Chef Automate no site do Chef.	
Chef Infra	Anteriormente chamado de Chef Server, o Chef Infra Server usa o agente (`chef-client`) do Chef Infra Client para aplicar continuamente configurações a nós gerenciados a fim de manter um estado desejado. Para obter mais informações sobre o Infra, consulte Chef Infra no site do Chef.	12.x

Nome do componente	Descrição	Versão instalada no AWS OpsWorks for Chef Automate servidor
Chef InSpec	O Chef InSpec descreve as regras de segurança e conformidade que podem ser compartilhadas entre engenheiros de software, operações e engenheiros de segurança. Conformidade, segurança e outros requisitos de política formam a estrutura de testes automatizados que o agente `chef-client` pode executar em nós gerenciados, garantindo a aplicação consistente de padrões. Para obter mais informações sobre InSpec, consulte Chef InSpec no site do Chef.	3.9.0

A versão mínima compatível do `chef-client` em nós associados a um servidor AWS OpsWorks for Chef Automate é 13.x. Recomendamos executar pelo menos 14.10.9 ou a versão estável `chef-client` mais atual.

Quando novas versões secundárias do software do Chef forem disponibilizadas, a manutenção do sistema é projetada para atualizar a versão secundária do Chef Automate e do Chef Server automaticamente no servidor, assim que ela passar nos testes da AWS. A AWS realiza testes extensivos para verificar se as atualizações do Chef estão prontas para produção e não interrompem os ambientes existentes dos clientes, portanto, pode haver atrasos entre os lançamentos do software Chef e sua disponibilidade para aplicação e os existentes nos servidores do Chef Automate. OpsWorks A manutenção do sistema também atualiza seu servidor para a versão mais recente do Amazon Linux.

Você pode conectar qualquer computador ou EC2 instância local que esteja executando um sistema operacional compatível e tenha acesso de rede a um AWS OpsWorks for Chef Automate servidor. Para consultar uma lista de sistemas operacionais compatíveis para nós que deseja gerenciar, acesse o site do Chef. O software do agente `chef-client` é instalado nos nós que você deseja gerenciar com um servidor Chef.

Tópicos

- Suporte regional para AWS OpsWorks for Chef Automate
- AWS OpsWorks para Chef Automate End of Life FAQs
- Atualize um AWS OpsWorks for Chef Automate servidor para o Chef Automate 2
- Começando com AWS OpsWorks for Chef Automate
- Crie um AWS OpsWorks for Chef Automate servidor usando AWS CloudFormation
- Atualizar um AWS OpsWorks for Chef Automate servidor para usar um domínio personalizado
- Regenere o kit inicial para um servidor AWS OpsWorks for Chef Automate
- Trabalhando com tags em AWS OpsWorks for Chef Automate recursos
- Fazer o backup e a restauração de um AWS OpsWorks for Chef Automate servidor
- Manutenção do sistema em AWS OpsWorks for Chef Automate
- Verificações de conformidade em AWS OpsWorks for Chef Automate
- Desassociar um nó de um servidor AWS OpsWorks for Chef Automate
- Excluir um AWS OpsWorks for Chef Automate servidor
- Redefinição das credenciais do painel do Chef Automate
- Registrando chamadas de AWS OpsWorks for Chef Automate API com AWS CloudTrail
- Solução de problemas AWS OpsWorks for Chef Automate

Suporte regional para AWS OpsWorks for Chef Automate

Os seguintes endpoints regionais oferecem suporte a AWS OpsWorks for Chef Automate servidores. AWS OpsWorks for Chef Automate cria recursos associados aos seus servidores Chef, como perfis de instância, usuários e funções de serviço, no mesmo endpoint regional do seu servidor Chef. O servidor do Chef deve estar em uma VPC. Você pode criar uma VPC, usar a que já tem ou optar pela VPC padrão.

- Região Leste dos EUA (Ohio)

- Região Leste dos EUA (Norte da Virgínia)
- Região Oeste dos EUA (Norte da Califórnia)
- Região Oeste dos EUA (Oregon)
- Região Ásia-Pacífico (Tóquio)
- Região Ásia-Pacífico (Singapura)
- Região Ásia-Pacífico (Sydney)
- Região Europa (Frankfurt)
- Região Europa (Irlanda)

AWS OpsWorks para Chef Automate End of Life FAQs

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa.

Tópicos

- Como os usuários existentes serão afetados por esse fim da vida útil?
- O que acontecerá aos meus servidores se eu não tomar nenhuma ação?
- Para quais alternativas posso fazer a transição?
- O serviço ainda está aceitando novos clientes?
- O fim da vida afetará tudo Regiões da AWS ao mesmo tempo?
- Qual nível de suporte técnico está disponível?
- Sou cliente atual do OpsWorks Chef Automate e preciso iniciar um servidor em uma conta que não estava usando o serviço anteriormente. Eu sou capaz de fazer isso?
- Haverá algum lançamento de recursos importantes no próximo ano?

Como os usuários existentes serão afetados por esse fim da vida útil?

Os clientes existentes não serão afetados até 5 de maio de 2024, a data de fim da vida útil do Chef OpsWorks Automate. Após a data de fim da vida útil, os clientes não poderão mais gerenciar seus servidores usando o OpsWorks console ou a API.

O que acontecerá aos meus servidores se eu não tomar nenhuma ação?

A partir de 5 de maio de 2024, você não poderá mais gerenciar seus servidores usando o OpsWorks console ou a API. Nesse momento, deixaremos de realizar quaisquer funções de gerenciamento contínuas de seus servidores, como backups ou manutenção. Para limitar o impacto para os clientes, deixaremos em execução todas as EC2 instâncias que estejam fazendo backup dos servidores do Chef Automate, mas suas licenças não serão mais válidas, pois o uso não é mais coberto (ou cobrado) pelo contrato de serviço do Chef Automate com o OpsWorks Chef. Você precisará entrar em contato com o Chef para obter uma nova licença. Ao entrar em contato com o Chef, certifique-se de dizer a eles que você já OpsWorks é cliente do Chef Automate e está fazendo a transição. OpsWorks

Para quais alternativas posso fazer a transição?

AWS e o Progress Chef recomendam que você migre para a nova oferta Chef SaaS para que você possa continuar se beneficiando de um serviço Chef Automate totalmente gerenciado. Para começar a usar o Chef SaaS, você pode entrar em contato com o Chef para obter documentação sobre como configurar uma conta do Chef SaaS e fazer a transição de seus dados e nós.

Se o Chef SaaS não atender às suas necessidades porque você prefere executar o Chef Automate em EC2 instâncias nas AWS contas que você controla, o Chef oferece várias opções, incluindo um modelo AWS Marketplace Bring Your Own License (BYOL) e hospedagem automática. EC2 Você pode entrar em contato com o Progress Chef para obter mais informações sobre como executar essa transição.

O serviço ainda está aceitando novos clientes?

Não. AWS OpsWorks for Chef Automate não está mais aceitando novos clientes.

O fim da vida afetará tudo Regiões da AWS ao mesmo tempo?

Sim. A API e o console chegarão ao fim da vida útil e ficarão inutilizáveis a partir de 5 de maio de 2024 em todas as Regiões da AWS. Para obter informações sobre Regiões da AWS onde AWS OpsWorks o Chef Automate está disponível, consulte a Lista de serviços AWS regionais.

Qual nível de suporte técnico está disponível?

AWS continuarão a fornecer o mesmo nível de suporte OpsWorks para o Chef Automate que os clientes têm hoje até a data de fim da vida útil. Se você tiver dúvidas ou preocupações, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support. Para obter suporte na transição, recomendamos que os clientes entrem em contato com o Progress Chef.

Sou cliente atual do OpsWorks Chef Automate e preciso iniciar um servidor em uma conta que não estava usando o serviço anteriormente. Eu sou capaz de fazer isso?

Geralmente não, a menos que haja circunstâncias excepcionais para fazer isso. Se você tiver uma situação especial, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support com os detalhes e a justificativa para isso e analisaremos sua solicitação.

Haverá algum lançamento de recursos importantes no próximo ano?

Não. Como o serviço está chegando ao fim da vida útil, não lançaremos nenhum atributo novo. No entanto, continuaremos fazendo melhorias na segurança e gerenciando os servidores conforme o esperado até a data de fim da vida útil.

Atualize um AWS OpsWorks for Chef Automate servidor para o Chef Automate 2

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Pré-requisitos para a atualização para o Chef Automate 2

Antes de começar, certifique-se de entender os novos recursos adicionados ao Chef Automate 2 e os recursos não compatíveis com o Chef Automate 2. Para obter informações sobre os recursos novos e não compatíveis no Chef Automate 2, consulte a documentação do Chef Automate 2 no site do Chef.

Um servidor que executa o Chef Automate 1 deve conter, pelo menos, uma execução de manutenção bem-sucedida após 1º de novembro de 2019 para se tornar elegível para atualização.

Como acontece com qualquer operação de manutenção em seu AWS OpsWorks for Chef Automate servidor, o servidor fica off-line durante a atualização. Programe um período de inatividade de até 3 horas durante o processo de atualização.

Você precisa das credenciais de entrada deste servidor para o site do painel do Chef Automate. Ao concluir a atualização, acesse o painel do Chef Automate e verifique se os nós e as informações da configuração não foram alteradas.

> **Important**
>
> Quando você estiver pronto para atualizar seu AWS OpsWorks for Chef Automate servidor para o Chef Automate 2, use apenas as instruções aqui para atualizar. Como AWS OpsWorks for Chef Automate automatiza muitos dos processos de atualização, como a criação de backup, não siga as instruções de atualização no site do Chef.

Sobre o processo de atualização

Durante o processo de atualização, o backup do servidor é feito antes de começar a atualização e após concluir a atualização. Os seguintes backups serão criados:

- Um backup do servidor quando ainda estiver executando o Chef Automate 1 (versão 12.17.33).
- Um backup do servidor após a atualização ter sido concluída e o servidor estiver executando o Chef Automate 2 (versão 2019-08).

O processo de atualização encerra a EC2 instância da Amazon que o servidor estava usando quando executou o Chef Automate 1. Uma nova instância será criada para executar o servidor do Chef Automate 2.

Atualizar para o Chef Automate 2 (console)

1. Faça login no AWS Management Console e abra o AWS OpsWorks console em https://console.aws.amazon.com/opsworks/.
2. No painel de navegação à esquerda, selecione AWS OpsWorks for Chef Automate.
3. Escolha um servidor para visualizar a página das propriedades. Um banner azul no topo da página indica se o servidor é elegível para a atualização para o Chef Automate 2.

   > **Note**
   >
   > Um servidor que executa o Chef Automate 1 deve conter, pelo menos, uma execução de manutenção bem-sucedida após 1º de novembro de 2019 para se tornar elegível para atualização.

4. Se o servidor for elegível para a atualização, escolha Start upgrade (Iniciar atualização).
5. Aguarde até três horas pela atualização. Durante o processo de atualização, a página de propriedades exibe o status do servidor como Under maintenance (Em manutenção).
6. Quando a atualização é concluída, a página de propriedades exibe as duas mensagens a seguir: Successfully upgraded to Automate 2 (Atualização para o Automate 2 concluída com êxito) e Maintenance completed successfully (Manutenção concluída com êxito). O status do servidor deve ser HEALTHY (Íntegro).
7. Faça login no painel do Chef Automate com as credenciais existentes e verifique se os nós estão relatando corretamente.

Atualizar para o Chef Automate 2 (CLI)

1. (Opcional) Se você não tiver certeza de quais dos seus AWS OpsWorks for Chef Automate servidores estão qualificados para atualização, execute o comando a seguir. Certifique-se de adicionar o `--region` parâmetro se quiser listar AWS OpsWorks for Chef Automate servidores em uma região da AWS diferente da sua região padrão da AWS.

   ```
   aws opsworks-cm describe-servers
   ```

   Nos resultados, procure o valor de `true` para o atributo `CHEF_MAJOR_UPGRADE_AVAILABLE`. Isso indica que o servidor está qualificado para a atualização para o Chef Automate 2. Anote

os nomes dos AWS OpsWorks for Chef Automate servidores que estão qualificados para atualização.

2. Execute o comando a seguir, *server_name* substituindo-o pelo nome de um AWS OpsWorks for Chef Automate servidor. Para atualizar para o Chef Automate 2 em vez de executar a manutenção de rotina do sistema, adicione o atributo do mecanismo `CHEF_MAJOR_UPGRADE`, conforme exibido no comando. Adicione o parâmetro `--region` se o servidor de destino não estiver na região da AWS padrão. É possível atualizar somente um servidor por comando.

```
aws opsworks-cm start-maintenance --server-name server_name --engine-attributes
 Name=CHEF_MAJOR_UPGRADE,Value=true --region region
```

Se AWS OpsWorks for Chef Automate não for possível atualizar o servidor por algum motivo, esse comando resultará em uma exceção de validação.

3. Aguarde até três horas pela atualização. Você pode verificar o status de atualização periodicamente ao executar o comando a seguir.

```
aws opsworks-cm describe-servers --server-name server_name
```

Nos resultados, procure o valor de `Status`. Um `Status` de `UNDER_MAINTENANCE` indica que a atualização ainda está em andamento. Uma atualização bem-sucedida retornará mensagens semelhantes às mensagens a seguir.

```
2019/10/24 00:27:56 UTC          Successfully upgraded to Automate 2.
2019/10/23 23:50:38 UTC          Upgrading Chef server from Automate 1 to Automate
 2
```

Se a atualização não for bem-sucedida, reverte AWS OpsWorks for Chef Automate automaticamente seu servidor para o Chef Automate 1.

Se a atualização tiver sido bem-sucedida, mas o servidor não estiver funcionando da mesma forma como antes da atualização (por exemplo, se os nós gerenciados não estiverem relatando), é possível reverter o servidor manualmente. Para obter informações sobre a reversão manual, consulte Reverta um AWS OpsWorks for Chef Automate servidor para o Chef Automate 1 (CLI).

Reverta um AWS OpsWorks for Chef Automate servidor para o Chef Automate 1 (CLI)

Se o processo de atualização falhar, reverte AWS OpsWorks for Chef Automate automaticamente seu servidor para o Chef Automate 1. Se a atualização foi bem-sucedida, mas o servidor não está funcionando da mesma forma que antes da atualização, você pode reverter seu AWS OpsWorks for Chef Automate servidor para o Chef Automate 1 manualmente usando o. AWS CLI

1. Execute o comando a seguir para exibir o `BackupId` do último backup realizado no servidor antes de tentar realizar a atualização. Adicione o parâmetro `--region` se o servidor estiver em uma região da AWS diferente da região da AWS padrão.

   ```
   aws opsworks-cm describe-backups server_name
   ```

   O backup IDs está no formato*`ServerName-yyyyMMddHHmmssSSS`*. Procure as seguintes propriedades do Chef Automate 1 nos resultados.

   ```
   "Engine": "Chef"
   "EngineVersion": "12.17.33"
   ```

2. Execute o comando a seguir, usando o ID do backup retornado na etapa 1 como o valor de `--backup-id`.

   ```
   aws opsworks-cm restore-server --server-name server_name --backup-id ServerName-
   yyyyMMddHHmmssSSS
   ```

   Aguarde entre 20 minutos e 3 horas para restaurar o servidor, dependendo da quantidade de dados armazenados no servidor. Durante a operação de restauração, o servidor apresentará um status de `RESTORING`. Esse status é exibido na página de propriedades do servidor no AWS Management Console, e retornado nos resultados do describe-servers comando.

3. Após a restauração ter sido concluída, o console exibirá a mensagem Restore completed successfully (Restauração concluída com êxito). Seu AWS OpsWorks for Chef Automate servidor está on-line e o mesmo de antes de você iniciar o processo de atualização.

Consulte também

- Manutenção do sistema em AWS OpsWorks for Chef Automate

- Restaurar um AWS OpsWorks for Chef Automate servidor a partir de um backup
- DescribeServers na Referência de API do AWS OpsWorks
- StartMaintenance na Referência de API do AWS OpsWorks

Começando com AWS OpsWorks for Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks for Chef Automate permite que você execute um servidor Chef Automate em AWS. Você pode provisionar um servidor do Chef em cerca de 15 minutos.

A partir de 3 de maio de 2021, AWS OpsWorks for Chef Automate armazena alguns atributos do servidor Chef Automate em AWS Secrets Manager. Para obter mais informações, consulte Integração com AWS Secrets Manager.

O passo a passo a seguir ajuda você a criar seu primeiro servidor Chef em. AWS OpsWorks for Chef Automate

Pré-requisitos

Antes de começar, conclua os pré-requisitos a seguir.

Tópicos

- Configuração de um VPC
- Pré-requisitos para usar um domínio personalizado (opcional)
- Configurar um par de EC2 chaves (opcional)

Configuração de um VPC

Seu AWS OpsWorks for Chef Automate servidor deve operar em uma Amazon Virtual Private Cloud. É possível adicioná-lo a um VPC existente, usar o VPC padrão ou criar um novo VPC para conter o servidor. Para obter informações sobre a Amazon VPC e como criar uma nova VPC, consulte o Guia de conceitos básicos da Amazon VPC.

Se você criar seu próprio VPC ou usar um existente, ele deve ter as configurações ou propriedades a seguir.

- A VPC deve ter no mínimo uma sub-rede.

  Se seu AWS OpsWorks for Chef Automate servidor estiver acessível ao público, torne a sub-rede pública e ative a atribuição automática de IP público.
- DNS resolution deve ser habilitada.
- Na sub-rede, ative Auto-assign public IP.

Pré-requisitos para usar um domínio personalizado (opcional)

Você pode configurar o servidor Chef Automate no seu próprio domínio especificando um endpoint público em um domínio personalizado para usar como endpoint do servidor. Quando você usa um domínio personalizado, todos os itens a seguir são necessários, conforme descrito em detalhes nesta seção.

Tópicos

- Configurar um domínio personalizado
- Obter um certificado
- Obter uma chave privada

Configurar um domínio personalizado

Para executar o servidor Chef Automate no seu próprio domínio personalizado, você precisará de um endpoint público de um servidor, como `https://aws.my-company.com`. Se você especificar um domínio personalizado, também deverá fornecer um certificado e uma chave privada, conforme descrito nas seções anteriores.

Para acessar o servidor depois de criá-lo, inclua um registro DNS CNAME em seu serviço DNS preferido. Esse registro deve apontar o domínio personalizado para o endpoint (o valor do atributo

`Endpoint` do servidor) que é gerado pelo processo de criação do servidor Chef Automate. Você não poderá acessar o servidor usando o valor de `Endpoint` gerado se o servidor estiver usando um domínio personalizado.

Obter um certificado

Para configurar o servidor Chef Automate no seu próprio domínio personalizado, você precisa de um certificado HTTPS no formato PEM. Ele pode ser um certificado autoassinado único ou uma cadeia de certificados. Ao concluir o fluxo de trabalho Create Chef Automate server (Criar servidor Chef Automate) se você especificar esse certificado, também deverá fornecer um domínio personalizado e uma chave privada.

Veja a seguir os requisitos para o valor do certificado:

- Você pode fornecer um certificado autoassinado personalizado ou a cadeia de certificados completa.
- O certificado deve ser um certificado X509 válido ou uma cadeia de certificados no formato PEM.
- O certificado deve ser válido no momento do upload. Não é possível usar um certificado antes de seu período de validade começar (a data `NotBefore` do certificado) ou após sua expiração (a data `NotAfter` do certificado).
- O nome comum do certificado ou os nomes alternativos do assunto (SANs), se presentes, devem corresponder ao valor do domínio personalizado.
- O certificado deve corresponder ao valor do campo Custom private key (Chave privada personalizada).

Obter uma chave privada

Para configurar o servidor Chef Automate em seu próprio domínio personalizado, você precisa de uma chave privada no formato PEM para se conectar ao servidor usando HTTPS. A chave privada não deve ser criptografada e não pode ser protegida por senha nem por frase secreta. Se especificar uma chave privada personalizada, também precisará fornecer um domínio personalizado e um certificado.

Configurar um par de EC2 chaves (opcional)

Uma conexão SSH não é necessária ou recomendada para o gerenciamento típico do servidor do Chef. É possível usar os comandos `knife` para executar a maioria das tarefas de gerenciamento em seu servidor do Chef.

É necessário um par de EC2 chaves para se conectar ao seu servidor usando SSH no caso de você perder ou querer alterar a senha de login do painel do Chef Automate. Você pode usar um par de chaves existente ou criar um novo par de chaves. Para obter mais informações sobre como criar um novo par de EC2 chaves, consulte Amazon EC2 Key Pairs.

Se você não precisar de um par de EC2 chaves, você está pronto para criar um servidor Chef.

Criar um servidor Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

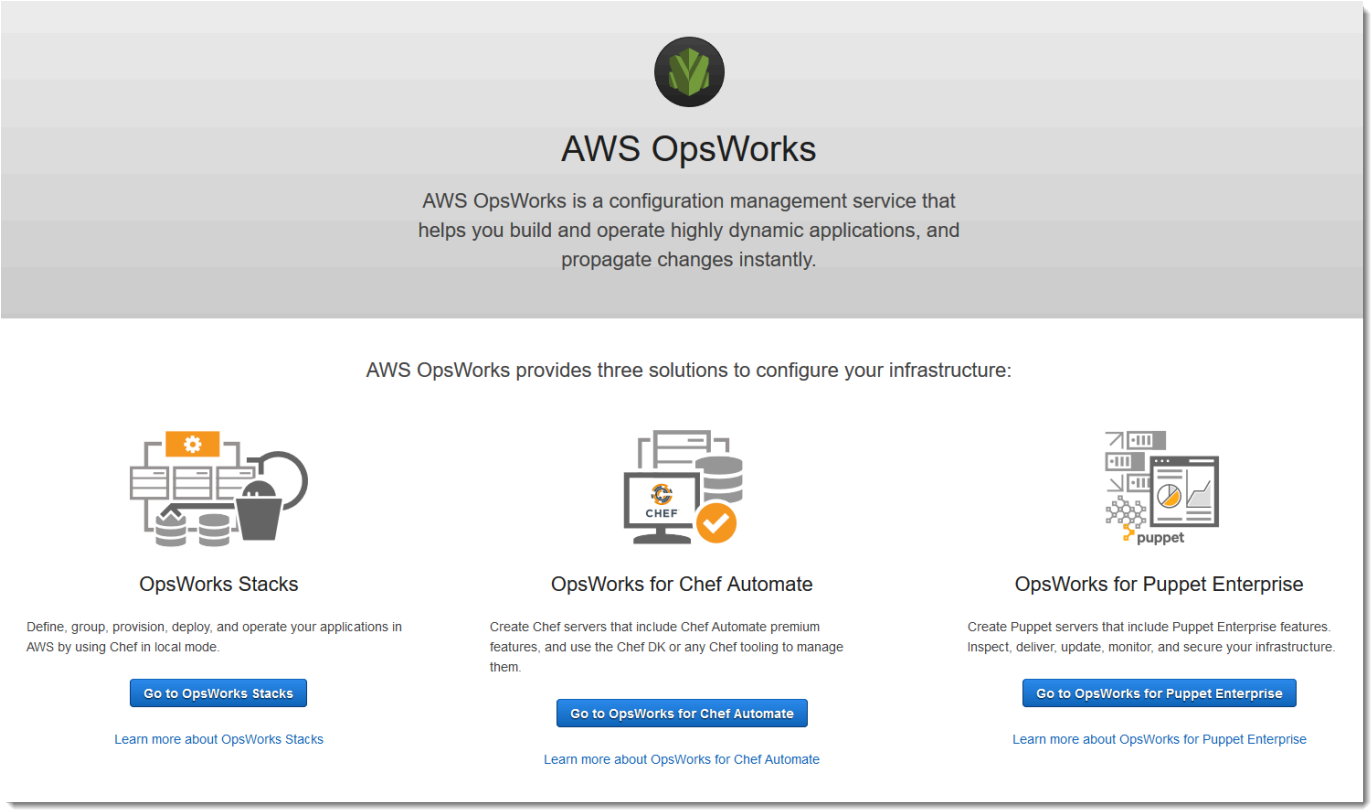
Você pode criar um servidor Chef usando o AWS OpsWorks for Chef Automate console ou AWS CLI o.

Tópicos

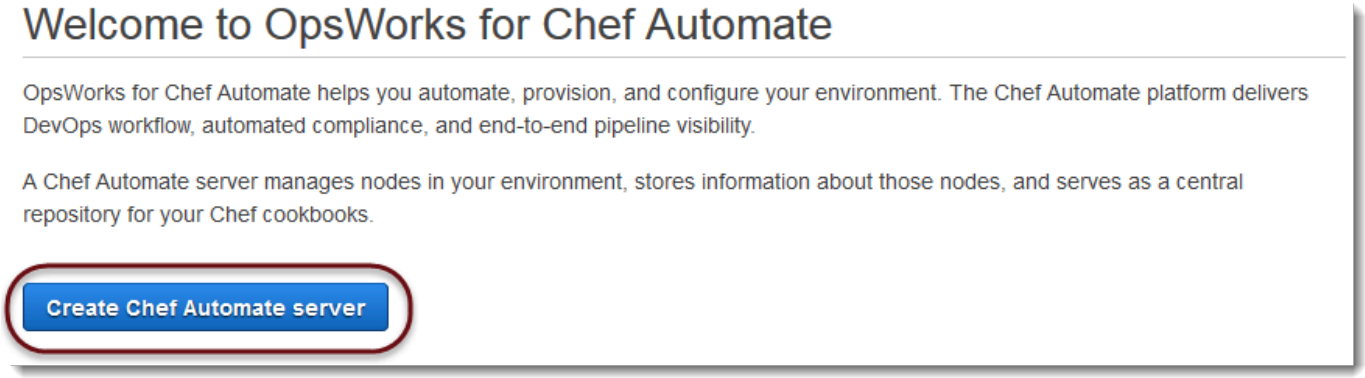
- Crie um servidor Chef Automate no AWS Management Console
- Crie um servidor Chef Automate usando o AWS CLI

Crie um servidor Chef Automate no AWS Management Console

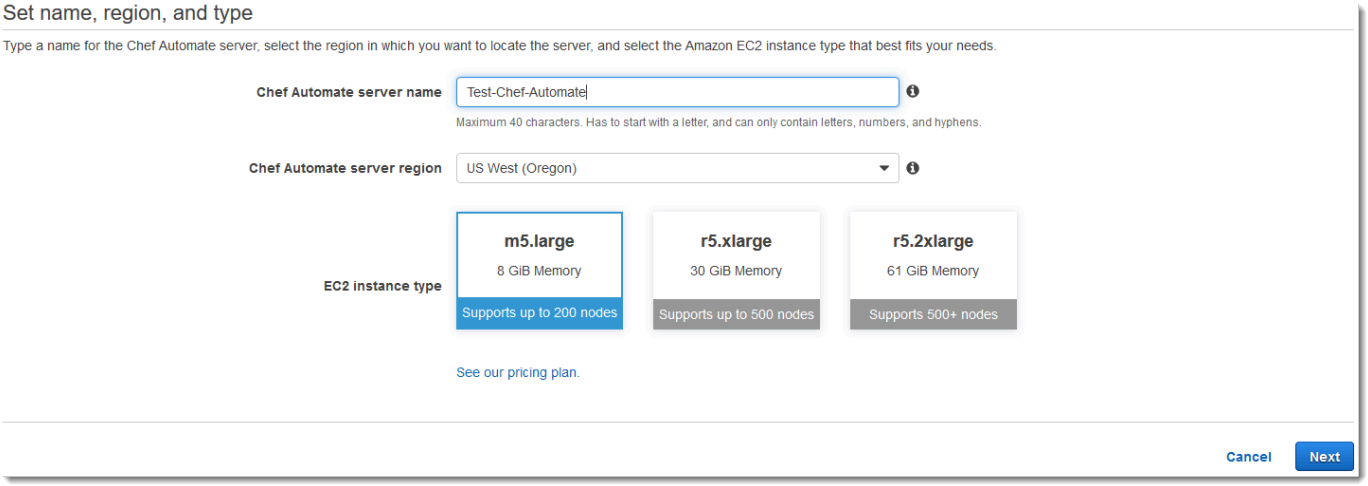
1. Faça login no AWS Management Console e abra o AWS OpsWorks console em https://console.aws.amazon.com/opsworks/.
2. Na página AWS OpsWorks inicial, escolha Ir OpsWorks para o Chef Automate.



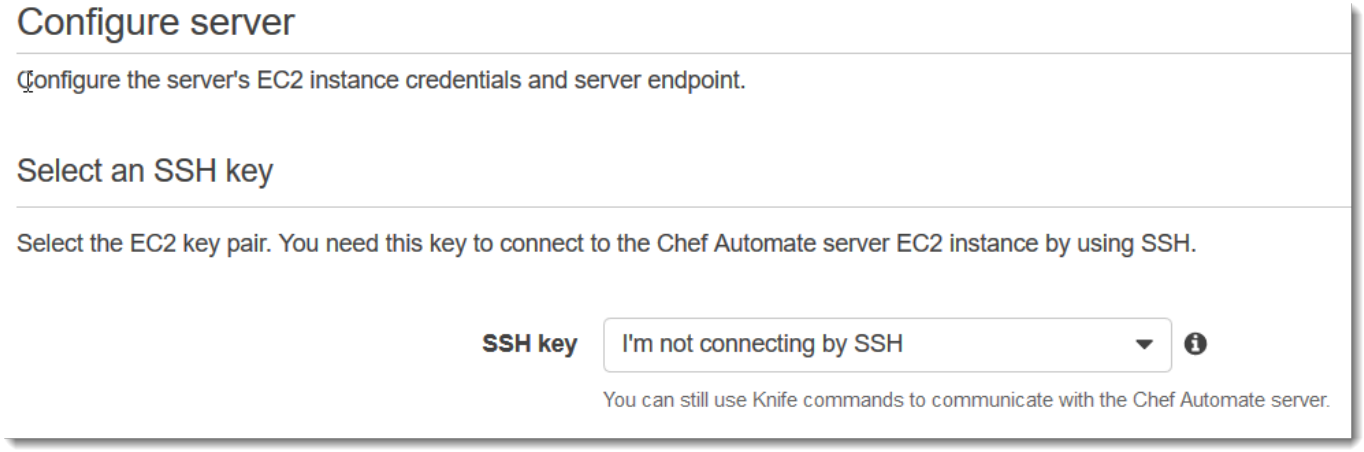
3. Na página AWS OpsWorks for Chef Automate inicial, escolha Criar servidor Chef Automate.



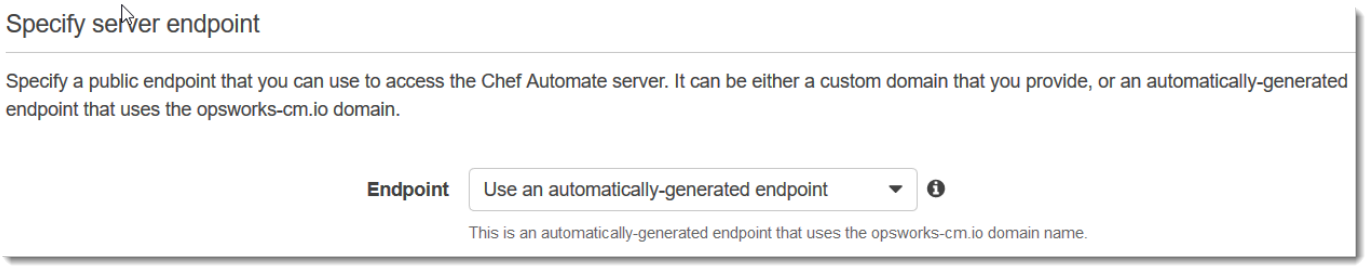
4. Na página Set name, region, and type, especifique um nome para o seu servidor. Os nomes de servidores do Chef podem ter no máximo 40 caracteres e conter somente caracteres alfanuméricos e traços. Selecione uma região compatível e, em seguida, escolha um tipo de instância compatível com o número de nós que deseja gerenciar. É possível alterar o tipo de instância após a criação do servidor, se necessário. Para este passo a passo, estamos criando um tipo de instância m5.large no Oeste dos EUA (Oregon). Escolha Próximo.



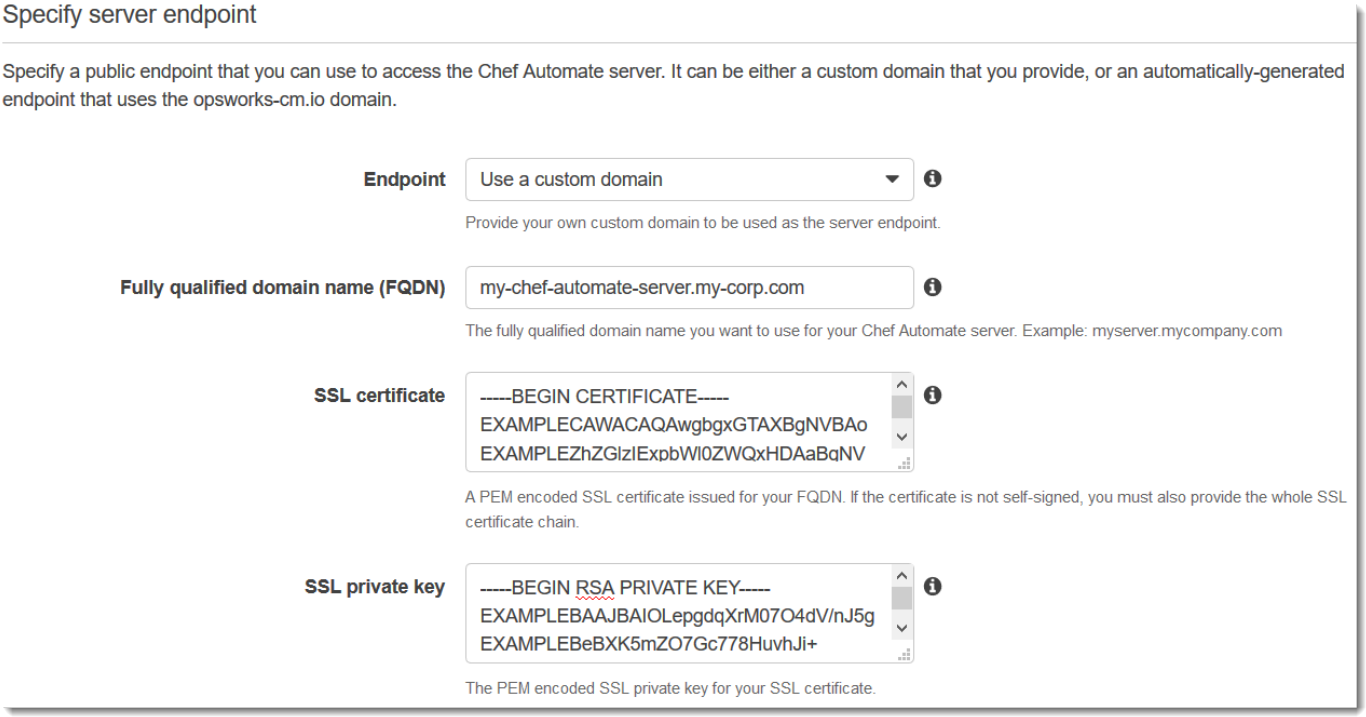
5. Na página Configure server (Configurar servidor), deixe a seleção padrão na lista suspensa SSH key (Chave SSH), a não ser que você deseje especificar um nome de par de chaves.



6. Em Specify server endpoint (Especificar endpoint do servidor), deixe o padrão, Use an automatically-generated endpoint (Usar um endpoint gerado automaticamente) e escolha Next (Avançar), a menos que você queira que o servidor esteja em um domínio personalizado. Para configurar um domínio personalizado, vá para a próxima etapa.



7. Para usar um domínio personalizado, em Specify server endpoint (Especificar endpoint do servidor), escolha Use a custom domain (Usar um domínio personalizado) na lista suspensa.



a. Em Fully qualified domain name (FQDN) (Nome de domínio totalmente qualificado [FQDN]), especifique um FQDN. Você deve ser proprietário do nome de domínio que usará.

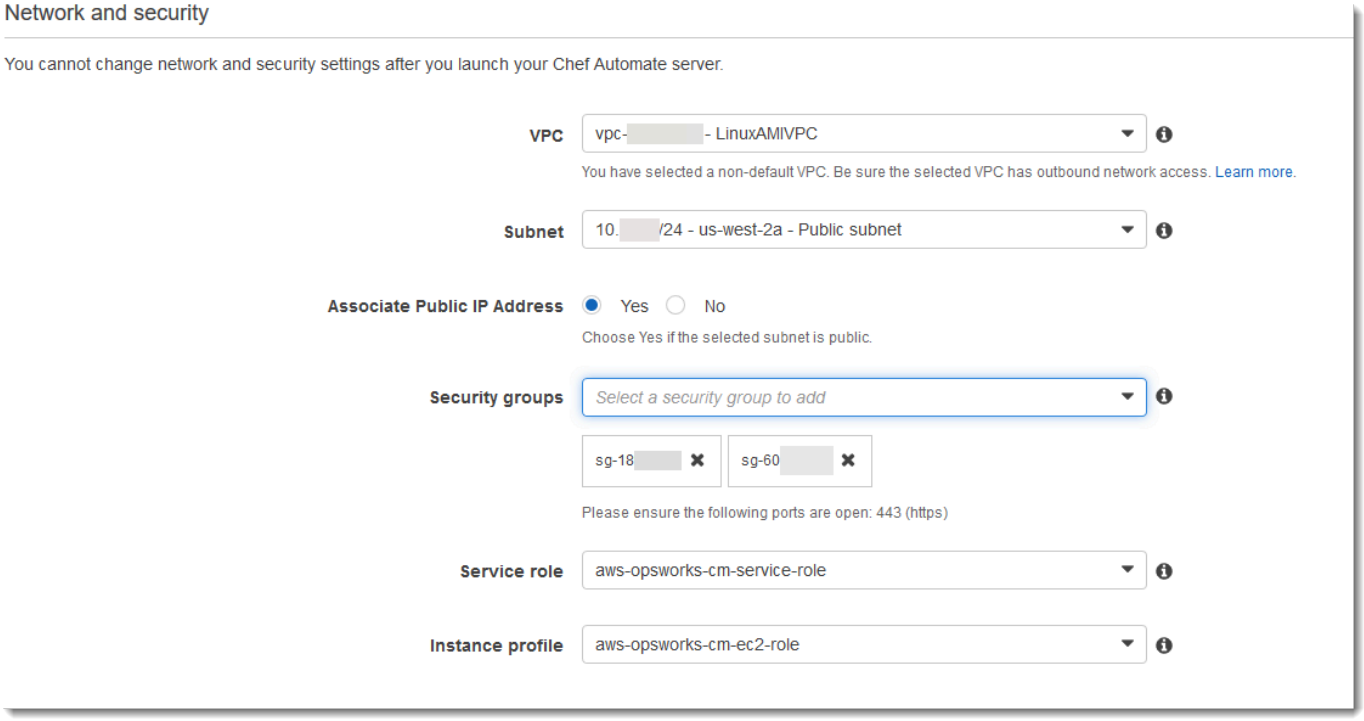
b. Em SSL certificate (Certificado SSL), cole todo o certificado em formato PEM, começando com `-----BEGIN CERTIFICATE-----` e terminando com `-----END CERTIFICATE-----`. O assunto do certificado SSL deve corresponder ao FQDN inserido na etapa anterior.

c. Em SSL private key (Chave privada SSL), cole toda a chave privada RSA, começando com `-----BEGIN RSA PRIVATE KEY-----` e terminando com `-----END RSA PRIVATE KEY-----`. A chave privada SSL precisa corresponder à chave pública do certificado SSL inserido na etapa anterior. Escolha Próximo.

8. Na página Configure advanced settings (Definir configurações avançadas), na seção Network and security (Rede e segurança), escolha uma VPC, uma sub-rede e um ou mais grupos de segurança. Veja a seguir os requisitos para a VPC:

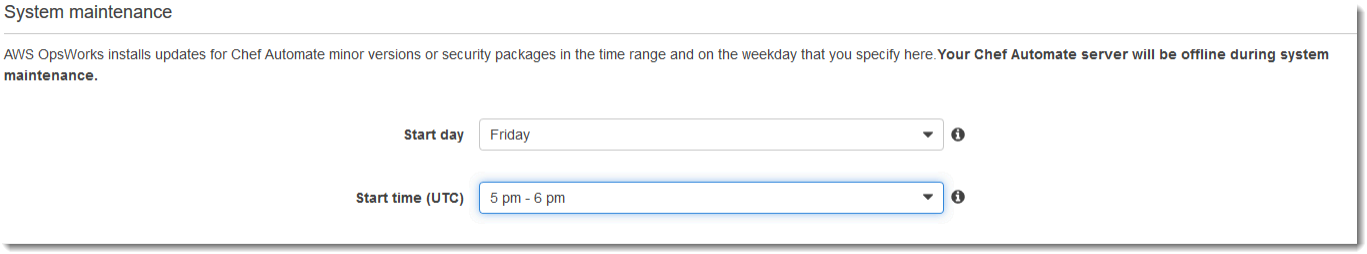
   - A VPC deve ter pelo menos uma sub-rede pública.
   - A resolução DNS deve estar habilitada.
   - Auto-assign public IP (Atribuir IP público de modo automático) deve estar habilitado em sub-redes públicas.

AWS OpsWorks pode gerar um grupo de segurança, uma função de serviço e um perfil de instância para você, caso ainda não tenha alguns que queira usar. O servidor pode ser membro de vários grupos de segurança. Não é possível alterar as configurações de rede e segurança do servidor do Chef depois de sair desta página.



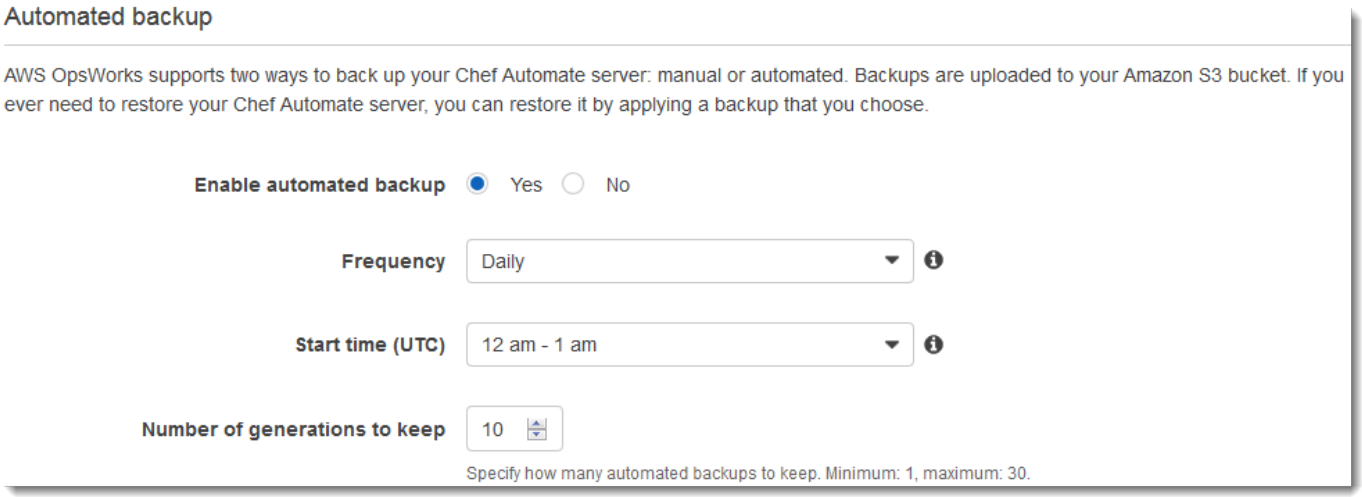
9. Na seção System maintenance, defina o dia e a hora em que você deseja que a manutenção do sistema se inicie. Como é recomendável que o servidor esteja offline durante a manutenção do sistema, escolha um horário de baixa demanda do servidor no horário comercial regular. Os nós conectados entram no estado `pending-server` até a manutenção ser concluída.

   A janela de manutenção é necessária. Você pode alterar o dia e a hora de início posteriormente usando o AWS Management Console AWS CLI, ou APIs o.



10. Configure os backups. Por padrão, os backups automáticos estão ativados. Defina uma preferência de frequência e hora para o início do backup automático e defina o número de

gerações de backup a armazenar no Amazon Simple Storage Service. Um máximo de 30 backups são mantidos; quando o máximo é atingido, AWS OpsWorks for Chef Automate exclui os backups mais antigos para abrir espaço para novos.



11. (Opcional) Em Tags, adicione tags ao servidor e aos recursos relacionados, como EC2 instância, endereço IP elástico, grupo de segurança, bucket do S3 e backups. Para obter mais informações sobre como marcar um AWS OpsWorks for Chef Automate servidor, consulteTrabalhando com tags em AWS OpsWorks for Chef Automate recursos.
12. Ao concluir a configuração das definições avançadas, selecione Next.
13. Na página Analisar, examine suas escolhas. Quando você estiver pronto para criar o servidor, escolha Launch.

    Enquanto você espera AWS OpsWorks para criar seu servidor Chef, acesse Configurar o servidor do Chef usando o Starter Kit e baixe o Starter Kit e as credenciais do painel do Chef Automate. Não espere até que o servidor esteja online para fazer o download desses itens.

    Quando a criação for concluída, o servidor do Chef estará disponível na página inicial do AWS OpsWorks for Chef Automate , com um status online. Depois que o servidor estiver online, o painel do Chef Automate estará disponível no domínio do servidor, em um URL no seguinte formato: `https://`*`your_server_name`*`-random.region.opsworks-cm.io`.

Crie um servidor Chef Automate usando o AWS CLI

Criar um AWS OpsWorks for Chef Automate servidor executando AWS CLI comandos é diferente de criar um servidor no console. No console, AWS OpsWorks cria uma função de serviço e um grupo de segurança para você, se você não especificar os existentes que deseja usar. No AWS CLI, AWS

OpsWorks pode criar um grupo de segurança para você se você não especificar um, mas ele não cria automaticamente uma função de serviço; você deve fornecer um ARN da função de serviço como parte do seu `create-server` comando. No console, enquanto AWS OpsWorks cria seu servidor Chef Automate, você baixa o kit inicial do Chef Automate e as credenciais de login para o painel do Chef Automate. Como você não pode fazer isso ao criar um AWS OpsWorks for Chef Automate servidor usando o AWS CLI, use um utilitário de processamento JSON para obter as credenciais de login e o kit inicial dos resultados do `create-server` comando depois que o novo servidor estiver on-line. AWS OpsWorks for Chef Automate Como alternativa, você pode gerar um novo conjunto de credenciais de login e um novo starter kit no console depois que o novo servidor do AWS OpsWorks for Chef Automate estiver online.

Se seu computador local ainda não estiver executando o AWS CLI, faça o download e instale-o AWS CLI seguindo as instruções de instalação no Guia do usuário da interface de linha de comando da AWS. Esta seção não descreve todos os parâmetros que você pode usar com o comando `create-server`. Para obter mais informações sobre os parâmetros do `create-server`, consulte `create-server` na Referência da AWS CLI .

1. Certifique-se de concluir os pré-requisitos, especialmente Configuração de um VPC ou verifique se você tem uma VPC existente que deseja usar. Para criar o servidor do Chef Automate, você precisa de um ID de sub-rede.
2. Opcionalmente, gere uma chave principal do Chef usando o OpenSSL e salve a chave em um arquivo seguro e conveniente em seu computador local. A chave principal será gerada automaticamente como parte do processo de criação do servidor, se você não fornecer uma no comando `create-server`. Se quiser ignorar esta etapa, você poderá obter a chave principal do Chef Automate nos resultados do comando `create-server`. Se optar por gerar a chave principal usando os comandos a seguir, certifique-se de incluir o parâmetro `-pubout`, porque o valor da chave principal do Chef Automate é a metade pública do par de chaves RSA. Para obter mais informações, consulte a etapa 6.

   ```
   umask 077
   openssl genrsa -out "pivotal" 2048
   openssl rsa -in "pivotal" -pubout
   ```

3. Crie uma função de serviço e um perfil de instância.
4. Encontre e copie as funções ARNs de serviço em sua conta.

   ```
   aws iam list-roles --path-prefix "/service-role/" --no-paginate
   ```

Nos resultados do comando `list-roles`, procure pelas entradas de ARN da função de serviço semelhantes às seguintes. Anote a função de serviço ARNs. Você precisa desses valores para criar o servidor Chef Automate.

```
{
    "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "ec2.amazonaws.com"
                }
            }
        ]
    },
    "RoleId": "AROZZZZZZZZZZQG6R22HC",
    "CreateDate": "2018-01-05T20:42:20Z",
    "RoleName": "aws-opsworks-cm-ec2-role",
    "Path": "/service-role/",
    "Arn": "arn:aws:iam::000000000000:role/service-role/aws-opsworks-cm-ec2-role"
},
{
    "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "opsworks-cm.amazonaws.com"
                }
            }
        ]
    },
    "RoleId": "AROZZZZZZZZZZZZZZZ6QE",
    "CreateDate": "2018-01-05T20:42:20Z",
    "RoleName": "aws-opsworks-cm-service-role",
    "Path": "/service-role/",
    "Arn": "arn:aws:iam::000000000000:role/service-role/aws-opsworks-cm-service-
role"
```

```
}
```

5. Encontre e copie os perfis ARNs da instância em sua conta.

```
aws iam list-instance-profiles --no-paginate
```

Nos resultados do comando `list-instance-profiles`, procure pelas entradas de ARN do perfil de instância semelhantes às seguintes. Anote o perfil da instância ARNs. Você precisa desses valores para criar o servidor Chef Automate.

```
{
    "Path": "/",
    "InstanceProfileName": "aws-opsworks-cm-ec2-role",
    "InstanceProfileId": "EXAMPLEDC6UR3LTUW7VHK",
    "Arn": "arn:aws:iam::123456789012:instance-profile/aws-opsworks-cm-ec2-role",
    "CreateDate": "2017-01-05T20:42:20Z",
    "Roles": [
        {
            "Path": "/service-role/",
            "RoleName": "aws-opsworks-cm-ec2-role",
            "RoleId": "EXAMPLEE4STNUQG6R22HC",
            "Arn": "arn:aws:iam::123456789012:role/service-role/aws-opsworks-cm-
ec2-role",
            "CreateDate": "2017-01-05T20:42:20Z",
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {
                            "Service": "ec2.amazonaws.com"
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            }
        }
    ]
},
```

6. Crie o AWS OpsWorks for Chef Automate servidor executando o `create-server` comando.

- O valor de `--engine` é `ChefAutomate`, de `--engine-model` é `Single` e de `--engine-version` é `12`.
- O nome do servidor deve ser exclusivo em sua AWS conta, em cada região. Os nomes dos servidores devem começar com uma letra; letras, números ou hífens (-) são permitidos, até um máximo de 40 caracteres.
- Use o ARN do perfil de instância e o ARN da função de serviço que você copiou nas etapas 4 e 5.
- Os tipos de instância válidos são `m5.large`, `r5.xlarge` ou `r5.2xlarge`. Para obter mais informações sobre as especificações desses tipos de instância, consulte Tipos de instância no Guia EC2 do usuário da Amazon.
- O parâmetro `--engine-attributes` é opcional; se você não especificar um dos valores ou ambos, o processo de criação do servidor gerará os valores para você. Se você adicionar `--engine-attributes`, especifique o valor da `CHEF_AUTOMATE_PIVOTAL_KEY` que você gerou na Etapa 2, uma `CHEF_AUTOMATE_ADMIN_PASSWORD` ou ambos.

  Se você não definir um valor para `CHEF_AUTOMATE_ADMIN_PASSWORD`, uma senha será gerada e retornada como parte da resposta do comando `create-server`. Você também pode fazer download do starter kit novamente no console, o que fará com que a senha seja gerada novamente. O comprimento da senha deve ter um mínimo de 8 e um máximo de 32 caracteres. A senha só pode conter letras, números e caracteres especiais (`!/@#$%^+=_`). Ela deve conter pelo menos uma letra minúscula, uma letra maiúscula, um número e um caractere especial.
- Um par de chaves SSH é opcional, mas pode ajudá-lo a se conectar ao servidor do Chef Automate se você precisar redefinir a senha de administrador do painel do Chef Automate. Para obter mais informações sobre a criação de um par de chaves SSH, consulte Amazon EC2 Key Pairs no Amazon EC2 User Guide.
- Para usar um domínio personalizado, adicione os seguintes parâmetros ao comando. Caso contrário, o processo de criação do servidor Chef Automate gerará automaticamente um endpoint para você. Todos os três parâmetros são necessários para configurar um domínio personalizado. Para obter informações sobre requisitos adicionais para usar esses parâmetros, consulte CreateServera Referência da API AWS OpsWorks CM.
  - `--custom-domain` – um endpoint público opcional de um servidor, como `https://aws.my-company.com`.

  - `--custom-certificate` – um certificado HTTPS no formato PEM. O valor pode ser um certificado autoassinado único ou uma cadeia de certificados.
  - `--custom-private-key` – uma chave privada no formato PEM para a conexão com o servidor usando HTTPS. A chave privada não deve ser criptografada e não pode ser protegida por senha nem por frase secreta.
- É necessário fazer a manutenção semanal do sistema. Os valores válidos devem sempre ser especificados no seguinte formato: `DDD:HH:MM`. A hora especificada está em formato de Tempo Universal Coordenado (UTC). Se você não especificar um valor para `--preferred-maintenance-window`, o valor padrão será aleatório, um período de uma hora na terça, quarta ou sexta-feira.
- Os valores válidos para `--preferred-backup-window` devem ser especificados em um dos seguintes formatos: `HH:MM` para backups diários ou `DDD:HH:MM` para backups semanais. A hora é especificada em UTC. O valor padrão é uma hora de início aleatória estabelecida diariamente. Para cancelar os backups automáticos, adicione o parâmetro `--disable-automated-backup`.
- Para`--security-group-ids`, insira um ou mais grupos de segurança IDs, separados por um espaço.
- Em `--subnet-ids`, insira um ID de sub-rede.

```
aws opsworks-cm create-server --engine "ChefAutomate" --engine-model "Single"
 --engine-version "12" --server-name "server_name" --instance-profile-arn
 "instance_profile_ARN" --instance-type "instance_type" --engine-attributes
 '{"CHEF_AUTOMATE_PIVOTAL_KEY":"pivotal_key","CHEF_AUTOMATE_ADMIN_PASSWORD":"password"}'
 --key-pair "key_pair_name" --preferred-maintenance-window
 "ddd:hh:mm" --preferred-backup-window "ddd:hh:mm" --security-group-
ids security_group_id1 security_group_id2 --service-role-arn "service_role_ARN" --
subnet-ids subnet_ID
```

Veja um exemplo a seguir.

```
aws opsworks-cm create-server --engine "ChefAutomate" --engine-
model "Single" --engine-version "12" --server-name "automate-06" --
instance-profile-arn "arn:aws:iam::12345678912:instance-profile/aws-
opsworks-cm-ec2-role" --instance-type "m5.large" --engine-attributes
 '{"CHEF_AUTOMATE_PIVOTAL_KEY":"MZZE...Wobg","CHEF_AUTOMATE_ADMIN_PASSWORD":"zZZzDj2DLYXSZF
 --key-pair "amazon-test" --preferred-maintenance-window "Mon:08:00" --preferred-
backup-window "Sun:02:00" --security-group-ids sg-b00000001 sg-b0000008 --service-
```

```
role-arn "arn:aws:iam::12345678912:role/service-role/aws-opsworks-cm-service-role"
 --subnet-ids subnet-300aaa00
```

O exemplo a seguir cria um servidor Chef Automate que usa um domínio personalizado.

```
aws opsworks-cm create-server --engine "ChefAutomate" --engine-model "Single" --
engine-version "12" \
    --server-name "my-custom-domain-server" \
    --instance-profile-arn "arn:aws:iam::12345678912:instance-profile/aws-opsworks-
cm-ec2-role" \
    --instance-type "m5.large" \
    --engine-attributes
 '{"CHEF_AUTOMATE_PIVOTAL_KEY":"MZZE...Wobg","CHEF_AUTOMATE_ADMIN_PASSWORD":"zZZzDj2DLYXSZF
 \
    --custom-domain "my-chef-automate-server.my-corp.com" \
    --custom-certificate "-----BEGIN CERTIFICATE----- EXAMPLEqEXAMPLE== -----END
 CERTIFICATE-----" \
    --custom-private-key "-----BEGIN RSA PRIVATE KEY----- EXAMPLEqEXAMPLE= -----END
 RSA PRIVATE KEY-----" \
    --key-pair "amazon-test" \
    --preferred-maintenance-window "Mon:08:00" \
    --preferred-backup-window "Sun:02:00" \
    --security-group-ids sg-b00000001 sg-b0000008 \
    --service-role-arn "arn:aws:iam::12345678912:role/service-role/aws-opsworks-cm-
service-role" \
    --subnet-ids subnet-300aaa00
```

O exemplo a seguir cria um servidor do Chef Automate que adiciona duas tags: `Stage: Production` e `Department: Marketing`. Para obter mais informações sobre como adicionar e gerenciar tags em AWS OpsWorks for Chef Automate servidores, consulte Trabalhando com tags em AWS OpsWorks for Chef Automate recursos este guia.

```
aws opsworks-cm create-server --engine "ChefAutomate" --engine-model "Single" --
engine-version "12" \
    --server-name "my-test-chef-server" \
    --instance-profile-arn "arn:aws:iam::12345678912:instance-profile/aws-opsworks-
cm-ec2-role" \
    --instance-type "m5.large" \
    --engine-attributes
 '{"CHEF_AUTOMATE_PIVOTAL_KEY":"MZZE...Wobg","CHEF_AUTOMATE_ADMIN_PASSWORD":"zZZzDj2DLYXSZF
 \
    --key-pair "amazon-test" \
```

```
    --preferred-maintenance-window "Mon:08:00" \
    --preferred-backup-window "Sun:02:00" \
    --security-group-ids sg-b00000001 sg-b0000008 \
    --service-role-arn "arn:aws:iam::12345678912:role/service-role/aws-opsworks-cm-
service-role" \
    --subnet-ids subnet-300aaa00 \
    --tags [{\"Key\":\"Stage\",\"Value\":\"Production\"},{\"Key\":\"Department\",
\"Value\":\"Marketing\"}]
```

7. AWS OpsWorks for Chef Automate leva cerca de 15 minutos para criar um novo servidor. Não descarte a saída do comando `create-server` ou feche a sessão de shell, pois a saída pode conter informações importantes que não serão exibidas novamente. Para obter as senhas e o starter kit dos resultados do comando `create-server`, vá para a próxima etapa.

   Se você estiver usando um domínio personalizado com o servidor, na saída do comando `create-server`, copie o valor do atributo `Endpoint`. Veja um exemplo a seguir.

```
"Endpoint": "automate-07-exampleexample.opsworks-cm.us-east-1.amazonaws.com"
```

8. Se você optou por AWS OpsWorks for Chef Automate gerar uma chave e uma senha para você, poderá extraí-las dos `create-server` resultados em formatos utilizáveis usando um processador JSON, como jq. Após instalar o jq, você pode executar os comandos a seguir para extrair a chave principal, a senha de administrador do painel do Chef Automate e o starter kit. Se você não forneceu sua própria chave principal e senha na etapa 4, salve a chave principal e a senha de administrador extraídas em locais seguros e convenientes.

```
#Get the Chef password:
cat resp.json | jq -r '.Server.EngineAttributes[] | select(.Name ==
 "CHEF_AUTOMATE_ADMIN_PASSWORD") | .Value'

#Get the Chef Pivotal Key:
cat resp.json | jq -r '.Server.EngineAttributes[] | select(.Name ==
 "CHEF_AUTOMATE_PIVOTAL_KEY") | .Value'

#Get the Chef Starter Kit:
cat resp.json | jq -r '.Server.EngineAttributes[] | select(.Name ==
 "CHEF_STARTER_KIT") | .Value' | base64 -D > starterkit.zip
```

9. Opcionalmente, se você não extraiu o kit inicial dos resultados do `create-server` comando, você pode baixar um novo kit inicial na página Propriedades do servidor no console.

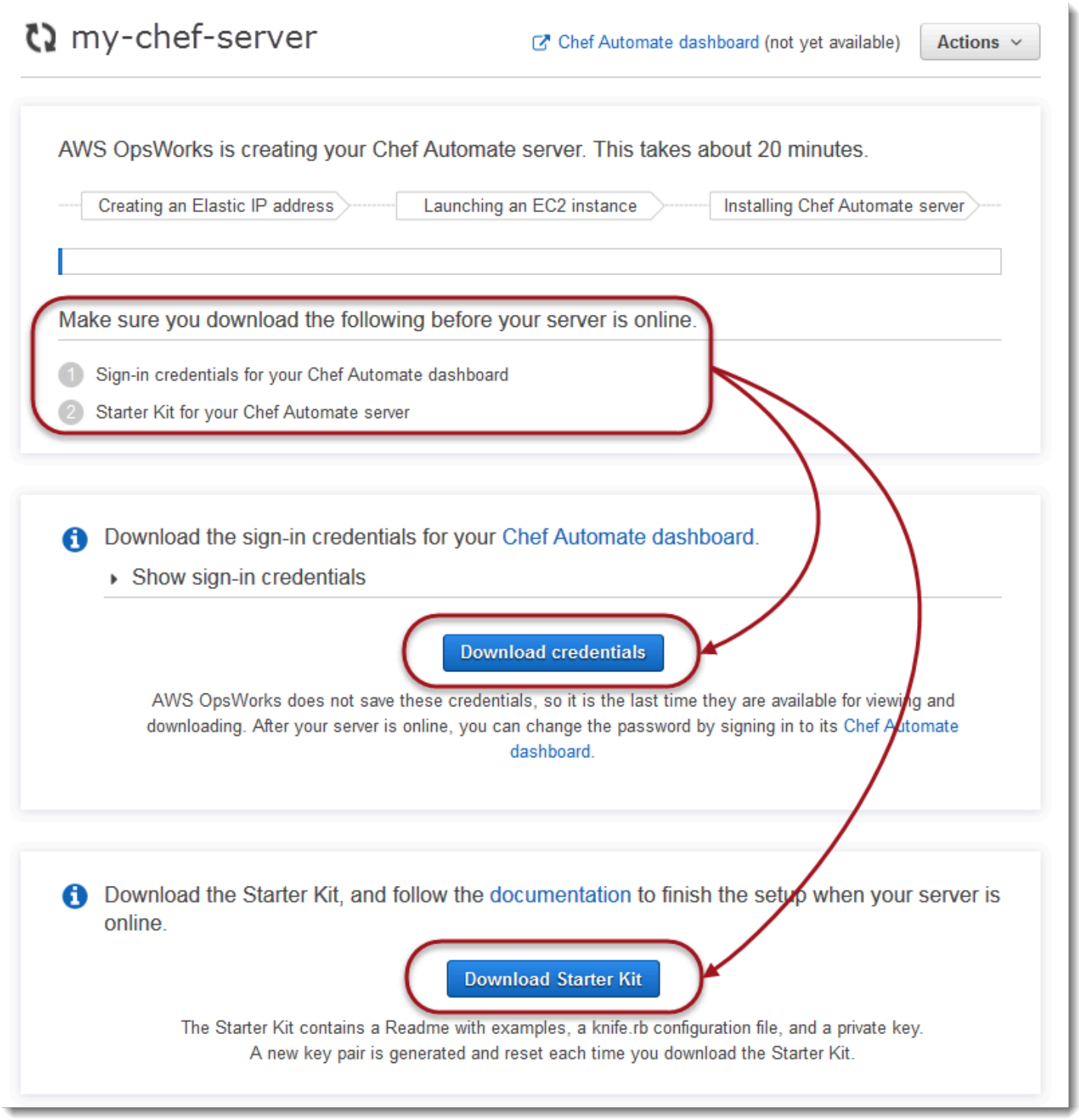
AWS OpsWorks for Chef Automate Ao fazer download de um novo starter kit, a senha de administrador do painel do Chef Automate é redefinida.

10. Se você não estiver usando um domínio personalizado, vá para a próxima etapa. Se você estiver usando um domínio personalizado com o servidor, crie uma entrada CNAME na ferramenta de gerenciamento de DNS da sua empresa para direcionar seu domínio personalizado para o AWS OpsWorks for Chef Automate endpoint que você copiou na etapa 7. Você não poderá acessar nem se conectar a um servidor com um domínio personalizado se não concluir esta etapa.
11. Quando o processo de criação do servidor for concluído, vá para the section called “Concluir a configuração e fazer o upload de livros de receitas”.

Configurar o servidor do Chef usando o Starter Kit

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Enquanto a criação do servidor do Chef ainda está em andamento, abra a página Properties dele no console do AWS OpsWorks for Chef Automate . Na primeira vez que você trabalhar com um novo servidor do Chef, a página Properties solicitará que você faça o download de dois itens necessários. Faça o download desses itens antes que o servidor do Chef fique online. Os botões de download não ficam disponíveis depois que um novo servidor está online.



- Sign-in credentials for the Chef server (Credenciais de login para o servidor do Chef). Você usará essas credenciais para entrar no painel do Chef Automate, onde trabalha com os recursos premium do Chef Automate, como fluxo de trabalho e verificações de conformidade. AWS OpsWorks não salva essas credenciais; essa é a última vez que elas estão disponíveis para

visualização e download. Se necessário, será possível alterar a senha fornecida com essas credenciais depois de fazer login.

- Starter Kit. O Starter Kit contém um arquivo README com exemplos, um arquivo de configuração `knife.rb` e uma chave privada para o usuário principal ou essencial. Um novo par de chaves será gerado (e a antiga chave será redefinida) sempre que você fizer o download do Starter Kit.

Além das credenciais que funcionam somente com o novo servidor, o arquivo.zip do Starter Kit inclui um exemplo simples de um repositório do Chef que funciona com qualquer servidor. AWS OpsWorks for Chef Automate No repositório do Chef, você armazena livros de receitas, funções, arquivos de configuração e outros artefatos para o gerenciamento de seus nós com o Chef. Recomendamos que você armazene esse repositório em um sistema de controle de versão, como Git, e trate-o como código-fonte. Para informações e exemplos que mostram como configurar um repositório do Chef controlado no Git, consulte Sobre o chef-repo na documentação do Chef.

Pré-requisitos

1. Embora a criação do servidor ainda esteja em andamento, faça o download das credenciais de login para o servidor do Chef e salve-as em um local seguro, mas conveniente.
2. Faça o download do Starter Kit e descompacte o arquivo Starter Kit..zip no diretório do espaço de trabalho. Não compartilhe a chave privada do Starter Kit. Se outros usuários forem gerenciar o servidor do Chef, adicione-os como administradores no painel do Chef Automate posteriormente.
3. Baixe e instale o Chef Workstation (anteriormente conhecido como Chef Development Kit, ou Chef DK) no computador que você usará para gerenciar seu servidor e nós do Chef. O utilitário `knife` faz parte do Chef Workstation. Para instruções, consulte instalar o Chef Workstation no site do Chef.

Explorar o conteúdo do Starter Kit

O Starter Kit tem o seguinte conteúdo.

- `cookbooks/` - Um diretório para os livros de receitas criados por você. A `cookbooks/` pasta contém o livro de receitas `opsworks-webserver`, um livro de receitas embalado que depende do livro de receitas `nginx` do site do Chef Supermarket. `Policyfile.rb` usa como padrão o Chef supermarket como fonte secundária se as dependências do livro de receitas não estiverem disponíveis no diretório `cookbooks/`.

- `Policyfile.rb` - Um arquivo de política baseado no Ruby que define os livros de receitas, as dependências e os atributos que se tornam a política para os nós.
- `userdata.sh` e `userdata.ps1` - Você poderá usar arquivos de dados de usuário para associar automaticamente nós depois de iniciar o servidor do Chef Automate. `userdata.sh` é para o bootstrapping de nós baseados no Linux e `userdata.ps1` é para nós baseados no Windows.
- `Berksfile` - Você poderá usar este arquivo se preferir usar o Berkshelf e os comandos `berks` para fazer o upload de livros de receitas e suas dependências. Neste passo a passo, usamos `Policyfile.rb` e comandos do Chef para fazer o upload de livros de receitas, dependências e atributos.
- `README.md`, um arquivo baseado no Markdown que descreve como usar o Starter Kit para configurar o servidor do Chef Automate pela primeira vez.
- `.chef` é um diretório oculto que contém um arquivo de configuração knife (`knife.rb`) e um arquivo de chave de autenticação secreta (.pem).
  - `.chef/knife.rb` - Um arquivo de configuração knife (`knife.rb`). O `knife.rb`arquivo é configurado para que as operações da `knife`ferramenta do Chef sejam executadas no AWS OpsWorks for Chef Automate servidor.
  - `.chef/ca_certs/opsworks-cm-ca-2020-root.pem` - Uma chave privada SSL assinada por uma autoridade de certificação (CA) fornecida pelo AWS OpsWorks. Essa chave permite que o servidor identifique-se para o agente cliente do Chef Infra em nós que ele gerencia.

Configurar seu repositório do Chef

Um repositório do Chef contém vários diretórios. Cada diretório no Starter Kit contém um arquivo README que descreve a finalidade de diretório e como usá-lo para gerenciar seus sistemas com o Chef. Há duas maneiras de obter livros de receitas instalados no servidor do Chef: executando comandos `knife` ou um comando do Chef para fazer o upload de um arquivo de política (`Policyfile.rb`) para o servidor que faz o download e instala livros de receitas especificados. Este passo a passo usa comandos do Chef e `Policyfile.rb` para instalar livros de receitas no servidor.

1. Crie um diretório no computador local para armazenar livros de receitas, como `chef-repo`. Depois de adicionar livros de receitas, funções e outros arquivos a esse repositório, recomendamos que você faça o upload ou o armazene em um sistema seguro e com controle de versão, como o CodeCommit Git ou o Amazon S3.
2. No diretório `chef-repo`, crie os seguintes diretórios:

- `cookbooks/`: armazena livros de receitas.
- `roles/`: armazena funções nos formatos `.rb` ou `.json`
- `environments/`: armazena ambientes nos formatos `.rb` ou `.json`.

Use Policyfile.rb para obter livros de receitas de uma fonte remota

Nesta seção, edite `Policyfile.rb` para especificar livros de receitas e executar um comando do Chef para fazer o upload do arquivo para o servidor e instalar livros de receitas.

1. Visualize `Policyfile.rb` no Starter Kit. Por padrão, `Policyfile.rb` inclui o livro de receitas wrapper `opsworks-webserver`, que depende do livro de receitas `nginx` disponível no site do Chef Supermarket. O livro de receitas `nginx` instala e configura um servidor web em nós gerenciados. O livro de receitas `chef-client` necessário, que instala o agente cliente do Chef Infra em nós gerenciados, também é especificado.

   `Policyfile.rb` também aponta para o livro de receitas do Chef Audit, que você pode usar para configurar verificações de conformidade em nós. Para obter mais informações sobre como configurar verificações de conformidade e obter resultados de conformidade de nós gerenciados, consulte Verificações de conformidade em AWS OpsWorks for Chef Automate. Se você não quiser configurar verificações de conformidade e auditoria agora, exclua `'audit'` da seção `run_list` e não especifique os atributos do livro de receitas `audit` no final do arquivo.

```
# Policyfile.rb - Describe how you want Chef to build your system.
#
# For more information about the Policyfile feature, visit

# https://docs.chef.io/policyfile.html


# A name that describes what the system you're building with Chef does.

name 'opsworks-demo-webserver'

# The cookbooks directory is the preferred source for external cookbooks

default_source :chef_repo, "cookbooks/" do |s|

  s.preferred_for "nginx", "windows", "chef-client", "yum-epel", "seven_zip",
```

```
                    "build-essential", "mingw", "ohai", "audit", "logrotate", "cron"

end
# Alternative source
default_source :supermarket


# run_list: chef-client runs these recipes in the order specified.

run_list  'chef-client',
          'opsworks-webserver',
          'audit'
# add 'ssh-hardening' to your runlist to fix compliance issues detected by the ssh-
baseline profile

# Specify a custom source for a single cookbook:

cookbook 'opsworks-webserver', path: 'cookbooks/opsworks-webserver'


# Policyfile defined attributes

# Define audit cookbook attributes
default["opsworks-demo"]["audit"]["reporter"] = "chef-server-automate"
default["opsworks-demo"]["audit"]["profiles"] = [
  {
    "name": "DevSec SSH Baseline",
    "compliance": "admin/ssh-baseline"
  }
]
```

Este é um exemplo de `Policyfile.rb` sem o livro de receitas `audit` e os atributos, caso você queira configurar apenas o servidor web `nginx` por enquanto.

```
# Policyfile.rb - Describe how you want Chef to build your system.
#
# For more information on the Policyfile feature, visit
# https://docs.chef.io/policyfile.html

# A name that describes what the system you're building with Chef does.
name 'opsworks-demo-webserver'

# Where to find external cookbooks:
```

```
default_source :supermarket

# run_list: chef-client will run these recipes in the order specified.
run_list  'chef-client',
          'opsworks-webserver'

# Specify a custom source for a single cookbook:
cookbook 'opsworks-webserver', path: 'cookbooks/opsworks-webserver'
```

Se você fizer alterações em `Policyfile.rb`, salve o arquivo.

2. Faça o download e instale os livros de receitas definidos em `Policyfile.rb`.

```
chef install
```

Todos os livros de receitas têm versionamento no arquivo `metadata.rb` do livro de receitas. Sempre que altera um livro de receitas, você deve aumentar a versão dele que está no `metadata.rb`.

3. Se você tiver optado por configurar verificações de conformidade e mantido as informações do livro de receitas `audit` no arquivo de política, envie a política `opsworks-demo` para o servidor.

```
chef push opsworks-demo
```

4. Se você concluiu a Etapa 3, verifique a instalação da política. Execute o seguinte comando:

```
chef show-policy
```

Os resultados devem ser semelhantes ao seguinte:

```
opsworks-demo-webserver
=======================
* opsworks-demo:  ec0fe46314
```

5. Você já está pronto para adicionar ou inicializar nós no servidor do Chef Automate. Você pode automatizar a associação de nós seguindo as etapas em Adicione nós automaticamente em AWS OpsWorks for Chef Automate ou adicionar nós, um de cada vez, seguindo as etapas em Adicione nós individualmente.

(Alternativa) Usar o Berkshelf para obter livros de receitas de uma fonte remota

O Berkshelf é uma ferramenta para o gerenciamento de livros de receitas e suas dependências. Se você preferir usar o Berkshelf em vez de `Policyfile.rb` instalar livros de receitas em armazenamento local, use o procedimento desta seção, e não da seção anterior. Você pode especificar quais livros de receitas e versões serão utilizados com o servidor do Chef e fazer o upload deles. O Starter Kit contém um arquivo chamado `Berksfile` que você pode usar para listar os livros de receitas.

1. Para começar, adicione o livro de receitas `chef-client` ao Berksfile incluído. O livro de receitas `chef-client` configura o software do agente cliente do Chef Intra em cada nó conectado ao servidor do Chef Automate. Para saber mais sobre esse livro de receitas, consulte Chef Client Cookbook no Chef Supermarket.
2. Usando um editor de texto, anexe outro livro de receitas ao Berksfile que instala um aplicativo de servidor Web; por exemplo, o livro de receitas `apache2`, que instala o servidor Web Apache. Seu Berksfile deve se parecer com o seguinte.

   ```
   source 'https://supermarket.chef.io'
   cookbook 'chef-client'
   cookbook 'apache2'
   ```

3. Faça o download e a instalação dos livros de receitas em seu computador local.

   ```
   berks install
   ```

4. Carregue o livro de receitas no servidor do Chef.

   No Linux, execute o seguinte.

   ```
   SSL_CERT_FILE='.chef/ca_certs/opsworks-cm-ca-2020-root.pem' berks upload
   ```

   No Windows, execute o seguinte comando do Chef Workstation em uma PowerShell sessão. Antes de executar o comando, certifique-se de definir a política de execução PowerShell como`RemoteSigned`. Adicione `chef shell-init` para disponibilizar os comandos do utilitário Chef Workstation para PowerShell.

   ```
   $env:SSL_CERT_FILE="ca_certs\opsworks-cm-ca-2020-root.pem"
   chef shell-init berks upload
   Remove-Item Env:\SSL_CERT_FILE
   ```

5. Verifique a instalação do livro de receitas mostrando uma lista de livros de receitas atualmente disponíveis no servidor Chef Automate. Você pode fazer isso executando o seguinte comando `knife`:

   Você está pronto para adicionar nós para gerenciar com o AWS OpsWorks for Chef Automate servidor.

   ```
   knife cookbook list
   ```

(Opcional) Configurar **`knife`** para trabalhar com um domínio personalizado

Se o seu servidor Chef Automate usa um domínio personalizado, talvez seja necessário adicionar o certificado PEM da CA raiz que assinou a cadeia de certificados do seu servidor ou o certificado PEM do servidor se o certificado for autoassinado. `ca_certs`é um subdiretório `chef/` que contém autoridades de certificação (CAs) nas quais o `knife` utilitário Chef confia.

Ignore esta seção se não estiver usando um domínio personalizado ou se o certificado personalizado estiver assinado por uma CA raiz que seja confiável para o sistema operacional. Caso contrário, configure o `knife` para confiar no certificado SSL do servidor Chef Automate, conforme descrito nas etapas a seguir.

1. Execute o seguinte comando:

   ```
   knife ssl check
   ```

   Se os resultados forem semelhantes aos seguintes, ignore o restante deste procedimento e vá para Adicionar nós para o servidor do Chef gerenciar.

   ```
   Connecting to host my-chef-automate-server.my-corp.com:443
             Successfully verified certificates from 'my-chef-automate-server.my-
   corp.com'
   ```

   Se receber uma mensagem de erro semelhante à seguinte, vá para a próxima etapa.

   ```
   Connecting to host my-chef-automate-server.my-corp.com:443
             ERROR: The SSL certificate of my-chef-automate-server.my-corp.com could
    not be verified.
             ...
   ```

2. Execute `knife ssl fetch` para confiar nos certificados do servidor do AWS OpsWorks for Chef Automate . Você também pode copiar manualmente o certificado no formato PEM da CA raiz do servidor no diretório que é o valor de `trusted_certs_dir` na saída de `knife ssl check`. Por padrão, esse diretório está em `.chef/ca_certs/` no Starter Kit. A saída deve ser semelhante à seguinte:

```
WARNING: Certificates from my-chef-automate-server.my-corp.com will be fetched and
 placed in your trusted_cert
          directory (/Users/username/starterkit/.chef/../.chef/ca_certs).

          Knife has no means to verify these are the correct certificates. You
 should
          verify the authenticity of these certificates after downloading.

          Adding certificate for my-chef-automate-server in /Users/users/
starterkit/.chef/../.chef/ca_certs/servv-aqtswxu20swzkjgz.crt
          Adding certificate for MyCorp_Root_CA in /Users/users/
starterkit/.chef/../.chef/ca_certs/MyCorp_Root_CA.crt
```

3. Execute `knife ssl check` novamente. A saída deve ser semelhante à seguinte:

```
Connecting to host my-chef-automate-server.my-corp.com:443
          Successfully verified certificates from 'my-chef-automate-server.my-
corp.com'
```

Você está pronto para usar o `knife` com o servidor Chef Automate.

Adicionar nós para o servidor do Chef gerenciar

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O agente `chef-client` executa receitas do Chef em computadores físicos ou virtuais, chamados de nós, que são associados ao servidor. Você pode conectar computadores ou instâncias localmente ao servidor do Chef para gerenciar, desde que os nós estejam executando sistemas operacionais compatíveis. O registro de nós junto ao servidor do Chef instala o software do agente `chef-client` nesses nós.

É possível usar os seguintes métodos para adicionar nós:

- Adicione notas individualmente executando um `knife` comando que adiciona ou inicializa uma EC2 instância para que o servidor Chef possa gerenciá-la. Para obter mais informações, consulte Adicione nós individualmente.
- Adicionar nós automaticamente usando um script para executar a associação automática de nós com o servidor do Chef. O código no Starter Kit mostra como adicionar nós automaticamente usando o método desassistido. Para obter mais informações, consulte, Adicione nós automaticamente em AWS OpsWorks for Chef Automate.

Tópicos

- Adicione nós individualmente
- Adicione nós automaticamente em AWS OpsWorks for Chef Automate

Adicione nós individualmente

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como executar um `knife` comando que adiciona ou inicializa uma EC2 instância para que o servidor Chef possa gerenciá-la.

A versão mínima compatível do `chef-client` em nós associados a um servidor AWS OpsWorks for Chef Automate é 13.x. Recomendamos executar a `chef-client` versão mais atual e estável.

Tópicos

- (Opcional) Especificar o URL da CA raiz do servidor Chef Automate
- Sistemas operacionais compatíveis
- Adicionar nós com knife

(Opcional) Especificar o URL da CA raiz do servidor Chef Automate

Se o servidor estiver usando um domínio e um certificado personalizados, talvez seja necessário editar a variável `ROOT_CA_URL` no script de dados de usuário com um URL público que você pode usar para obter o certificado no formato PEM da CA raiz do seu servidor. Os comandos da AWS CLI a seguir carregam sua CA raiz para um bucket do Amazon S3 e geram um pre-signed URL que você pode usar por uma hora.

1. Carregue o certificado no formato PEM da CA raiz no S3.

```
aws s3 cp ROOT_CA_PEM_FILE_PATH s3://bucket_name/
```

2. Gere um pre-signed URL que você pode usar por uma hora (3.600 segundos, neste exemplo) para fazer download da CA raiz.

```
aws s3 presign s3://bucket_name/ROOT_CA_PEM_FILE_NAME --expires-in 3600
```

3. Edite a variável `ROOT_CA_URL` no script de dados de usuário com o valor do pre-signed URL.

Sistemas operacionais compatíveis

Para consultar a lista de sistemas operacionais compatíveis para nós, acesse Website do Chef

Adicionar nós com knife

O plug-in `knife-ec2` está incluído no Chef Workstation. Se você estiver mais familiarizado`knife-ec2`, poderá usá-lo em vez de `knife bootstrap` provisionar e inicializar novos EC2instances. Caso contrário, execute uma nova EC2 instância e siga as etapas nesta seção.

Para adicionar nós a gerenciar

1. Execute o seguinte comando `knife bootstrap`. Esse comando inicializa uma EC2 instância nos nós que seu servidor Chef gerenciará. Observe que você está instruindo o servidor do Chef a executar receitas do livro de receitas `nginx` instalado em the section called “Use Policyfile.rb

para obter livros de receitas de uma fonte remota". Para obter mais informações sobre a adição de nós ao executar o comando `knife bootstrap`, consulte Ação de bootstrap em um nó na documentação do Chef.

A tabela a seguir mostra os nomes de usuário válidos para os sistemas operacionais do nó no comando `knife` nesta etapa. Se nem `root` nem `ec2-user` funcionar, verifique com o seu provedor de AMI. Para obter mais informações sobre a conexão com instâncias baseadas em Linux, consulte a seção Conexão à sua instância Linux usando SSH na documentação da AWS.

Valores válidos para nomes de usuário nos sistemas operacionais do nó

Sistema operacional	Nomes de usuário válidos
Amazon Linux	`ec2-user`
Red Hat Enterprise Linux 5	`root` ou `ec2-user`
Ubuntu	`ubuntu`
Fedora	`fedora` ou `ec2-user`
SUSE Linux	`root` ou `ec2-user`

```
knife bootstrap INSTANCE_IP_ADDRESS -N INSTANCE_NAME -x USER_NAME --sudo --run-list
 "recipe[nginx]"
```

2. Verifique se o novo nó foi adicionado executando os comandos a seguir, *INSTANCE_NAME* substituindo-os pelo nome da instância que você acabou de adicionar.

```
knife client show INSTANCE_NAME
knife node show INSTANCE_NAME
```

Adicione nós automaticamente em AWS OpsWorks for Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes

migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico descreve como adicionar nós do Amazon Elastic Compute Cloud (Amazon EC2) ao seu servidor Chef automaticamente. O código no Starter Kit mostra como adicionar nós automaticamente usando o método desassistido. O método recomendado de associação desassistida ou automática de novos nós é configurar o Livro de receitas do Chef Client. Você pode usar o script `userdata` no Starter Kit e alterar a seção `run_list` do script `userdata` ou seu `Policyfile.rb` com os livros de receitas que deseja aplicar aos nós. Antes de executar o agente `chef-client`, instale o livro de receitas do Chef Client no servidor do Chef e o agente `chef-client` em modo de serviço com, por exemplo, uma função HTTPD, conforme mostrado no comando de exemplo a seguir.

```
chef-client -r "chef-client,role[httpd]"
```

Para se comunicar com o software de servidor do Chef, o agente `chef-client` deve ter acesso à chave pública do nó do cliente. Você pode gerar um par de chaves pública-privada na Amazon EC2 e depois passar a chave pública para a chamada de AWS OpsWorks `associate-node` API com o nome do nó. O script mostrado incluído no Starter Kit reúne o nome da organização, nome do servidor e endpoint do servidor. Isso garante que o nó esteja associado ao servidor do Chef, e o software do agente `chef-client` que é executado no nó possa se comunicar com o servidor depois de vincular a chave privada.

A versão mínima compatível do `chef-client` em nós associados a um servidor AWS OpsWorks for Chef Automate é 13.x. Recomendamos executar a `chef-client` versão mais atual e estável.

Para obter informações sobre como desassociar um nó, consulte Desassociar um nó de um servidor AWS OpsWorks for Chef Automate este guia e `disassociate-node`a documentação da AWS OpsWorks for Chef Automate API.

Tópicos

- Sistemas operacionais compatíveis
- Etapa 1: criar um perfil do IAM para usar como o perfil de instância
- Etapa 2: instalar o livro de receitas Chef Client
- Etapa 3: Criar instâncias com um script de associação autônoma
- Outros métodos de automatizar execuções de chef-client repetidas

- Related Topics

Sistemas operacionais compatíveis

Para consultar a lista de sistemas operacionais compatíveis para nós, acesse Website do Chef

Etapa 1: criar um perfil do IAM para usar como o perfil de instância

Crie uma função AWS Identity and Access Management (IAM) para usar como seu perfil de EC2 instância e anexe a política a seguir à função do IAM. Essa política permite que a API AWS OpsWorks for Chef Automate (`opsworks-cm`) se comunique com a EC2 instância durante o registro do nó. Para obter mais informações sobre perfis de instância, consulte Como usar perfis de instância na EC2 documentação da Amazon. Para obter informações sobre como criar uma função do IAM, consulte Como criar uma função do IAM no console na EC2 documentação da Amazon.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "opsworks-cm:AssociateNode",
                "opsworks-cm:DescribeNodeAssociationStatus",
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

Etapa 2: instalar o livro de receitas Chef Client

Se ainda não tiver feito isso, siga as etapas em (Alternativa) Usar o Berkshelf para obter livros de receitas de uma fonte remota para garantir que o Berksfile ou o arquivo `Policyfile.rb` faça referência ao livro de receitas do Chef Client e instale o livro de receitas.

Etapa 3: Criar instâncias com um script de associação autônoma

1. Para criar EC2 instâncias, você pode copiar o `userdata` script do Starter Kit para a `userdata` seção de instruções da EC2 instância, configurações de lançamento em grupo do Amazon EC2 Auto Scaling ou um modelo. AWS CloudFormation Para obter mais informações sobre como

adicionar scripts aos dados do usuário, consulte Executando comandos na sua instância Linux no lançamento na EC2 documentação da Amazon.

Este script executa o comando da `opsworks-cm`API `associate-node` para associar um novo nó com o servidor do Chef.

Por padrão, o nome do novo nó registrado é o ID da instância, mas é possível alterar o nome modificando o valor da variável `NODE_NAME` no script `userdata`. Atualmente, não é possível alterar o nome da organização na interface do console do Chef, por isso, defina `CHEF_AUTOMATE_ORGANIZATION` como `default`.

2. Siga o procedimento em Iniciando uma instância na EC2 documentação, com modificações aqui. No assistente de execução da EC2 instância, escolha uma Amazon Linux AMI.
3. Na página Configure Instance Details (Configurar detalhes da instância), selecione a função que você criou em Etapa 1: criar um perfil do IAM para usar como o perfil de instância, como sua função do IAM.
4. Na área Advanced Details (Detalhes avançados), carregue o script `userdata.sh` que você criou anteriormente nesse procedimento.
5. Nenhuma alteração é necessária na página Add Storage. Vá para Add Tags.
6. Na página Configure Security Group (Configurar security group), selecione Add Rule (Adicionar regra) e selecione o tipo HTTP para abrir os números de porta 443 e 80 para o servidor web Apache neste exemplo.
7. Escolha Review and Launch e, em seguida, selecione Launch. Quando o novo nó é iniciado, ele aplica as configurações especificadas pelas receitas que você especificou no parâmetro `RUN_LIST`.
8. Opcional: se você tiver adicionado o livro de receitas `nginx` à sua lista de execuções, quando você abrir a página da web vinculada ao DNS público do novo nó, você deve ver um site hospedado pelo servidor web nginx.

Outros métodos de automatizar execuções de **`chef-client`** repetidas

Embora seja mais difícil de realizar e não recomendado, você pode executar o script neste tópico somente como parte dos dados do usuário da instância autônoma, usar um AWS CloudFormation modelo para adicioná-lo aos dados do usuário da nova instância, configurar um `cron` trabalho para executar o script regularmente ou executar `chef-client` dentro de um serviço. No entanto, recomendamos o método Chef Client Cookbook por causa de algumas desvantagens de outras técnicas de automação.

Para uma lista completa de parâmetros que você pode fornecer para `chef-client`, consulte a documentação do Chef.

Related Topics

As postagens de AWS blog a seguir oferecem mais informações sobre a associação automática de nós ao seu servidor Chef Automate, usando grupos de Auto Scaling ou em várias contas.

- Usando o AWS OpsWorks for Chef Automate para gerenciar EC2 instâncias com Auto Scaling
- OpsWorks para Chef Automate — Inicializando automaticamente nós em contas diferentes

Faça login no painel do Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Depois de ter feito o download das credenciais de login na página Properties do servidor do Chef e o servidor estiver online, faça login no painel do Chef Automate. Nesta demonstração, nós instruímos você, primeiramente, a fazer o upload de livros de receitas e adicionar pelo menos um nó para gerenciar. Isso permite que você consulte informações sobre os livros de receitas e nós no painel.

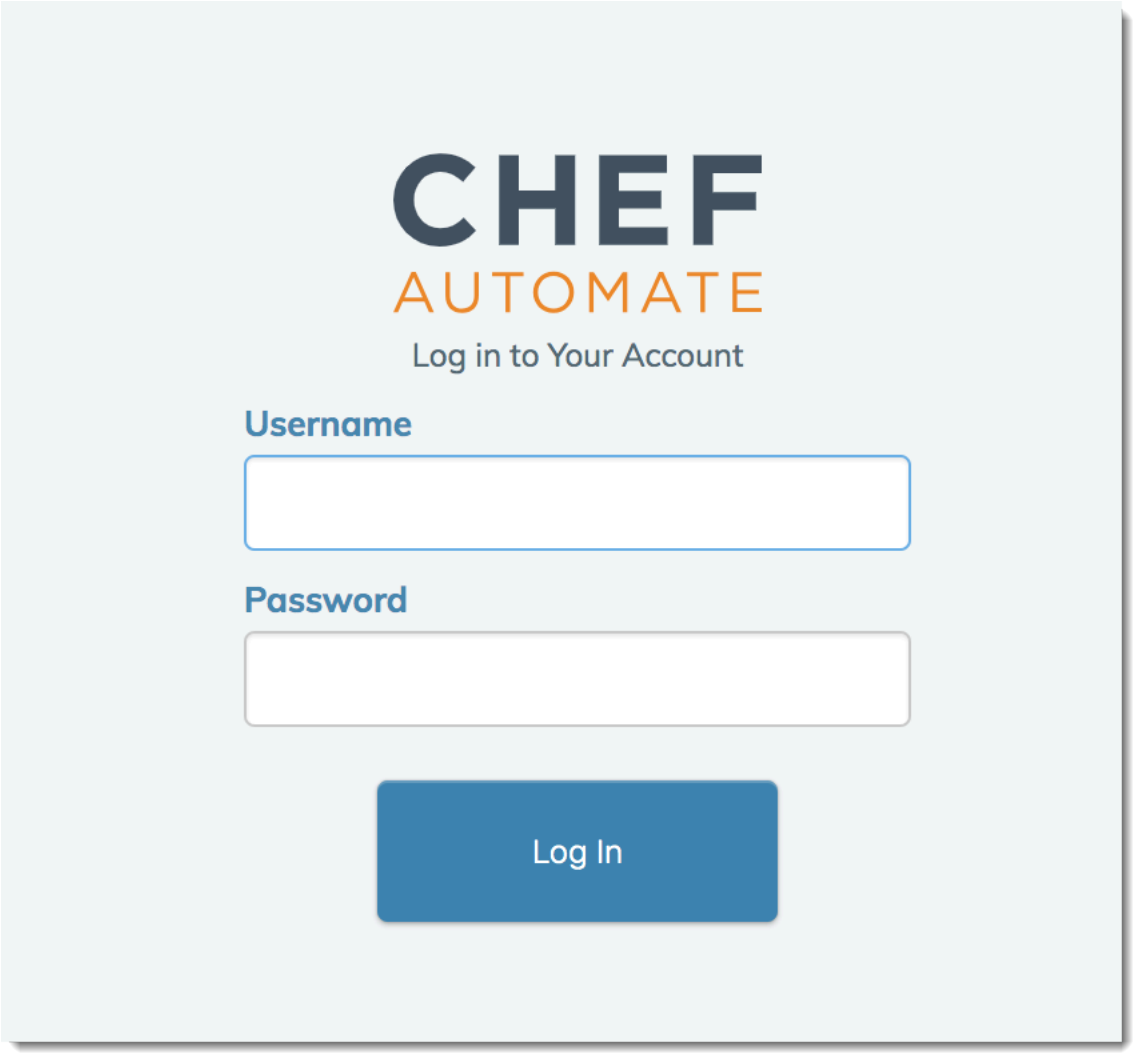
Quando você tenta se conectar à página do painel, os avisos de certificado aparecem em seu navegador até que você instale um certificado SSL AWS OpsWorks específico assinado por CA no computador cliente que você está usando para gerenciar seu servidor Chef. Se você preferir não ver os avisos antes de continuar para a página da web do painel, instale o certificado SSL antes de fazer login.

Depois de ter instalado o certificado SSL do lado do cliente, é possível fazer login no painel do Chef Automate sem visualizar mensagens de aviso.

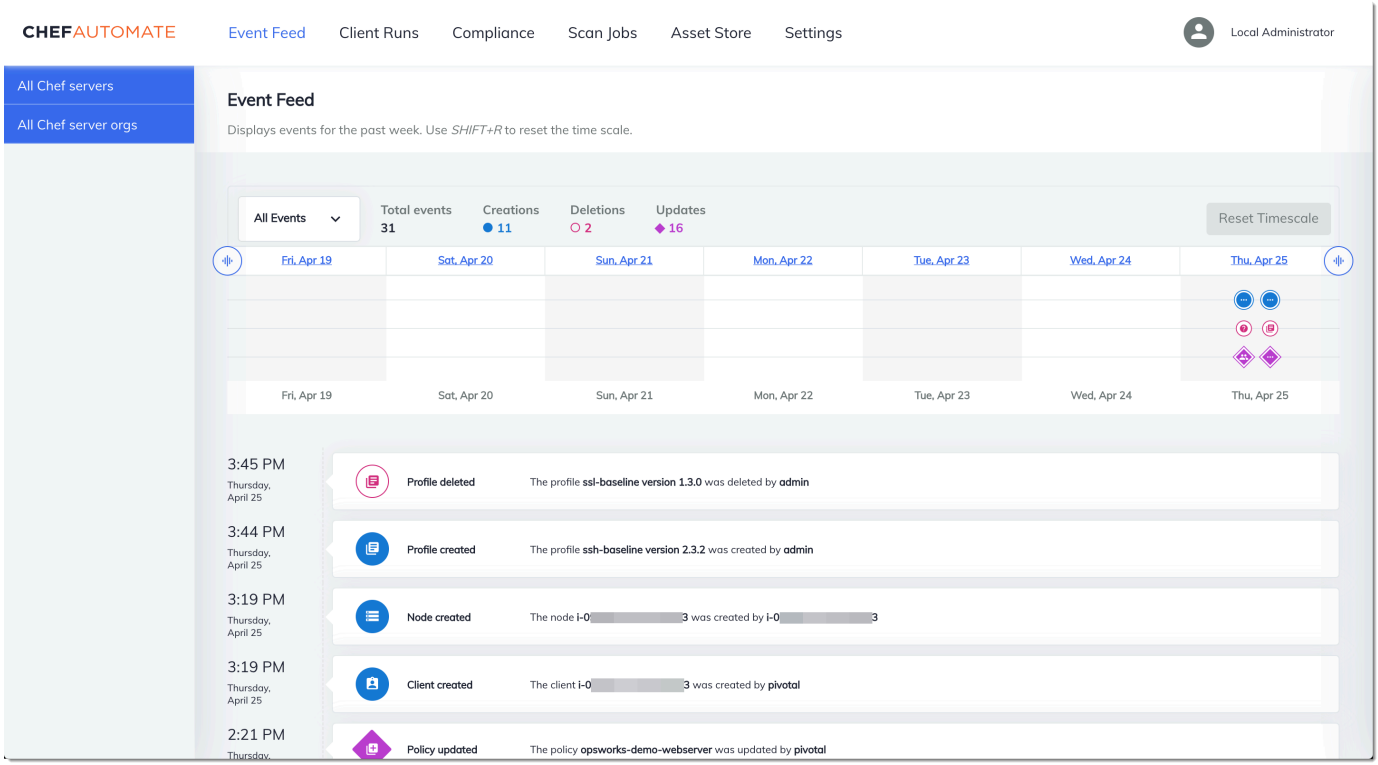
> **Note**
>
> Os usuários do Google Chrome nos sistemas operacionais Ubuntu e Linux Mint podem ter dificuldade para fazer login. Recomendamos que você use o Mozilla Firefox ou outros navegadores para acessar e usar o painel do Chef Automate nesses sistemas operacionais. Nenhum problema foi encontrado usando o Google Chrome no Windows ou MacOS.

Para fazer login no painel do Chef Automate

1. Descompacte e abra as credenciais do Chef Automate credenciais que você baixou em Pré-requisitos. Você precisará essas credenciais para fazer login.
2. Abra a página Properties do servidor do Chef.
3. No canto superior direito da página Properties, escolha Open Chef Automate dashboard.
4. Faça login usando as credenciais da Etapa 1.



5. No painel do Chef Automate, você pode exibir informações detalhadas sobre os nós em que realizou ações de bootstrap, progresso e eventos de execução do livro de receitas, nível de compatibilidade de nós e muito mais. Para obter mais informações sobre os recursos do painel do Chef Automate e como usá-los, consulte a Documentação do Chef Automate.



> **Note**
>
> Para informações sobre como alterar a senha que você usa para fazer login no painel do Chef Automate, consulte Redefinição das credenciais do painel do Chef Automate.

Crie um AWS OpsWorks for Chef Automate servidor usando AWS CloudFormation

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks for Chef Automate permite que você execute um servidor Chef Automate em AWS. Você pode provisionar um servidor Chef Automate em cerca de 15 minutos.

A partir de 3 de maio de 2021, AWS OpsWorks for Chef Automate armazena alguns atributos do servidor Chef Automate em AWS Secrets Manager. Para obter mais informações, consulte Integração com AWS Secrets Manager.

O passo a passo a seguir ajuda você a criar um servidor em AWS OpsWorks for Chef Automate criando uma pilha em. AWS CloudFormation

Tópicos

- Pré-requisitos
- Criar um servidor Chef Automate no AWS CloudFormation

Pré-requisitos

Antes de criar um servidor Chef Automate, crie fora do AWS OpsWorks for Chef Automate os recursos que serão necessários para acessar e gerenciar o servidor Chef. Para obter mais informações, consulte Pré-requisitos na seção de Conceitos básicos deste guia.

Consulte a seção OpsWorks -CM da Referência do Modelo AWS CloudFormation do Guia do Usuário para saber mais sobre os valores suportados e necessários no AWS CloudFormation modelo que você usa para criar seu servidor.

Se você estiver criando um servidor que use um domínio personalizado, precisará de um domínio, um certificado e uma chave privada personalizados. Você deve especificar valores para todos esses três parâmetros em seu AWS CloudFormation modelo. Para obter mais informações sobre os requisitos dos `CustomPrivateKey` parâmetros `CustomDomainCustomCertificate`,, e, consulte CreateServera Referência da API AWS OpsWorks CM.

Crie um valor de senha para o atributo do mecanismo `CHEF_AUTOMATE_ADMIN_PASSWORD`. O comprimento da senha deve ter um mínimo de 8 e um máximo de 32 caracteres. A senha só pode conter letras, números e caracteres especiais (`!/@#$%^+=_`). Ela deve conter pelo menos uma letra minúscula, uma letra maiúscula, um número e um caractere especial. Você especifica essa senha em seu AWS CloudFormation modelo ou como o valor do `CHEF_AUTOMATE_ADMIN_PASSWORD` parâmetro ao criar sua pilha.

Gere um par de chaves RSA codificado em base64 antes de começar a criar um servidor Chef Automate em. AWS CloudFormation A chave pública do par é o valor

deCHEF_AUTOMATE_PIVOTAL_KEY, o Chef específico EngineAttributesda CreateServerAPI. Essa chave é fornecida como o valor de Parâmetros no AWS CloudFormation console ou no create-stack comando no AWS CLI. Para gerar essa chave, sugerimos os métodos a seguir.

- Em computadores baseados em Linux, você pode gerar essa chave executando o comando OpenSSL a seguir.

  ```
  openssl genrsa -out pivotal_key_file_name.pem 2048
  ```

  Em seguida, exporte a parte da chave pública RSA do par para um arquivo. A chave pública se torna o valor de CHEF_AUTOMATE_PIVOTAL_KEY.

  ```
  openssl rsa -in pivotal_key_file_name.pem -pubout -out public.pem -outform PEM
  ```

- Em computadores baseados em Windows, você pode usar o TTYgen utilitário Pu para gerar um par de chaves RSA codificado em base64. Para obter mais informações, consulte Pu TTYgen - Gerador de chaves para PuTTY no Windows em SSH.com.

Criar um servidor Chef Automate no AWS CloudFormation

Esta seção descreve como usar um AWS CloudFormation modelo para criar uma pilha que cria um AWS OpsWorks for Chef Automate servidor. Você pode fazer isso usando o AWS CloudFormation console ou AWS CLI o. Um AWS CloudFormation modelo de exemplo está disponível para você usar para criar uma pilha de AWS OpsWorks for Chef Automate servidores. Certifique-se de atualizar o modelo de exemplo com suas próprias informações de nome do servidor, perfis do IAM, perfil de instância, descrição do servidor, contagem de retenção de backup, opções de manutenção e tags opcionais. Se seu servidor usar um domínio personalizado, você deverá especificar valores para os CustomPrivateKey parâmetros CustomDomainCustomCertificate, e em seu AWS CloudFormation modelo. Você pode especificar os atributos CHEF_AUTOMATE_ADMIN_PASSWORD e o CHEF_AUTOMATE_PIVOTAL_KEY mecanismo e seus valores no AWS CloudFormation modelo, ou fornecer apenas os atributos e, em seguida, especificar valores para os atributos no assistente ou create-stack comando AWS CloudFormation Create Stack. Para obter mais informações sobre esses atributos, consulte the section called “Crie um servidor Chef Automate no AWS Management Console” na seção de Conceitos básicos deste guia.

Tópicos

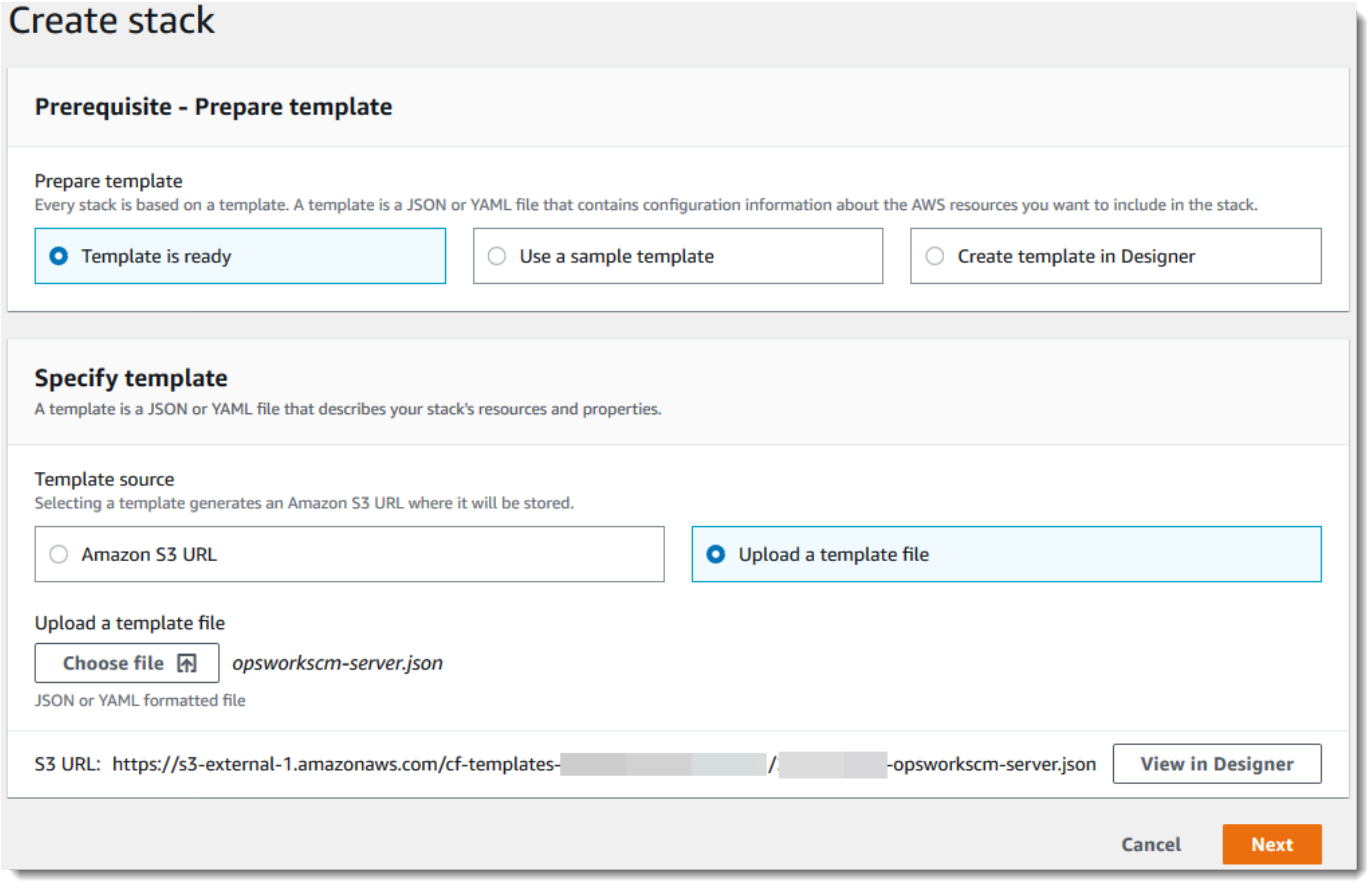
- Criar um servidor Chef Automate usando o AWS CloudFormation (console)

- Criar um servidor Chef Automate usando o AWS CloudFormation (CLI)

Criar um servidor Chef Automate usando o AWS CloudFormation (console)

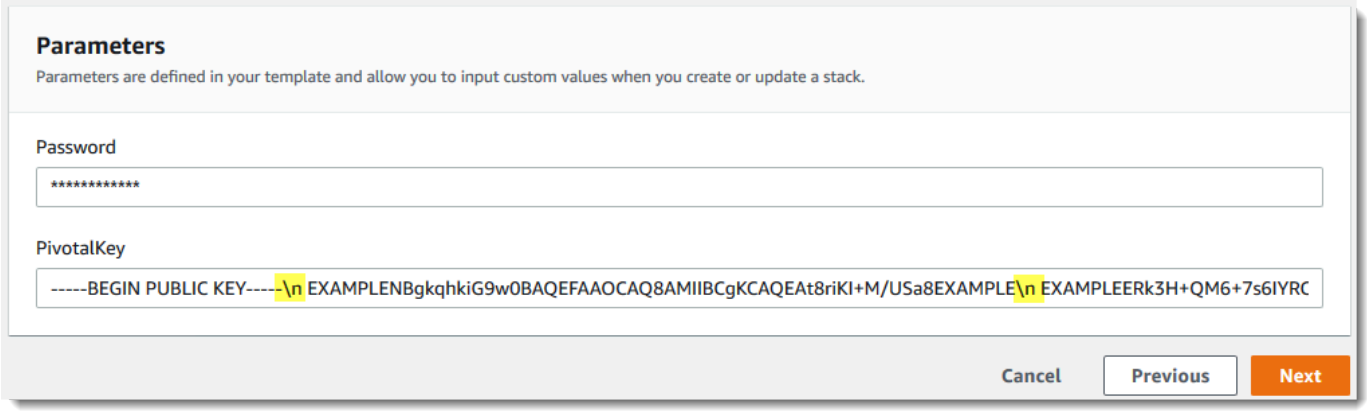
1. Faça login no AWS Management Console e abra o AWS CloudFormation console em https://console.aws.amazon.com/cloudformation.
2. Na página AWS CloudFormation inicial, escolha Criar pilha.
3. Em Pré-requisito - Preparar modelo, se você estiver usando o AWS CloudFormation modelo de exemplo, escolha O modelo está pronto.
4. Em Specify template (Especificar modelo), escolha a origem do seu modelo. Para este passo a passo, escolha Carregar um arquivo de modelo e carregue um AWS CloudFormation modelo que crie um servidor Chef Automate. Procure seu arquivo de modelo e escolha Next (Próximo).

   Um AWS CloudFormation modelo pode estar no formato YAML ou JSON. Um AWS CloudFormation modelo de exemplo está disponível para você usar; certifique-se de substituir os valores de exemplo pelos seus. Você pode usar o designer AWS CloudFormation de modelos para criar um novo modelo ou validar um existente. Para obter mais informações sobre como fazer isso, consulte Visão geral da interface do AWS CloudFormation Designer no Guia do usuário do AWS CloudFormation .



5. Na página Specify stack details (Especificar detalhes da pilha), insira um nome para a pilha. Ele não será o mesmo que o nome do servidor, apenas o nome de uma pilha. Na área Parameters (Parâmetros), cole os valores criados em the section called “Pré-requisitos”. Insira a senha em Password (Senha).

   Cole o conteúdo do arquivo de chave RSA em PivotalKey. No AWS CloudFormation console, você deve adicionar caracteres newline (**\n**) no final de cada linha do valor da chave principal, conforme mostrado na captura de tela a seguir. Escolha Próximo.



6. Na página Configurar opções da pilha, você poderá adicionar tags ao servidor que está criando com a pilha e escolher um perfil do IAM para a criação de recursos, caso ainda não tenha especificado um perfil do IAM a ser usada no modelo. Quando terminar de especificar as opções, selecione Next (Próximo). Para obter mais informações sobre opções avançadas, como acionadores de reversão, consulte Configurando opções de AWS CloudFormation pilha no Guia do usuário.AWS CloudFormation

7. Na página Analisar, examine suas escolhas. Quando estiver pronto para criar a pilha do servidor, selecione Create (Criar).

   Enquanto você espera para criar AWS CloudFormation a pilha, veja o status de criação da pilha. Se a criação da pilha falhar, analise as mensagens de erro mostradas no console para ajudá-lo a resolver os problemas. Para obter mais informações sobre como solucionar problemas de erros em pilhas do AWS CloudFormation , consulte Solução de problemas de erros no Guia do usuário do AWS CloudFormation .

   Quando a criação for concluída, o servidor Chef Automate estará disponível na página inicial do AWS OpsWorks for Chef Automate , com um status online. Gere um novo Starter Kit e novas credenciais do painel do Chef Automate na página Properties (Propriedades) do servidor. Depois que o servidor estiver online, o painel do Chef Automate estará disponível no domínio do servidor, em um URL no seguinte formato: `https://your_server_name-randomID.region.opsworks-cm.io`.

   > **Note**
   >
   > Se você especificou um domínio, certificado e chave privada personalizados para seu servidor, crie uma entrada CNAME na ferramenta de gerenciamento de DNS da sua empresa que mapeie seu domínio personalizado para o endpoint gerado AWS OpsWorks for Chef Automate automaticamente para o servidor. Não é possível gerenciar o servidor nem se conectar ao painel do Chef Automate para o servidor até que o endpoint gerado seja mapeado para o valor de domínio personalizado.
   > Para obter o valor do endpoint gerado, execute o seguinte AWS CLI comando depois que seu servidor estiver on-line:
   >
   > ```
   > aws opsworks describe-servers --server-name server_name
   > ```

Criar um servidor Chef Automate usando o AWS CloudFormation (CLI)

Se o seu computador local ainda não estiver executando o AWS CLI, faça o download e instale-o AWS CLI seguindo as instruções de instalação no Guia do usuário da interface de linha de comando da AWS. Esta seção não descreve todos os parâmetros que você pode usar com o comando create-stack. Para obter mais informações sobre os parâmetros do create-stack, consulte create-stack na Referência da AWS CLI .

1. Certifique-se de concluir o Pré-requisitos para criar um servidor AWS OpsWorks for Chef Automate .
2. Crie uma função de serviço e um perfil de instância.

   Encontre e copie as funções ARNs de serviço em sua conta.

   ```
   aws iam list-roles --path-prefix "/service-role/" --no-paginate
   ```

   Nos resultados do comando `list-roles`, procure pelas entradas da função de serviço e do perfil de instância que sejam semelhantes aos seguintes. Anote a função ARNs de serviço e o perfil da instância e adicione-os ao AWS CloudFormation modelo que você está usando para criar sua pilha de servidores.

   ```
   {
       "AssumeRolePolicyDocument": {
           "Version": "2012-10-17",
           "Statement": [
               {
                   "Action": "sts:AssumeRole",
                   "Effect": "Allow",
                   "Principal": {
                       "Service": "ec2.amazonaws.com"
                   }
               }
           ]
       },
       "RoleId": "AROZZZZZZZZZZQG6R22HC",
       "CreateDate": "2018-01-05T20:42:20Z",
       "RoleName": "aws-opsworks-cm-ec2-role",
       "Path": "/service-role/",
       "Arn": "arn:aws:iam::000000000000:role/service-role/aws-opsworks-cm-ec2-role"
   },
   {
   ```

```
    "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Action": "sts:AssumeRole",
                "Effect": "Allow",
                "Principal": {
                    "Service": "opsworks-cm.amazonaws.com"
                }
            }
        ]
    },
    "RoleId": "AROZZZZZZZZZZZZZZZ6QE",
    "CreateDate": "2018-01-05T20:42:20Z",
    "RoleName": "aws-opsworks-cm-service-role",
    "Path": "/service-role/",
    "Arn": "arn:aws:iam::000000000000:role/service-role/aws-opsworks-cm-service-
role"
}
```

3. Crie o AWS OpsWorks for Chef Automate servidor executando o create-stack comando novamente.

   - *stack_name*Substitua pelo nome da sua pilha. Este é o nome da AWS CloudFormation pilha, não do seu servidor Chef Automate. O nome do servidor Chef Automate é o valor de `ServerName` em seu AWS CloudFormation modelo.
   - *template*Substitua pelo caminho do arquivo de modelo e pela extensão *yaml or json* por `.yaml` ou `.json` conforme apropriado.
   - Os valores de `--parameters` correspondem aos EngineAttributesda CreateServerAPI. Para o Chef, os atributos do mecanismo fornecido pelo usuário para criar um servidor são `CHEF_AUTOMATE_PIVOTAL_KEY`, uma chave pública RSA codificada em base64 gerada usando utilitários descritos em the section called “Pré-requisitos” e `CHEF_AUTOMATE_ADMIN_PASSWORD`, uma senha criada por você que tem entre oito e 32 caracteres. Para obter mais informações sobre o `CHEF_AUTOMATE_ADMIN_PASSWORD`, consulte Crie um servidor Chef Automate usando o AWS CLI. Você pode fornecer um ponteiro para o arquivo PEM que contém a chave principal como o valor do parâmetro `PivotalKey`, conforme mostrado no exemplo. Se os valores para `CHEF_AUTOMATE_ADMIN_PASSWORD` e não `CHEF_AUTOMATE_PIVOTAL_KEY` estiverem especificados em seu modelo, você deverá fornecer os valores em seu AWS CLI comando.

```
aws cloudformation create-stack --stack-name stack_name
 --template-body file://template.yaml or json --parameters
 ParameterKey=PivotalKey,ParameterValue="base64_encoded_RSA_public_key_value"
```

Veja a seguir um exemplo que inclui amostras de valores para os atributos `CHEF_AUTOMATE_ADMIN_PASSWORD` e `CHEF_AUTOMATE_PIVOTAL_KEY`. Execute um comando semelhante se você não especificou valores para esses atributos em seu AWS CloudFormation modelo.

```
aws cloudformation create-stack --stack-name "OpsWorksCMChefServerStack"
 --template-body file://opsworkscm-server.yaml --parameters
 ParameterKey=PivotalKey,ParameterValue="$(openssl rsa -in "pivotalKey.pem" -
pubout)" ParameterKey=Password,ParameterValue="SuPer\$ecret890"
```

4. Quando a criação da pilha estiver concluída, abra a página Propriedades do novo servidor no AWS OpsWorks for Chef Automate console e baixe um kit inicial. Ao fazer download de um novo starter kit, a senha de administrador do painel do Chef Automate é redefinida.
5. Se o servidor usar um domínio personalizado, um certificado e uma chave privada, siga as etapas para configurar `knife.rb` no (Opcional) Configurar `knife` para trabalhar com um domínio personalizado e, depois, vá para a etapa 7.
6. Para usar comandos `knife` no novo servidor, atualize as configurações do arquivo de configuração `knife.rb` do Chef. Um exemplo do arquivo `knife.rb` está incluso no starter kit. O exemplo a seguir mostra como configurar `knife.rb` em um servidor que não usa um domínio personalizado. Se você estiver usando um domínio personalizado, consulte (Opcional) Configurar `knife` para trabalhar com um domínio personalizado para obter instruções de configuração para `knife`.

   - *`ENDPOINT`*Substitua pelo valor do endpoint do servidor. Isso faz parte da saída da operação de criação da pilha. É possível obter o endpoint executando o comando a seguir.

     ```
     aws cloudformation describe-stacks --stack-name stack_name
     ```

   - Substitua *`key_pair_file.pem`* na `client_key` configuração pelo nome do arquivo PEM que contém o `CHEF_AUTOMATE_PIVOTAL_KEY` que você usou para criar seu servidor.

     ```
     base_dir = File.join(File.dirname(File.expand_path(__FILE__)), '..')
     ```

```
log_level                :info
log_location             STDOUT
node_name                'pivotal'
client_key               File.join(base_dir, '.chef', 'key_pair_file.pem')
syntax_check_cache_path  File.join(base_dir, '.chef', 'syntax_check_cache')
cookbook_path            [File.join(base_dir, 'cookbooks')]

chef_server_url          'ENDPOINT/organizations/default'
ssl_ca_file              File.join(base_dir, '.chef', 'ca_certs', 'opsworks-cm-
ca-2020-root.pem')
trusted_certs_dir        File.join(base_dir, '.chef', 'ca_certs')
```

7. Quando o processo de criação do servidor for concluído, vá para the section called "Concluir a configuração e fazer o upload de livros de receitas". Se a criação da pilha falhar, analise as mensagens de erro mostradas no console para ajudá-lo a resolver os problemas. Para obter mais informações sobre como solucionar erros em AWS CloudFormation pilhas, consulte Solução de problemas no Guia do AWS CloudFormation usuário.

Atualizar um AWS OpsWorks for Chef Automate servidor para usar um domínio personalizado

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como atualizar um AWS OpsWorks for Chef Automate servidor existente para usar um domínio e um certificado personalizados usando um backup do servidor para criar um novo servidor. Basicamente, você está copiando um servidor AWS OpsWorks for Chef Automate 2.0 existente criando um novo servidor a partir de um backup e, em seguida, configurando o novo servidor para usar um domínio, certificado e chave privada personalizados.

Tópicos

- Pré-requisitos

- Limitações
- Atualizar um servidor para usar um domínio personalizado
- Consulte também

Pré-requisitos

A seguir estão os requisitos para atualizar um AWS OpsWorks for Chef Automate servidor existente para usar um domínio e um certificado personalizados.

- O servidor que você deseja atualizar (ou copiar) deve estar executando o Chef Automate 2.0.
- Decida qual backup você deseja usar para criar um novo servidor. É necessário ter pelo menos um backup de segurança disponível do servidor que você quer atualizar. Para obter mais informações sobre backups em AWS OpsWorks for Chef Automate, consulteFaça backup de um AWS OpsWorks for Chef Automate servidor.
- Prepare a função de serviço e o perfil da instância ARNs que você usou para criar o servidor existente que é a origem do seu backup.
- Certifique-se de que você esteja executando a versão mais atual da AWS CLI. Para obter mais informações sobre como atualizar suas AWS CLI ferramentas, consulte Instalando o AWS CLI no Guia do usuário da interface de linha de comando da AWS.

Limitações

Quando você atualiza um servidor existente criando um novo servidor a partir de um backup, o novo servidor não pode ser exatamente igual ao AWS OpsWorks for Chef Automate servidor existente.

- Você só pode concluir esse procedimento usando o AWS CLI ou um dos AWS SDKs. Não é possível criar um novo servidor de um backup utilizando o AWS Management Console.
- O novo servidor não pode usar o mesmo nome do servidor existente em uma conta e em uma região da AWS. O nome deve ser diferente do servidor existente que você usou como a origem do backup.
- Os nós que foram conectados ao servidor existente não são gerenciados pelo novo servidor. Você deve fazer um dos seguintes procedimentos.
  - Associe nós diferentes, porque os nós não podem ser gerenciados por mais de um servidor Chef Automate.

- Migre os nós do servidor existente (a origem do backup) para o novo servidor e o novo endpoint de domínio personalizado. Para obter mais informações sobre como migrar nós, consulte a documentação do Chef.

Atualizar um servidor para usar um domínio personalizado

Para atualizar um servidor Chef Automate 2.0 existente, faça uma cópia dele executando o comando `create-server`, adicionando parâmetros para especificar um backup, além de um domínio, um certificado e uma chave privada personalizados.

1. Se você não tiver uma função de serviço ou um perfil de instância ARNs disponível para especificar em seu `create-server` comando, siga as etapas de 1 a 5 Crie um servidor Chef Automate usando o AWS CLI para criar uma função de serviço e um perfil de instância que você possa usar.
2. Se ainda não tiver feito isso, localize o backup do servidor Chef Automate 2.0 existente no qual você deseja basear um novo servidor com um domínio personalizado. Execute o comando a seguir para mostrar informações sobre todos os AWS OpsWorks for Chef Automate backups em sua conta e em uma região. Anote o ID do backup que você deseja usar.

   ```
   aws opsworks-cm --region region name describe-backups
   ```

3. Crie o AWS OpsWorks for Chef Automate servidor executando o `create-server` comando.
   - O valor de `--engine` é `ChefAutomate`, de `--engine-model` é `Single` e de `--engine-version` é `12`.
   - O nome do servidor deve ser exclusivo em sua AWS conta, em cada região. Os nomes dos servidores devem começar com uma letra; letras, números ou hífens (-) são permitidos, até um máximo de 40 caracteres.
   - Use o ARN do perfil de instância e o ARN da função de serviço da etapa 1.
   - Os tipos de instância válidos são `m5.large`, `r5.xlarge` ou `r5.2xlarge`. Para obter mais informações sobre as especificações desses tipos de instância, consulte Tipos de instância no Guia EC2 do usuário da Amazon.
   - O parâmetro `--engine-attributes` é opcional; se você não especificar um dos valores ou ambos, o processo de criação do servidor gerará os valores para você. Se você adicionar `--engine-attributes`, especifique o valor da `CHEF_AUTOMATE_PIVOTAL_KEY` que você gerou na Etapa 2, uma `CHEF_AUTOMATE_ADMIN_PASSWORD` ou ambos.

Se você não definir um valor para `CHEF_AUTOMATE_ADMIN_PASSWORD`, uma senha será gerada e retornada como parte da resposta do comando `create-server`. Você também pode fazer download do starter kit novamente no console, o que fará com que a senha seja gerada novamente. O comprimento da senha deve ter um mínimo de 8 e um máximo de 32 caracteres. A senha só pode conter letras, números e caracteres especiais (`!/@#$%^+=_`). Ela deve conter pelo menos uma letra minúscula, uma letra maiúscula, um número e um caractere especial.

- Um par de chaves SSH é opcional, mas pode ajudá-lo a se conectar ao servidor do Chef Automate se você precisar redefinir a senha de administrador do painel do Chef Automate. Para obter mais informações sobre a criação de um par de chaves SSH, consulte Amazon EC2 Key Pairs no Amazon EC2 User Guide.
- Para usar um domínio personalizado, adicione os seguintes parâmetros ao comando. Caso contrário, o processo de criação do servidor Chef Automate gerará automaticamente um endpoint para você. Todos os três parâmetros são necessários para configurar um domínio personalizado. Para obter informações sobre requisitos adicionais para usar esses parâmetros, consulte CreateServera Referência da API AWS OpsWorks CM.
  - `--custom-domain` – um endpoint público opcional de um servidor, como `https://aws.my-company.com`.
  - `--custom-certificate` – um certificado HTTPS no formato PEM. O valor pode ser um certificado autoassinado único ou uma cadeia de certificados.
  - `--custom-private-key` – uma chave privada no formato PEM para a conexão com o servidor usando HTTPS. A chave privada não deve ser criptografada e não pode ser protegida por senha nem por frase secreta.
- É necessário fazer a manutenção semanal do sistema. Os valores válidos devem sempre ser especificados no seguinte formato: `DDD:HH:MM`. A hora especificada está em formato de Tempo Universal Coordenado (UTC). Se você não especificar um valor para `--preferred-maintenance-window`, o valor padrão será aleatório, um período de uma hora na terça, quarta ou sexta-feira.
- Os valores válidos para `--preferred-backup-window` devem ser especificados em um dos seguintes formatos: `HH:MM` para backups diários ou `DDD:HH:MM` para backups semanais. A hora é especificada em UTC. O valor padrão é uma hora de início aleatória estabelecida diariamente. Para cancelar os backups automáticos, adicione o parâmetro `--disable-automated-backup`.

- Para `--security-group-ids`, insira um ou mais grupos de segurança IDs, separados por um espaço.
- Em `--subnet-ids`, insira um ID de sub-rede.
- Em `--backup-id`, insira o ID do backup que você copiou na etapa 2.

```
aws opsworks-cm create-server --engine "ChefAutomate" --engine-model "Single"
 --engine-version "12" --server-name "server_name" --instance-profile-arn
 "instance_profile_ARN" --instance-type "instance_type" --engine-attributes
 '{"CHEF_AUTOMATE_PIVOTAL_KEY":"pivotal_key","CHEF_AUTOMATE_ADMIN_PASSWORD":"password"}'
 --key-pair "key_pair_name" --preferred-maintenance-window
 "ddd:hh:mm" --preferred-backup-window "ddd:hh:mm" --security-group-
ids security_group_id1 security_group_id2 --service-role-arn "service_role_ARN" --
subnet-ids subnet_ID --backup-id backup_ID
```

O exemplo a seguir cria um servidor Chef Automate que usa um domínio personalizado.

```
aws opsworks-cm create-server --engine "ChefAutomate" --engine-model "Single" --
engine-version "12" \
    --server-name "my-custom-domain-server" \
    --instance-profile-arn "arn:aws:iam::12345678912:instance-profile/aws-opsworks-
cm-ec2-role" \
    --instance-type "m5.large" \
    --engine-attributes
 '{"CHEF_AUTOMATE_PIVOTAL_KEY":"MZZE...Wobg","CHEF_AUTOMATE_ADMIN_PASSWORD":"zZZzDj2DLYXSZF
 \
    --custom-domain "my-chef-automate-server.my-corp.com" \
    --custom-certificate "-----BEGIN CERTIFICATE----- EXAMPLEqEXAMPLE== -----END
 CERTIFICATE-----" \
    --custom-private-key "-----BEGIN RSA PRIVATE KEY----- EXAMPLEqEXAMPLE= -----END
 RSA PRIVATE KEY-----" \
    --key-pair "amazon-test" \
    --preferred-maintenance-window "Mon:08:00" \
    --preferred-backup-window "Sun:02:00" \
    --security-group-ids sg-b00000001 sg-b0000008 \
    --service-role-arn "arn:aws:iam::12345678912:role/service-role/aws-opsworks-cm-
service-role" \
    --subnet-ids subnet-300aaa00 \
    --backup-id MyChefServer-20191004122143125
```

4. AWS OpsWorks for Chef Automate leva cerca de 15 minutos para criar um novo servidor. Na saída do comando `create-server`, copie o valor do atributo `Endpoint`. Veja um exemplo a seguir.

```
"Endpoint": "automate-07-exampleexample.opsworks-cm.us-east-1.amazonaws.com"
```

   Não descarte a saída do comando `create-server` ou feche a sessão de shell, pois a saída pode conter informações importantes que não serão exibidas novamente. Para obter as senhas e o starter kit dos resultados do comando `create-server`, vá para a próxima etapa.

5. Se você optou por AWS OpsWorks for Chef Automate gerar uma chave e uma senha para você, poderá extraí-las dos `create-server` resultados em formatos utilizáveis usando um processador JSON, como jq. Após instalar o jq, você pode executar os comandos a seguir para extrair a chave principal, a senha de administrador do painel do Chef Automate e o starter kit. Se você não forneceu sua própria chave principal e senha na etapa 3, salve a chave principal e a senha de administrador extraídas em locais seguros e convenientes.

```
#Get the Chef password:
cat resp.json | jq -r '.Server.EngineAttributes[] | select(.Name ==
 "CHEF_AUTOMATE_ADMIN_PASSWORD") | .Value'

#Get the Chef Pivotal Key:
cat resp.json | jq -r '.Server.EngineAttributes[] | select(.Name ==
 "CHEF_AUTOMATE_PIVOTAL_KEY") | .Value'

#Get the Chef Starter Kit:
cat resp.json | jq -r '.Server.EngineAttributes[] | select(.Name ==
 "CHEF_STARTER_KIT") | .Value' | base64 -D > starterkit.zip
```

6. Opcionalmente, se você não extraiu o kit inicial dos resultados do `create-server` comando, você pode baixar um novo kit inicial na página Propriedades do servidor no console. AWS OpsWorks for Chef Automate Ao fazer download de um novo starter kit, a senha de administrador do painel do Chef Automate é redefinida.

7. Crie uma entrada CNAME na ferramenta de gerenciamento de DNS da sua empresa para direcionar seu domínio personalizado para o AWS OpsWorks for Chef Automate endpoint que você copiou na etapa 4. Você não poderá acessar nem se conectar ao servidor se não concluir esta etapa.

8. Quando o processo de criação do servidor for concluído, vá para the section called “Concluir a configuração e fazer o upload de livros de receitas”.

Consulte também

- Crie um servidor Chef Automate usando o AWS CLI
- Restaurar um AWS OpsWorks for Chef Automate servidor a partir de um backup
- CreateServerna referência da API AWS OpsWorks CM
- create-server na AWS CLI Command Reference

Regenere o kit inicial para um servidor AWS OpsWorks for Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O kit inicial AWS OpsWorks for Chef Automate contém um arquivo README com exemplos, um arquivo de `knife.rb` configuração e uma chave privada para o usuário principal ou principal. Um novo par de chaves será gerado (e a antiga chave será redefinida) sempre que você fizer o download do starter kit. Você pode regenerar o kit inicial para um AWS OpsWorks for Chef Automate servidor de duas maneiras:

- No AWS OpsWorks console, no menu Ações da página de detalhes de um AWS OpsWorks for Chef Automate servidor. Você será solicitado a confirmar se deseja regenerar e redefinir a chave central antiga.
- Ao executar comandos no AWS CLI.

Para obter mais informações sobre como usar a interface do starter kit, consulte Configurar o servidor do Chef usando o Starter Kit.

Regenere o kit AWS OpsWorks for Chef Automate inicial com o AWS CLI

> **Note**
>
> Ao regenerar o starter kit, você também regenera e redefine o par de chaves de autenticação do seu servidor Chef Automate e exclui o par de chaves atual.

Regenere o starter kit executando o comando update-server-engine-attributes. Em uma AWS CLI sessão, execute o comando a seguir. Especifique o nome do seu servidor como o valor de `--server-name`. Para definir sua própria chave pública como valor de `CHEF_AUTOMATE_PIVOTAL_KEY`, especifique o valor da chave pública em `--attribute-value`. Caso contrário, defina `--attribute-value` como nulo.

```
aws opsworks-cm update-server-engine-attributes \
   --server-name server_name \
   --attribute-name "CHEF_AUTOMATE_PIVOTAL_KEY" \
   --attribute-value your_public_key
```

O comando a seguir é um exemplo que especifica um valor de chave pública que o administrador do servidor deseja usar.

```
aws opsworks-cm update-server-engine-attributes \
   --server-name your-test-server \
   --attribute-name "CHEF_AUTOMATE_PIVOTAL_KEY" \
   --attribute-value "-----BEGIN PUBLIC KEY-----ExamplePublicKey-----END PUBLIC
 KEY-----"
```

O comando a seguir é um exemplo que permite AWS OpsWorks for Chef Automate regenerar a chave pública.

```
aws opsworks-cm update-server-engine-attributes \
   --server-name your-test-server \
   --attribute-name "CHEF_AUTOMATE_PIVOTAL_KEY" \
   --attribute-value null
```

A saída desse comando são informações sobre o servidor e um arquivo ZIP codificado em base64. O arquivo ZIP contém um starter kit do Chef, que inclui um LEIAME, um arquivo de configuração

e a chave privada RSA necessária. Salve esse arquivo, descompacte-o e, em seguida, vá para o diretório em que você descompactou o conteúdo do arquivo. Nesse diretório, você pode executar comandos `knife`.

Trabalhando com tags em AWS OpsWorks for Chef Automate recursos

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As tags são palavras ou frases que atuam como metadados para identificar e organizar os recursos da AWS. Em AWS OpsWorks for Chef Automate, um recurso pode ter até 50 tags aplicadas pelo usuário. Cada tag consiste em uma chave e em um valor opcional. É possível aplicar tags aos seguintes recursos no AWS OpsWorks for Chef Automate:

- AWS OpsWorks for Chef Automate servidores
- Backups de AWS OpsWorks for Chef Automate servidores

As tags nos AWS recursos podem ajudar você a monitorar custos, controlar o acesso aos recursos, agrupar recursos para automatizar tarefas ou organizar recursos por finalidade ou estágio do ciclo de vida. Para obter mais informações sobre os benefícios das tags, consulte Estratégias de marcação da AWS no AWS Answers e Usar tags de alocação de custos no Guia do usuário do Gerenciamento de Faturamento e Custos da AWS .

Para usar tags para controlar o acesso a AWS OpsWorks for Chef Automate servidores ou backups, você cria ou edita declarações de política no AWS Identity and Access Management (IAM). Para obter mais informações, consulte Controlar o acesso aos recursos da AWS usando tags de recurso no Guia do usuário do AWS Identity and Access Management .

Quando você aplica tags a um AWS OpsWorks for Chef Automate servidor, as tags também são aplicadas aos backups do servidor, ao bucket Amazon S3 que armazena os backups, à EC2

instância Amazon do servidor, aos segredos do servidor que estão armazenados e ao endereço IP elástico usado pelo servidor. AWS Secrets Manager As tags não são propagadas para a AWS CloudFormation pilha AWS OpsWorks usada para criar seu servidor.

Tópicos

- Como as tags funcionam em AWS OpsWorks for Chef Automate
- Adicionar e gerenciar tags no AWS OpsWorks for Chef Automate (console)
- Adicionar e gerenciar tags em AWS OpsWorks for Chef Automate (CLI)
- Consulte também

Como as tags funcionam em AWS OpsWorks for Chef Automate

Nesta versão, é possível adicionar e gerenciar tags usando a API do AWS OpsWorks CM ou o AWS Management Console. AWS OpsWorks O CM também tenta adicionar tags que você adiciona a um servidor aos AWS recursos associados ao servidor, incluindo a EC2 instância, os segredos no Secrets Manager, o endereço IP elástico, o grupo de segurança, o bucket do S3 e os backups. A tabela a seguir fornece uma visão geral de como adicionar e gerenciar tags no AWS OpsWorks for Chef Automate.

Ação	O que usar
Adicione tags a um novo AWS OpsWorks for Chef Automate servidor ou backup que você está criando manualmente.	• Escolha Create Chef Automate server (Criar servidor Chef Automate) e adicione tags na página Configure advanced settings (Definir configurações avançadas). • Escolha Create backup (Criar backup) na página Backups para um servidor existente e adicione tags na página Create a backup of your Chef Automate 2 server (Criar um backup do servidor Chef Automate 2). • Adicione um parâmetro `Tags` aos comandos `CreateServer` ou `CreateBackup`.
Visualize tags em um recurso.	• Na página de detalhes do servidor, escolha Tags no painel de navegação.

Ação	O que usar
	• Na página Backups do servidor, selecione um backup e escolha Edit backup (Editar backup). • Execute o comando `ListTagsForResource`.
Adicione tags a um AWS OpsWorks for Chef Automate servidor ou backup existente, independentemente de o backup ter sido criado manual ou automaticamente.	• Na página de detalhes do servidor, escolha Tags no painel de navegação e escolha Edit (Editar). • Na página Backups do servidor, selecione um backup e escolha Edit backup (Editar backup). • Execute o comando `TagResource`.
Exclua tags de um recurso.	• Na página de detalhes do servidor, escolha Tags no painel de navegação e escolha Edit (Editar). Escolha X ao lado das tags que deseja excluir. • Na página Backups do servidor, selecione um backup e escolha Edit backup (Editar backup). Escolha X ao lado das tags que deseja excluir. • Execute o comando `UntagResource`.

As respostas `DescribeServers` e `DescribeBackups` não incluem informações de tag. Para mostrar tags, use a API `ListTagsForResource`.

Adicionar e gerenciar tags no AWS OpsWorks for Chef Automate (console)

Os procedimentos nesta seção são realizados na AWS Management Console.

Se você adicionar tags, uma chave de tag não poderá ficar vazia. A chave pode ter, no máximo, 127 caracteres e pode conter somente letras, números ou separadores Unicode, ou os seguintes caracteres especiais: `+ - = . _ : / @`. Um valor de tag é opcional. É possível adicionar uma tag que tenha uma chave, mas nenhum valor. O valor pode ter, no máximo, 255 caracteres e conter

somente letras, números ou separadores Unicode, ou os seguintes caracteres especiais: `+ - = . _ : / @`

Tópicos

- Adicionar tags a um novo AWS OpsWorks for Chef Automate servidor (console)
- Adicionar tags a um novo backup (console)
- Adicionar ou visualizar tags em um servidor existente (console)
- Adicionar ou visualizar tags em um backup existente (console)
- Excluir tags de um servidor (console)
- Excluir tags de um backup (console)

Adicionar tags a um novo AWS OpsWorks for Chef Automate servidor (console)

1. Certifique-se de preencher todos os pré-requisitos para criar um servidor. AWS OpsWorks for Chef Automate
2. Siga as etapas de 1 a 10 em Criar um servidor Chef Automate.
3. Depois que especificar as configurações de backup automatizado, adicione as tags na área Tags da página Configure advanced settings (Definir configurações avançadas). É possível adicionar um máximo de 50 tags. Quando terminar de adicionar tags, escolha Próximo.
4. Vá para a etapa 13 de Criar um servidor Chef Automate e revise as configurações escolhidas para o novo servidor.

Adicionar tags a um novo backup (console)

1. Na página AWS OpsWorks for Chef Automate inicial, escolha um servidor Chef Automate existente.
2. Na página de detalhes do servidor, escolha Backups no painel de navegação.
3. Na página Backups, escolha Create backup (Criar backup).
4. Adicione as tags. Escolha Create (Criar) quando terminar de adicionar as tags.

Adicionar ou visualizar tags em um servidor existente (console)

1. Na página AWS OpsWorks for Chef Automate inicial, escolha um servidor Chef Automate existente para abrir sua página de detalhes.

2. Escolha Tags no painel de navegação ou, na parte inferior da página de detalhes, escolha View all tags (Visualizar todas as tags).
3. Na página Tags, escolha Edit (Editar).
4. Adicione ou edite as tags no servidor. Escolha Salvar quando terminar.

> **Note**
>
> Esteja ciente de que alterar as tags em seu servidor Chef Automate também altera as tags nos recursos associados ao servidor, como EC2 instância, endereço IP elástico, grupo de segurança, bucket S3 e backups.

Adicionar ou visualizar tags em um backup existente (console)

1. Na página AWS OpsWorks for Chef Automate inicial, escolha um servidor Chef Automate existente para abrir sua página de detalhes.
2. Escolha Backups no painel de navegação ou, na área Recent backups (Backups recentes) da página de detalhes, escolha View all backups (Visualizar todos os backups).
3. Na página Backups, escolha um backup para gerenciar e, depois, escolha Edit backup (Editar backup).
4. Adicione ou edite as tags no backup. Escolha Update (Atualizar) quando terminar.

Excluir tags de um servidor (console)

1. Na página AWS OpsWorks for Chef Automate inicial, escolha um servidor Chef Automate existente para abrir sua página de detalhes.
2. Escolha Tags no painel de navegação ou, na parte inferior da página de detalhes, escolha View all tags (Visualizar todas as tags).
3. Na página Tags, escolha Edit (Editar).
4. Escolha X ao lado de uma tag para excluí-la. Escolha Salvar quando terminar.

> **Note**
>
> Esteja ciente de que alterar as tags em seu servidor Chef Automate também altera as tags nos recursos associados ao servidor, como EC2 instância, endereço IP elástico, grupo de segurança, bucket S3 e backups.

Excluir tags de um backup (console)

1. Na página AWS OpsWorks for Chef Automate inicial, escolha um servidor Chef Automate existente para abrir sua página de detalhes.
2. Escolha Backups no painel de navegação ou, na área Recent backups (Backups recentes) da página de detalhes, escolha View all backups (Visualizar todos os backups).
3. Na página Backups, escolha um backup para gerenciar e, depois, escolha Edit backup (Editar backup).
4. Escolha X ao lado de uma tag para excluí-la. Escolha Update (Atualizar) quando terminar.

Adicionar e gerenciar tags em AWS OpsWorks for Chef Automate (CLI)

Os procedimentos nesta seção são realizados na AWS CLI. Verifique se você está executando a versão mais recente do AWS CLI antes de começar a trabalhar com tags. Para obter mais informações sobre como instalar ou atualizar o AWS CLI, consulte Instalando o AWS CLI no Guia AWS Command Line Interface do Usuário.

Se você adicionar tags, uma chave de tag não poderá ficar vazia. A chave pode ter, no máximo, 127 caracteres e pode conter somente letras, números ou separadores Unicode, ou os seguintes caracteres especiais: `+ - = . _ : / @`. Um valor de tag é opcional. É possível adicionar uma tag que tenha uma chave, mas nenhum valor. O valor pode ter, no máximo, 255 caracteres e conter somente letras, números ou separadores Unicode, ou os seguintes caracteres especiais: `+ - = . _ : / @`

Tópicos

- Adicionar tags a um novo AWS OpsWorks for Chef Automate servidor (CLI)
- Adicionar tags a um novo backup (CLI)
- Adicionar tags a backups ou servidores existentes (CLI)

- Listar tags de recurso
- Excluir tags de um recurso

Adicionar tags a um novo AWS OpsWorks for Chef Automate servidor (CLI)

Você pode usar o AWS CLI para adicionar tags ao criar um AWS OpsWorks for Chef Automate servidor. Este procedimento não descreve na íntegra como criar um servidor. Para obter informações detalhadas sobre como criar um AWS OpsWorks for Chef Automate servidor usando o, AWS CLI consulte Crie um servidor Chef Automate usando o AWS CLI neste guia. É possível adicionar até 50 tags a um servidor.

1. Certifique-se de preencher todos os pré-requisitos para criar um servidor. AWS OpsWorks for Chef Automate
2. Conclua as etapas 1 a 5 de Crie um servidor Chef Automate usando o AWS CLI.
3. Na etapa 6, ao executar o comando `create-server`, adicione o parâmetro `--tags` ao comando, conforme mostrado no exemplo a seguir.

   ```
   aws opsworks-cm create-server ... --tags Key=Key1,Value=Value1
    Key=Key2,Value=Value2
   ```

   Veja a seguir um exemplo que mostra somente a parte de tags do comando `create-server`.

   ```
   aws opsworks-cm create-server ... --tags Key=Stage,Value=Production
    Key=Department,Value=Marketing
   ```

4. Conclua as etapas restantes em Crie um servidor Chef Automate usando o AWS CLI. Para verificar se as tags foram adicionadas ao novo servidor, siga as etapas em Listar tags de recurso neste tópico.

Adicionar tags a um novo backup (CLI)

Você pode usar o AWS CLI para adicionar tags ao criar um novo backup manual de um AWS OpsWorks for Chef Automate servidor. Este procedimento não descreve na íntegra como criar um backup manual. Para obter informações detalhadas sobre como criar um backup manual, consulte “Para realizar um backup manual no AWS CLI" emFaça backup de um AWS OpsWorks for Chef Automate servidor. É possível adicionar até 50 tags a um backup. Se um servidor tiver tags, novos backups serão marcados automaticamente com as tags do servidor.

Por padrão, quando você cria um novo AWS OpsWorks for Chef Automate servidor, os backups automatizados são habilitados. É possível adicionar tags a um backup automatizado executando o comando `tag-resource`, descrito em Adicionar tags a backups ou servidores existentes (CLI) neste tópico.

- Para adicionar tags a um backup manual à medida que cria o backup, execute o comando a seguir. Somente a parte de tags do comando é exibida. Para obter um exemplo do comando `create-backup` completo, consulte "Como executar um backup manual na AWS CLI" em Faça backup de um AWS OpsWorks for Chef Automate servidor.

  ```
  aws opsworks-cm create-backup ... --tags Key=Key1,Value=Value1
   Key=Key2,Value=Value2
  ```

  O exemplo a seguir mostra somente a parte de tags do comando `create-backup`.

  ```
  aws opsworks-cm create-backup ... --tags Key=Stage,Value=Production
   Key=Department,Value=Marketing
  ```

Adicionar tags a backups ou servidores existentes (CLI)

É possível executar o comando `tag-resource` para adicionar tags a servidores do AWS OpsWorks for Chef Automate ou backups existentes (se os backups foram criados manual ou automaticamente). Especifique o número de recurso da Amazon (ARN) de um recurso de destino para adicionar tags a ele.

1. Para obter o ARN do recurso ao qual deseja aplicar tags:

   - Para um servidor, execute `describe-servers --server-name` *`server_name`*. Os resultados do comando mostram o ARN do servidor.
   - Para um backup, execute `describe-backups --backup-id` *`backup_ID`*. Os resultados do comando mostram o ARN do backup. Você também pode executar `describe-backups --server-name` *`server_name`* para mostrar informações sobre todos os backups de um AWS OpsWorks for Chef Automate servidor específico.

   O exemplo a seguir mostra somente o `ServerArn` nos resultados de um comando `describe-servers --server-name opsworks-cm-test`. O valor `ServerArn` é adicionado a um comando `tag-resource` para adicionar tags ao servidor.

```
{
    "Servers": [
        {
            ...
            "ServerArn": "arn:aws:opsworks-cm:us-west-2:123456789012:server/
opsworks-cm-test/EXAMPLEd-66b0-4196-8274-d1a2bEXAMPLE"
        }
    ]
}
```

2. Execute o comando `tag-resource` com o ARN retornado na etapa 1.

```
aws opsworks-cm tag-resource --resource-arn "server_or_backup_ARN" --tags
 Key=Key1,Value=Value1 Key=Key2,Value=Value2
```

Veja um exemplo a seguir.

```
aws opsworks-cm tag-resource --resource-arn "arn:aws:opsworks-cm:us-
west-2:123456789012:server/opsworks-cm-test/EXAMPLEd-66b0-4196-8274-d1a2bEXAMPLE"
 --tags Key=Stage,Value=Production Key=Department,Value=Marketing
```

3. Para verificar se as tags foram adicionadas com êxito, vá para o próximo procedimento, Listar tags de recurso.

Listar tags de recurso

Você pode executar o `list-tags-for-resource` comando para mostrar as tags que estão anexadas aos AWS OpsWorks for Chef Automate servidores ou backups. Especifique o ARN de um recurso de destino para exibir suas tags.

1. Para obter o ARN do recurso para o qual deseja listar tags:

   - Para um servidor, execute `describe-servers --server-name` *`server_name`*. Os resultados do comando mostram o ARN do servidor.
   - Para um backup, execute `describe-backups --backup-id` *`backup_ID`*. Os resultados do comando mostram o ARN do backup. Você também pode executar `describe-backups --server-name` *`server_name`* para mostrar informações sobre todos os backups de um AWS OpsWorks for Chef Automate servidor específico.

2. Execute o comando `list-tags-for-resource` com o ARN retornado na etapa 1.

```
aws opsworks-cm list-tags-for-resource --resource-arn "server_or_backup_ARN"
```

Veja um exemplo a seguir.

```
aws opsworks-cm tag-resource --resource-arn "arn:aws:opsworks-cm:us-
west-2:123456789012:server/opsworks-cm-test/EXAMPLEd-66b0-4196-8274-d1a2bEXAMPLE"
```

Se houver tags no recurso, o comando retornará resultados como os seguintes.

```
{
    "Tags": [
        {
            "Key": "Stage",
            "Value": "Production"
        },
        {
            "Key": "Department",
            "Value": "Marketing"
        }
    ]
}
```

Excluir tags de um recurso

É possível executar o comando `untag-resource` para excluir tags de servidores do AWS OpsWorks for Chef Automate ou de backups. Se o recurso for excluído, as tags no recurso também serão excluídas. Especifique o número de recurso da Amazon (ARN) de um recurso de destino para remover as tags dele.

1. Para obter o ARN do recurso do qual deseja remover tags:

   - Para um servidor, execute `describe-servers --server-name` *`server_name`*. Os resultados do comando mostram o ARN do servidor.
   - Para um backup, execute `describe-backups --backup-id` *`backup_ID`*. Os resultados do comando mostram o ARN do backup. Você também pode executar `describe-backups --server-name` *`server_name`* para mostrar informações sobre todos os backups de um AWS OpsWorks for Chef Automate servidor específico.

2. Execute o comando `untag-resource` com o ARN retornado na etapa 1. Especifique somente as tags que deseja excluir.

```
aws opsworks-cm untag-resource --resource-arn "server_or_backup_ARN" --tags
 Key=Key1,Value=Value1 Key=Key2,Value=Value2
```

   Neste exemplo, o comando `untag-resource` remove somente a tag com uma chave de `Stage` e um valor de `Production`.

```
aws opsworks-cm untag-resource --resource-arn "arn:aws:opsworks-cm:us-
west-2:123456789012:server/opsworks-cm-test/EXAMPLEd-66b0-4196-8274-d1a2bEXAMPLE"
 --tags Key=Stage,Value=Production
```

3. Para verificar se as tags foram excluídas com êxito, siga as etapas em Listar tags de recurso neste tópico.

Consulte também

- Crie um servidor Chef Automate usando o AWS CLI
- Faça backup de um AWS OpsWorks for Chef Automate servidor
- Estratégias de marcação da AWS
- Controlando o acesso aos AWS recursos usando tags de recursos no Guia AWS Identity and Access Management do usuário
- Usar tags de alocação de custos no Guia do usuário do Gerenciamento de Faturamento e Custos da AWS .
- CreateBackup na Referência de API do AWS OpsWorks CM
- CreateServer na Referência de API do AWS OpsWorks CM
- TagResource na Referência de API do AWS OpsWorks CM
- ListTagsForResource na Referência de API do AWS OpsWorks CM
- UntagResource na Referência de API do AWS OpsWorks CM

Fazer o backup e a restauração de um AWS OpsWorks for Chef Automate servidor

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como fazer o backup e a restauração de um AWS OpsWorks for Chef Automate servidor e como excluir backups.

Tópicos

- Faça backup de um AWS OpsWorks for Chef Automate servidor
- Restaurar um AWS OpsWorks for Chef Automate servidor a partir de um backup

Faça backup de um AWS OpsWorks for Chef Automate servidor

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode definir um backup recorrente diário ou semanal de um AWS OpsWorks for Chef Automate servidor e fazer com que o serviço armazene os backups no Amazon Simple Storage Service (Amazon S3) em seu nome. Como alternativa, você pode fazer backups manuais sob demanda.

Como os backups são armazenados no Amazon S3, há a cobrança de taxas adicionais. É possível definir um período de retenção de backups de até 30 gerações. Você pode enviar uma solicitação de

serviço para alterar esse limite usando os canais de AWS suporte. O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

É possível adicionar tags aos backups de um AWS OpsWorks for Chef Automate servidor. Se você adicionou tags a um servidor AWS OpsWorks for Chef Automate , os backups automatizados do servidor herdarão essas tags. Para obter mais informações sobre como adicionar e gerenciar tags em backups, consulte Trabalhando com tags em AWS OpsWorks for Chef Automate recursos neste guia.

Tópicos

- Backups automatizados
- Backups manuais
- Excluir backups

Backups automatizados

Quando você configura o AWS OpsWorks for Chef Automate servidor é possível escolher backups manuais ou automáticos. AWS OpsWorks for Chef Automate inicia os backups automáticos na hora e no dia que você escolher na seção Backup automático da página de configurações Configurar opções avançadas de Configuração. Após o servidor ficar online, você poderá alterar as configurações de backup executando as seguintes etapas, no bloco do servidor na página inicial dos servidores do Chef Automate ou na página Properties do servidor.

Para alterar configurações de backup automático

1. No menu Actions do bloco do servidor na página inicial Chef servers, escolha Change settings
2. Para desativar backups automatizados, escolha No para a opção Enable automated backups. Salve as alterações; não é necessário ir para a próxima etapa.
3. Na seção Automated Backup, altere a frequência, o horário de início ou as gerações que serão mantidas. Salve as alterações.

Backups manuais

Você pode iniciar um backup manual a qualquer momento no AWS Management Console, ou executando o comando AWS CLI create-backup. Os backups manuais não são incluídos no limite

máximo de 30 gerações de backups automáticos que estão armazenados; no máximo 10 backups manuais são armazenados e devem ser excluídos manualmente do Amazon S3.

É possível adicionar tags ao criar um backup manual de um AWS OpsWorks for Chef Automate servidor. Para obter mais informações sobre como adicionar tags ao criar um backup manual, consulte Adicionar tags a um novo backup (CLI).

Para realizar um backup manual no AWS Management Console

1. Na página Chef Automate servers, selecione o servidor para o qual você deseja fazer backup.
2. Na página de propriedades do servidor, no painel de navegação esquerdo, selecione Backups.
3. Escolha Create backup.
4. O backup manual é finalizado quando a página mostra uma marca de seleção verde na coluna Status do backup.

Para realizar um backup manual no AWS CLI

- Para iniciar um backup manual, execute o AWS CLI comando a seguir.

```
aws opsworks-cm --region region name create-backup --server-name "Chef server name"
 --description "optional descriptive string"
```

Excluir backups

A exclusão de um backup permanentemente o exclui do bucket do S3 em que os backups estão armazenados.

Para excluir um backup no. AWS Management Console

1. Na página Chef Automate servers, selecione o servidor para o qual você deseja fazer backup.
2. Na página de propriedades do servidor, no painel de navegação esquerdo, selecione Backups.
3. Escolha o backup que você deseja excluir e, em seguida, selecione Delete backup. Você pode selecionar somente um backup por vez.
4. Quando você for solicitado a confirmar a exclusão, marque a caixa de seleção Delete the backup, which is stored in an S3 bucket e, em seguida, escolha Yes, Delete.

Para excluir um backup no. AWS CLI

- Para excluir um backup, execute o AWS CLI comando a seguir, `--backup-id` substituindo pelo ID do backup que você deseja excluir. O backup IDs está no formato*ServerName-yyyyMMddHHmmssSSS*. Por exemplo, .**`test-chef-server-20171218132604388`**

```
aws opsworks-cm --region region name delete-backup --backup-id ServerName-
yyyyMMddHHmmssSSS
```

Restaurar um AWS OpsWorks for Chef Automate servidor a partir de um backup

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Após navegar pelos backups disponíveis, você poderá escolher um ponto no tempo para restaurar o AWS OpsWorks for Chef Automate servidor. Os backups do servidor contêm apenas os dados persistentes do software de gerenciamento de configuração (livros de receitas, nós registrados, etc). Realizar uma restauração in-loco de um servidor (ou seja, restaurar o AWS OpsWorks for Chef Automate servidor existente para uma nova EC2 instância) registrará novamente os nós que foram registrados no momento do backup usado para restaurar o servidor e alternará o tráfego para a nova instância se a restauração for bem-sucedida e o estado do servidor restaurado será. AWS OpsWorks for Chef Automate `Healthy` Restaurar para um servidor do AWS OpsWorks for Chef Automate recém-criado não mantém conexões de nó. A restauração de um servidor não atualiza versões secundárias de software do Chef; ela aplica as mesmas versões do Chef e dados de gerenciamento de configuração que estão disponíveis no backup que você escolher.

A restauração de um servidor normalmente leva mais tempo do que a criação de um novo servidor; o tempo depende do tamanho do backup escolhido. Após a conclusão da restauração, a EC2 instância antiga permanece no `Stopped` estado `Running` ou, mas apenas temporariamente. Ela será encerrada por fim.

Nesta versão, você pode usar o AWS CLI para restaurar um servidor Chef em AWS OpsWorks for Chef Automate.

> **Note**
>
> Você também pode executar o comando restore-server para alterar o tipo de instância atual ou para restaurar ou definir a chave SSH se ela for perdida ou comprometida.

Para restaurar um servidor a partir de um backup

1. No AWS CLI, execute o comando a seguir para retornar uma lista dos backups disponíveis e seus IDs. Anote o ID do backup que você deseja usar. O backup IDs está no formato*myServerName-yyyyMMddHHmmssSSS*.

   ```
   aws opsworks-cm --region region name describe-backups
   ```

2. Execute o seguinte comando:

   ```
   aws opsworks-cm --region region name restore-server --backup-id "myServerName-
   yyyyMMddHHmmssSSS" --instance-type "Type of instance" --key-pair "name of your EC2
    key pair" --server-name "name of Chef server"
   ```

   Veja um exemplo a seguir.

   ```
   aws opsworks-cm --region us-west-2 restore-server --backup-id
    "MyChefServer-20161120122143125" --server-name "MyChefServer"
   ```

3. Aguarde até que a restauração esteja concluída.

Manutenção do sistema em AWS OpsWorks for Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar

em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A manutenção obrigatória do sistema garante que as versões secundárias mais recentes do Chef Server e do Chef Automate Server, incluindo atualizações de segurança, estejam sempre em execução em um AWS OpsWorks for Chef Automate servidor. É necessário fazer a manutenção do sistema pelo menos uma vez por semana. Ao usar o AWS CLI, você pode configurar a manutenção automática diária, se desejar. Você também pode usar o AWS CLI para realizar a manutenção do sistema sob demanda, além da manutenção programada do sistema.

Quando novas versões secundárias do software do Chef forem disponibilizadas, a manutenção do sistema é projetada para atualizar a versão secundária do Chef Automate e do Chef Server automaticamente no servidor, assim que ela passar nos testes da AWS. A AWS realiza testes extensivos para verificar se as atualizações do Chef estão prontas para produção e não interrompem os ambientes existentes dos clientes, portanto, pode haver atrasos entre os lançamentos do software Chef e sua disponibilidade para aplicação e os existentes nos servidores do Chef Automate. OpsWorks Para atualizar versões secundárias de software do Chef disponíveis sob demanda, consulte Iniciar a manutenção do sistema sob demanda neste tópico.

A manutenção do sistema inicia uma nova instância a partir de um backup que é executado como parte do processo de manutenção, o que ajuda a reduzir o risco de EC2 instâncias Amazon degradadas ou danificadas que passam por manutenção periódica.

> **Important**
>
> A manutenção do sistema exclui arquivos ou configurações personalizadas que você adicionou ao servidor AWS OpsWorks for Chef Automate . Para obter mais informações sobre como reparar a configuração ou a perda de arquivos, consulte Restaurar configurações e arquivos personalizados após a manutenção neste tópico.

Tópicos

- Garantir que os nós confiem na Autoridade AWS OpsWorks de Certificação
- Configurar a manutenção do sistema
- Iniciar a manutenção do sistema sob demanda
- Restaurar configurações e arquivos personalizados após a manutenção

Garantir que os nós confiem na Autoridade AWS OpsWorks de Certificação

> **Note**
>
> As etapas desta seção não são necessárias se você estiver usando um domínio e um certificado personalizados com seu AWS OpsWorks for Chef Automate servidor.

Os nós que você está gerenciando com um AWS OpsWorks for Chef Automate servidor devem se autenticar com o servidor usando certificados. Durante a manutenção do sistema, AWS OpsWorks substitui a instância do servidor e regenera novos certificados por meio da autoridade de AWS OpsWorks certificação (CA). Para restaurar automaticamente a comunicação com seus nós gerenciados após a conclusão da manutenção, os nós devem confiar na AWS OpsWorks CA que acompanha o kit inicial e está hospedada nas regiões que são suportadas pela AWS OpsWorks for Chef Automate. Quando você usa a AWS OpsWorks CA para estabelecer a confiança entre os nós e o servidor, os nós se reconectam à nova instância do servidor após a manutenção. Se você estiver adicionando EC2 nós usando o EC2 `userdata` script descrito em[Adicione nós automaticamente em AWS OpsWorks for Chef Automate](), os nós já estão configurados para confiar na AWS OpsWorks CA.

- Para os nós baseados no Linux, o local do bucket do S3 da CA é `https://opsworks-cm-${REGION}-prod-default-assets.s3.amazonaws.com/misc/opsworks-cm-ca-2020-root.pem`. A CA AWS OpsWorks confiável deve ser armazenada no caminho`/etc/chef/opsworks-cm-ca-2020-root.pem`.
- Para os nós baseados no Windows, o local do bucket do S3 da CA é `https://opsworks-cm-$env:AWS_REGION-prod-default-assets.s3.amazonaws.com/misc/opsworks-cm-ca-2020-root.pem`. A AWS OpsWorks CA deve ser armazenada na pasta raiz do Chef; por exemplo, `C:\chef\opsworks-cm-ca-2020-root.pem`

Nos dois caminhos, a variável de região resulta em uma das seguintes:

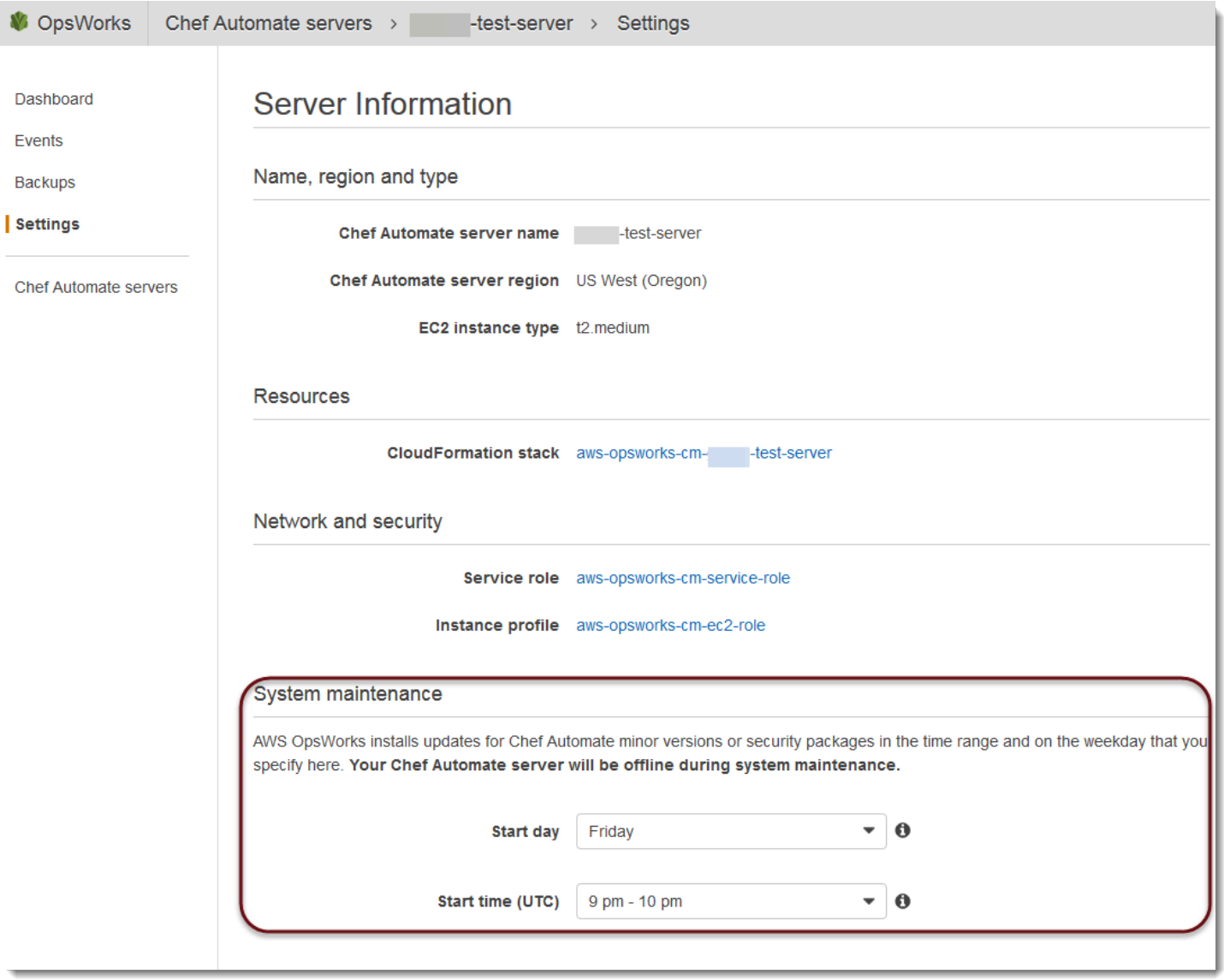
- `us-east-2`
- `us-east-1`
- `us-west-1`
- `us-west-2`
- `ap-northeast-1`

- `ap-southeast-1`
- `ap-southeast-2`
- `eu-central-1`
- `eu-west-1`

Configurar a manutenção do sistema

Ao criar um novo AWS OpsWorks for Chef Automate servidor, você pode configurar um dia e um horário da semana, no Horário Universal Coordenado (UTC), para iniciar a manutenção do sistema. A manutenção começa na hora que você especificar. Como é recomendável que o servidor esteja offline durante a manutenção do sistema, escolha um horário de baixa demanda do servidor no horário comercial regular. Enquanto a manutenção estiver em andamento, o status do servidor será `UNDER_MAINTENANCE`.

Você também pode alterar as configurações de manutenção do sistema em um AWS OpsWorks for Chef Automate servidor existente, alterando as configurações na área Manutenção do sistema da página Configurações do seu servidor, conforme mostrado na captura de tela a seguir.



Na seção System maintenance, defina o dia e a hora em que você deseja que a manutenção do sistema se inicie.

Configurando a manutenção do sistema usando o AWS CLI

Você também pode configurar a hora do início automático da manutenção do sistema usando a AWS CLI. O AWS CLI permite que você configure a manutenção automática diária, se desejar, omitindo o prefixo de três caracteres do dia da semana.

Em um comando `create-server`, adicione o parâmetro `--preferred-maintenance-window`, depois de especificar os requisitos para a criação da instância do servidor (como o tipo de instância e o Nome de região da Amazon (ARN) do perfil da instância e da função de serviço). No exemplo de `create-server` a seguir, `--preferred-maintenance-window` está definido como `Mon:08:00`.

Isso significa que você configurou a manutenção para começar toda segunda-feira às 8:00 da manhã. UTC.

```
aws opsworks-cm create-server --engine "Chef" --engine-model "Single" --
engine-version "12" --server-name "automate-06" --instance-profile-arn
 "arn:aws:iam::1019881987024:instance-profile/aws-opsworks-cm-ec2-role"
 --instance-type "t2.medium" --key-pair "amazon-test" --service-role-arn
 "arn:aws:iam::044726508045:role/aws-opsworks-cm-service-role" --preferred-maintenance-
window "Mon:08:00"
```

Em um comando `update-server`, você pode atualizar o valor `--preferred-maintenance-window` sozinho, se desejar. No exemplo a seguir, a janela de manutenção está configurada para toda sexta-feira às 18h15. UTC.

```
aws opsworks-cm update-server --server-name "shiny-kitchen" --preferred-maintenance-
window "Fri:18:15"
```

Para alterar o horário de início da janela de manutenção para todos os dias às 18h15, UTC, omita o prefixo de dia da semana de três caracteres, como mostrado no exemplo a seguir.

```
aws opsworks-cm update-server --server-name "shiny-kitchen" --preferred-maintenance-
window "18:15"
```

Para obter mais informações sobre como definir a janela preferencial de manutenção do sistema usando o AWS CLI, consulte create-server e update-server.

Iniciar a manutenção do sistema sob demanda

Para iniciar a manutenção do sistema sob demanda, fora da manutenção automática semanal ou diária configurada, execute o AWS CLI comando a seguir. Não é possível iniciar a manutenção sob demanda no AWS Management Console.

```
aws opsworks-cm start-maintenance --server-name server_name
```

Para obter mais informações sobre esse comando, consulte start-maintenance.

Restaurar configurações e arquivos personalizados após a manutenção

A manutenção do sistema pode excluir ou alterar arquivos ou configurações personalizados que você adicionou ao seu AWS OpsWorks for Chef Automate servidor.

Se, após uma execução de manutenção, faltarem arquivos ou configurações que você adicionou usando `RunCommand` ou SSH em seu servidor Chef, você pode usar uma Amazon Machine Image (AMI) para iniciar uma nova EC2 instância da Amazon. AMIs estão disponíveis e são criados a partir da configuração de pré-manutenção de um servidor.

A nova instância estará no mesmo estado em que o servidor do Chef estava antes da manutenção e deve incluir os arquivos e configurações que estão faltando.

> **Important**
>
> Não é possível usar a nova instância para restaurar seu servidor, pois ela não pode ser executada como um servidor do Chef. Você pode usar a instância apenas para recuperar seus arquivos e configurações.

Para iniciar uma EC2 instância a partir de uma AMI, no EC2 console da Amazon, abra o assistente Launch, escolha My e AMIs, em seguida, escolha a AMI que tem o nome do seu servidor. Siga as etapas do EC2 assistente da Amazon como você faria para iniciar qualquer outra instância.

Verificações de conformidade em AWS OpsWorks for Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As verificações de conformidade permitem acompanhar a conformidade de nós gerenciados na infraestrutura com base em políticas predefinidas, também chamadas de regras. As visualizações de conformidade permitem que você audite regularmente os aplicativos em busca de vulnerabilidades e configurações fora de conformidade. O Chef oferece mais de 100 perfis de conformidade predefinidos – coleções de regras que se aplicam a configurações de nó específicas – que você pode

usar nas verificações de conformidade. Você também pode usar a InSpec linguagem Chef para criar seus próprios perfis personalizados.

Se o servidor ainda não estiver executando o Chef Automate 2.0, você poderá configurar o Chef Compliance manualmente instalando o livro de receitas de auditoria.

> **Note**
>
> A versão mínima suportada do software de agente cliente Chef Infra (`chef-client`) nos nós associados a um AWS OpsWorks for Chef Automate servidor é 13. x. Recomendamos executar a `chef-client` versão mais atual e estável, ou pelo menos a 14.10.9.

Tópicos

- Conformidade no Chef Automate 2.0
- Conformidade no Chef Automate 1.x
- Atualizações feitas na conformidade
- Perfis de conformidade e comunidade personalizados
- Consulte também

Conformidade no Chef Automate 2.0

Se o seu AWS OpsWorks for Chef Automate servidor estiver executando o Chef Automate 2.0, configure o Chef Compliance usando os procedimentos desta seção.

Executar tarefas de verificação de conformidade com o Chef Automate 2.0

O Chef Automate 2.0 inclui o recurso de InSpec verificação de conformidade do Chef que anteriormente exigia configuração manual e configuração do livro de receitas. Você pode executar trabalhos de digitalização em um AWS OpsWorks for Chef Automate servidor que esteja executando o Chef Automate 2.0. As tarefas podem ser executadas imediatamente (uma vez), programadas para um horário posterior ou programadas para serem executadas em intervalos específicos, como diariamente ou a cada duas horas. Os resultados de uma tarefa de verificação são enviados para geração de relatórios de conformidade. Você pode visualizar e tomar atitudes com base nos resultados de verificação de conformidade no painel do Chef Automate. Para abrir a guia Compliance (Conformidade) e visualizar relatórios, na guia Scan Jobs (Tarefas de verificação) do painel do Chef Automate, escolha Report (Relatório) à direita de uma linha de nós gerenciados.

Para executar tarefas de verificação em nós gerenciados, você deve ter os itens a seguir.

- Pelo menos um perfil de conformidade instalado no namespace.
- Pelo menos um nó de destino, adicionado manualmente ou uma EC2 instância adicionada automaticamente.

Em AWS OpsWorks for Chef Automate, os trabalhos de digitalização são suportados nos seguintes alvos.

- Nós adicionados manualmente
- Instâncias `aws-ec2`
- Regiões da AWS

Para obter instruções detalhadas sobre como executar tarefas de verificação, consulte Tarefas de verificação do Chef Automate na documentação do Chef.

(Opcional, Chef Automate 2.0) Configuração de conformidade com o livro de receitas de auditoria

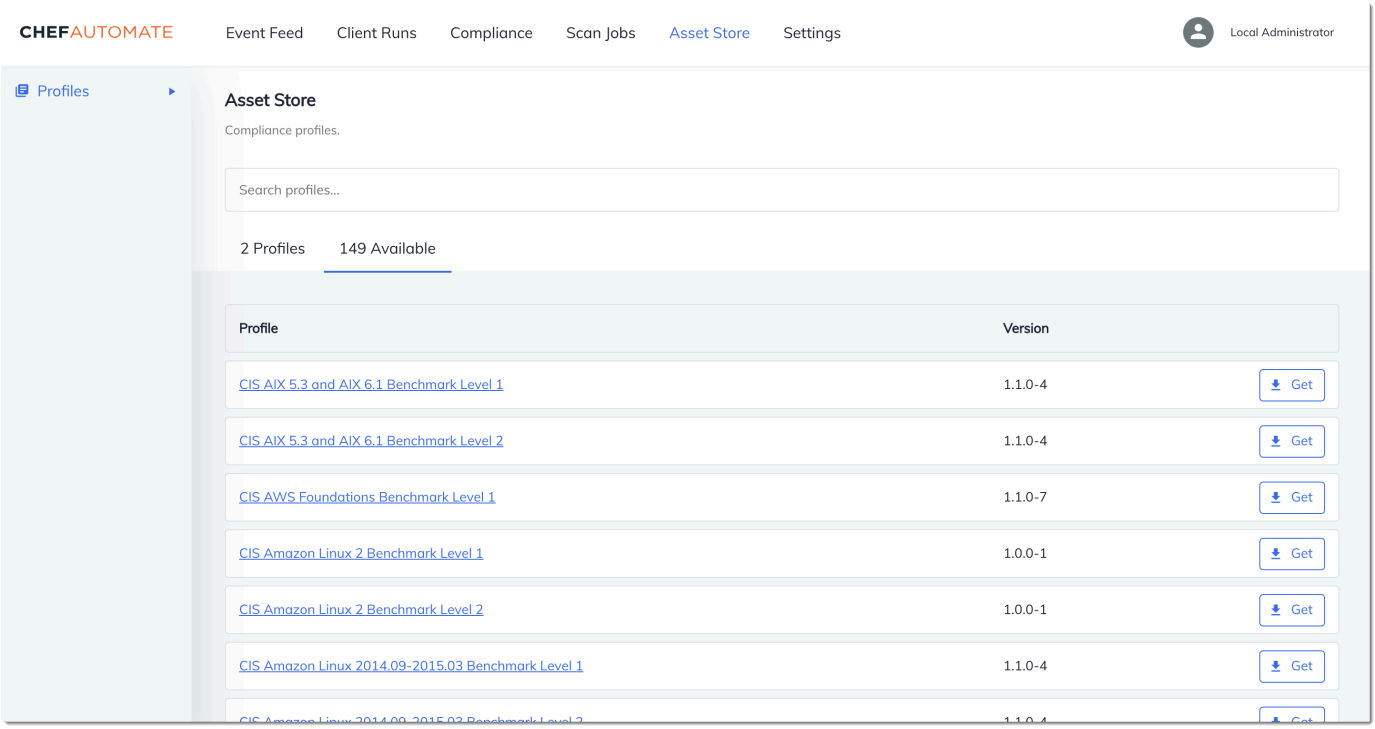
Você pode configurar a conformidade em qualquer AWS OpsWorks for Chef Automate servidor. Depois de iniciar um servidor do AWS OpsWorks for Chef Automate , você poderá instalar perfis pelo painel do Chef Automate ou adicionar perfis desejados a atributos do livro de receitas de auditoria no arquivo de política `Policyfile.rb`. Um arquivo `Policyfile.rb` pré-preenchido está incluído no Starter Kit.

Depois de editar `Policyfile.rb` com perfis como atributos do livro de receitas de auditoria, execute comandos `chef push` para fazer o upload do livro de receitas de auditoria e outros livros de receitas especificados em `Policyfile.rb` no servidor do Chef Automate. A instalação do livro de receitas de auditoria também instala a gema do Chef InSpec, uma estrutura de teste e auditoria de código aberto produzida pelo Chef. Para o Chef Automate 2.0, escolha a versão 7.1.0 ou posterior do livro de receitas de auditoria. O InSpec gem deve ser da versão 2.2.102 ou posterior.

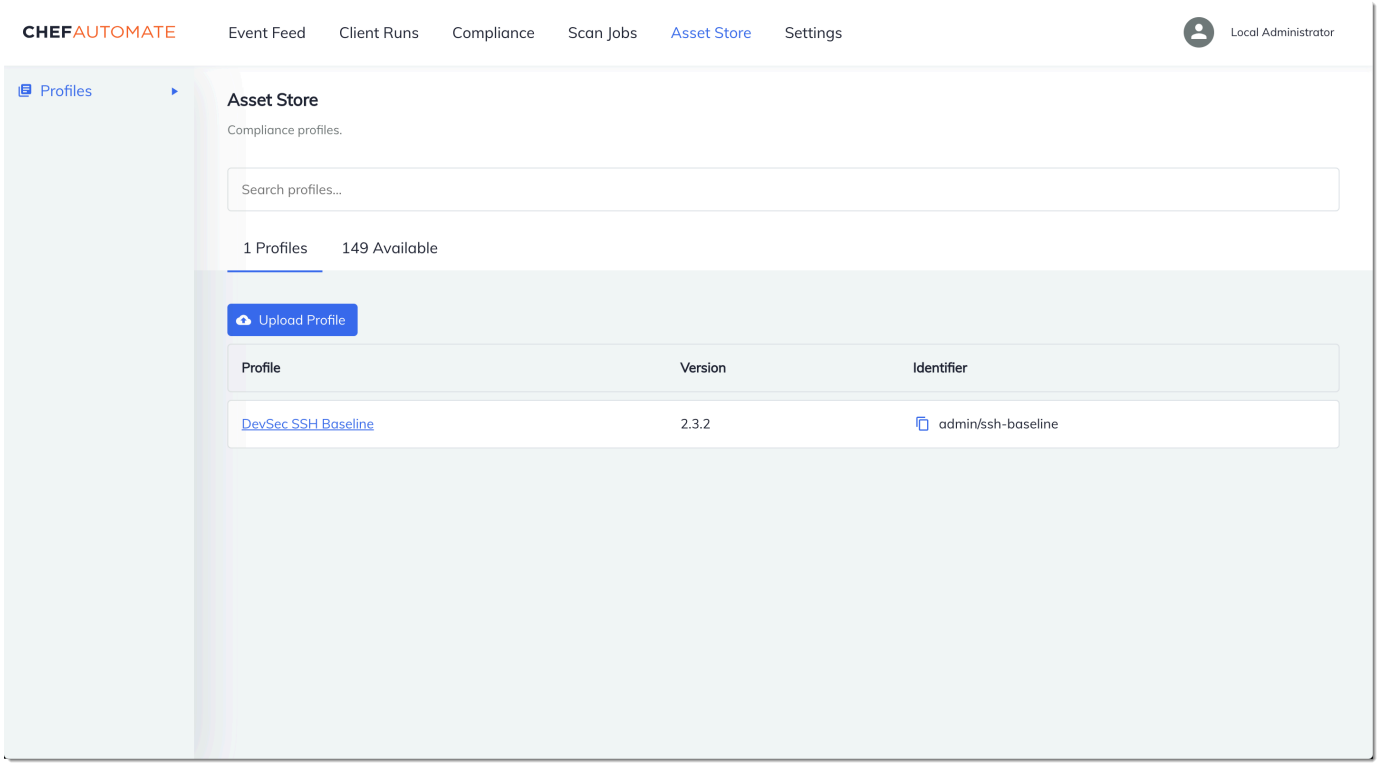
As instruções nesta seção mostram como implementar o livro de receitas `opsworks-audit`. O livro de receitas de auditoria baixa perfis especificados do servidor Chef Automate, avalia os nós em relação ao perfil DevSec SSH Baseline e relata o resultado das verificações de conformidade em cada execução. `chef-client`

Para instalar perfis de conformidade

1. Se você ainda não tiver feito isso, acesse o painel baseado na Web do Chef Automate. Use as credenciais que você recebeu quando fez download do Starter Kit ao criar o servidor do AWS OpsWorks for Chef Automate .
2. No painel do Chef Automate, escolha a guia Asset Store (Loja de ativos).



3. Escolha a guia Disponível para ver perfis predefinidos.
4. Pesquise a lista de perfis. Escolha um perfil que corresponda ao sistema operacional e à configuração de pelo menos um de seus nós gerenciados. Para visualizar detalhes sobre o perfil, incluindo uma descrição das violações direcionadas do perfil e o código da regra subjacente, escolha > à direita da entrada do perfil. Você pode escolher vários perfis. Se você estiver configurando o exemplo no Starter Kit, escolha DevSec SSH Baseline.



5. Para instalar os perfis selecionados no seu servidor Chef Automate, escolha Get (Obter).
6. Depois que você instalar os perfis, eles serão mostrados na guia Profiles (Perfis) do painel do Chef Automate.

Para instalar livros de receita com **`Policyfile.rb`**

1. Visualize `Policyfile.rb` no Starter Kit para ver quais atributos do livro de receitas de auditoria especificam o perfil `ssh-baseline` em `['profiles']`.

```
# Define audit cookbook attributes
default["opsworks-demo"]["audit"]["reporter"] = "chef-server-automate"
default["opsworks-demo"]["audit"]["profiles"] = [
  {
    "name": "DevSec SSH Baseline",
    "compliance": "admin/ssh-baseline"
  }
]
```

2. Faça o download e instale os livros de receitas definidos em `Policyfile.rb`.

```
chef install
```

Todos os livros de receitas têm versionamento no arquivo `metadata.rb` do livro de receitas. Sempre que altera um livro de receitas, você deve aumentar a versão dele que está no `metadata.rb`.

3. Envie a política `opsworks-demo`, definida em `Policyfile.rb`, para o servidor.

```
chef push opsworks-demo
```

4. Verifique a instalação da política. Execute o seguinte comando:

```
chef show-policy
```

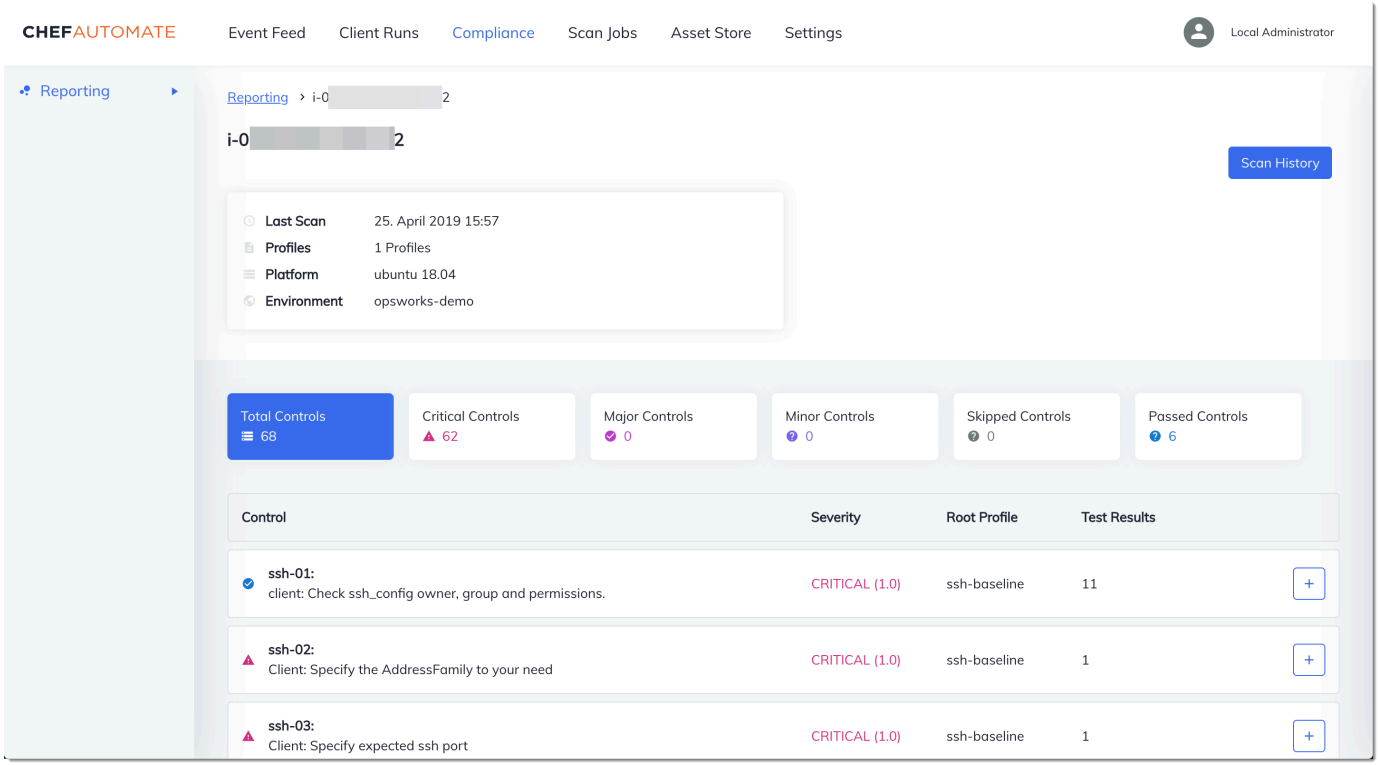
Os resultados devem ser semelhantes ao seguinte:

```
opsworks-demo-webserver
=======================
* opsworks-demo:  ec0fe46314
```

5. Adicione nós ao servidor para gerenciar, se você ainda não tiver feito isso. Para conectar seu primeiro nó ao AWS OpsWorks for Chef Automate servidor, use o `userdata.sh` script incluído neste Starter Kit. Ele usa a AWS OpsWorks `AssociateNode` API para conectar um nó ao seu servidor.

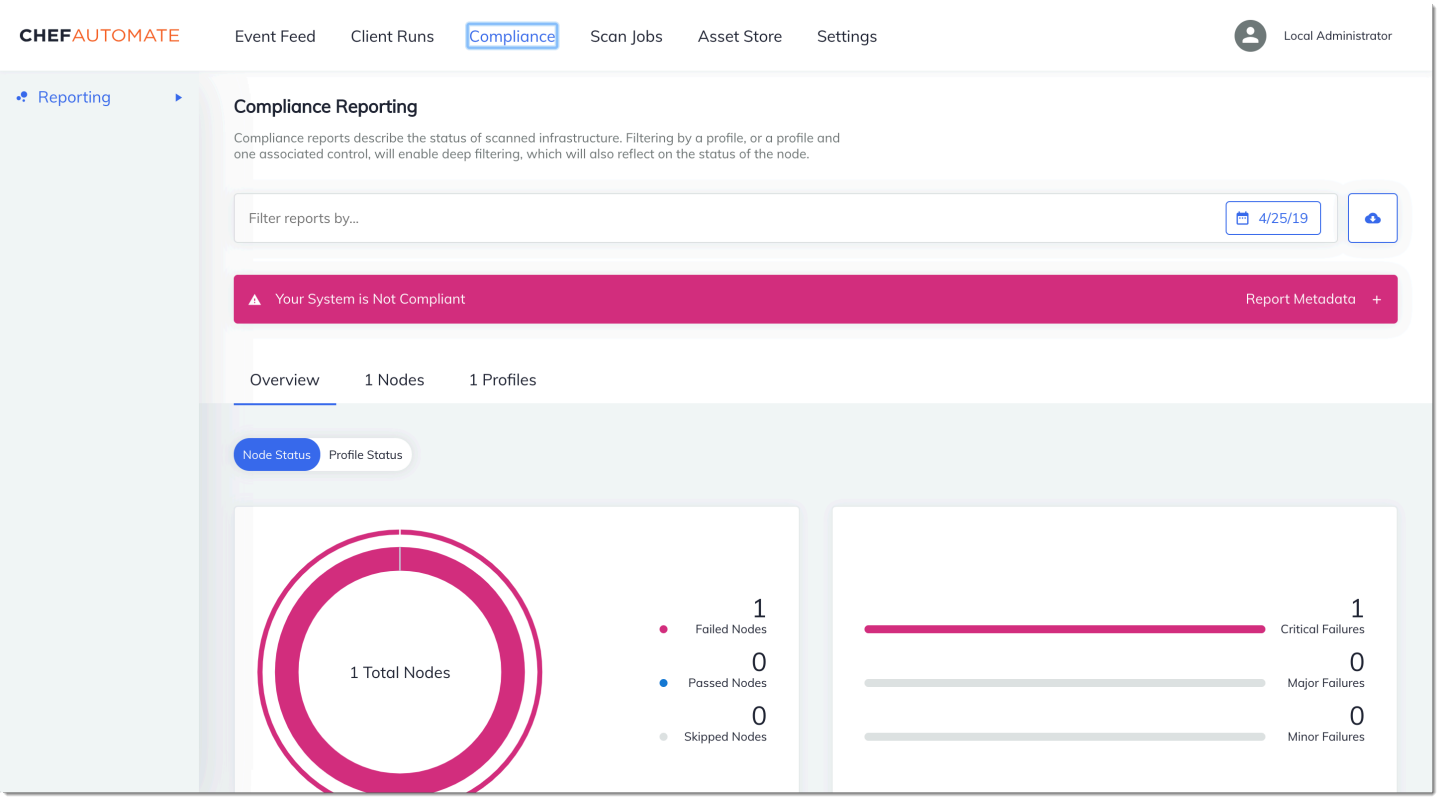
   Você pode automatizar a associação de nós seguindo as etapas em Adicione nós automaticamente em AWS OpsWorks for Chef Automate ou adicionar nós, um de cada vez, seguindo as etapas em Adicione nós individualmente.

6. Depois de atualizar a lista de execução de seus nós, o agente `chef-client` executará as receitas especificadas na próxima execução. Por padrão, isso ocorre a cada 1.800 segundos (30 minutos). Depois da execução, você poderá visualizar e tomar medidas com base nos resultados da guia Compliance (Conformidade) no painel do Chef Automate.

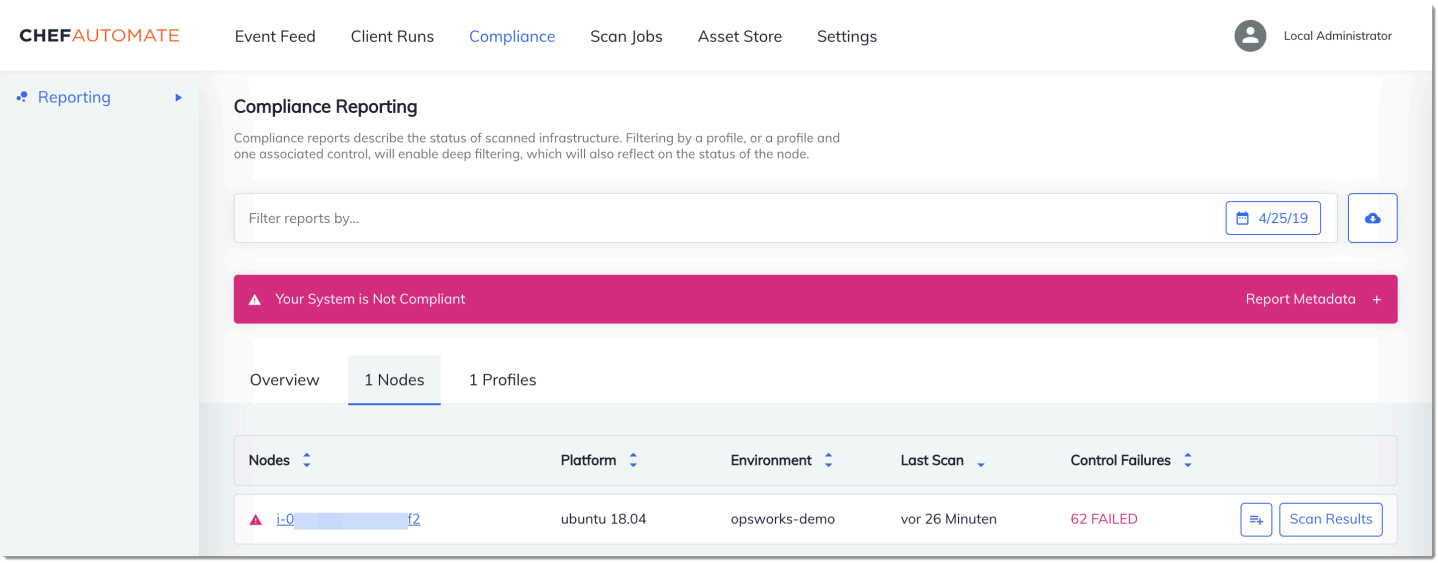


Execução de uma verificação de conformidade

Você deverá ver os resultados de verificação de conformidade no painel do Chef Automate logo depois da primeira execução do agente ocorrida depois de configurar listas de execução de nós.



No painel do Chef Automate, escolha a guia Compliance (Conformidade). No painel de navegação à esquerda, escolha Relatórios. Escolha a guia Profiles (Perfis), selecione Scan Results (Resultados da verificação) e escolha um nó com falhas de verificação para saber mais sobre as regras nas quais o nó falhou.



Normalmente, você vê resultados de varredura incompatíveis, porque os novos nós ainda não satisfazem todas as regras no perfil DevSec SSH Baseline. O DevSec Hardening Framework, um

projeto baseado na comunidade, oferece livros de receitas para corrigir problemas que violam as regras do perfil SSH Baseline. DevSec

(Opcional) resolução de resultados não compatíveis

O kit inicial inclui um livro de receitas de código aberto,**`ssh-hardening`**, que você pode executar para corrigir resultados não compatíveis de execuções no perfil SSH Baseline. DevSec

> **Note**
>
> O `ssh-hardening` livro de receitas faz alterações em seus nós para cumprir as regras de linha de base do DevSec SSH. Antes de executar este livro de receitas em qualquer nó de produção, revise os detalhes sobre o perfil DevSec SSH Baseline no console do Chef Automate para entender as violações de regras que o livro de receitas visa. Revise as informações sobre o livro de receitas de código-fonte aberto `ssh-hardening` antes de executá-lo em qualquer um dos nós de produção.

Para executar o livro de receitas **`ssh-hardening`**

1. Em um editor de texto, anexe o livro de receitas `ssh-hardening` à lista de execução de `Policyfile.rb`. A lista de execução `Policyfile.rb` deve corresponder ao seguinte.

   ```
   run_list  'chef-client', 'opsworks-webserver', 'audit', 'ssh-hardening'
   ```

2. Atualize `Policyfile.rb` e o envie para o servidor do AWS OpsWorks for Chef Automate .

   ```
   chef update Policyfile.rb
      chef push opsworks-demo
   ```

3. Os nós associados à política `opsworks-demo` atualizam a lista de execução automaticamente e aplicarão o livro de receitas `ssh-hardening` na próxima execução de `chef-client`.

   Como você está usando o livro de receitas `chef-client`, o nó é verificado em intervalos regulares (por padrão, a cada 30 minutos). No próximo check-in, o `ssh-hardening` livro de receitas é executado e ajuda a melhorar a segurança dos nós para atender às regras do perfil DevSec SSH Baseline.

4. Após a execução inicial do livro de receitas `ssh-hardening`, aguarde 30 minutos para que uma verificação de conformidade seja executada novamente. Visualize os resultados no painel

do Chef Automate. Os resultados não compatíveis que ocorreram na execução inicial da verificação de linha de base do DevSec SSH devem ser resolvidos.

Conformidade no Chef Automate 1.x

Se o seu AWS OpsWorks for Chef Automate servidor estiver executando o Chef Automate 1. x, configure o Chef Compliance usando os procedimentos desta seção.

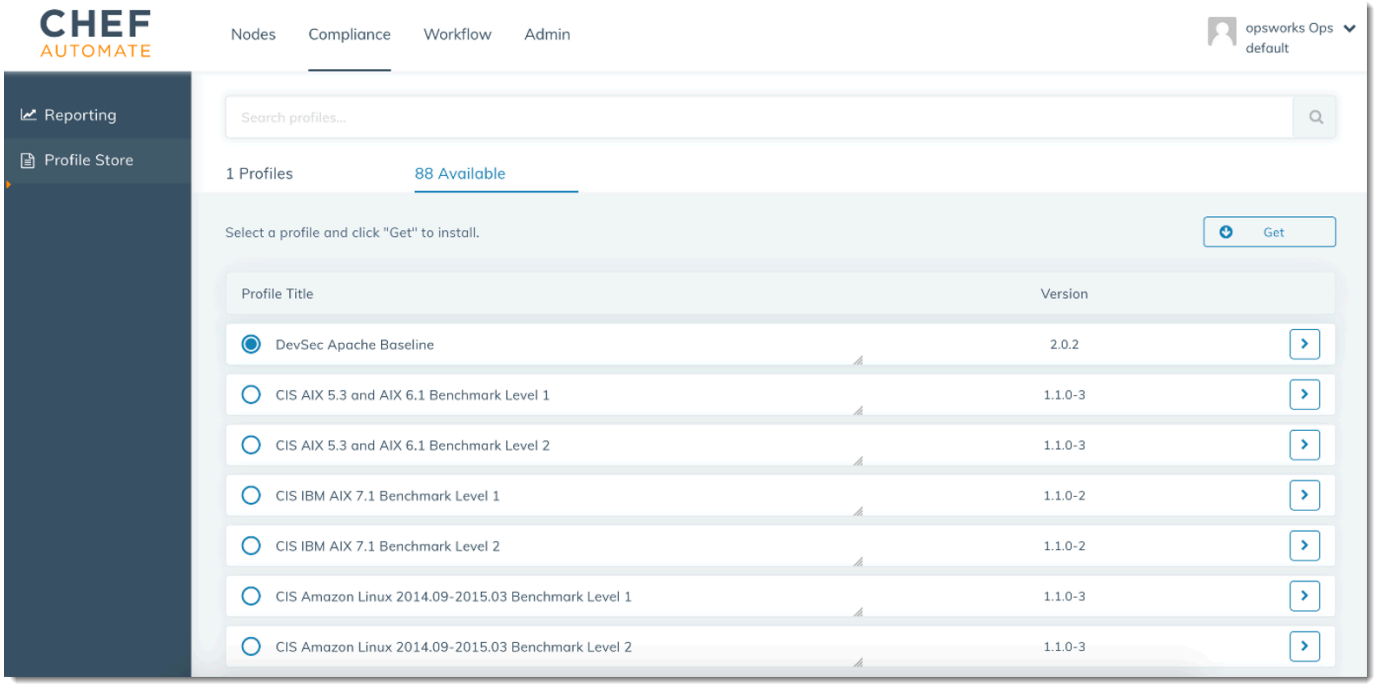
(Opcional, Chef Automate 1.x) Configuração do Chef Compliance

Você pode configurar o Chef Compliance em qualquer AWS OpsWorks for Chef Automate servidor. Depois de iniciar um servidor AWS OpsWorks for Chef Automate , escolha os perfis que você deseja executar a partir dos perfis no painel do Chef Automate. Depois de instalar perfis, execute comandos `berks` para carregar o livro de receitas de auditoria para o servidor Chef Automate. A instalação do livro de receitas de auditoria também instala o gem for InSpec, uma estrutura de teste de código aberto produzida pelo Chef que permite integrar testes automatizados em qualquer estágio do seu pipeline de implantação. Para o Chef Automate 1.x, escolha a versão 5.0.1 ou posterior do livro de receitas de auditoria. O InSpec gem deve ser da versão 1.24.0 ou posterior.

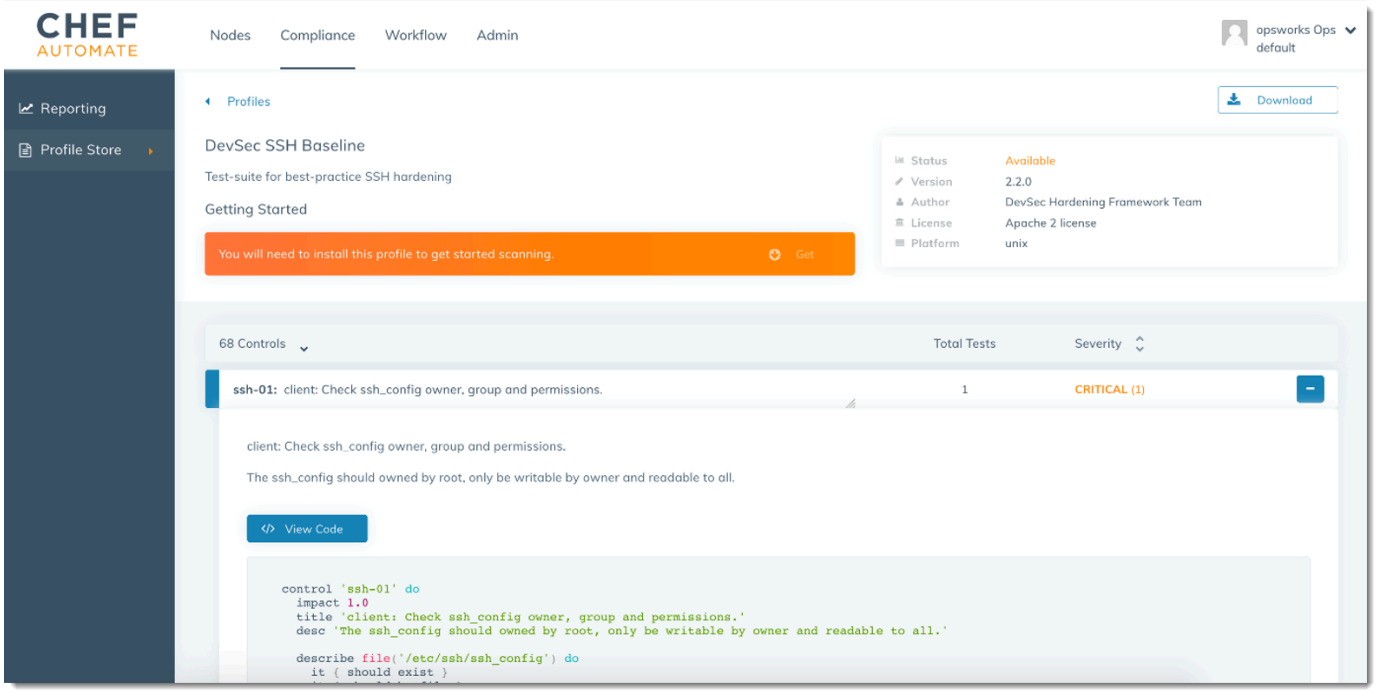
O kit AWS OpsWorks for Chef Automate inicial inclui um livro de receitas embaladoopsworks-`audit`, que baixa e instala a versão correta do livro de receitas do Chef's Audit para você. O `opsworks-audit` livro de receitas também instrui o `chef-client` agente a avaliar os nós em relação ao perfil DevSecSSH Baseline que você instala a partir do console de conformidade do Chef posteriormente neste tópico. Você pode configurar o Compliance usando o livro de receitas para atender suas preferências. As instruções nesta seção mostram como implementar o livro de receitas `opsworks-audit`.

Para instalar perfis do Compliance

1. Se você ainda não tiver feito isso, acesse o painel baseado na Web do Chef Automate. Use as credenciais que você recebeu ao baixar o Starter Kit ao criar seu AWS OpsWorks for Chef Automate servidor.
2. No painel do Chef Automate, escolha a guia Compliance (Conformidade).



3. Na barra de navegação à esquerda, escolha Armazenamento de perfis e, em seguida, selecione a guia Disponível para ver os perfis predefinidos.
4. Pesquise a lista de perfis. Escolha um perfil que corresponda ao sistema operacional e à configuração de pelo menos um de seus nós gerenciados. Para visualizar detalhes sobre o perfil, incluindo uma descrição das violações direcionadas do perfil e o código da regra subjacente, escolha > à direita da entrada do perfil. Você pode escolher vários perfis.



5. Para instalar os perfis selecionados no seu servidor Chef Automate, escolha Get (Obter).
6. Quando o download estiver concluído, vá para o próximo procedimento.

Para instalar e configurar o livro de receitas **`opsworks-audit`**

1. Esta etapa é opcional, mas economiza tempo na Etapa 6, quando você está adicionando receitas às listas de execução de nós. Edite o arquivo `roles/opsworks-example-role.rb` que está incluído no starter kit que você baixou durante a criação do servidor AWS OpsWorks for Chef Automate . Adicione as linhas a seguir. A última linha é comentada, pois é opcional a adição do livro de receitas `ssh-hardening` e da receita para resolver nós não compatíveis após as execuções de verificação do Compliance.

```
run_list(
     "recipe[chef-client]",
     "recipe[apache2]",
     "recipe[opsworks-audit]"
     # "recipe[ssh-hardening]"
       )
```

2. Use um editor de texto para especificar os livros de receita desejados no seu Berksfile. Um Berksfile de exemplo é fornecido para você no starter kit. Neste exemplo, instalamos o livro de receitas cliente do Chef Infra (`chef-client`), o livro de receitas `apache2` e o livro de receitas `opsworks-audit`. Seu Berksfile deve se parecer com o seguinte.

```
source 'https://supermarket.chef.io
     cookbook 'chef-client'
     cookbook 'apache2', '~> 5.0.1'
     cookbook 'opsworks-audit', path: 'cookbooks/opsworks-audit', '~> 1.0.0'
```

   Todos os livros de receitas têm versionamento no arquivo `metadata.rb` do livro de receitas. Sempre que altera um livro de receitas, você deve aumentar a versão dele que está no `metadata.rb`.

3. Execute o comando a seguir para baixar e instalar os livros de receitas na pasta `cookbooks` do seu computador local ou de trabalho.

```
berks vendor cookbooks
```

4. Execute o comando a seguir para carregar os livros de receitas do fornecedor no servidor AWS OpsWorks for Chef Automate .

```
knife upload .
```

5. Execute o comando a seguir para verificar a instalação do livro de receitas `opsworks-audit` mostrando uma lista de livros de receitas que estão disponíveis atualmente no servidor.

```
knife cookbook list
```

6. Adicione nós ao servidor para gerenciar, se você ainda não tiver feito isso. Você pode automatizar a associação de nós seguindo as etapas em Adicione nós automaticamente em AWS OpsWorks for Chef Automate ou adicionar nós, um de cada vez, seguindo as etapas em Adicione nós individualmente. Edite a lista de execução de seus nós para adicionar a função que você especificou na Etapa 1, `opsworks-example-role`. Neste exemplo, edite o atributo `RUN_LIST` no script `userdata` que você usa para automatizar a associação de nós.

```
RUN_LIST="role[opsworks-example-role]"
```

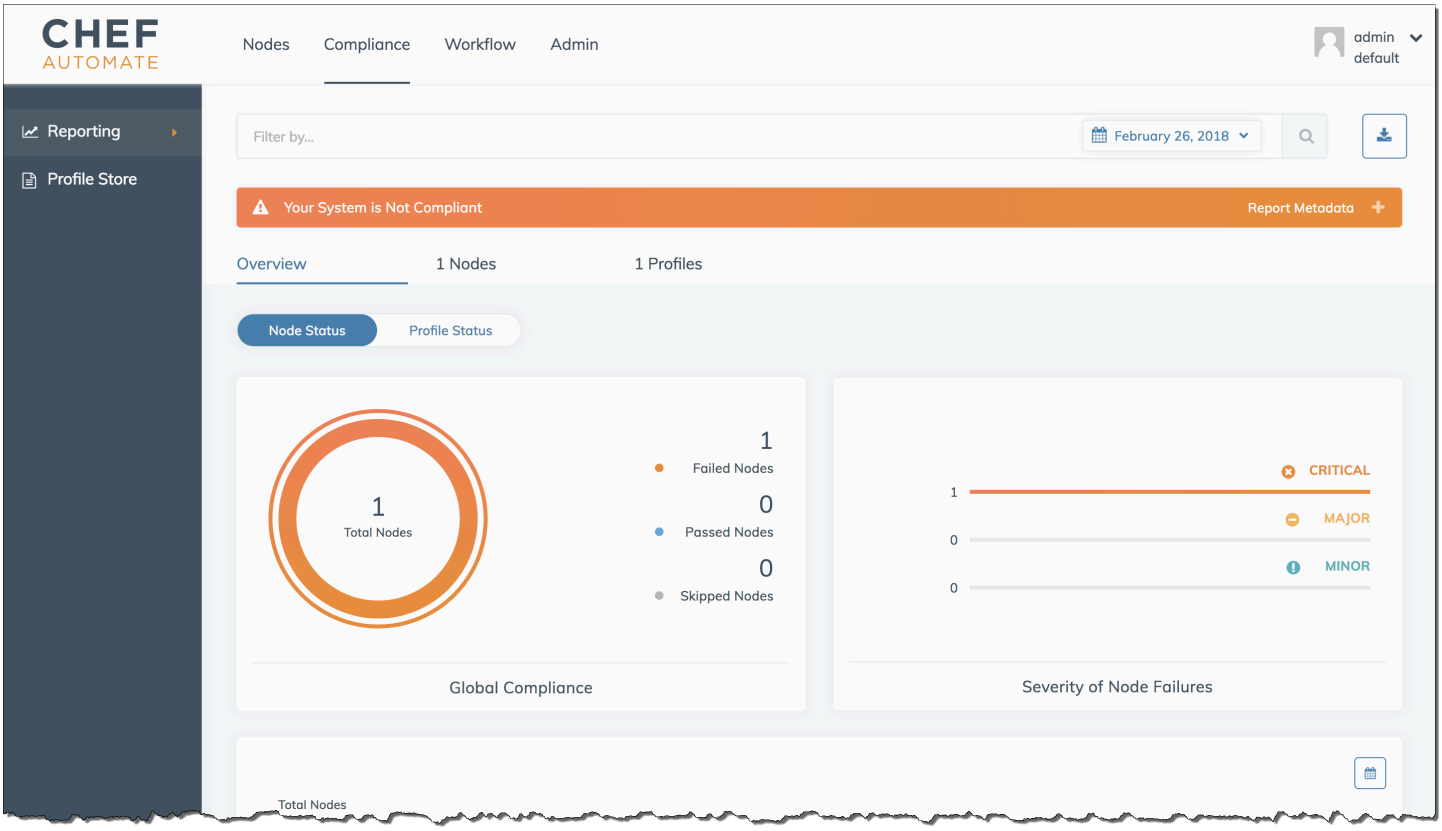
   Se você ignorou a Etapa 1 e não configurou a função, adicione os nomes das receitas individuais à lista de execuções. Salve suas alterações e siga as etapas Etapa 3: Criar instâncias com um script de associação autônoma para aplicar seu script de dados de usuário às EC2 instâncias da Amazon.

```
RUN_LIST="recipe[chef-client],recipe[apache2],recipe[opworks-audit]"
```

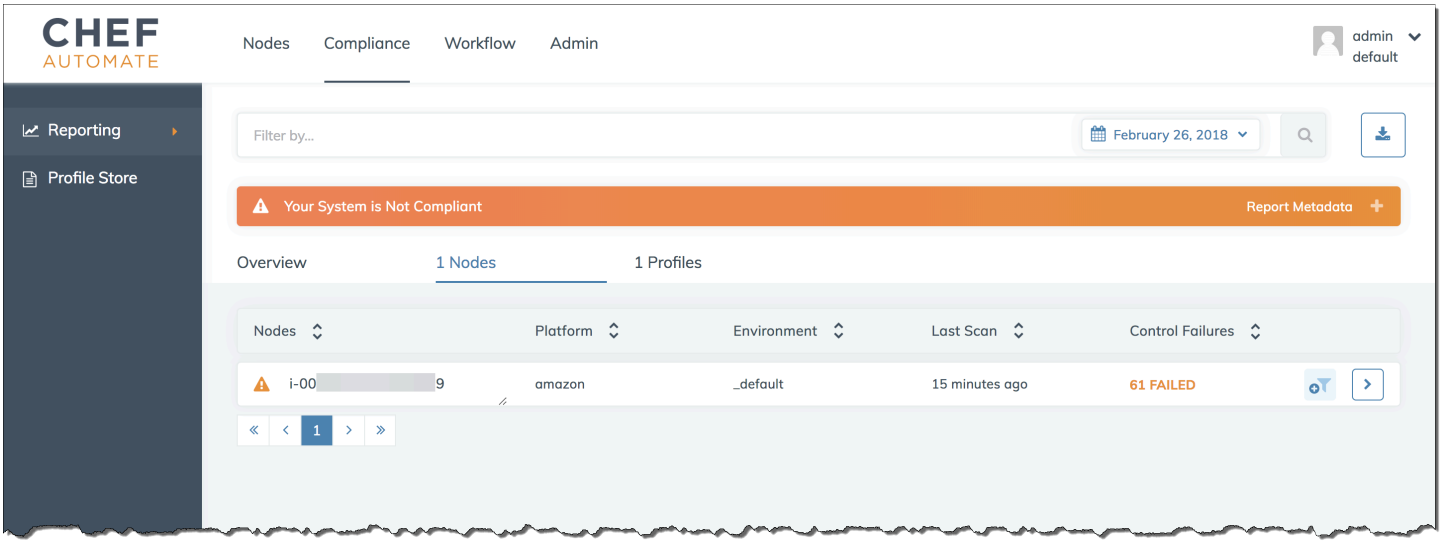
7. Depois de atualizar a lista de execução de seus nós, o agente `chef-client` executará as receitas especificadas na próxima execução. Por padrão, isso ocorre a cada 1.800 segundos (30 minutos). Após a execução, os seus resultados do Compliance ficam visíveis no painel do Chef Automate.

Execução de uma verificação de conformidade

Você deve ver os resultados de verificação de conformidade no painel do Chef Automate logo após a primeira execução do agente daemon que ocorre depois de configurar listas de execução de nós.



No painel do Chef Automate, escolha a guia Compliance (Conformidade). No painel de navegação à esquerda, escolha Relatórios. Escolha a guia Profiles (Perfis), selecione Scan Results (Resultados da verificação) e escolha um nó com falhas de verificação para saber mais sobre as regras nas quais o nó falhou.



Normalmente, você vê resultados de varredura incompatíveis, porque os novos nós ainda não satisfazem todas as regras no perfil DevSec SSH Baseline. O DevSec Hardening Framework, um

projeto baseado na comunidade, oferece livros de receitas para corrigir problemas que violam as regras do perfil SSH Baseline. DevSec

(Opcional) resolução de resultados não compatíveis

O kit inicial inclui um livro de receitas de código aberto,**`ssh-hardening`**, que você pode executar para corrigir resultados não compatíveis de execuções no perfil SSH Baseline. DevSec

> **Note**
>
> O `ssh-hardening` livro de receitas faz alterações em seus nós para cumprir as regras de linha de base do DevSec SSH. Antes de executar este livro de receitas em qualquer nó de produção, revise os detalhes sobre o perfil DevSec SSH Baseline no console do Chef Automate para entender as violações de regras que o livro de receitas visa. Revise as informações sobre o livro de receitas de código-fonte aberto `ssh-hardening` antes de executá-lo em qualquer um dos nós de produção.

Para executar o livro de receitas **`ssh-hardening`**

1. Em um editor de texto, anexe o livro de receitas `ssh-hardening` ao seu Berksfile. Seu Berksfile deve se parecer com o seguinte.

   ```
   source 'https://supermarket.chef.io'
        cookbook 'chef-client'
        cookbook 'apache2', '~> 5.0.1'
        cookbook 'opsworks-audit', path: 'cookbooks/opsworks-audit', '~> 1.0.0' #
    optional
        cookbook 'ssh-hardening'
   ```

2. Execute os comandos a seguir para baixar o livro de receitas `ssh-hardening` na sua pasta local de livros de receitas e, em seguida, carregue-o no servidor AWS OpsWorks for Chef Automate .

   ```
   berks vendor cookbooks
   knife upload .
   ```

3. Adicione a receita `ssh-hardening` à lista de execução de nós conforme descrito nas Etapas 1 e 6 de Para instalar e configurar o livro de receitas `opsworks-audit`.

Se você atualizar o arquivo `opsworks-example-role.rb`, carregue as alterações no servidor executando o comando a seguir.

```
knife upload .
```

Se você atualizar a lista de execuções diretamente, carregue as alterações no servidor executando o comando a seguir. O nome do nó geralmente é o ID da instância.

```
knife node run_list add <node name> 'recipe[ssh-hardening]'
```

4. Como você está usando o livro de receitas `chef-client`, o nó é verificado em intervalos regulares (por padrão, a cada 30 minutos). No próximo check-in, o `ssh-hardening` livro de receitas é executado e ajuda a melhorar a segurança dos nós para atender às regras do perfil DevSec SSH Baseline.
5. Após a execução inicial do livro de receitas `ssh-hardening`, aguarde 30 minutos para que uma verificação do Compliance seja executada novamente. Visualize os resultados no painel do Chef Automate. Os resultados não compatíveis que ocorreram na execução inicial da verificação de linha de base do DevSec SSH devem ser resolvidos.

Atualizações feitas na conformidade

Em um AWS OpsWorks for Chef Automate servidor, a funcionalidade de conformidade é atualizada automaticamente pela manutenção programada do sistema. À medida que as versões atualizadas do Chef Automate, Chef Infra Server e Chef InSpec se tornam disponíveis para seu AWS OpsWorks for Chef Automate servidor, talvez seja necessário verificar e atualizar as versões suportadas do livro de receitas Audit e do Chef InSpec gem que estão sendo executadas em seu servidor. Os perfis que você já instalou no seu AWS OpsWorks for Chef Automate servidor não são atualizados como parte da manutenção.

Perfis de conformidade e comunidade personalizados

O Chef atualmente inclui mais de 100 perfis de verificação de conformidade. Você pode adicionar perfis personalizados e de comunidade à lista, fazer o download e executar verificações de conformidade com base nesses perfis, da mesma forma que faria para perfis incluídos. Os perfis de conformidade baseados em comunidade estão disponíveis no Chef Supermarket. Os perfis

personalizados são programas baseados no Ruby que incluem uma pasta de controles que especificam as regras de verificação.

Consulte também

- Post do blog de anúncios do Chef Compliance
- Treinamento online do Chef Automate Compliance
- InSpecSite do Chef
- InSpec Tutoriais do Chef

Desassociar um nó de um servidor AWS OpsWorks for Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como desassociar ou remover um nó gerenciado do gerenciamento por um AWS OpsWorks for Chef Automate servidor. Essa operação é executada na linha de comando; você não pode desassociar nós no console AWS OpsWorks for Chef Automate de gerenciamento. Atualmente, a AWS OpsWorks for Chef Automate API não permite a remoção em lote de vários nós. O comando nesta seção dissociará um nó por vez.

Recomendamos que você dissocie os nós de um servidor Chef antes de excluir o servidor, de forma que os nós continuem a operar sem tentar se reconectar ao servidor. Para fazer isso, execute o `disassociate-node` AWS CLI comando.

Como desassociar nós

1. No AWS CLI, execute o comando a seguir para dissociar os nós. *`Node_name`*é o nome do nó que você deseja desassociar; para EC2 instâncias da Amazon, esse é o ID da instância.

*Server_name*é o nome do servidor Chef do qual você deseja desassociar o nó. `--engine-attributes`especifica seu `CHEF_AUTOMATE_ORGANIZATION` nome padrão. Todos os três parâmetros são obrigatórios.

O parâmetro `--region` não será necessário, a menos que você queira dissociar um nó de um servidor Chef que não esteja na região padrão.

```
aws opsworks-cm --region Region_name disassociate-node --node-
name Node_name --server-name Server_name --engine-attributes
 "Name=CHEF_AUTOMATE_ORGANIZATION,Value='default'"
```

O comando a seguir é um exemplo.

```
aws opsworks-cm --region us-west-2 disassociate-node --node-name
 i-0010zzz00d66zzz90 --server-name opsworkstest --engine-attributes
 "Name=CHEF_AUTOMATE_ORGANIZATION,Value='default'"
```

2. Aguarde pela mensagem de resposta indicando que a dissociação foi concluída.

   Depois de desassociar com sucesso um nó de um AWS OpsWorks for Chef Automate servidor, ele ainda pode estar visível no painel do Chef Automate. Por padrão, o Chef impõe um período de retenção para informações de estados de nós e limpa o nó automaticamente após alguns dias.

Para obter mais informações sobre como excluir um AWS OpsWorks for Chef Automate servidor, consulteExcluir um AWS OpsWorks for Chef Automate servidor.

Related Topics

As postagens de AWS blog a seguir oferecem mais informações sobre a associação automática de nós ao seu servidor Chef Automate, usando grupos de Auto Scaling ou em várias contas.

- Usando o AWS OpsWorks for Chef Automate para gerenciar EC2 instâncias com Auto Scaling
- OpsWorks para Chef Automate — Inicializando automaticamente nós em contas diferentes

Excluir um AWS OpsWorks for Chef Automate servidor

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como excluir um AWS OpsWorks for Chef Automate servidor. A exclusão de um servidor também exclui seus eventos, logs e todos livros de receitas que foram armazenados no servidor. Os recursos de suporte (instância do Amazon Elastic Compute Cloud, volume do Amazon Elastic Block Store etc.) também são excluídos, junto com todos os backups automatizados.

Embora a exclusão de um servidor não exclua os nós, eles não serão mais gerenciados pelo servidor excluído e tentarão continuamente a se reconectar. Por esse motivo, recomendamos que você desassocie os nós gerenciados antes de excluir um servidor do Chef. Nesta versão, você pode dissociar os nós executando um AWS CLI comando.

Etapa 1: Dissociação dos nós gerenciados

Desassocie os nós do servidor Chef antes de excluir o servidor, de forma que os nós continuem a operar sem tentar a reconexão com o servidor. Para fazer isso, execute o `disassociate-node` AWS CLI comando.

Como desassociar nós

1. No AWS CLI, execute o comando a seguir para dissociar os nós. *`Server_name`*é o nome do servidor Chef do qual você deseja desassociar o nó.

   ```
   aws opsworks-cm --region Region_name disassociate-node --node-name Node_name --
   server-name Server_name
   ```

2. Aguarde pela mensagem de resposta indicando que a dissociação foi concluída.

Etapa 2: Exclusão do servidor

1. No bloco do servidor no painel, expanda o menu Actions.
2. Escolha Delete server.
3. Quando você for solicitado a confirmar a exclusão, escolha Yes.

Redefinição das credenciais do painel do Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Periodicamente, você pode querer alterar a senha com a qual faz login no painel do Chef Automate. Você também pode usar os AWS CLI comandos do Amazon EC2 Systems Manager mostrados nesta seção para alterar a senha do painel do Chef Automate, caso a tenha perdido. O comando que você usa depende se o seu servidor Chef Automate está executando a versão 1 ou a versão 2 do Chef Automate.

1. Para retornar o ID da instância do seu servidor Chef, abra AWS Management Console a página a seguir.

   https://console.aws.amazon.com/ec2/v2/home? region= #instâncias:search= *region_of_your_server* - aws-opsworks-cm *server_name*

   Por exemplo, para um servidor Chef chamado MyChefServerna região Oeste dos EUA (Oregon), o URL do console seria o seguinte.

   https://console.aws.amazon.com/ec2/v2/home? region=us-west-2 #instâncias:search= - aws-opsworks-cm MyChefServer

   Anote o ID da instância que é exibido no console;, pois você precisará dele para alterar sua senha.

2. Para redefinir a senha de login do painel do Chef Automate, execute um dos seguintes comandos AWS CLI , dependendo se o seu servidor está executando o Chef Automate 1 ou o Chef Automate 2. *enterprise_name*Substitua pelo nome da sua empresa ou organização, *user_name* pelo nome de usuário de um administrador no servidor, *new_password* pela senha que você deseja usar e *region_name* pela região na qual seu servidor está localizado. Se você não especificar um nome de empresa, o nome da empresa será `default`. Por padrão, *enterprise_name* é `default` (esse é o nome da organização que está sempre provisionada). Para*user_name*, cria AWS OpsWorks for Chef Automate apenas um usuário chamado`admin`. Anote a nova senha e guarde-a em um local seguro, mas de fácil acesso.

   Para Chef Automate 1:

```
aws ssm send-command --document-name "AWS-RunShellScript" --comment "reset admin
 password" --instance-ids "instance_id"
--parameters commands="sudo delivery-ctl reset-
password enterprise_name user_name new_password" --region region_name --output text
```

   Para Chef Automate 2:

```
aws ssm send-command --document-name "AWS-RunShellScript" --comment "reset admin
 password" --instance-ids "instance_id"
--parameters commands="sudo chef-automate iam admin-access restore new_password" --
region region_name --output text
```

3. Aguarde o texto de saída (neste caso, o ID de comando) para mostrar que a alteração da senha foi concluída.

Registrando chamadas de AWS OpsWorks for Chef Automate API com AWS CloudTrail

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks for Chef Automate é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por uma identidade do IAM ou por um AWS serviço em AWS OpsWorks for Chef Automate. CloudTrail captura todas as chamadas de API para eventos AWS OpsWorks for Chef Automate as, incluindo chamadas do AWS OpsWorks for Chef Automate console e de chamadas de código para o. AWS OpsWorks for Chef Automate APIs Se você criar uma trilha, poderá habilitar a entrega contínua de CloudTrail eventos para um bucket do Amazon S3, incluindo eventos para. AWS OpsWorks for Chef Automate Se você não configurar uma trilha, ainda poderá ver os eventos mais recentes no CloudTrail console no Histórico de eventos. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita AWS OpsWorks for Chef Automate, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais sobre isso CloudTrail, consulte o Guia AWS CloudTrail do usuário.

AWS OpsWorks for Chef Automate Informações em CloudTrail

CloudTrail é ativado em sua AWS conta quando você cria a conta. Quando a atividade ocorre em AWS OpsWorks for Chef Automate, essa atividade é registrada em um CloudTrail evento junto com outros eventos AWS de serviço no histórico de eventos. Você pode visualizar, pesquisar e baixar eventos recentes em sua AWS conta. Para obter mais informações, consulte Visualização de eventos com histórico de CloudTrail eventos.

Para um registro contínuo dos eventos em sua AWS conta, incluindo eventos para AWS OpsWorks for Chef Automate, crie uma trilha. Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Por padrão, ao criar uma trilha no console, ela é aplicada a todas as regiões da . A trilha registra eventos de todas as regiões na AWS partição e entrega os arquivos de log ao bucket do Amazon S3 que você especificar. Além disso, você pode configurar outros AWS serviços para analisar e agir com base nos dados de eventos coletados nos CloudTrail registros. Para obter mais informações, consulte:

- Visão geral da criação de uma trilha
- CloudTrail Serviços e integrações compatíveis
- Configurando notificações do Amazon SNS para CloudTrail
- Recebendo arquivos de CloudTrail log de várias regiões e recebendo arquivos de CloudTrail log de várias contas

Todas AWS OpsWorks for Chef Automate as ações são registradas CloudTrail e documentadas na Referência da AWS OpsWorks for Chef Automate API. Por exemplo, chamadas para as DescribeServersações CreateServerCreateBackup, e geram entradas nos arquivos de CloudTrail log.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário-raiz ou usuário do IAM.
- Se a solicitação foi feita com credenciais de segurança temporárias de uma função ou de um usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte Elemento userIdentity do CloudTrail .

Entendendo as entradas do arquivo de AWS OpsWorks for Chef Automate log

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log para um bucket do Amazon S3 que você especificar. CloudTrail os arquivos de log contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros da solicitação e assim por diante. CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas de API, portanto, eles não aparecem em nenhuma ordem específica.

O exemplo a seguir mostra uma entrada de CloudTrail registro para a AWS OpsWorks for Chef Automate `CreateServer` ação.

```
{"eventVersion":"1.05",
"userIdentity":{
    "type":"AssumedRole",
    "principalId":"ID number:OpsWorksCMUser",
    "arn":"arn:aws:sts::831000000000:assumed-role/Admin/OpsWorksCMUser",
    "accountId":"831000000000","accessKeyId":"ID number",
    "sessionContext":{
        "attributes":{
            "mfaAuthenticated":"false",
            "creationDate":"2017-01-05T22:03:47Z"
            },
```

```
        "sessionIssuer":{
            "type":"Role",
            "principalId":"ID number",
            "arn":"arn:aws:iam::831000000000:role/Admin",
            "accountId":"831000000000",
            "userName":"Admin"
            }
        }
    },
"eventTime":"2017-01-05T22:18:23Z",
"eventSource":"opsworks-cm.amazonaws.com",
"eventName":"CreateServer",
"awsRegion":"us-west-2",
"sourceIPAddress":"101.25.190.51",
"userAgent":"console.amazonaws.com",
"requestParameters":{
    "serverName":"OpsChef-test-server",
    "engineModel":"Single",
    "engine":"Chef",
    "instanceProfileArn":"arn:aws:iam::831000000000:instance-profile/aws-opsworks-cm-
ec2-role",
    "backupRetentionCount":3,"serviceRoleArn":"arn:aws:iam::831000000000:role/service-
role/aws-opsworks-cm-service-role",
    "engineVersion":"12",
    "preferredMaintenanceWindow":"Fri:21:00",
    "instanceType":"t2.medium",
    "subnetIds":["subnet-1e111f11"],
    "preferredBackupWindow":"Wed:08:00"
    },
"responseElements":{
    "server":{
        "endpoint":"OpsChef-test-server-thohsgreckcnwgz3.us-west-2.opsworks-cm.io",
        "createdAt":"Jan 5, 2017 10:18:22 PM",
        "serviceRoleArn":"arn:aws:iam::831000000000:role/service-role/aws-opsworks-cm-
service-role",
        "preferredBackupWindow":"Wed:08:00",
        "status":"CREATING",
        "subnetIds":["subnet-1e111f11"],
        "engine":"Chef",
        "instanceType":"t2.medium",
        "serverName":"OpsChef-test-server",
        "serverArn":"arn:aws:opsworks-cm:us-west-2:831000000000:server/OpsChef-test-
server/8epp7f6z-e91f-4z10-89z5-8c6219cdb09f",
        "engineModel":"Single",
```

```
        "backupRetentionCount":3,
        "engineAttributes":[
            {"name":"CHEF_STARTER_KIT","value":"*** Redacted ***"},
            {"name":"CHEF_PIVOTAL_KEY","value":"*** Redacted ***"},
            {"name":"CHEF_DELIVERY_ADMIN_PASSWORD","value":"*** Redacted ***"}],
        "engineVersion":"12.11.1",
        "instanceProfileArn":"arn:aws:iam::831000000000:instance-profile/aws-opsworks-
cm-ec2-role",
        "preferredMaintenanceWindow":"Fri:21:00"
        }
    },
"requestID":"de7f64f9-d394-12ug-8081-7bb0386fbcb6",
"eventID":"8r7b18df-6c90-47be-87cf-e8346428cfc3",
"eventType":"AwsApiCall",
"recipientAccountId":"831000000000"
}
```


Solução de problemas AWS OpsWorks for Chef Automate

> **Important**
>
> AWS OpsWorks O Chef Automate chegou ao fim da vida útil em 5 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes existentes migrem para o Chef SaaS ou uma solução alternativa. Se você tiver dúvidas, pode entrar em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico contém alguns AWS OpsWorks for Chef Automate problemas comuns e sugestões de soluções para esses problemas.

Tópicos

- Dicas para solução de problemas em geral
- Solução de problemas de erros específicos
- Ajuda e suporte adicionais

Dicas para solução de problemas em geral

Caso não consiga criar ou trabalhar com um servidor do Chef, você pode exibir mensagens de erro ou logs para ajudar a solucionar o problema. As tarefas a seguir descrevem locais em geral para iniciar quando você está solucionando um problema no servidor do Chef. Para obter mais informações sobre erros específicos e soluções, consulte a seção Solução de problemas de erros específicos deste tópico.

- Use o AWS OpsWorks for Chef Automate console para visualizar mensagens de erro se um servidor Chef falhar ao iniciar. Na página de detalhes do servidor do Chef, as mensagens de erro relacionadas à inicialização e à execução do servidor são mostradas na parte superior da página. Os erros podem vir de AWS OpsWorks for Chef Automate AWS CloudFormation, ou da Amazon EC2, serviços usados para criar um servidor Chef. Na página de detalhes, você também pode visualizar eventos que ocorrem em um servidor em execução, que podem conter mensagens de evento de falha.
- Para ajudar a resolver EC2 problemas, conecte-se à instância do seu servidor usando SSH e visualize os registros. EC2 os registros da instância são armazenados no `/var/log/aws/opsworks-cm` diretório. Esses registros capturam as saídas do comando enquanto AWS OpsWorks for Chef Automate inicia um servidor Chef.

Solução de problemas de erros específicos

Tópicos

- O servidor está em um estado de conexão perdida
- Nó gerenciado aparece no painel Automate do Chef, na coluna Missing
- Não é possível criar um cofre do Chef; falha no comando knife vault com erros
- Falha na criação do servidor com a mensagem "requested configuration is currently not supported"
- O servidor do Chef não reconhece os nomes da organização adicionados no painel do Chef Automatize
- Não é possível criar a EC2 instância Amazon do servidor
- Erro na função de serviço impede a criação do servidor
- Limite de endereço IP elástico excedido
- Não é possível fazer logon no painel do Chef Automatize
- Falha na associação do nó autônomo

- Falha na manutenção do sistema

O servidor está em um estado de conexão perdida

Problema: o status de um servidor é exibido como Conexão perdida.

Causa: Isso geralmente ocorre quando uma entidade externa AWS OpsWorks faz alterações em um AWS OpsWorks for Chef Automate servidor ou em seus recursos de suporte. AWS OpsWorks não pode se conectar aos servidores do Chef Automate nos estados perdidos do Connection para lidar com tarefas de manutenção, como criar backups, aplicar patches do sistema operacional ou atualizar o Chef Automate. Como resultado, seu servidor pode estar perdendo atualizações importantes, suscetível a problemas de segurança ou não funcionando conforme o esperado.

Solução: tente as etapas a seguir para restaurar a conexão do servidor.

1. Verifique se seu perfil de serviço tem todas as permissões necessárias.
   a. Na página Configurações do seu servidor, em Rede e segurança, escolha o link para o perfil de serviço que o servidor está usando. Isso abrirá o perfil de serviço para visualização no console do IAM.
   b. Na guia Permissões, verifique se `AWSOpsWorksCMServiceRole` está na lista de Políticas de permissões. Se ela não estiver listada, adicione a política `AWSOpsWorksCMServiceRole` gerenciada manualmente ao perfil.
   c. Na guia Relações de confiança, verifique se o perfil de serviço tem uma política de confiança que confia no serviço `opsworks-cm.amazonaws.com` para assumir perfis em seu nome. Para obter mais informações sobre como usar políticas de confiança com funções, consulte Modificação de uma função (console) ou a postagem do blog de AWS segurança, Como usar políticas de confiança com funções do IAM.
2. Verifique se seu perfil de instância tem todas as permissões necessárias.
   a. Na página Configurações do seu servidor, em Rede e segurança, escolha o link para o perfil de instância que o servidor está usando. Isso abrirá o perfil de instância para visualização no console do IAM.
   b. Na guia Permissões, verifique se `AmazonEC2RoleforSSM` e `AWSOpsWorksCMInstanceProfileRole` estão na lista de Políticas de permissões. Se uma ou ambas não estiverem listadas, adicione essas políticas gerenciadas manualmente ao perfil.

   c. Na guia Relações de confiança, verifique se o perfil de serviço tem uma política de confiança que confia no serviço `ec2.amazonaws.com` para assumir perfis em seu nome. Para obter mais informações sobre como usar políticas de confiança com funções, consulte Modificação de uma função (console) ou a postagem do blog de AWS segurança, Como usar políticas de confiança com funções do IAM.

3. No EC2 console da Amazon, verifique se você está na mesma região do AWS OpsWorks for Chef Automate servidor e, em seguida, reinicie a EC2 instância que seu servidor está usando.

   a. Escolha a EC2 instância nomeada `aws-opsworks-cm-instance-`*`server-name`*.

   b. Escolha Estado da instância e Reinicializar instância no menu.

   c. Aguarde até 15 minutos para que seu servidor reinicie e fique totalmente on-line.

4. No AWS OpsWorks for Chef Automate console, na página de detalhes do servidor, verifique se o status do servidor agora está íntegro.

Se o status do servidor ainda for Conexão perdida após a execução das etapas anteriores, tente uma das opções a seguir.

- Substitua o servidor criando um novo e excluindo o original. Se os dados do servidor atual forem importantes para você, restaure o servidor a partir de um backup recente e verifique se os dados estão atualizados antes de excluir o servidor original que não responde.
- Entre em contato com o suporte da AWS.

Nó gerenciado aparece no painel Automate do Chef, na coluna Missing

Problema: um nó gerenciado aparece na coluna Missing do painel do Chef Automate.

Motivo: quando um nó não se conecta ao servidor do Chef Automate por mais de 12 horas, e `chef-client` não pode ser executado no nó, ele muda do estado em que estava antes do período de 12 horas e avança para a coluna Missing do painel Chef Automate.

Solução: Verifique se o nó está online. Tente executar `knife node show` *`node_name`* `--run-list` para saber se `chef-client` pode ser executado no nó, ou `knife node show -l` *`node_name`* para exibir todas as informações sobre o nó. O nó pode estar off-line ou desconectado da rede.

Não é possível criar um cofre do Chef; falha no comando **`knife vault`** com erros

Problema: Você está tentando criar um cofre no servidor Chef Automate (como um cofre para armazenar credenciais para nós baseados no Windows de ingresso no domínio), executando o comando `knife vault`. O comando retorna uma mensagem de erro semelhante à seguinte.

```
WARN: Auto inflation of JSON data is deprecated. Please pass in the class to inflate or
 use #edit_hash (CHEF-1)
at /opt/chefdk/embedded/lib/ruby/2.3.0/forwardable.rb:189:in `edit_data'.Please see
 https://docs.chef.io/deprecations_json_auto_inflate.html
for further details and information on how to correct this problem.
WARNING: pivotal not found in users, trying clients.
ERROR: ChefVault::Exceptions::AdminNotFound: FATAL: Could not find pivotal in users or
 clients!
```

O usuário essencial não é retornado quando você executa `knife user list` remotamente, embora possa vê-lo em resultados ao executar o comando `chef-server-ctl user-show` localmente no servidor do Chef Automatize. Em outras palavras, o comando `knife vault` não pode encontrar o usuário essencial, mas você sabe que ele existe.

Causa: Embora seja considerado o superusuário no Chef e tenha permissões completas, o usuário essencial não é membro de nenhuma organização, inclusive a organização `default` usada no AWS OpsWorks for Chef Automate. O comando `knife user list` retorna todos os usuários presentes na organização atual na configuração do Chef. O comando `chef-server-ctl user-show` retorna todos os usuários, independentemente da organização, inclusive o usuário essencial.

Solução: Para corrigir o problema, adicione o usuário essencial à organização padrão executando `knife opc`.

Primeiro, você precisará instalar o plug-in knife-opc.

```
chef gem install knife-opc
```

Depois de instalar o plug-in, execute o comando a seguir para adicionar o usuário essencial à organização padrão.

```
knife opc org user add default pivotal
```

Você pode verificar se o usuário essencial faz parte da organização padrão executando o `knife user list` novamente. O `pivotal` deve ser listado nos resultados. Em seguida, tente reexecutar `knife vault`.

Falha na criação do servidor com a mensagem "requested configuration is currently not supported"

Problema: Você está tentando criar um servidor do Chef Automatize, mas a criação do servidor falha com uma mensagem de erro semelhante a "The requested configuration is currently not supported. Please check the documentation for supported configurations."

Causa: Um tipo de instância não compatível pode ter sido especificado para o servidor do Chef Automatize. Se você optar por criar o servidor do Chef Automatize em uma VPC que tenha uma locação não padrão, como uma para instâncias dedicadas, todas as instâncias dentro da VPC especificada também devem ser de locação dedicada ou host. Como alguns tipos de instância, como t2, só estão disponíveis com a locação padrão, o tipo de instância do servidor do Chef Automatize talvez não seja compatível com a VPC especificada, e a criação do servidor falha.

Solução: Se você escolher uma VPC que tenha uma locação não padrão, use um tipo de instância m4, que pode dar suporte à locação dedicada.

O servidor do Chef não reconhece os nomes da organização adicionados no painel do Chef Automatize

Problema: Você adicionou novos nomes de organização de fluxo de trabalho no painel do Chef Automatize ou especificou um valor `CHEF_AUTOMATE_ORGANIZATION` diferente de `"default"` no script de associação de nó autônomo, mas a associação do nó falha. O servidor do AWS OpsWorks for Chef Automate não reconhece os novos nomes de organização.

Causa: Os nomes de organização do fluxo de trabalho e os nomes de organização do servidor do Chef não são iguais. Você pode criar novas organizações de fluxo de trabalho no painel do Chef Automate baseado na web, mas não nomes de organização do servidor do Chef. Você só pode usar o painel Chef Automatize para visualizar as organizações de servidor do Chef existentes. Uma nova organização que você cria no painel Chef Automate é uma organização de fluxo de trabalho, e não é reconhecida pelo servidor do Chef. Você não pode criar novos nomes de organização especificando-os no script de associação do nó. A consulta ao nome de uma organização em um script de associação do nó quando a organização não tiver sido adicionada primeiro ao servidor do Chef fará a associação do nó falhar.

Solução: Para criar novas organizações reconhecidas no servidor do Chef, use o comando `knife opc org create` ou execute `chef-server-ctl org-create`.

Não é possível criar a EC2 instância Amazon do servidor

Problema: a criação do servidor falhou com uma mensagem de erro semelhante à seguinte: “Falha na criação dos seguintes recursos: [EC2Instância]. Failed to receive 1 resource signal(s) within the specified duration."

Causa: Isso provavelmente ocorre porque a EC2 instância não tem acesso à rede.

Solução: certifique-se de que a instância tenha acesso de saída à Internet e que o agente AWS de serviço seja capaz de emitir comandos. Certifique-se de que a VPC (uma VPC com uma única sub-rede pública) tenha DNS resolution ativado e que a sub-rede tenha a configuração Auto-assign Public IP ativada.

Erro na função de serviço impede a criação do servidor

Problema: A criação do servidor falha com uma mensagem de erro que diz: “Não autorizado a realizar sts:”AssumeRole.

Causa: Isso pode ocorrer quando a função de serviço que você está usando não tem permissões adequadas para criar um novo servidor.

Solução: abra o AWS OpsWorks for Chef Automate console; use o console para gerar uma nova função de serviço e uma função de perfil de instância. Se você preferir usar sua própria função de serviço, anexe a AWSOps política de CMServiceFunção de Trabalho à função. Verifique se opsworks-cm.amazonaws.com está listado entre os serviços nas Relações de confiança do perfil. Verifique se a função de serviço associada ao servidor Chef tem a política gerenciada do AWSOpsWorks CMService Role anexada.

Limite de endereço IP elástico excedido

Problema: a criação do servidor falha com uma mensagem de erro que diz: “Falha na criação dos seguintes recursos: [EIP, EC2 Instância]. Resource creation cancelled, the maximum number of addresses has been reached."

Causa: Isso ocorre quando a conta usou o número máximo de endereços Elastic IP (EIP – IP elástico). O limite de endereços EIP padrão é cinco.

Solução: você pode liberar endereços EIP existentes ou excluir aqueles que sua conta não está usando ativamente, ou pode entrar em contato com o Suporte AWS ao Cliente para aumentar o limite de endereços EIP associados à sua conta.

Não é possível fazer logon no painel do Chef Automatize

Problema: O painel do Chef Automate mostra um erro semelhante ao seguinte: “Solicitação de origem cruzada bloqueada: a mesma política de origem não permite a leitura do recurso remoto na https://myserver-name.region.opsworks-cm.io/api/v0/e/default/verify-token. (Reason: CORS header 'Access-Control-Allow-Origin' missing)". The error can also be similar to "The User Id / Password com binação inserida e está incorreta”.

Causa: O painel do Chef Automate define explicitamente o FQDN e não aceita o relativo. URLs No momento, você não pode fazer logon usando o endereço IP do servidor do Chef; você só pode fazer logon usando o nome DNS do servidor.

Solução: Faça login no painel do Chef Automatize apenas usando a entrada de nome DNS do servidor do Chef, e não o endereço IP. Você também pode tentar redefinir as credenciais do painel do Chef Automate executando um comando da AWS CLI , conforme descrito em Redefinição das credenciais do painel do Chef Automate.

Falha na associação do nó autônomo

Problema: a associação autônoma ou automática de novos EC2 nós da Amazon está falhando. Os nós que devem ter sido adicionados ao servidor do Chef não estão aparecendo no painel do Chef Automate, e não são listados em resultados dos comandos `knife client show` ou `knife node show`.

Causa: isso pode ocorrer quando você não tem uma função do IAM configurada como um perfil de instância que permita que as chamadas de `opsworks-cm` API se comuniquem com novas EC2 instâncias.

Solução: anexe uma política ao seu perfil de EC2 instância que permita que as chamadas de `DescribeNodeAssociationStatus` API `AssociateNode` e as chamadas de API funcionem EC2, conforme descrito emAdicione nós automaticamente em AWS OpsWorks for Chef Automate.

Falha na manutenção do sistema

AWS OpsWorks CM realiza manutenção semanal do sistema para garantir que as versões secundárias mais recentes do Chef Server e do Chef Automate Server, incluindo atualizações de segurança, estejam sempre em execução em um servidor AWS OpsWorks for Chef Automate. Se,

por algum motivo, a manutenção do sistema falhar, AWS OpsWorks CM notifica você sobre a falha. Para obter mais informações sobre manutenção do sistema, consulte Manutenção do sistema em AWS OpsWorks for Chef Automate.

Esta seção descreve os possíveis motivos da falha e sugere soluções.

Tópicos

- Erro no perfil de serviço ou no perfil de instância impede a manutenção do sistema

Erro no perfil de serviço ou no perfil de instância impede a manutenção do sistema

Problema: A manutenção do sistema falha com uma mensagem de erro que diz: “Não autorizado a executar sts: AssumeRole “ou uma mensagem de erro semelhante sobre permissões.

Causa: isso pode ocorrer quando o perfil de serviço ou o perfil de instância que você está usando não têm permissões adequadas para realizar a manutenção do sistema no servidor.

Solução: verifique se seu perfil de serviço e perfil de instância têm todas as permissões necessárias.

1. Verifique se seu perfil de serviço tem todas as permissões necessárias.
   a. Na página Configurações do seu servidor, em Rede e segurança, escolha o link para o perfil de serviço que o servidor está usando. Isso abrirá o perfil de serviço para visualização no console do IAM.
   b. Na guia Permissões, verifique se `AWSOpsWorksCMServiceRole` está anexado ao perfil de serviço. Se `AWSOpsWorksCMServiceRole` não estiver listada, adicione essa política ao perfil.
   c. Verifique se opsworks-cm.amazonaws.com está listado entre os serviços nas Relações de confiança do perfil. Para obter mais informações sobre como usar políticas de confiança com funções, consulte Modificação de uma função (console) ou a postagem do blog de AWS segurança, Como usar políticas de confiança com funções do IAM.
2. Verifique se seu perfil de instância tem todas as permissões necessárias.
   a. Na página Configurações do seu servidor, em Rede e segurança, escolha o link para o perfil de instância que o servidor está usando. Isso abrirá o perfil de instância para visualização no console do IAM.
   b. Na guia Permissões, verifique se `AmazonEC2RoleforSSM` e `AWSOpsWorksCMInstanceProfileRole` estão na lista de Políticas de permissões. Se

uma ou ambas não estiverem listadas, adicione essas políticas gerenciadas manualmente ao perfil.

c. Na guia Relações de confiança, verifique se o perfil de serviço tem uma política de confiança que confia no serviço `ec2.amazonaws.com` para assumir perfis em seu nome. Para obter mais informações sobre como usar políticas de confiança com funções, consulte Modificação de uma função (console) ou a postagem do blog de AWS segurança, Como usar políticas de confiança com funções do IAM.

Ajuda e suporte adicionais

Caso você não veja o problema específico descrito neste tópico ou tenha tentado as sugestões neste tópico e ainda esteja enfrentando problemas, visite os fóruns do AWS OpsWorks.

Você também pode acessar o AWS Support Center. O AWS Support Center é o hub para criar e gerenciar casos de AWS suporte. O AWS Support Center também inclui links para outros recursos úteis, como fóruns, informações técnicas FAQs, status de integridade do serviço AWS Trusted Advisor e.

Segurança no gerenciamento de AWS OpsWorks configuração (CM)

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de uma arquitetura de data center e rede criada para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O modelo de responsabilidade compartilhada descreve isto como segurança da nuvem e segurança na nuvem:

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços na AWS nuvem. AWS também fornece serviços que você pode usar com segurança. Auditores de terceiros testam e verificam regularmente a eficácia da nossa segurança como parte dos programas de conformidade da AWS. Para saber mais sobre os programas de conformidade que se aplicam ao AWS OpsWorks CM, consulte Serviços da AWS no escopo por programa de conformidade.
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da empresa e as leis e regulamentos aplicáveis.

Esta documentação ajuda você a entender como aplicar o modelo de responsabilidade compartilhada ao usar o AWS OpsWorks CM. Os tópicos a seguir mostram como configurar o AWS OpsWorks CM para atender aos seus objetivos de segurança e conformidade. Você também aprende a usar outros serviços da AWS que ajudam você a monitorar e proteger seus recursos de AWS OpsWorks CM.

Tópicos

- Proteção de dados no AWS OpsWorks CM
- Criptografia de dados
- Identity and Access Management para AWS OpsWorks CM
- Privacidade do tráfego entre redes
- Registro e monitoramento no AWS OpsWorks CM
- Validação de conformidade do AWS OpsWorks CM
- Resiliência em CM AWS OpsWorks

- Segurança da infraestrutura no AWS OpsWorks CM
- Análise de configuração e vulnerabilidade no AWS OpsWorks CM
- Melhores práticas de segurança para AWS OpsWorks CM

Proteção de dados no AWS OpsWorks CM

O modelo de responsabilidade AWS compartilhada de se aplica à proteção de dados no AWS OpsWorks Configuration Management. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa todos os Nuvem AWS. Você é responsável por manter o controle sobre o conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para obter mais informações sobre a privacidade de dados, consulte as Data Privacy FAQ. Para obter mais informações sobre a proteção de dados na Europa, consulte a postagem do blog AWS Shared Responsibility Model and RGPD no Blog de segurança da AWS .

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com os recursos. AWS Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail. Para obter informações sobre o uso de CloudTrail trilhas para capturar AWS atividades, consulte Como trabalhar com CloudTrail trilhas no Guia AWS CloudTrail do usuário.
- Use soluções de AWS criptografia, juntamente com todos os controles de segurança padrão Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sigilosos armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-3 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para obter mais informações sobre os endpoints FIPS disponíveis, consulte Federal Information Processing Standard (FIPS) 140-3.

É altamente recomendável que nunca sejam colocadas informações confidenciais ou sigilosas, como endereços de e-mail de clientes, em tags ou campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com o OpsWorks CM ou outro Serviços da AWS usando o console, a API ou AWS SDKs. AWS CLI Quaisquer dados inseridos em tags ou em campos de texto de formato livre usados para nomes podem ser usados para logs de faturamento ou de diagnóstico. Se você fornecer um URL para um servidor externo, é fortemente recomendável que não sejam incluídas informações de credenciais no URL para validar a solicitação nesse servidor.

Os nomes dos servidores OpsWorks CM não são criptografados.

OpsWorks O CM coleta os seguintes dados do cliente durante a criação e manutenção de seus AWS OpsWorks for Chef Automate AWS OpsWorks for Puppet Enterprise servidores.

- OpsWorks Para o Puppet Enterprise, coletamos chaves privadas que o Puppet Enterprise usa para permitir a comunicação entre seu mestre do Puppet e os nós gerenciados.
- Pois AWS OpsWorks for Chef Automate, coletamos chaves privadas para certificados que você anexa ao serviço se estiver usando um domínio personalizado. A chave privada que você fornece quando está criando um servidor do Chef Automatize com um domínio personalizado é transmitida ao servidor.

OpsWorks Os servidores CM armazenam seu código de configuração, como livros de receitas do Chef ou módulos do Puppet Enterprise. Embora esse código esteja armazenado nos backups do servidor, AWS não tem acesso a ele. Esse conteúdo é criptografado e somente os administradores da sua AWS conta podem acessá-lo. Recomendamos que você proteja o código de configuração do Chef ou do Puppet usando protocolos recomendados para os repositórios de origem. Por exemplo, você pode restringir as permissões aos repositórios ou seguir as diretrizes do GitHub site para proteger GitHub os repositórios. AWS CodeCommit

OpsWorks O CM não usa conteúdo fornecido pelo cliente para manter o serviço ou manter registros do cliente. Os registros sobre seus servidores OpsWorks CM são armazenados em sua conta, nos buckets do Amazon S3. Os endereços IP dos usuários que se conectam aos seus servidores OpsWorks CM são registrados por AWS.

Integração com AWS Secrets Manager

A partir de 3 de maio de 2021, quando você cria um novo servidor no OpsWorks CM, o OpsWorks CM armazena segredos do servidor em AWS Secrets Manager. Para novos servidores, os atributos a seguir são armazenados como segredos no Secrets Manager.

- Servidor Chef Automate
  - Chave privada HTTPS (somente servidores que não usam um domínio personalizado)
  - Senha administrativa do Chef Automate (CHEF_AUTOMATE_ADMIN_PASSWORD)
- Master da Puppet Enterprise
  - Chave privada HTTPS (somente servidores que não usam um domínio personalizado)
  - Senha administrativa do Puppet (PUPPET_ADMIN_PASSWORD)
  - Controle remoto Puppet r10k (PUPPET_R10K_REMOTE)

Para servidores existentes que não usam um domínio personalizado, o único segredo armazenado no Secrets Manager, para os servidores Chef Automate e Puppet Enterprise, é a chave privada HTTPS, porque ela é gerada durante a manutenção automática semanal do sistema.

OpsWorks O CM armazena segredos no Secrets Manager automaticamente, e esse comportamento não é configurável pelo usuário.

Criptografia de dados

AWS OpsWorks O CM criptografa os backups do servidor e a comunicação entre AWS usuários autorizados e seus servidores AWS OpsWorks CM. No entanto, os volumes raiz do Amazon EBS dos servidores AWS OpsWorks CM não são criptografados.

Criptografia em repouso

AWS OpsWorks Os backups do servidor CM são criptografados. No entanto, os volumes raiz do Amazon EBS dos servidores AWS OpsWorks CM não são criptografados. Isso não é configurável pelo usuário.

Criptografia em trânsito

AWS OpsWorks O CM usa HTTP com criptografia TLS. AWS OpsWorks O CM usa certificados autoassinados como padrão para provisionar e gerenciar servidores, se nenhum certificado assinado for fornecido pelos usuários. Recomendamos que utilize um certificado assinado por uma autoridade de certificação (CA).

Gerenciamento de chaves

AWS Key Management Service Atualmente, as chaves gerenciadas pelo cliente e as chaves gerenciadas pela AWS não são suportadas pelo AWS OpsWorks CM.

Identity and Access Management para AWS OpsWorks CM

AWS Identity and Access Management (IAM) é um AWS serviço que ajuda o administrador a controlar com segurança o acesso aos AWS recursos. Os administradores do IAM controlam quem pode ser autenticado (conectado) e autorizado (tem permissões) a usar os recursos do OpsWorks CM. O IAM é um AWS serviço que você pode usar sem custo adicional.

Tópicos

- Público
- Autenticação com identidades
- Gerenciamento do acesso usando políticas
- Como o AWS OpsWorks CM funciona com o IAM
- AWS OpsWorks Exemplos de políticas baseadas em identidade do CM
- Solução de problemas AWS OpsWorks de identidade e acesso do CM
- AWS políticas gerenciadas para gerenciamento de AWS OpsWorks configuração
- Prevenção confusa de delegados entre serviços em AWS OpsWorks CM

Público

A forma como você usa AWS Identity and Access Management (IAM) difere, dependendo do trabalho que você faz no OpsWorks CM.

Usuário do serviço — Se você usar o serviço OpsWorks CM para fazer seu trabalho, seu administrador fornecerá as credenciais e as permissões de que você precisa. À medida que você usa mais recursos do OpsWorks CM para fazer seu trabalho, talvez precise de permissões adicionais. Compreenda como o acesso é gerenciado pode ajudar a solicitar as permissões corretas ao administrador. Se você não conseguir acessar um recurso no OpsWorks CM, consulteSolução de problemas AWS OpsWorks de identidade e acesso do CM.

Administrador de serviços — Se você é responsável pelos recursos de OpsWorks CM em sua empresa, provavelmente tem acesso total ao OpsWorks CM. É seu trabalho determinar quais

recursos e recursos do OpsWorks CM seus usuários do serviço devem acessar. Envie as solicitações ao administrador do IAM para alterar as permissões dos usuários de serviço. Revise as informações nesta página para compreender os conceitos básicos do IAM. Para saber mais sobre como sua empresa pode usar o IAM com o OpsWorks CM, consulteComo o AWS OpsWorks CM funciona com o IAM.

Administrador do IAM — Se você for administrador do IAM, talvez queira saber detalhes sobre como criar políticas para gerenciar o acesso ao OpsWorks CM. Para ver exemplos de políticas baseadas em identidade do OpsWorks CM que você pode usar no IAM, consulte. AWS OpsWorks Exemplos de políticas baseadas em identidade do CM

Autenticação com identidades

A autenticação é a forma como você faz login AWS usando suas credenciais de identidade. Você deve estar autenticado (conectado AWS) como usuário do Usuário raiz da conta da AWS IAM ou assumindo uma função do IAM.

Você pode entrar AWS como uma identidade federada usando credenciais fornecidas por meio de uma fonte de identidade. AWS IAM Identity Center Usuários (IAM Identity Center), a autenticação de login único da sua empresa e suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como identidade federada, o administrador já configurou anteriormente a federação de identidades usando perfis do IAM. Ao acessar AWS usando a federação, você está assumindo indiretamente uma função.

Dependendo do tipo de usuário que você é, você pode entrar no AWS Management Console ou no portal de AWS acesso. Para obter mais informações sobre como fazer login em AWS, consulte Como fazer login Conta da AWS no Guia do Início de Sessão da AWS usuário.

Se você acessar AWS programaticamente, AWS fornece um kit de desenvolvimento de software (SDK) e uma interface de linha de comando (CLI) para assinar criptograficamente suas solicitações usando suas credenciais. Se você não usa AWS ferramentas, você mesmo deve assinar as solicitações. Para obter mais informações sobre como usar o método recomendado para designar solicitações por conta própria, consulte Versão 4 do AWS Signature para solicitações de API no Guia do usuário do IAM.

Independente do método de autenticação usado, também pode ser necessário fornecer informações adicionais de segurança. Por exemplo, AWS recomenda que você use a autenticação multifator (MFA) para aumentar a segurança da sua conta. Para saber mais, consulte Autenticação multifator

no Guia do usuário do AWS IAM Identity Center e Usar a autenticação multifator da AWS no IAM no Guia do usuário do IAM.

Conta da AWS usuário root

Ao criar uma Conta da AWS, você começa com uma identidade de login que tem acesso completo a todos Serviços da AWS os recursos da conta. Essa identidade é chamada de usuário Conta da AWS raiz e é acessada fazendo login com o endereço de e-mail e a senha que você usou para criar a conta. É altamente recomendável não usar o usuário-raiz para tarefas diárias. Proteja as credenciais do usuário-raiz e use-as para executar as tarefas que somente ele puder executar. Para obter a lista completa das tarefas que exigem login como usuário-raiz, consulte Tarefas que exigem credenciais de usuário-raiz no Guia do Usuário do IAM.

Grupos e usuários do IAM

Um usuário do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas para uma única pessoa ou aplicativo. Sempre que possível, é recomendável contar com credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM, é recomendável alternar as chaves de acesso. Para obter mais informações, consulte Alternar as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo no Guia do Usuário do IAM.

Um grupo do IAM é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível usar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdminse conceder a esse grupo permissões para administrar recursos do IAM.

Usuários são diferentes de perfis. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas um perfil pode ser assumido por qualquer pessoa que precisar dele. Os usuários têm credenciais permanentes de longo prazo, mas os perfis fornecem credenciais temporárias. Para saber mais, consulte Casos de uso para usuários do IAM no Guia do usuário do IAM.

> **Warning**
>
> Os usuários do IAM têm credenciais de longo prazo, o que representa um risco de segurança. Para ajudar a reduzir esse risco, recomendamos que você forneça a esses

usuários somente as permissões necessárias para realizar a tarefa e que você os remova quando não forem mais necessários.

Perfis do IAM

Uma função do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas. Ele é semelhante a um usuário do IAM, mas não está associado a uma pessoa específica. Para assumir temporariamente uma função do IAM no AWS Management Console, você pode alternar de um usuário para uma função do IAM (console). Você pode assumir uma função chamando uma operação de AWS API AWS CLI ou usando uma URL personalizada. Para obter mais informações sobre métodos para usar perfis, consulte Métodos para assumir um perfil no Guia do usuário do IAM.

Perfis do IAM com credenciais temporárias são úteis nas seguintes situações:

- Acesso de usuário federado: para atribuir permissões a identidades federadas, é possível criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas por ele. Para ter mais informações sobre perfis para federação, consulte Criar um perfil para um provedor de identidade de terceiros (federação) no Guia do usuário do IAM. Se usar o Centro de Identidade do IAM, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o Centro de Identidade do IAM correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte Conjuntos de Permissões no Guia do Usuário do AWS IAM Identity Center .
- Permissões temporárias para usuários do IAM: um usuário ou um perfil do IAM pode presumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso entre contas: é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, com alguns Serviços da AWS, você pode anexar uma política diretamente a um recurso (em vez de usar uma função como proxy). Para conhecer a diferença entre perfis e políticas baseadas em recurso para acesso entre contas, consulte Acesso a recursos entre contas no IAM no Guia do usuário do IAM.
- Acesso entre serviços — Alguns Serviços da AWS usam recursos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicativos na Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso usando as permissões da entidade principal da chamada, usando um perfil de serviço ou um perfil vinculado ao serviço.

- Sessões de acesso direto (FAS) — Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Sessões de acesso direto.
- Perfil de serviço: um perfil de serviço é um perfil do IAM que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um AWS service (Serviço da AWS) no Guia do Usuário do IAM.
- Função vinculada ao serviço — Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode presumir o perfil de executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não editar as permissões para perfis vinculados ao serviço.

- Aplicativos em execução na Amazon EC2 — Você pode usar uma função do IAM para gerenciar credenciais temporárias para aplicativos que estão sendo executados em uma EC2 instância e fazendo solicitações AWS CLI de AWS API. Isso é preferível a armazenar chaves de acesso na EC2 instância. Para atribuir uma AWS função a uma EC2 instância e disponibilizá-la para todos os aplicativos, você cria um perfil de instância anexado à instância. Um perfil de instância contém a função e permite que programas em execução na EC2 instância recebam credenciais temporárias. Para obter mais informações, consulte Usar uma função do IAM para conceder permissões a aplicativos executados em EC2 instâncias da Amazon no Guia do usuário do IAM.

Gerenciamento do acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política é um objeto AWS que, quando associada a uma identidade ou recurso, define suas permissões. AWS avalia essas políticas quando um principal (usuário, usuário raiz ou sessão de função) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas é armazenada AWS como documentos JSON. Para obter mais

informações sobre a estrutura e o conteúdo de documentos de políticas JSON, consulte Visão geral das políticas JSON no Guia do usuário do IAM.

Os administradores podem usar políticas AWS JSON para especificar quem tem acesso ao quê. Ou seja, qual entidade principal pode executar ações em quais recursos e em que condições.

Por padrão, usuários e perfis não têm permissões. Para conceder permissão aos usuários para executar ações nos recursos que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis e os usuários podem assumir os perfis.

As políticas do IAM definem permissões para uma ação independentemente do método usado para executar a operação. Por exemplo, suponha que você tenha uma política que permite a ação `iam:GetRole`. Um usuário com essa política pode obter informações de função da AWS Management Console AWS CLI, da ou da AWS API.

Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e perfis podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Definir permissões personalizadas do IAM com as políticas gerenciadas pelo cliente no Guia do Usuário do IAM.

As políticas baseadas em identidade podem ser categorizadas como políticas em linha ou políticas gerenciadas. As políticas em linha são anexadas diretamente a um único usuário, grupo ou perfil. As políticas gerenciadas são políticas autônomas que você pode associar a vários usuários, grupos e funções em seu Conta da AWS. As políticas AWS gerenciadas incluem políticas gerenciadas e políticas gerenciadas pelo cliente. Para saber como escolher entre uma política gerenciada ou uma política em linha, consulte Escolher entre políticas gerenciadas e políticas em linha no Guia do usuário do IAM.

OpsWorks O CM oferece suporte a políticas personalizadas que você cria no IAM e anexa a usuários, funções ou grupos.

Políticas baseadas em recurso

Políticas baseadas em recursos são documentos de políticas JSON que você anexa a um recurso. São exemplos de políticas baseadas em recursos as políticas de confiança de perfil do IAM e as

políticas de bucket do Amazon S3. Em serviços compatíveis com políticas baseadas em recursos, os administradores de serviço podem usá-las para controlar o acesso a um recurso específico. Para o atributo ao qual a política está anexada, a política define quais ações uma entidade principal especificado pode executar nesse atributo e em que condições. Você deve especificar uma entidade principal em uma política baseada em recursos. Os diretores podem incluir contas, usuários, funções, usuários federados ou. Serviços da AWS

Políticas baseadas em recursos são políticas em linha localizadas nesse serviço. Você não pode usar políticas AWS gerenciadas do IAM em uma política baseada em recursos.

OpsWorks O CM não oferece suporte a políticas baseadas em recursos.

Listas de controle de acesso (ACLs)

As listas de controle de acesso (ACLs) controlam quais diretores (membros da conta, usuários ou funções) têm permissões para acessar um recurso. ACLs são semelhantes às políticas baseadas em recursos, embora não usem o formato de documento de política JSON.

O Amazon S3 e o AWS WAF Amazon VPC são exemplos de serviços que oferecem suporte. ACLs Para saber mais ACLs, consulte a visão geral da lista de controle de acesso (ACL) no Guia do desenvolvedor do Amazon Simple Storage Service.

OpsWorks CM não usa ACLs.

Outros tipos de política

OpsWorks O CM não oferece suporte aos seguintes outros tipos de política.

AWS oferece suporte a tipos de políticas adicionais menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- Limites de permissões: Um limite de permissões é um atributo avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou função). É possível definir um limite de permissões para uma entidade. As permissões resultantes são a interseção das políticas baseadas em identidade da entidade e seus limites de permissões. As políticas baseadas em recursos que especificam o usuário ou o perfil no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações sobre limites de permissões, consulte Limites de permissões para identidades do IAM no Guia do usuário do IAM.

- Políticas de controle de serviço (SCPs) — SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (OU) em AWS Organizations. AWS Organizations é um serviço para agrupar e gerenciar centralmente várias AWS contas que sua empresa possui. Se você habilitar todos os recursos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as suas contas. O SCP limita as permissões para entidades nas contas dos membros, incluindo cada uma Usuário raiz da conta da AWS. Para obter mais informações sobre Organizations e SCPs, consulte How SCPs Work no Guia AWS Organizations do Usuário.
- Políticas de sessão: são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para um perfil ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou do perfil e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em recursos. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para obter mais informações, consulte Políticas de sessão no Guia do usuário do IAM.

Vários tipos de política

Quando vários tipos de política são aplicáveis a uma solicitação, é mais complicado compreender as permissões resultantes. Para saber como AWS determinar se uma solicitação deve ser permitida quando vários tipos de políticas estão envolvidos, consulte Lógica de avaliação de políticas no Guia do usuário do IAM.

Como o AWS OpsWorks CM funciona com o IAM

Antes de usar o IAM para gerenciar o acesso ao AWS OpsWorks CM, você deve entender quais recursos do IAM estão disponíveis para uso com o AWS OpsWorks CM. Para ter uma visão de alto nível de como o AWS OpsWorks CM e outros AWS serviços funcionam com o IAM, consulte AWS os serviços que funcionam com o IAM no Guia do usuário do IAM.

Tópicos

- AWS OpsWorks Políticas baseadas em identidade do CM
- AWS OpsWorks CM e políticas baseadas em recursos
- Autorização baseada em tags AWS OpsWorks CM
- AWS OpsWorks Funções do CM IAM

AWS OpsWorks Políticas baseadas em identidade do CM

Com as políticas baseadas em identidade do IAM, você pode especificar ações e recursos permitidos ou negados, bem como as condições sob as quais as ações são permitidas ou negadas. AWS OpsWorks O CM oferece suporte a ações, recursos e chaves de condição específicos. Para saber mais sobre todos os elementos usados em uma política JSON, consulte Referência de elementos de política JSON do IAM no Guia do usuário do IAM.

No AWS OpsWorks CM, você pode anexar uma declaração de política personalizada a um usuário, função ou grupo.

Ações

O elemento `Action` de uma política baseada em identidade do IAM descreve a ação ou ações específicas que serão permitidas ou negadas pela política. As ações de política geralmente têm o mesmo nome da operação de AWS API associada. A ação é usada em uma política para conceder permissões para executar a operação associada.

As ações políticas no AWS OpsWorks CM usam o seguinte prefixo antes da ação:`opsworks-cm:`. Por exemplo, para conceder a alguém permissão para criar um servidor do AWS OpsWorks CM usando uma operação de API, inclua a ação `opsworks-cm:CreateServer` na política da pessoa. As declarações de política devem incluir um `NotAction` elemento `Action` ou. AWS OpsWorks O CM define seu próprio conjunto de ações que descrevem as tarefas que você pode executar com esse serviço.

Para especificar várias ações em uma única instrução, separe-as com vírgulas, como segue:

```
"Action": [
      "opsworks-cm:action1",
      "opsworks-cm:action2"
```

Você também pode especificar várias ações usando caracteres curinga (*). Por exemplo, para especificar todas as ações que começam com a palavra `Describe`, inclua a seguinte ação:

```
"Action": "opsworks-cm:Describe*"
```

Ao usar caracteres curinga para permitir várias ações em uma instrução de política, tenha cuidado para permitir essas ações somente a serviços ou a usuários autorizados.

Para ver uma lista de ações do AWS OpsWorks CM, consulte Ações, recursos e chaves de condição para a AWS OpsWorks no Guia do usuário do IAM.

Recursos

O elemento `Resource` especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou um elemento `NotResource`. Você especifica um recurso usando um ARN ou usando o caractere curinga (*) para indicar que a instrução se aplica a todos os recursos.

Você pode obter o Amazon Resource Number (ARN) de um servidor AWS OpsWorks CM ou backup executando as DescribeServersoperações de DescribeBackupsAPI e basear as políticas em nível de recurso nesses recursos.

Um recurso de servidor AWS OpsWorks CM tem um ARN no seguinte formato:

```
arn:aws:opsworks-cm:{Region}:${Account}:server/${ServerName}/${UniqueId}
```

Um recurso de backup do AWS OpsWorks CM tem um ARN no seguinte formato:

```
arn:aws:opsworks-cm:{Region}:${Account}:backup/${ServerName}-{Date-and-Time-Stamp-of-
Backup}
```

Para obter mais informações sobre o formato de ARNs, consulte Amazon Resource Names (ARNs) e AWS Service Namespaces.

Por exemplo, para especificar o servidor `test-chef-automate` do Chef Automate na instrução, use o seguinte ARN:

```
"Resource": "arn:aws:opsworks-cm:us-west-2:123456789012:server/test-chef-automate/
EXAMPLE-d1a2bEXAMPLE"
```

Para especificar todos os servidores AWS OpsWorks CM que pertencem a uma conta específica, use o caractere curinga (*):

```
"Resource": "arn:aws:opsworks-cm:us-west-2:123456789012:server/*"
```

O exemplo a seguir especifica um backup do servidor AWS OpsWorks CM como um recurso:

```
"Resource": "arn:aws:opsworks-cm:us-west-2:123456789012:backup/test-chef-automate-
server-2018-05-20T19:06:12.399Z"
```

Algumas ações do AWS OpsWorks CM, como aquelas para criar recursos, não podem ser executadas em um recurso específico. Nesses casos, você deve utilizar o caractere curinga (*).

```
"Resource": "*"
```

Muitas ações da API do envolvem vários recursos. Para especificar vários recursos em uma única instrução, separe-os ARNs com vírgulas.

```
"Resource": [
      "resource1",
      "resource2"
```

Para ver uma lista dos tipos de recursos do AWS OpsWorks CM e seus ARNs, consulte Ações, recursos e chaves de condição para o AWS OpsWorks CM no Guia do usuário do IAM. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte Ações, recursos e chaves de condição para o AWS OpsWorks CM no Guia do usuário do IAM.

Chaves de condição

AWS OpsWorks O CM não tem chaves de contexto específicas do serviço que possam ser usadas no `Condition` elemento das declarações de política. Para obter a lista das chaves de contexto globais que estão disponíveis para todos os serviços, consulte chaves de contexto de condição globais da AWS na Referência de política do IAM. Para ver todas as chaves de condição AWS globais, consulte as chaves de contexto de condição AWS global no Guia do usuário do IAM.

O elemento `Condition` (ou bloco `Condition`) permite que você especifique condições nas quais uma instrução estiver em vigor. O elemento `Condition` é opcional. É possível criar expressões condicionais que usam operadores de condição, como "igual a" ou "menor que", para fazer a condição da política corresponder aos valores na solicitação.

Se você especificar vários elementos de `Condition` em uma declaração ou várias chaves em um único elemento de `Condition`, a AWS os avaliará usando uma operação lógica `AND`. Se você especificar vários valores para uma única chave de condição, AWS avalia a condição usando uma `OR` operação lógica. Todas as condições devem ser atendidas antes que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar condições. Por exemplo, você pode conceder a um usuário permissão para acessar um recurso somente se ele estiver marcado com seu nome de usuário. Para obter mais informações, consulte Elementos de política do IAM: variáveis e tags no Guia do usuário do IAM.

Exemplos

Para ver exemplos de políticas baseadas em identidade do AWS OpsWorks CM, consulte. AWS OpsWorks Exemplos de políticas baseadas em identidade do CM

AWS OpsWorks CM e políticas baseadas em recursos

AWS OpsWorks O CM não oferece suporte a políticas baseadas em recursos.

As políticas baseadas em recursos são documentos de políticas JSON que especificam quais ações uma entidade principal especificada pode executar em um recurso e sob quais condições.

Autorização baseada em tags AWS OpsWorks CM

Você pode anexar tags aos recursos do AWS OpsWorks CM ou passar tags em uma solicitação para o AWS OpsWorks CM. Para controlar o acesso com base em tags, forneça informações sobre as tags no elemento de condição de uma política usando as chaves de condição `aws:RequestTag/key-name` ou `aws:TagKeys`. Para obter mais informações sobre a marcação de recursos de AWS OpsWorks CM, consulte Trabalhando com tags em AWS OpsWorks for Chef Automate recursos ou Trabalhando com tags em AWS OpsWorks for Puppet Enterprise recursos neste guia.

AWS OpsWorks Funções do CM IAM

Uma função do IAM é uma entidade dentro da sua AWS conta que tem permissões específicas.

AWS OpsWorks O CM usa duas funções:

- Uma função de serviço que concede ao serviço AWS OpsWorks CM permissões para trabalhar na AWS conta de um usuário. Se você usar a função de serviço padrão fornecida pelo OpsWorks CM, o nome dessa função será`aws-opsworks-cm-service-role`.
- Uma função de perfil de instância que permite que o serviço AWS OpsWorks CM chame a API OpsWorks CM. Essa função concede acesso ao Amazon S3 e AWS CloudFormation à criação do servidor e do bucket do S3 para backups. Se você usar o perfil de instância padrão fornecido pelo OpsWorks CM, o nome dessa função de perfil de instância será`aws-opsworks-cm-ec2-role`.

AWS OpsWorks O CM não usa funções vinculadas a serviços.

Usar credenciais temporárias com o AWS OpsWorks CM

AWS OpsWorks O CM suporta o uso de credenciais temporárias e herda esse recurso de. AWS Security Token Service

É possível usar credenciais temporárias para fazer login com federação, assumir um perfil do IAM ou assumir um perfil entre contas. Você obtém credenciais de segurança temporárias chamando operações de AWS STS API, como AssumeRoleou GetFederationToken.

Funções vinculadas ao serviço

AWS OpsWorks O CM não usa funções vinculadas a serviços.

As funções vinculadas ao serviço permitem que AWS os serviços acessem recursos em outros serviços para concluir uma ação em seu nome. Os perfis vinculados a serviço aparecem em sua conta do IAM e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para perfis vinculados a serviço.

Perfis de serviço

Esse atributo permite que um serviço assuma um perfil de serviço em seu nome. O perfil permite que o serviço acesse recursos em outros serviços para concluir uma ação em seu nome. Os perfis de serviço aparecem em sua conta do IAM e são de propriedade da conta. Isso significa que um administrador do IAM pode alterar as permissões para esse perfil. Porém, fazer isso pode alterar a funcionalidade do serviço.

AWS OpsWorks O CM usa duas funções:

- Uma função de serviço que concede ao serviço AWS OpsWorks CM permissões para trabalhar na AWS conta de um usuário. Se você usar a função de serviço padrão fornecida pelo OpsWorks CM, o nome dessa função será`aws-opsworks-cm-service-role`.
- Uma função de perfil de instância que permite que o serviço AWS OpsWorks CM chame a API OpsWorks CM. Essa função concede acesso ao Amazon S3 e AWS CloudFormation à criação do servidor e do bucket do S3 para backups. Se você usar o perfil de instância padrão fornecido pelo OpsWorks CM, o nome dessa função de perfil de instância será`aws-opsworks-cm-ec2-role`.

Selecionar um perfil do IAM no AWS OpsWorks CM

Ao criar um servidor no AWS OpsWorks CM, você deve escolher uma função para permitir que o AWS OpsWorks CM acesse a Amazon EC2 em seu nome. Se você já criou uma função de serviço, o AWS OpsWorks CM fornece uma lista de funções para você escolher. OpsWorks O CM pode criar a função para você, se você não especificar uma. É importante escolher uma função que permita o acesso para iniciar e interromper EC2 instâncias da Amazon. Para ter mais informações, consulte Criar um servidor Chef Automate ou Criar um Puppet Enterprise master.

AWS OpsWorks Exemplos de políticas baseadas em identidade do CM

Por padrão, usuários ou funções não têm permissão para criar ou modificar recursos do AWS OpsWorks CM. Eles também não podem realizar tarefas usando a AWS API AWS Management Console AWS CLI, ou. Um administrador do IAM deve criar políticas do IAM que concedam às identidades de IAM permissão para executarem operações de API específicas nos recursos especificados de que precisam. O administrador deve anexar essas políticas aos usuários ou grupos que exigem essas permissões.

Para saber como criar uma política baseada em identidade do IAM usando esses exemplos de documentos de política JSON, consulte Criar políticas do IAM no Guia do usuário do IAM.

No AWS OpsWorks CM, você pode atribuir a `AWSOpsWorksCMServiceRole` política a um usuário para permitir que o usuário crie e gerencie os servidores Chef Automate ou Puppet Enterprise usando o AWS Management Console ou. AWS CLI

Tópicos

- Melhores práticas de políticas
- Permitir que os usuários visualizem suas próprias permissões
- Visualizando servidores AWS OpsWorks CM com base em tags

Melhores práticas de políticas

As políticas baseadas em identidade determinam se alguém pode criar, acessar ou excluir recursos de OpsWorks CM em sua conta. Essas ações podem incorrer em custos para sua Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

- Comece com as políticas AWS gerenciadas e avance para as permissões de privilégios mínimos — Para começar a conceder permissões aos seus usuários e cargas de trabalho, use as políticas

AWS gerenciadas que concedem permissões para muitos casos de uso comuns. Eles estão disponíveis no seu Conta da AWS. Recomendamos que você reduza ainda mais as permissões definindo políticas gerenciadas pelo AWS cliente que sejam específicas para seus casos de uso. Para obter mais informações, consulte Políticas gerenciadas pela AWS ou Políticas gerenciadas pela AWS para funções de trabalho no Guia do usuário do IAM.

- Aplique permissões de privilégio mínimo: ao definir permissões com as políticas do IAM, conceda apenas as permissões necessárias para executar uma tarefa. Você faz isso definindo as ações que podem ser executadas em recursos específicos sob condições específicas, também conhecidas como permissões de privilégio mínimo. Para obter mais informações sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no IAM no Guia do usuário do IAM.
- Use condições nas políticas do IAM para restringir ainda mais o acesso: você pode adicionar uma condição às políticas para limitar o acesso a ações e recursos. Por exemplo, você pode escrever uma condição de política para especificar que todas as solicitações devem ser enviadas usando SSL. Você também pode usar condições para conceder acesso às ações de serviço se elas forem usadas por meio de uma ação específica AWS service (Serviço da AWS), como AWS CloudFormation. Para obter mais informações, consulte Elementos da política JSON do IAM: condição no Guia do usuário do IAM.
- Use o IAM Access Analyzer para validar suas políticas do IAM a fim de garantir permissões seguras e funcionais: o IAM Access Analyzer valida as políticas novas e existentes para que elas sigam a linguagem de política do IAM (JSON) e as práticas recomendadas do IAM. O IAM Access Analyzer oferece mais de cem verificações de política e recomendações práticas para ajudar a criar políticas seguras e funcionais. Para obter mais informações, consulte Validação de políticas do IAM Access Analyzer no Guia do Usuário do IAM.
- Exigir autenticação multifator (MFA) — Se você tiver um cenário que exija usuários do IAM ou um usuário root, ative Conta da AWS a MFA para obter segurança adicional. Para exigir MFA quando as operações de API forem chamadas, adicione condições de MFA às suas políticas. Para obter mais informações, consulte Configuração de acesso à API protegido por MFA no Guia do Usuário do IAM.

Para obter mais informações sobre as práticas recomendadas do IAM, consulte Práticas recomendadas de segurança no IAM no Guia do usuário do IAM.

Permitir que os usuários visualizem suas próprias permissões

Este exemplo mostra como você pode criar uma política que permite que os usuários visualizem as políticas gerenciadas e embutidas anexadas a sua identidade de usuário. Essa política inclui

permissões para concluir essa ação no console ou programaticamente usando a API AWS CLI ou AWS .

```
{
       "Version": "2012-10-17",
       "Statement": [
           {
               "Sid": "ViewOwnUserInfo",
               "Effect": "Allow",
               "Action": [
                   "iam:GetUserPolicy",
                   "iam:ListGroupsForUser",
                   "iam:ListAttachedUserPolicies",
                   "iam:ListUserPolicies",
                   "iam:GetUser"
               ],
               "Resource": [
                   "arn:aws:iam::*:user/${aws:username}"
               ]
           },
           {
               "Sid": "NavigateInConsole",
               "Effect": "Allow",
               "Action": [
                   "iam:GetGroupPolicy",
                   "iam:GetPolicyVersion",
                   "iam:GetPolicy",
                   "iam:ListAttachedGroupPolicies",
                   "iam:ListGroupPolicies",
                   "iam:ListPolicyVersions",
                   "iam:ListPolicies",
                   "iam:ListUsers"
               ],
               "Resource": "*"
           }
       ]
   }
```

Visualizando servidores AWS OpsWorks CM com base em tags

Você pode usar condições em sua política baseada em identidade para controlar o acesso aos servidores e backups do AWS OpsWorks CM com base em tags. Este exemplo mostra como você pode criar uma política que permita visualizar um servidor AWS OpsWorks CM. No entanto, a

permissão é concedida somente se a tag do servidor AWS OpsWorks CM `Owner` tiver o valor do nome de usuário desse usuário. Essa política também concede as permissões necessárias concluir essa ação no console.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ListServersInConsole",
            "Effect": "Allow",
            "Action": "opsworks-cm:DescribeServers",
            "Resource": "*"
        },
        {
            "Sid": "ViewServerIfOwner",
            "Effect": "Allow",
            "Action": "opsworks-cm:DescribeServers",
            "Resource": "arn:aws:opsworks-cm:region:master-account-ID:server/server-
name",
            "Condition": {
                "StringEquals": {"opsworks-cm:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Você pode anexar essa política aos usuários na sua conta. Se um usuário chamado `richard-roe` tentar visualizar um servidor AWS OpsWorks CM, o servidor deverá ser marcado como `Owner=richard-roe` ou`owner=richard-roe`. Caso contrário, ele terá o acesso negado. A chave da tag de condição `Owner` corresponde a `Owner` e a `owner` porque os nomes das chaves de condição não fazem distinção entre maiúsculas e minúsculas. Para obter mais informações, consulte Elementos de política JSON do IAM: condição no Guia do usuário do IAM.

Solução de problemas AWS OpsWorks de identidade e acesso do CM

Use as seguintes informações para ajudar a diagnosticar e corrigir problemas comuns que podem ser encontrados ao trabalhar com o IAM. Para obter informações sobre solução de problemas específicas do AWS OpsWorks CM, consulte Solução de problemas AWS OpsWorks for Chef Automate Solução de problemas OpsWorks do Puppet Enterprise e.

Tópicos

- Não estou autorizado a realizar uma ação no AWS OpsWorks CM
- Não estou autorizado a realizar o meu pedido: PassRole
- Quero permitir que pessoas fora da minha AWS conta acessem meus recursos AWS OpsWorks de CM

Não estou autorizado a realizar uma ação no AWS OpsWorks CM

Se isso AWS Management Console indicar que você não está autorizado a realizar uma ação, entre em contato com o administrador para obter ajuda. Caso seu administrador seja a pessoa que forneceu suas credenciais de início de sessão.

O exemplo de erro a seguir ocorre quando o usuário `mateojackson` tenta usar o console para visualizar detalhes sobre um servidor AWS OpsWorks CM, mas não tem `opsworks-cm:DescribeServers` permissões.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 opsworks-cm:DescribeServers on resource: test-chef-automate-server
```

Nesse caso, Mateo pede ao administrador para atualizar suas políticas a fim de obter acesso ao recurso `test-chef-automate-server` usando a ação `opsworks-cm:DescribeServers`.

Não estou autorizado a realizar o meu pedido: PassRole

Se você receber uma mensagem de erro informando que você não está autorizado a executar a ação `iam:PassRole`, entre em contato com o administrador para obter assistência. Caso seu administrador seja a pessoa que forneceu suas credenciais de início de sessão. Peça a essa pessoa que atualize suas políticas para permitir que você passe uma função para o OpsWorks CM.

Alguns AWS serviços permitem que você passe uma função existente para esse serviço, em vez de criar uma nova função de serviço ou uma função vinculada ao serviço. Para fazer isso, é preciso ter permissões para passar o perfil para o serviço.

O exemplo de erro a seguir ocorre quando um usuário chamado `marymajor` tenta usar o console para realizar uma ação no OpsWorks CM. No entanto, a ação exige que o serviço tenha permissões concedidas por um perfil de serviço. Mary não tem permissões para passar a função para o serviço.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Neste caso, Mary pede ao administrador para atualizar suas políticas para permitir que ela execute a ação `iam:PassRole`.

Quero permitir que pessoas fora da minha AWS conta acessem meus recursos AWS OpsWorks de CM

É possível criar um perfil que os usuários de outras contas ou pessoas fora da sua organização podem usar para acessar seus recursos. É possível especificar quem é confiável para assumir o perfil. Para serviços que oferecem suporte a políticas baseadas em recursos ou listas de controle de acesso (ACLs), você pode usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte:

- AWS OpsWorks O CM oferece suporte à concessão de acesso a usuários de mais de uma conta para gerenciar um servidor AWS OpsWorks CM.
- Para saber como fornecer acesso aos seus recursos em todas AWS as contas que você possui, consulte Como fornecer acesso a um usuário do IAM em outra AWS conta que você possui no Guia do usuário do IAM.
- Para saber como fornecer acesso aos seus recursos para AWS contas de terceiros, consulte Como fornecer acesso a AWS contas pertencentes a terceiros no Guia do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte Conceder acesso a usuários autenticados externamente (federação de identidades) no Guia do usuário do IAM.
- Para saber a diferença entre usar perfis e políticas baseadas em recursos para acesso entre contas, consulte Como os perfis do IAM diferem de políticas baseadas em recursos no Guia do usuário do IAM.

AWS políticas gerenciadas para gerenciamento de AWS OpsWorks configuração

Para adicionar permissões a usuários, grupos e funções, é mais fácil usar políticas AWS gerenciadas do que escrever políticas você mesmo. É necessário tempo e experiência para criar políticas gerenciadas pelo cliente do IAM que fornecem à sua equipe apenas as permissões de que precisam. Para começar rapidamente, você pode usar nossas políticas AWS gerenciadas. Essas políticas abrangem casos de uso comuns e estão disponíveis em sua AWS conta. Para obter mais informações sobre políticas AWS gerenciadas, consulte políticas AWS gerenciadas no Guia do usuário do IAM.

AWS os serviços mantêm e atualizam as políticas AWS gerenciadas. Você não pode alterar as permissões nas políticas AWS gerenciadas. Os serviços ocasionalmente acrescentam permissões adicionais a uma política gerenciada pela AWS para oferecer suporte a novos recursos. Esse tipo de atualização afeta todas as identidades (usuários, grupos e funções) em que a política está anexada. É mais provável que os serviços atualizem uma política gerenciada pela AWS quando um novo recurso for iniciado ou novas operações se tornarem disponíveis. Os serviços não removem as permissões de uma política AWS gerenciada, portanto, as atualizações de políticas não violarão suas permissões existentes.

Além disso, AWS oferece suporte a políticas gerenciadas para funções de trabalho que abrangem vários serviços. Por exemplo, a política ReadOnlyAccess AWS gerenciada fornece acesso somente de leitura a todos os AWS serviços e recursos. Quando um serviço lança um novo recurso, AWS adiciona permissões somente de leitura para novas operações e recursos. Para obter uma lista e descrições das políticas de perfis de trabalho, consulte Políticas gerenciadas pela AWS para perfis de trabalho no Guia do usuário do IAM.

Política gerenciada pela AWS: **`AWSOpsWorksCMServiceRole`**

Você pode anexar `AWSOpsWorksCMServiceRole` às suas entidades do IAM. OpsWorks O CM também anexa essa política a uma função de serviço que permite que o OpsWorks CM execute ações em seu nome.

Essa política concede *`administrative`* permissões que permitem aos administradores do OpsWorks CM criar, gerenciar e excluir servidores e backups do OpsWorks CM.

Detalhes das permissões

Esta política inclui as seguintes permissões.

- `opsworks-cm`: permite que os diretores excluam os servidores existentes e iniciem as operações de manutenção.
- `acm`— Permite que os diretores excluam ou importem certificados AWS Certificate Manager que permitem que os usuários se conectem a um servidor OpsWorks CM.
- `cloudformation`— Permite que o OpsWorks CM crie e gerencie AWS CloudFormation pilhas quando os principais criam, atualizam ou excluem servidores OpsWorks CM.
- `ec2`— Permite que o OpsWorks CM inicie, provisione, atualize e encerre instâncias do Amazon Elastic Compute Cloud quando os diretores criam, atualizam ou OpsWorks excluem servidores CM.

- `iam`— Permite que o OpsWorks CM crie funções de serviço necessárias para criar e gerenciar servidores OpsWorks CM.
- `tag`— permite que os diretores apliquem e removam tags dos recursos do OpsWorks CM, incluindo servidores e backups.
- `s3`— Permite que o OpsWorks CM crie buckets Amazon S3 para armazenar backups de servidores, gerenciar objetos em buckets S3 mediante solicitação principal (por exemplo, excluir um backup) e excluir buckets.
- `secretsmanager`— Permite que o OpsWorks CM crie e gerencie segredos do Secrets Manager e aplique ou remova tags dos segredos.
- `ssm`— Permite que o OpsWorks CM use o Systems Manager Run Command nas instâncias que são servidores OpsWorks CM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket*"
            ],
            "Action": [
                "s3:CreateBucket",
                "s3:DeleteObject",
                "s3:DeleteBucket",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:PutBucketPolicy",
                "s3:PutObject",
                "s3:GetBucketTagging",
                "s3:PutBucketTagging"
            ]
        },
        {
            "Effect": "Allow",
            "Resource": [
                "*"
            ],
            "Action": [
                "tag:UntagResources",
                "tag:TagResources"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Resource": [
                "*"
            ],
            "Action": [
                "ssm:DescribeInstanceInformation",
                "ssm:GetCommandInvocation",
                "ssm:ListCommandInvocations",
                "ssm:ListCommands"
            ]
        },
        {
            "Effect": "Allow",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringLike": {
                    "ssm:resourceTag/aws:cloudformation:stack-name": "aws-opsworks-cm-
*"
                }
            },
            "Action": [
                "ssm:SendCommand"
            ]
        },
        {
            "Effect": "Allow",
            "Resource": [
                "arn:aws:ssm:*::document/*",
                "arn:aws:s3:::amzn-s3-demo-bucket*"
            ],
            "Action": [
                "ssm:SendCommand"
            ]
        },
        {
            "Effect": "Allow",
            "Resource": [
                "*"
            ],
```

```
            "Action": [
                "ec2:AllocateAddress",
                "ec2:AssociateAddress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateImage",
                "ec2:CreateSecurityGroup",
                "ec2:CreateSnapshot",
                "ec2:CreateTags",
                "ec2:DeleteSecurityGroup",
                "ec2:DeleteSnapshot",
                "ec2:DeregisterImage",
                "ec2:DescribeAccountAttributes",
                "ec2:DescribeAddresses",
                "ec2:DescribeImages",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstances",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSnapshots",
                "ec2:DescribeSubnets",
                "ec2:DisassociateAddress",
                "ec2:ReleaseAddress",
                "ec2:RunInstances",
                "ec2:StopInstances"
            ]
        },
        {
            "Effect": "Allow",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringLike": {
                    "ec2:ResourceTag/aws:cloudformation:stack-name": "aws-opsworks-cm-
*"
                }
            },
            "Action": [
                "ec2:TerminateInstances",
                "ec2:RebootInstances"
            ]
        },
        {
            "Effect": "Allow",
            "Resource": [
```

```
                "arn:aws:opsworks-cm:*:*:server/*"
            ],
            "Action": [
                "opsworks-cm:DeleteServer",
                "opsworks-cm:StartMaintenance"
            ]
        },
        {
            "Effect": "Allow",
            "Resource": [
                "arn:aws:cloudformation:*:*:stack/aws-opsworks-cm-*"
            ],
            "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStackEvents",
                "cloudformation:DescribeStackResources",
                "cloudformation:DescribeStacks",
                "cloudformation:UpdateStack"
            ]
        },
        {
            "Effect": "Allow",
            "Resource": [
                "arn:aws:iam::*:role/aws-opsworks-cm-*",
                "arn:aws:iam::*:role/service-role/aws-opsworks-cm-*"
            ],
            "Action": [
                "iam:PassRole"
            ]
        },
        {
            "Effect": "Allow",
            "Resource": "*",
            "Action": [
                "acm:DeleteCertificate",
                "acm:ImportCertificate"
            ]
        },
        {
            "Effect": "Allow",
            "Resource": "arn:aws:secretsmanager:*:*:opsworks-cm!aws-opsworks-cm-
secrets-*",
            "Action": [
```

```
                "secretsmanager:CreateSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:UpdateSecret",
                "secretsmanager:DeleteSecret",
                "secretsmanager:TagResource",
                "secretsmanager:UntagResource"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteTags",
            "Resource": [
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:elastic-ip/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        }
    ]
}
```

Política gerenciada pela AWS: **`AWSOpsWorksCMInstanceProfileRole`**

Você pode anexar `AWSOpsWorksCMInstanceProfileRole` às suas entidades do IAM. OpsWorks O CM também anexa essa política a uma função de serviço que permite que o OpsWorks CM execute ações em seu nome.

Essa política concede *`administrative`* permissões que permitem que as EC2 instâncias da Amazon usadas como servidores OpsWorks CM obtenham informações de AWS CloudFormation e AWS Secrets Manager armazenem backups de servidores em buckets do Amazon S3.

Detalhes das permissões

Esta política inclui as seguintes permissões.

- `acm`— Permite que EC2 as instâncias do servidor OpsWorks CM obtenham certificados AWS Certificate Manager que permitem que os usuários se conectem a um servidor OpsWorks CM.
- `cloudformation`— permite que as EC2 instâncias do servidor OpsWorks CM obtenham informações sobre AWS CloudFormation pilhas durante o processo de criação ou atualização da instância e enviem sinais AWS CloudFormation sobre seu status.

- `s3`— permite que as EC2 instâncias do servidor OpsWorks CM carreguem e armazenem backups do servidor em buckets do S3, interrompam ou revertam os uploads, se necessário, e excluam os backups dos buckets do S3.
- `secretsmanager`— Permite que as EC2 instâncias do servidor OpsWorks CM obtenham os valores dos segredos do Secrets Manager relacionados ao OpsWorks CM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "cloudformation:DescribeStackResource",
                "cloudformation:SignalResource"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
            ]
        },
        {
            "Action": [
                "s3:AbortMultipartUpload",
                "s3:DeleteObject",
                "s3:GetObject",
                "s3:ListAllMyBuckets",
                "s3:ListBucket",
                "s3:ListMultipartUploadParts",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::amzn-s3-demo-bucket*",
            "Effect": "Allow"
        },
        {
            "Action": "acm:GetCertificate",
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:*:*:opsworks-cm!aws-opsworks-cm-
secrets-*",
```

```
            "Effect": "Allow"
        }
    ]
}
```

OpsWorks Atualizações do CM para políticas AWS gerenciadas

Veja detalhes sobre as atualizações das políticas AWS gerenciadas do OpsWorks CM desde que esse serviço começou a rastrear essas alterações. Para alertas automáticos sobre alterações nesta página, assine o feed RSS na página de histórico do documento OpsWorks CM.

Alteração	Descrição	Data
AWSOpsFunciona CMInstance ProfileRole - Política gerenciada atualizada	OpsWorks O CM atualizou a política gerenciada que permite que as EC2 instâncias usadas como servidores OpsWorks CM compartilhem informações com CloudFormation o Secrets Manager e gerenciem backups. A alteração é adicionada `opsworks-cm!` ao nome do recurso para segredos do Secrets Manager, para que o OpsWorks CM possa possuir os segredos.	23 de abril de 2021
AWSOpsCMServiceFunção de trabalho - Política gerenciada atualizada	OpsWorks O CM atualizou a política gerenciada que permite aos administradores do OpsWorks CM criar, gerenciar e excluir servidores e backups do OpsWorks CM. A alteração é adicionada a `opsworks-cm!` ao nome do recurso para segredos do Secrets Manager, para que o	23 de abril de 2021

Alteração	Descrição	Data
	OpsWorks CM possa possuir os segredos.	
OpsWorks O CM começou a rastrear as mudanças	OpsWorks O CM começou a monitorar as mudanças em suas políticas AWS gerenciad as.	23 de abril de 2021

Prevenção confusa de delegados entre serviços em AWS OpsWorks CM

O problema de "confused deputy" é uma questão de segurança em que uma entidade que não tem permissão para executar uma ação pode coagir uma entidade mais privilegiada a executá-la. Em AWS, a falsificação de identidade entre serviços pode resultar no problema confuso do deputado. A personificação entre serviços pode ocorrer quando um serviço (o serviço de chamada) chama outro serviço (o serviço chamado). O serviço de chamada pode ser manipulado de modo a usar suas permissões para atuar nos recursos de outro cliente de uma forma na qual ele não deveria ter permissão para acessar. Para evitar isso, a AWS fornece ferramentas que ajudam você a proteger seus dados para todos os serviços com entidades principais de serviço que receberam acesso aos recursos em sua conta.

Recomendamos usar as chaves de contexto de condição `aws:SourceAccount`global `aws:SourceArn`e as chaves de contexto nas políticas de recursos para limitar as permissões que AWS OpsWorks CM concedem outro serviço ao recurso. Se o valor de `aws:SourceArn` não contém ID da conta, como um ARN do bucket do Amazon S3, você deve usar ambas as chaves de contexto de condição global para limitar as permissões. Se você usa ambas as chaves de contexto de condição global, e o valor `aws:SourceArn` contém o ID da conta, o valor `aws:SourceAccount` e a conta no valor `aws:SourceArn` deverão utilizar a mesma ID de conta quando na mesma declaração de política. Use `aws:SourceArn` se quiser apenas um recurso associado a acessibilidade de serviço. Use `aws:SourceAccount` se quiser permitir que qualquer recurso nessa conta seja associado ao uso entre serviços.

O valor de `aws:SourceArn` deve ser o ARN de um servidor OpsWorks CM Chef ou Puppet.

A maneira mais eficaz de se proteger contra o problema do substituto confuso é usar a chave de contexto de condição global `aws:SourceArn` com o ARN completo do servidor AWS OpsWorks CM . Se você não souber o ARN completo ou se estiver especificando vários servidores ARNs,

use a chave de condição de contexto `aws:SourceArn` global com curingas (*) para as partes desconhecidas do ARN. Por exemplo, `arn:aws:servicename:*:123456789012:*`.

A seção a seguir mostra como você pode usar as chaves de contexto de condição `aws:SourceAccount` global `aws:SourceArn` e as chaves de contexto AWS OpsWorks CM para evitar o confuso problema substituto.

Evite explorações confusas de delegados em AWS OpsWorks CM

Esta seção descreve como você pode ajudar a evitar explorações confusas de adjuntos e inclui exemplos de políticas de permissões que você pode anexar à função do IAM que você está usando para acessar AWS OpsWorks CM. AWS OpsWorks CM Como prática recomendada de segurança, sugerimos adicionar as chaves de condição `aws:SourceArn` e `aws:SourceAccount` às relações de confiança que seu perfil do IAM possui com outros serviços. As relações de confiança AWS OpsWorks CM permitem assumir a função de realizar ações em outros serviços necessários para criar ou gerenciar seus AWS OpsWorks CM servidores.

Para editar relações de confiança para adicionar chaves de condição **`aws:SourceArn`** e **`aws:SourceAccount`**

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação à esquerda, selecione Perfis.
3. Na caixa Pesquisar, pesquise a função que você usa para acessar AWS OpsWorks CM. A função AWS gerenciada é`aws-opsworks-cm-service-role`.
4. Na página Resumo do perfil, escolha a guia Relações de confiança.
5. Na guia Trust relationships (Relações de confiança), escolha Edit trust relationship (Editar relação de confiança).
6. No Documento de política, adicione pelo menos uma das chaves de condição `aws:SourceArn` ou `aws:SourceAccount` à política. Use `aws:SourceArn` para restringir a relação de confiança entre serviços cruzados (como AWS Certificate Manager AWS OpsWorks CM a Amazon EC2) e AWS OpsWorks CM servidores específicos, o que é mais restritivo. Adicione `aws:SourceAccount` para restringir a relação de confiança entre serviços cruzados e AWS OpsWorks CM servidores em uma conta específica, que é menos restritiva. Veja um exemplo a seguir. Observe que, se você usar as duas chaves de condição, a conta IDs deverá ser a mesma.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "opsworks-cm.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:opsworks-cm:us-east-2:123456789012:server/my-
opsworks-server/EXAMPLEabcd-1234-efghEXAMPLE-ID"
        }
      }
    }
  ]
}
```

7. Quando terminar de adicionar as permissões à política, escolha Atualizar política de confiança.

Veja a seguir exemplos adicionais de funções que limitam o acesso aos AWS OpsWorks CM servidores usando `aws:SourceArn` `aws:SourceAccount` e.

Tópicos

- Exemplo: acessar AWS OpsWorks CM servidores em uma região específica
- Exemplo: adicionar mais de um ARN de servidor ao aws:SourceArn

Exemplo: acessar AWS OpsWorks CM servidores em uma região específica

A declaração de relação de confiança de funções a seguir acessa qualquer AWS OpsWorks CM servidor na região Leste dos EUA (Ohio) (`us-east-2`). Observe que a região está especificada no valor ARN de `aws:SourceArn`, mas o valor da ID do servidor é um curinga (*).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Principal": {
        "Service": "opsworks-cm.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:opsworks-cm:us-east-2:123456789012:server/*"
        }
      }
    }
  ]
}
```

Exemplo: adicionar mais de um ARN de servidor ao **`aws:SourceArn`**

O exemplo a seguir limita o acesso a uma matriz de dois AWS OpsWorks CM servidores na ID da conta 123456789012.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "opsworks-cm.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnEquals": {
          "aws:SourceArn": [
             "arn:aws:opsworks-cm:us-east-2:123456789012:server/my-chef-
server/unique_ID",
             "arn:aws:opsworks-cm:us-east-2:123456789012:server/my-puppet-
server/unique_ID"
           ]
       }
      }
```

```
    }
  ]
}
```

Privacidade do tráfego entre redes

AWS OpsWorks O CM usa os mesmos protocolos de segurança de transmissão geralmente usados por AWS: HTTPS ou HTTP com criptografia TLS.

Registro e monitoramento no AWS OpsWorks CM

AWS OpsWorks O CM registra todas as ações da API em CloudTrail. Para obter mais informações, consulte os tópicos a seguir.

- Registro OpsWorks de chamadas de API do Puppet Enterprise com AWS CloudTrail
- Registrando chamadas de AWS OpsWorks for Chef Automate API com AWS CloudTrail

Validação de conformidade do AWS OpsWorks CM

AWS OpsWorks O CM oferece suporte aos seguintes programas e regulamentações de conformidade:

- PCI (Payment Card Industry)
- Health Insurance Portability and Accountability Act of 1996 (HIPAA – Lei de portabilidade e responsabilidade de provedores de saúde de 1996 dos EUA).
- AWS Controles de sistema e organização (SOC) 1, 2 e 3
- Regulamento Geral sobre a Proteção de Dados (General Data Protection Regulation, ou GDPR)

Auditores terceirizados avaliam a segurança e a conformidade do AWS OpsWorks CM como parte de vários programas de AWS conformidade. Isso inclui SOC, PCI, FedRAMP, HIPAA e outros.

Para obter uma lista de AWS serviços no escopo de programas de conformidade específicos, consulte AWS Services in Scope by Compliance Program. Para obter informações gerais, consulte Programas de conformidade da AWS.

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact. Para obter mais informações, consulte Download de relatórios no AWS Artifact.

Sua responsabilidade de conformidade ao usar o AWS OpsWorks CM é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade da sua empresa e pelas leis e regulamentos aplicáveis. AWS fornece os seguintes recursos para ajudar na conformidade:

- Guias de início rápido de segurança e conformidade – Esses guias de implantação discutem considerações sobre arquitetura e fornecem medidas para implantar ambientes de linha de base focados em segurança e conformidade na AWS.
- Documento técnico sobre arquitetura para segurança e conformidade com a HIPAA — Este whitepaper descreve como as empresas podem usar para criar aplicativos compatíveis com a HIPAA. AWS
- AWS Recursos de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- AWS Config— Esse AWS serviço avalia se suas configurações de recursos estão em conformidade com as práticas internas, as diretrizes e os regulamentos do setor.
- AWS Security Hub— Esse AWS serviço fornece uma visão abrangente do seu estado de segurança interno, AWS que ajuda você a verificar sua conformidade com os padrões e as melhores práticas do setor de segurança.

Resiliência em CM AWS OpsWorks

AWS OpsWorks O CM permite backups diários de servidores por padrão quando você cria um servidor. Os backups são criptografados e armazenados em um bucket do Amazon S3. Por padrão, esse bucket é acessível somente para a conta que criou o servidor. É possível adicionar acesso ao bucket para outros usuários ou configurar backups entre regiões no Amazon S3 a seu critério. O Chef e o Puppet oferecem suporte à criptografia entre regiões, pois ambos os produtos criptografam o tráfego entre o servidor do AWS OpsWorks CM e os nós gerenciados.

AWS OpsWorks O CM não oferece suporte a configurações de alta disponibilidade (HA).

A infraestrutura AWS global é construída em torno de AWS regiões e zonas de disponibilidade. AWS As regiões fornecem várias zonas de disponibilidade fisicamente separadas e isoladas, conectadas a redes de baixa latência, alta taxa de transferência e alta redundância. Com as zonas de disponibilidade, é possível projetar e operar aplicações e bancos de dados que executam o failover automaticamente entre as zonas de disponibilidade sem interrupção. As zonas de disponibilidade são mais altamente disponíveis, tolerantes a falhas e escaláveis que uma ou várias infraestruturas de data center tradicionais.

Para obter mais informações sobre como fazer backup e restaurar servidores no AWS OpsWorks CM, consulte o seguinte:

- Fazer o backup e a restauração de um servidor do OpsWorks para Puppet Enterprise
- Fazer o backup e a restauração de um AWS OpsWorks for Chef Automate servidor

Para obter mais informações sobre AWS regiões e zonas de disponibilidade, consulte Infraestrutura AWS global.

Segurança da infraestrutura no AWS OpsWorks CM

Como um serviço gerenciado, o AWS OpsWorks Configuration Management é protegido pela segurança de rede AWS global. Para obter informações sobre serviços AWS de segurança e como AWS proteger a infraestrutura, consulte AWS Cloud Security. Para projetar seu AWS ambiente usando as melhores práticas de segurança de infraestrutura, consulte Proteção de infraestrutura no Security Pillar AWS Well-Architected Framework.

Você usa chamadas de API AWS publicadas para acessar o OpsWorks CM pela rede. Os clientes devem oferecer compatibilidade com:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com perfect forward secrecy (PFS) como DHE (Ephemeral Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Além disso, as solicitações devem ser assinadas usando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou é possível usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

AWS OpsWorks O CM não oferece suporte a links privados nem a endpoints privados de VPC.

AWS OpsWorks O CM não oferece suporte a políticas baseadas em recursos. Para obter mais informações, consulte Serviços da AWS que trabalham com o IAM no Guia do usuário do AWS Identity and Access Management .

Análise de configuração e vulnerabilidade no AWS OpsWorks CM

AWS OpsWorks O CM executa atualizações periódicas de kernel e segurança no sistema operacional que está sendo executado no seu servidor AWS OpsWorks CM. Os usuários podem definir uma janela de tempo para que as atualizações automáticas ocorram por até duas semanas a partir da data atual. AWS OpsWorks O CM envia atualizações automáticas das versões secundárias do Chef e do Puppet Enterprise. Para obter mais informações sobre como configurar atualizações para AWS OpsWorks for Chef Automate, consulte Manutenção do sistema (Chef) neste guia. Para obter mais informações sobre como configurar atualizações OpsWorks para o Puppet Enterprise, consulte Manutenção do sistema (Puppet) neste guia.

Melhores práticas de segurança para AWS OpsWorks CM

AWS OpsWorks O CM, como todos os AWS serviços, oferece recursos de segurança a serem considerados ao desenvolver e implementar suas próprias políticas de segurança. As práticas recomendadas a seguir são diretrizes gerais e não representam uma solução completa de segurança. Como essas práticas recomendadas podem não ser adequadas ou suficientes para o seu ambiente, trate-as como considerações úteis em vez de prescrições.

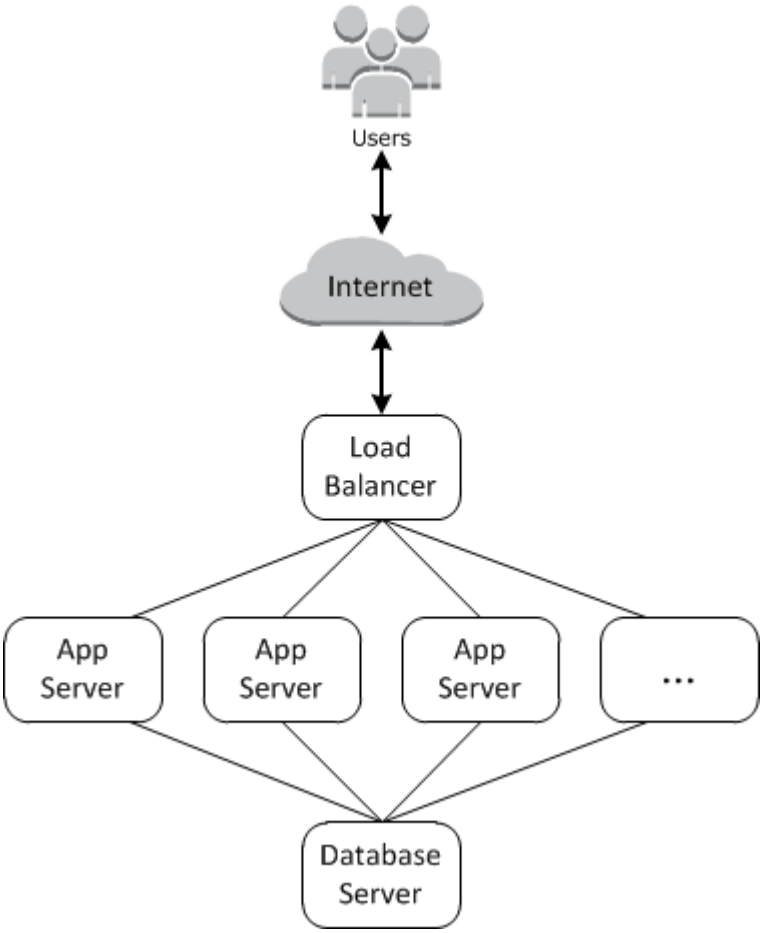
- Proteja o Starter kit e as credenciais de login obtidas por download. Ao criar um novo servidor AWS OpsWorks CM ou baixar um novo Starter Kit e credenciais do console AWS OpsWorks CM, armazene esses itens em um local seguro que exija pelo menos um fator de autenticação. As credenciais concedem acesso de nível de administrador ao servidor.
- Proteja o código de configuração. Proteja o código de configuração do Chef ou do Puppet (livros de receitas e módulos) usando os protocolos recomendados para os repositórios de origem. Por exemplo, você pode restringir as permissões aos repositórios ou seguir as diretrizes do GitHub site para proteger GitHub os repositórios. AWS CodeCommit
- Use certificados assinados pela CA para se conectar aos nós. Embora você possa usar certificados autoassinados ao registrar ou inicializar nós em seu servidor AWS OpsWorks CM, como prática recomendada, use certificados assinados pela CA. Recomendamos que utilize um certificado assinado por uma autoridade de certificação (CA).
- Não compartilhe as credenciais de login do console de gerenciamento do Chef ou do Puppet com outros usuários. Um administrador deve criar usuários separados para cada usuário dos sites do console do Chef ou do Puppet.
  - Gerenciar usuários no Chef Automate
  - Gerenciar usuários no Puppet Enterprise

- Configurar backups e atualizações de manutenção do sistema automáticos. Configurar atualizações de manutenção automática no servidor do AWS OpsWorks CM ajuda a garantir que o servidor esteja executando as atualizações mais atuais do sistema operacional relacionadas à segurança. A configuração de backups automáticos ajuda a facilitar a recuperação de desastres e a acelerar o tempo de restauração em caso de incidente ou falha. Limite o acesso ao bucket do Amazon S3 que armazena seus backups do servidor AWS OpsWorks CM; não conceda acesso a todos. Conceda acesso de leitura ou de gravação a outros usuários individualmente, conforme necessário, ou crie um grupo de segurança no IAM para esses usuários e atribua acesso ao grupo de segurança.
  - Manutenção do sistema (Chef)
  - Manutenção do Sistema (Puppet)
  - Fazer o backup e a restauração de um AWS OpsWorks for Chef Automate servidor
  - Fazer o backup e a restauração de um servidor do OpsWorks para Puppet Enterprise
  - Criar o primeiro usuário e grupo delegados do IAM no Guia do usuário do AWS Identity and Access Management
  - Melhores práticas de segurança do Amazon S3 no Guia do desenvolvedor do Amazon Simple Storage Service

AWS OpsWorks Pilhas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

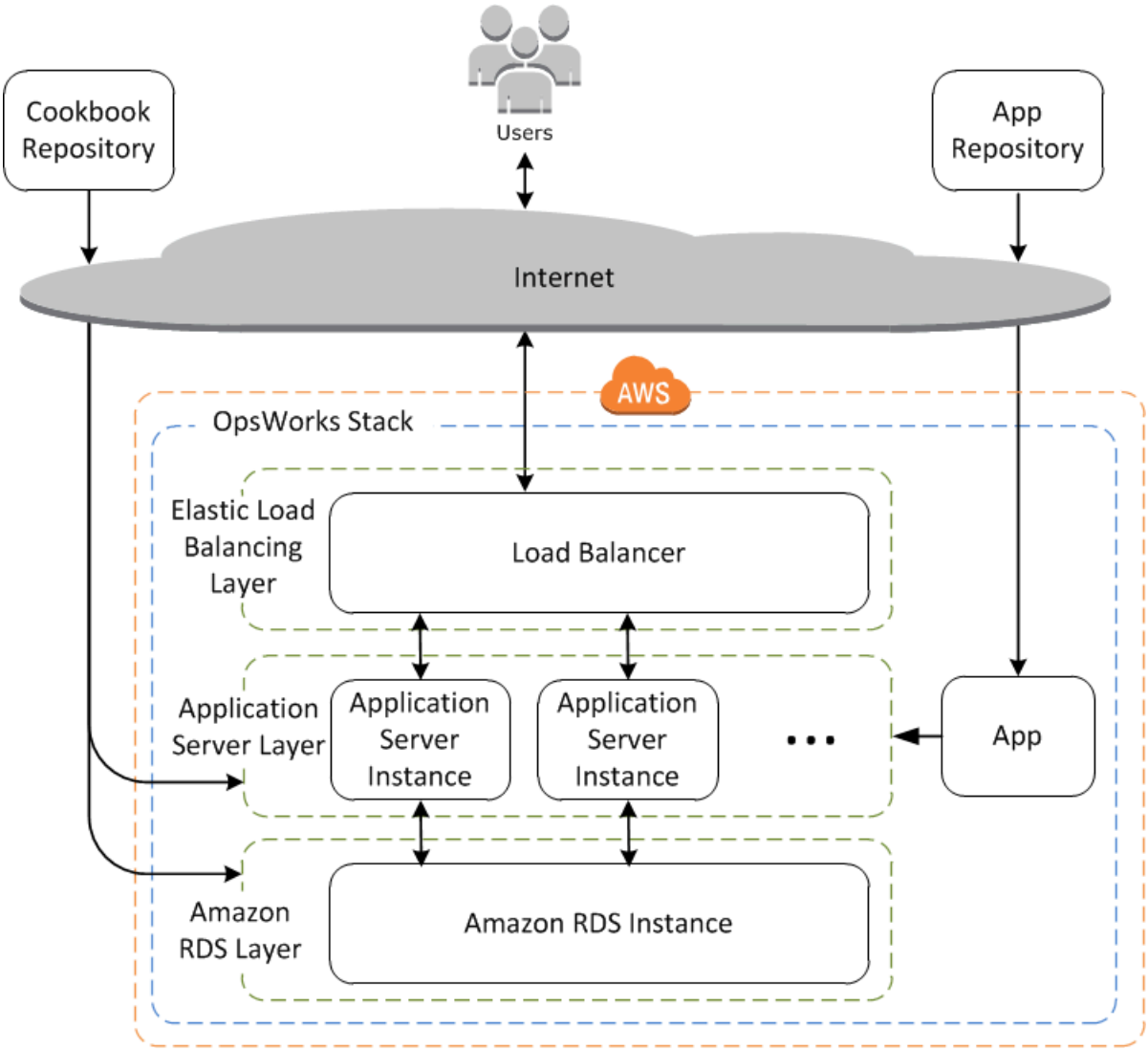
A computação em nuvem geralmente envolve grupos de recursos do AWS, como EC2 instâncias da Amazon e do Amazon Relational Database Service (RDS), que devem ser criadas e gerenciadas coletivamente. Por exemplo, um aplicativo web geralmente requer servidores de aplicativos, servidores de banco de dados, load balancers e assim por diante. Esse grupo de instâncias é geralmente chamado de uma pilha; uma simples pilha de servidor de aplicativo pode ter a seguinte aparência.



Além de criar as instâncias e instalar os pacotes necessários, normalmente você precisa de uma maneira de distribuir aplicativos para os servidores de aplicativos, monitorar o desempenho da pilha, gerenciar segurança e permissões, e assim por diante.

AWS OpsWorks O Stacks oferece uma maneira simples e flexível de criar e gerenciar pilhas e aplicativos.

Veja como uma pilha básica de servidores de aplicativos pode ficar com o AWS OpsWorks Stacks. Ele consiste em um grupo de servidores de aplicativos em execução atrás de um balanceador de carga do Elastic Load Balancing, com um servidor de banco de dados back-end do Amazon RDS.



Embora relativamente simples, essa pilha mostra todos os principais recursos do AWS OpsWorks Stacks. Veja como eles são unidos.

Tópicos

- Pilhas
- Camadas
- Receitas e LifeCycle eventos
- Instâncias
- Apps
- Personalização de sua pilha
- Gerenciamento de recursos

- Segurança e permissões
- Monitoramento e registro em log
- CLI, SDK e modelos AWS CloudFormation
- AWS OpsWorks Stacks Fim da vida FAQs
- Migrando seus AWS OpsWorks Stacks aplicativos para o AWS Systems Manager Application Manager
- Usando a AWS OpsWorks Stacks ferramenta Detach in Place
- Começando a usar o AWS OpsWorks Stacks
- AWS OpsWorks Práticas recomendadas do Stacks
- Pilhas
- Camadas
- Instâncias
- Apps
- Livros de receitas e receitas
- Gerenciamento de recursos
- Tags
- Monitoramento
- Segurança e permissões
- AWS OpsWorks Suporte Stacks para Chef 12 Linux
- Support para versões anteriores do Chef em AWS OpsWorks Stacks
- Usando AWS OpsWorks Stacks com outros serviços da AWS
- Usando a CLI do AWS OpsWorks Stacks
- Guia de depuração e solução de problemas
- AWS OpsWorks CLI do Stacks Agent
- AWS OpsWorks Referência do Stacks Data Bag
- OpsWorks Mudanças no agente

Pilhas

A pilha é o componente principal das AWS OpsWorks pilhas. É basicamente um contêiner para recursos da AWS (instâncias da Amazon EC2 , instâncias de banco de dados do Amazon RDS, e

assim por diante) que têm um propósito comum e deve ser logicamente gerenciados em conjunto. A pilha ajuda você a gerenciar esses recursos como um grupo e também define algumas opções de configuração padrão, como o sistema operacional das instâncias e região da AWS. Se quiser isolar alguns componentes da pilha de interação de usuário direta, você pode executar a pilha em uma VPC.

Camadas

Você define os constituintes adicionando uma ou mais camadas. Uma camada representa um conjunto de EC2 instâncias da Amazon que atende a um propósito específico, como servir aplicativos ou hospedar de um banco de dados.

Você pode personalizar ou estender camadas modificando as configurações padrão dos pacotes, adicionando receitas do Chef para realizar tarefas como instalar pacotes adicionais e muito mais.

Para todas as pilhas, as AWS OpsWorks pilhas incluem camadas de serviço, que representam os seguintes serviços da AWS.

- Amazon Relational Database Service
- Elastic Load Balancing
- Amazon Elastic Container Service

As camadas oferecem controle total sobre quais pacotes são instalados, como eles são configurados, como as aplicações são implantadas e muito mais.

Receitas e LifeCycle eventos

As camadas dependem de receitas do Chef para gerenciar tarefas como instalação de pacotes nas instâncias, implantação de aplicativos, execução de scripts e assim por diante. Um dos principais recursos do AWS OpsWorks Stacks é um conjunto de eventos de ciclo de vida (Setup, Configure, Deploy, Undeploy, e Shutdown) que executam automaticamente um conjunto especificado de receitas no momento apropriado em cada instância.

Cada camada pode ter um conjunto de receitas atribuído a cada evento de ciclo de vida, que lida com uma variedade de tarefas desse evento e camada. Por exemplo, após a conclusão da inicialização de uma instância que pertence a uma camada de servidor web, o AWS OpsWorks Stacks faz o seguinte.

1. Executa as receitas de configuração da camada, que pode realizar tarefas como instalar e configurar um servidor web.
2. Executa as receitas de implantação da camada, que implanta os aplicativos da camada de um repositório à instância e realiza tarefas relacionadas, como reiniciar o serviço.
3. Executa o Configurar receita em cada instância na pilha para que cada instância possa ajustar suas configurações conforme necessário para acomodar a nova instância.

   Por exemplo, em uma instância executando um load balancer, um Configurar receita pode modificar a configuração do load balancer para incluir a nova instância.

Se uma instância pertencer a várias camadas, o AWS OpsWorks Stacks executa as receitas de cada camada para que você possa, por exemplo, ter uma instância compatível com um servidor de aplicativos PHP e um servidor de banco de dados MySQL.

Se você implementou receitas, você pode atribuir cada receita à camada e ao evento apropriados, e o AWS OpsWorks Stacks as executa automaticamente para você no momento apropriado. Você também pode executar as receitas manualmente a qualquer momento.

Instâncias

Uma instância representa um recurso de computação, como uma EC2 instância da Amazon. Define a configuração básica do recurso, como o sistema operacional e tamanho. Outras opções de configuração, como endereços IP elástico ou volumes do Amazon EBS são definidos pelas camadas da instância. As receitas de camada completam a configuração realizando tarefas como pacotes de instalação e configuração, além de implantação de aplicativos.

Você pode usar AWS OpsWorks pilhas para criar instâncias e adicioná-las a uma camada. Quando você inicia a instância, o AWS OpsWorks Stacks inicia uma EC2 instância da Amazon usando as configurações especificadas pela instância e sua camada. Após a finalização do processo de inicialização da EC2 instância da Amazon, o AWS OpsWorks Stacks instala um agente que lida com a comunicação entre a instância e o serviço e executa as receitas apropriadas em resposta a eventos de ciclo de vida.

AWS OpsWorks O Stacks é compatível com os seguintes tipos de instância, que são caracterizados pela forma como são iniciados e interrompidos.

- As instâncias ininterruptas são iniciadas manualmente e executadas até que sejam interrompidas por você.

- As instâncias baseadas em tempo são executadas pelo AWS OpsWorks Stacks em uma programação diária e semanal especificada.

  Eles permitem que sua pilha ajuste automaticamente o número de instâncias para acomodar uso de padrões previsíveis.
- As instâncias baseadas em carga são iniciadas e interrompidas automaticamente pelas AWS OpsWorks pilhas, com base em métricas de carga especificadas, como a utilização da CPU.

  Eles permitem que sua pilha ajuste automaticamente o número de instâncias para acomodar variações em tráfico recebido. Instâncias baseadas em carga estão disponíveis apenas para pilhas baseadas em Linux.

AWS OpsWorks O Stacks oferece suporte à recuperação automática de instâncias. Se um agente parar de se comunicar com o serviço, o AWS OpsWorks Stacks interrompe e reinicia automaticamente a instância.

Você também pode incorporar recursos de computação baseados em Linux em uma pilha criada fora das pilhas. AWS OpsWorks

- EC2 As instâncias Amazon criadas diretamente utilizando o EC2 console Amazon, o CLI ou a API.
- Instâncias on-premises executadas em seu próprio hardware, inclusive instâncias executadas em máquinas virtuais.

Depois de registrar uma dessas instâncias, ela se torna uma instância do AWS OpsWorks Stacks e você pode gerenciá-la da mesma forma que as instâncias que você cria com o AWS OpsWorks Stacks.

Apps

Você armazena aplicativos e relaciona arquivos em um repositório, por exemplo, um bucket do Amazon S3. Cada aplicativo é representado por um aplicativo, que especifica o tipo de aplicativo e contém as informações necessárias para implantar o aplicativo do repositório a suas instâncias, como a URL do repositório e senha. Quando você implanta um aplicativo, o AWS OpsWorks Stacks dispara um evento Implantar, que executa Implantar receitas nas instâncias da pilha.

Você pode implantar aplicativos das seguintes maneiras:

- Automaticamente — Quando você inicia instâncias, o AWS OpsWorks Stacks executa automaticamente as receitas de implantação da instância.
- Manualmente: se você tiver um novo aplicativo ou deseja atualizar um existente, pode executar manualmente o Implantar receitas das instâncias online.

Normalmente, você faz com que AWS OpsWorks as pilhas executem as receitas de implantação em toda a pilha, o que permite que as instâncias das outras camadas modifiquem suas configurações adequadamente. No entanto, você pode limitar a implantação a um subconjunto de instâncias se, por exemplo, deseja testar um novo aplicativo antes de implantá-lo em todas as instâncias de servidor do aplicativo.

Personalização de sua pilha

O AWS OpsWorks Stacks fornece uma variedade de formas para personalizar camadas para atender aos seus requisitos específicos:

- Você pode modificar a forma como o AWS OpsWorks Stacks configura os pacotes substituindo os atributos que representam as várias configurações ou até mesmo substituindo os modelos usados para criar arquivos de configuração.
- Você pode estender uma camada existente fornecendo suas próprias receitas para realizar tarefas como executar scripts ou instalar e configurar pacotes não padrão.

Todos as pilhas podem incluir uma ou mais camadas, que começam com apenas um conjunto mínimo de receitas. Você adiciona funcionalidade à camada implementando receitas para lidar com tarefas como instalação de pacotes, implantação de aplicativos e assim por diante. Você empacota suas receitas personalizadas e arquivos relacionados em um ou mais livros de receitas e armazenar esses livros de receitas em um repositório, como o Amazon S3 ou o Git.

Você pode executar receitas manualmente, mas o AWS OpsWorks Stacks também permite automatizar o processo dando suporte a um conjunto de cinco eventos do ciclo de vida:

- Configuração ocorre em uma nova instância após reinicialização bem-sucedida.
- Configuração ocorre em todas instâncias de pilhas quando uma instância entra ou deixa o estado online.
- Implantar ocorre quando você implanta um aplicativo.
- Cancelar implantação ocorre quando você exclui um aplicativo.

- Desligar ocorre quando você interrompe uma instância.

Cada camada pode ter qualquer número de receitas atribuído a cada evento. Quando ocorre um evento de ciclo de vida na instância de uma camada, o AWS OpsWorks Stacks executa as receitas associadas. Por exemplo, quando um evento Deploy ocorre em uma instância do servidor de aplicativos, o AWS OpsWorks Stacks executa as receitas de implantação da camada para baixar o aplicativo ou realizar tarefas relacionadas.

Gerenciamento de recursos

É possível incorporar outros recursos da AWS, como Endereços IP elásticos, em sua pilha. Você pode usar o console ou a API do AWS OpsWorks Stacks para registrar recursos com uma pilha, anexar recursos registrados ou separá-los das instâncias e mover recursos de uma instância para outra.

Segurança e permissões

O AWS OpsWorks Stacks integra-se com o AWS Identity and Access Management (IAM) para fornecer formas robustas de controlar como os usuários acessam o AWS OpsWorks Stacks, incluindo o seguinte:

- Como usuários individuais podem interagir com cada pilha, como se eles podem criar recursos da pilha como camadas e instâncias, ou se eles podem usar SSH ou RDP para conectar às instâncias da Amazon da pilha. EC2
- Como o AWS OpsWorks Stacks podem atuar em seu nome para interagir com recursos da AWS, como EC2 instâncias da Amazon.
- Como os aplicativos executados em instâncias do AWS OpsWorks Stacks podem acessar recursos da AWS, como buckets do Amazon S3.
- Como gerenciar chaves SSH de usuários públicos e senhas de RDP e conectar a uma instância.

Monitoramento e registro em log

AWS OpsWorks O Stacks fornece vários recursos para ajudá-lo a monitorar sua pilha e solucionar problemas com sua pilha e com qualquer receita. Para todas as pilhas:

- AWS OpsWorks O Stacks fornece um conjunto de CloudWatch métricas personalizadas para pilhas do Linux, que são resumidas para sua conveniência na página Monitoramento.

  AWS OpsWorks O Stacks oferece suporte para as CloudWatch métricas padrão para pilhas do Windows. Você pode monitorá-las com o CloudWatch console.
- CloudTrail logs, que registram chamadas de API feitas por ou em nome do AWS OpsWorks Stacks na sua conta da AWS.
- Um registro de evento, que lista todos os eventos em sua pilha.
- Chef registra os detalhes acontecidos para cada evento de ciclo de vida em cada instância, como quais receitas foram executadas e quais erros ocorreram.

Pilhas baseadas em Linux também podem incluir uma camada mestre do Ganglia, que você pode usar para coletar e exibir dados de monitoramento detalhado para as instâncias em sua pilha.

CLI, SDK e modelos AWS CloudFormation

Além do console, o AWS OpsWorks Stacks também oferece suporte a uma interface de linha de comando (CLI) e a vários idiomas que podem ser usados SDKs para realizar qualquer operação. Considere estes recursos:

- O AWS OpsWorks Stacks CLI faz parte da CLI da AWS e pode ser usado para executar qualquer operação na linha de comando.

  O CLI da AWS oferece suporte a diversos serviços da AWS e pode ser instalado em sistemas Windows, Linux ou OS X.
- AWS OpsWorks O Stacks está incluído nas Ferramentas da AWS para Windows PowerShell e pode ser usado para executar qualquer operação em uma linha de PowerShell comando do Windows.
- O AWS OpsWorks Stacks SDK está incluído na AWS SDKs, que pode ser usado por aplicativos implementados em: Java JavaScript(baseado em navegador e Node.js), .NET, PHP,Python (boto) ou Ruby.

Você também pode usar AWS CloudFormation modelos para provisionar pilhas. Para alguns exemplos, consulte AWS OpsWorks Snippets.

AWS OpsWorks Stacks Fim da vida FAQs

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível.

Tópicos

- Como os clientes existentes serão afetados por esse fim da vida útil?
- Está AWS OpsWorks Stacks aceitando novos clientes?
- Para onde devo migrar minhas pilhas existentes?
- Como posso manter minhas EC2 instâncias existentes da Amazon após o fim da vida útil?
- O fim da vida afetará tudo Regiões da AWS ao mesmo tempo?
- Para qual nível de suporte técnico está disponível AWS OpsWorks Stacks?
- Haverá algum lançamento de novos recursos para AWS OpsWorks Stacks?

Como os clientes existentes serão afetados por esse fim da vida útil?

Os clientes existentes não serão afetados até 26 de maio de 2024, a data de fim da vida útil do AWS OpsWorks Stacks. Depois de 26 de maio de 2024, os clientes não poderão usar o OpsWorks console, a API, a CLI e os recursos. CloudFormation

Está AWS OpsWorks Stacks aceitando novos clientes?

Não. AWS OpsWorks Stacks não está mais aceitando novos clientes e somente os clientes existentes podem criar novas pilhas no momento.

Para onde devo migrar minhas pilhas existentes?

Recomendamos que AWS OpsWorks Stacks os clientes migrem suas cargas de trabalho para AWS Systems Manager onde possam aproveitar os seguintes recursos:

- Versões do Chef moderno

- SSM Agent
- Application Load Balancers
- Atributos de escalonamento aprimorados por meio de grupos do Auto Scaling
- Capacidade de definir as características desejadas do host usando modelos de EC2 lançamento
- Tipos de instância mais recentes
- Tipos de volume do EBS mais novos

Para obter informações sobre o Systems Manager, consulte o Guia do usuário do AWS Systems Manager. Para obter informações sobre a migração para AWS Systems Manager, consulte Migrando seus AWS OpsWorks Stacks aplicativos para o AWS Systems Manager Application Manager

Como posso manter minhas EC2 instâncias existentes da Amazon após o fim da vida útil?

Depois que a data de fim da vida útil for atingida, suas EC2 instâncias da Amazon permanecerão em sua conta, mas você não poderá mais usar o serviço OpsWorks Stacks para controlar e gerenciar as instâncias.

Você pode usar a ferramenta AWS OpsWorks Stacks Detach in Place para separar suas OpsWorks instâncias do serviço OpsWorks Stacks. Após a separação, você pode usar a Amazon EC2 ou qualquer abordagem EC2 compatível para configurar e gerenciar as instâncias. AWS Systems Manager Para obter mais informações, consulte Usando a AWS OpsWorks Stacks ferramenta Detach in Place.

O fim da vida afetará tudo Regiões da AWS ao mesmo tempo?

Sim. O OpsWorks console, a API, a CLI e CloudFormation os recursos serão descontinuados Regiões da AWS simultaneamente em 26 de maio de 2024. Para obter uma lista de Regiões da AWS onde AWS OpsWorks Stacks está disponível, consulte Lista de serviços AWS regionais.

Para qual nível de suporte técnico está disponível AWS OpsWorks Stacks?

AWS continuarão a fornecer o mesmo nível de suporte AWS OpsWorks Stacks que os clientes têm hoje até a data de fim da vida útil. Se você tiver dúvidas ou preocupações, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Haverá algum lançamento de novos recursos para AWS OpsWorks Stacks?

Não. Como o serviço está chegando ao fim da vida útil, não lançaremos nenhum atributo novo. No entanto, continuaremos fazendo melhorias de segurança e gerenciando as EC2 instâncias da Amazon conforme o esperado até a data de fim da vida útil.

Migrando seus AWS OpsWorks Stacks aplicativos para o AWS Systems Manager Application Manager

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível.

Agora você pode migrar seus AWS OpsWorks Stacks aplicativos para o Application Manager, um recurso de AWS Systems Manager, usando um script de migração. A migração de seus aplicativos Stacks para o Systems Manager Application Manager permite que você use AWS recursos que não estão disponíveis no AWS OpsWorks Stacks, como novos tipos de EC2 instância da Amazon, como Graviton, novos volumes do Amazon Elastic Block Store (EBS), como gp3, novos sistemas operacionais, integrações com grupos do Auto Scaling e balanceadores de carga de aplicativos.

Com esta versão, agora você pode monitorar e executar operações em suas instâncias migradas usando uma nova guia Instâncias disponível no Systems Manager Application Manager. Você pode usar a guia Instâncias para visualizar várias AWS instâncias em um só lugar. Usando essa guia, você pode ver informações sobre a integridade da instância e solucionar problemas. Para obter mais informações sobre como trabalhar com a guia Instâncias, consulte Como trabalhar com as instâncias do seu aplicativo no Guia do usuário AWS Systems Manager .

Tópicos

- Como o script funciona
- Pré-requisitos
- Limitações
- Conceitos básicos

- Perguntas frequentes
- Solução de problemas

Como o script funciona

AWS OpsWorks fornece um script que você pode executar para migrar seus AWS OpsWorks Stacks aplicativos para o Systems Manager Application Manager usando um CloudFormation modelo. O script obtém informações sobre uma OpsWorks camada existente e, dependendo do valor do `--provision-application` parâmetro do script, provisiona um clone do seu aplicativo ou fornece um CloudFormation modelo inicial que você pode modificar usando. AWS CloudFormation

Pré-requisitos

- Certifique-se de que o AWS CLI esteja instalado e configurado. Para obter mais informações sobre a instalação do AWS CLI, consulte Instalando ou atualizando a versão mais recente do AWS CLI no Guia AWS Command Line Interface do Usuário.

> **Note**
>
> Se você não quiser configurar o AWS CLI, você também pode executar comandos usando AWS CloudShell o. Para obter mais informações sobre como trabalhar com CloudShell, consulte Trabalhando com AWS CloudShell no Guia AWS CloudShell do Usuário.

- Garanta que o Python versão 3.6 ou mais recente esteja instalado ou venha com a Imagem de máquina da Amazon (AMI).
- Garanta que seu sistema operacional seja compatível. Você pode baixar e executar o script de migração nos seguintes sistemas operacionais.
  - Amazon Linux e Amazon Linux 2
  - Ubuntu 18.04 LTS, 20.04 LTS, 22.04 LTS
  - Red Hat Enterprise Linux 8
  - Windows Server 2019, Windows 10 Enterprise

> **Note**
>
> O Windows Server 2022 não é compatível.

Limitações

A nova OpsWorks arquitetura é diferente da arquitetura para AWS OpsWorks Stacks. Esta seção descreve as limitações conhecidas dessa arquitetura.

Os itens a seguir não são compatíveis com a nova OpsWorks arquitetura.

- Executando receitas do Chef em instâncias do Windows e CentOS
- Chef 11 camadas e Berkshelf integrados
- Atributos e pacotes de dados do Chef
- Instâncias on-premises
- Instâncias importadas de EC2
- Não há suporte para instalar uma lista especificada pelo usuário de pacotes do sistema operacional
- Os aplicativos não são compatíveis nem migrados

Estes a seguir são compatíveis, mas com limitações:

- O script de migração clona as informações do volume do EBS, mas exclui os pontos de montagem e os dados reais contidos nos volumes.
- As instâncias escalonadas com base no tempo e na carga são migradas, mas as regras de escalabilidade associadas a essas instâncias não são migradas. Você pode modificar o grupo do Auto Scaling para obter resultados semelhantes.
- As entidades do IAM definidas na página de permissões da pilha no OpsWorks console não são criadas nem geradas.
- O script de migração só é capaz de provisionar aplicativos de camada única no Systems Manager. Por exemplo, se você executar o script duas vezes para duas camadas na mesma pilha, obterá dois aplicativos diferentes no Systems Manager.

Conceitos básicos

O script de migração,`stack_exporter.py`, é um script Python que você pode executar localmente ou em uma EC2 instância. Antes de executar o script, verifique se todos os pré-requisitos foram atendidos. Para mais sobre os pré-requisitos, consulte Pré-requisitos.

As etapas nas seções a seguir mostram como migrar suas OpsWorks pilhas para o Systems Manager Application Manager.

Tópicos

- Etapa 1: preparar o ambiente para executar o script
- Etapa 2: fazer download do script de migração
- Etapa 3: configurar o ambiente para executar o script
- Etapa 4: executar o script
- Etapa 5: provisionar uma CloudFormation pilha
- Etapa 6: revisar os recursos provisionados
- Etapa 7: iniciar uma instância
- Etapa 8: revisar a instância
- Etapa 9: monitore e execute operações em suas instâncias usando o Systems Manager Application Manager

Etapa 1: preparar o ambiente para executar o script

Preparar o ambiente executando os comandos apropriados para o sistema operacional.

Tópicos

- Amazon Linux 2
- Amazon Linux
- Ubuntu 18.04, 20.04, 22.04
- Red Hat Enterprise Linux 8
- Windows Server 2019, Windows 10 Enterprise

Amazon Linux 2

```
sudo su
python3 -m pip install pipenv
PATH="$PATH:/usr/local/bin"
yum update
yum install git
```

Amazon Linux

```
sudo su
PATH="$PATH:/usr/local/bin"
export LC_ALL=en_US.utf-8
export LANG=en_US.utf-8
yum update
yum list | grep python3
yum install python36 // Any python version
yum install git
```

Para o Python versão 3.6, execute também:

```
python3 -m pip install pipenv==2022.4.8
```

Para Python versão 3.7 e versões mais recentes, execute também:

```
python3 -m pip install pipenv
```

Ubuntu 18.04, 20.04, 22.04

```
sudo su
export PATH="${HOME}/.local/bin:$PATH"
apt-get update
apt install python3-pip
apt-get install git // if git is not installed
python3 -m pip install --user pipenv==2022.4.8
```

Red Hat Enterprise Linux 8

```
sudo su
sudo dnf install python3
PATH="$PATH:/usr/local/bin"
yum update
yum install git
python3 -m pip install pipenv==2022.4.8
```

Windows Server 2019, Windows 10 Enterprise

> **Note**
>
> Para o Windows Server 2019, instale o Python versão 3.6.1 ou mais recente.

```
pip install pipenv
```

Se o Git ainda não estiver instalado, baixe e instale o Git.

Se você usa o Git como fonte do livro de receitas, adicione seu servidor Git a um arquivo `known_hosts` antes de executar o script no Windows. Você pode usar PowerShell para criar a seguinte função.

```
function add_to_known_hosts($server){
    $new_host=$(ssh-keyscan $server 2> $null)
    $existing_hosts=''
    if (!(test-path "$env:userprofile\.ssh")) {
        md "$env:userprofile\.ssh"
    }
    if ((test-path "$env:userprofile\.ssh\known_hosts")) {
        $existing_hosts=Get-Content "$env:userprofile\.ssh\known_hosts"
    }
    $host_added=0
    foreach ($line in $new_host) {
        if (!($existing_hosts -contains $line)) {
            Add-Content -Path "$env:userprofile\.ssh\known_hosts" -Value $line
            $host_added=1
    }
   }
   if ($host_added) {
       echo "$server has been added to known_hosts."
   } else {
       echo "$server already exists in known_hosts."
   }
}
```

Em seguida, você pode fornecer seu servidor Git (por exemplo, github.com, git-codecommit). *`repository_region`*.amazonaws.com) quando você executa a função.

```
add_to_known_hosts "myGitServer"
```

Etapa 2: fazer download do script de migração

Faça o download do arquivo zip contendo o script de migração e todos os arquivos relevantes executando o comando a seguir.

```
aws s3api get-object \
    --bucket export-opsworks-stacks-bucket-prod-us-east-1 \
    --key export_opsworks_stacks_script.zip export_opsworks_stacks_script.zip
```

Se você estiver usando Linux, instale o utilitário de descompactação usando os seguintes comandos.

```
sudo apt-get install unzip
sudo yum install unzip
```

Descompacte os arquivos usando o comando apropriado para seu sistema operacional.

No Linux, use o comando a seguir:

```
unzip export_opsworks_stacks_script.zip
```

Para Windows, use o `Expand-Archive` comando em PowerShell.

```
Expand-Archive -LiteralPath PathToZipFile -DestinationPath PathToDestination
```

Depois que o arquivo for descompactado, os seguintes diretórios e arquivos estarão disponíveis.

- README.md
- LICENSE
- NOTICE
- requirements.txt
- templates/
  - OpsWorksCFNTemplate.yaml
  - Monte EBSVolumes .yaml
- opsworks/
- cloudformation/

- instances_tab/
- cfn_stack_deployer.py
- s3.py
- stack_exporter_context.py
- stack_exporter.py

Etapa 3: configurar o ambiente para executar o script

Configure seu ambiente para executar o script usando o comando a seguir.

```
pipenv install -r requirements.txt
pipenv shell
```

> **Note**
>
> Atualmente, o script só pode provisionar aplicativos de camada única no Application Manager. Por exemplo, se você executar o script duas vezes para duas camadas na mesma pilha, o script cria dois aplicativos diferentes no Application Manager.

Depois de configurar seu ambiente, revise os parâmetros do script. Você pode ver as opções disponíveis para o script de migração executando o comando `python3 stack_exporter.py --help`.

Parameter	Descrição	Obrigatório	Tipo	Valor padrão
`--layer-id`	Exporta um CloudFormation modelo para essa ID OpsWorks de camada.	Sim	string	
`--region`	A AWS região da OpsWorks pilha. Se a região da OpsWorks pilha e a região do endpoint da API forem diferentes, use a região da pilha. Essa é a mesma região que os outros recursos que fazem	Não	string	us-east-1

Parameter	Descrição	Obrigatório	Tipo	Valor padrão
	parte da sua OpsWorks pilha (por exemplo, EC2 instâncias e sub-redes).			
`--provision-application`	Por padrão, o script provision a o aplicativo exportado pelo CloudFormation modelo. Passe esse parâmetro para o script com o valor FALSE para ignorar o provisionamento do modelo. CloudFormation	Não	Booliano	`TRUE`

Parameter	Descrição	Obrigatório	Tipo	Valor padrão
`--launch-template`	Esse parâmetro define se vai usar um modelo de inicializ ação existente ou criar um novo modelo de inicialização. Você pode criar um novo modelo de execução que use as proprieda des de instância recomenda das ou que use propriedades de instância que correspondam a uma instância on-line. Os valores válidos são: • `RECOMMENDED` - Usa caracterí sticas de instância da AMI mais recente para o sistema operacional da OpsWorks pilha e um tamanho de instância c5.large. • `MATCH_LAST_INSTANCE` : usa as características mais recentes da instância on-line disponível. • *`LaunchTemplateID`* / [*`LaunchTemplateVers ion]`* : para selecionar um modelo de execução existente Como opção, você pode fornecer uma versão de modelo. Se você não fornecer uma versão de modelo, o script usará a versão padrão.	Não	string	`RECOMMENDED`

Parameter	Descrição	Obrigatório	Tipo	Valor padrão
`--system-updates`	Define se vai atualizar o kernel e o pacote vai atualizar quando a instância é inicializada. Os valores válidos são: • `ALL_UPDATES` : executa atualizações do sistema para o kernel e os pacotes quando a instância é inicializada. • `NO_UPDATES` : não realiza atualizações do sistema quando a instância é inicializ ada. • `MATCH_LAYER_SETTINGS` - Usa a `InstallUpdatesOnBoot` propriedade da OpsWorks camada ou da instância para determinar se as atualizações do sistema devem ser instaladas.	Não	string	`ALL_UPDATES`
`--http-username`	O nome do parâmetro `SecureString` do Systems Manager que armazena o nome de usuário usado para autentica ção no arquivo HTTP que contém os livros de receitas personali zados.	Não	string	

Parameter	Descrição	Obrigatório	Tipo	Valor padrão
`--http-password`	O nome do parâmetro `SecureString` do Systems Manager que armazena a senha usada para autenticação no arquivo HTTP que contém os livros de receitas personalizados.	Não	string	
`--repo-private-key`	O nome do parâmetro `SecureString` do Systems Manager que armazena a chave SSH usada para autenticação no repositório que contém os livros de receitas personalizados. Se o repositório estiver ativado GitHub, você deverá gerar uma nova chave `Ed25519` SSH. Se você não gerar uma nova chave `Ed25519` SSH, a conexão com o GitHub repositório falhará.	Não	string	
`--lb-type`	O tipo de balanceador de carga, se houver, a ser criado ao migrar seu balanceador de carga existente. Os valores válidos são: • `ALB` (Application load balancer) • `Classis` (Classic load balancer) • `None` (Se você não quiser criar um balanceador de carga)	Não	string	`ALB`

Parameter	Descrição	Obrigatório	Tipo	Valor padrão
`--lb-access-logs-path`	O caminho para um bucket S3 existente e o prefixo para armazenar os registros de acesso do balanceador de carga. O bucket e o balanceador de carga devem estar na mesma Região. Se você não fornecer um valor e o valor do parâmetro `--lb-type` for definido como `None`, o script criará um novo bucket e prefixo do S3. Verifique se há uma política de bucket apropriada para esse prefixo.	Não	string	
`--enable-instance-protection`	Se definido como`TRUE`, o script cria uma política de encerramento personalizada (função Lambda) para seu grupo de Auto Scaling. EC2instâncias com uma `protected_instance` tag são protegidas de eventos de expansão. Adicione uma `protected_instance` tag a cada EC2 instância que você deseja proteger de eventos de escalabilidade.	Não	Booliano	`FALSE`
`--command-logs-bucket`	O nome de um bucket do S3 existente para armazenar os logs `AWS ApplyChefRecipe` e `MountEBSVolumes`. Se você não fornecer um valor, o script criará um novo bucket do S3.	Não	string	`aws-opsworks-application-manager-logs-`*`account-id`

Parameter	Descrição	Obrigatório	Tipo	Valor padrão
`--custom-json-bucket`	O nome de um bucket do S3 existente para armazenar JSON personalizado. Se você não fornecer um valor, o script criará um novo bucket do S3.	Não	string	`aws-apply-chef-application-manager-transition-data-`*`account-id`

Observações:

- Se você usar um GitHub repositório privado, deverá criar uma nova chave de `Ed25519` host para SSH. Isso ocorre porque GitHub alterou quais chaves são suportadas no SSH e removeu o protocolo Git não criptografado. Para obter mais informações sobre a chave do `Ed25519` host, consulte a postagem do GitHub blog Melhorando a segurança do protocolo Git em. GitHub Depois de gerar uma nova chave de host `Ed25519`, crie um parâmetro `SecureString` do Systems Manager para a chave SSH e use o nome do parâmetro `SecureString` como o valor do parâmetro `--repo-private-key`. Para obter mais informações sobre como criar um `SecureString` parâmetro do Systems Manager, consulte Create a SecureString parameter (AWS CLI) ou Create a Systems Manager parameter (console) no Guia do AWS Systems Manager Usuário.
- Os parâmetros `--http-username`, `--http-password` e `--repo-private-key` referem-se ao nome do parâmetro `SecureString` do Systems Manager. O script de migração usa esses parâmetros quando você executa o documento `AWS-ApplyChefRecipes`.
- O parâmetro `--http-username` exige que você também especifique um valor para o `--http-password` parâmetro.
- O parâmetro `--http-password` exige que você também especifique um valor para o `--http-username` parâmetro.
- Não defina valores para `--http-password` e `--repo-private-key`. Forneça um nome de parâmetro `SecureString` do Systems Manager de uma chave SSH (`--repo-private-key`) ou um nome de usuário (`--http-username`) e senha (`--http-password`) do repositório.

Etapa 4: executar o script

Ao executar `python3 stack_exporter.py`, você pode provisionar o aplicativo ou criar um modelo inicial definindo o valor do parâmetro `--provision-application` como `FALSE`.

Exemplo 1: provisionar um aplicativo Systems Manager Application Manager

O comando a seguir obtém informações sobre uma OpsWorks camada existente e provisiona um aplicativo usando a OpsWorks arquitetura mais recente, que obtém um resultado semelhante à versão do Chef configurada para a pilha. O script provisiona todos os recursos necessários, como grupos de Auto Scaling usando e CloudFormation, em seguida, registra o aplicativo no Systems Manager Application Manager.

Substitua *`stack-region`* e *`layer-id`* pelos valores de sua OpsWorks pilha e camada.

```
python3 stack_exporter.py \
     --layer-id layer-id \
     --region stack-region
```

Exemplo 2: gerar um modelo

O comando a seguir obtém informações sobre uma OpsWorks camada existente e gera um CloudFormation modelo. O modelo, se provisionado, obtém um resultado semelhante ao uso do Chef 14. Neste exemplo, nenhum recurso é provisionado, porque o `--provision-application` parâmetro está definido como. `FALSE`

Substitua *`stack-region`* e *`layer-id`* pelos valores de sua OpsWorks pilha e camada.

```
python3 stack_exporter.py \
    --layer-id layer-id \
    --region stack-region \
    --provision-application FALSE
```

Depois de executar o comando, você pode revisar o modelo na biblioteca de modelos do Application Manager no Systems Manager e também pode provisionar o modelo. Para obter mais informações sobre como visualizar a biblioteca de modelos, consulte Como trabalhar com a biblioteca de modelos no Guia do usuário AWS Systems Manager .

Etapa 5: provisionar uma CloudFormation pilha

> **Note**
>
> Você só precisará concluir esta etapa se definir o parâmetro `--provision-application` do script como `FALSE`.

Quando você especifica o `--provision-application` parâmetro com um valor de`FALSE`, a saída do script fornece o nome e a URL do CloudFormation modelo. Esse modelo representa uma proposta de substituição para sua OpsWorks pilha e camada existentes.

Você pode provisionar o modelo usando a biblioteca de modelos do Application Manager (recomendado) ou usando CloudFormation. Para obter mais informações sobre como trabalhar com a biblioteca de modelos, consulte Como trabalhar com a biblioteca de modelos no Guia do usuário AWS Systems Manager .

Etapa 6: revisar os recursos provisionados

Agora, você está pronto para revisar os recursos provisionados.

1. Analise os recursos da pilha provisionada usando o console. AWS CloudFormation
   a. Abra o AWS CloudFormation console em https://console.aws.amazon.com/cloudformation e escolha Stacks.
   b. Na página Pilhas, escolha a pilha, e depois escolha a guia Recursos.
   c. Na guia Recursos, revise os recursos listados para sua pilha. A lista de recursos inclui um grupo de EC2 Auto Scaling, que você pode analisar no console do Auto Scaling, ou. AWS CLI
2. Analise os recursos do aplicativo usando o Systems Manager Application Manager.
   a. Abra o console do Systems Manager em https://console.aws.amazon.com/systems-manager/.
   b. No painel de navegação, escolha Application Manager.
   c. Na seção Aplicações, escolha a aplicação personalizada. O Application Manager abre a guia Visão geral.
   d. Escolha a guia Recursos. A guia Recursos mostra todos os recursos que foram migrados para sua OpsWorks pilha e camada. O nome do aplicativo inclui o nome da OpsWorks

pilha e é formatado como *app* - *stack-name* -, *suffix* onde *suffix* representa os primeiros seis caracteres do ID da pilha. Para obter mais informações sobre a visualização de recursos no Application Manager, consulte Visualização de recursos do aplicativo no Guia do usuário AWS Systems Manager .

Etapa 7: iniciar uma instância

Depois de provisionar uma instância, você estará pronto para testar a instância. Neste momento, não há instâncias em execução.

Para colocar suas instâncias on-line, ajuste os valores `Min`, `Max`, e `Desired capacity` do grupo do Auto Scaling para um número que faça sentido para seu aplicativo. Inicialmente, talvez você queira definir esses valores como 1, para colocar uma única instância on-line e verificar se a instância executa todas as ações esperadas, incluindo a execução de suas receitas personalizadas do Chef.

Etapa 8: revisar a instância

Depois de iniciar uma instância, verifique se ela é executada conforme o esperado.

1. Revise os logs `startup` e `terminate` do Chef, localizados no bucket do S3 especificado pelo parâmetro `--command-logs-bucket` do script. Por padrão, os registros são armazenados em um bucket com o nome `aws-opsworks-application-manager-logs-`*`account-id`*.

   a. Faça login no AWS Management Console e abra o console do Amazon S3 em. https://console.aws.amazon.com/s3/

   b. Escolha o bucket que contém seus logs.

   c. Navegue até o `ApplyChefRecipes` prefixo para ver seus registros.

2. Verifique a conectividade e a integridade do Application Load Balancer.

   Siga as seguintes etapas para visualizar os logs de acesso do seu balanceador de carga. É possível especificar o bucket do S3 no qual deseja armazenar os logs de acesso do balanceador de carga usando o parâmetro `--lb-access-logs-path` do script.

   a. Faça login no AWS Management Console e abra o console do Amazon S3 em. https://console.aws.amazon.com/s3/

   b. Escolha seu bucket do S3 e, em seguida, navegue até o prefixo que contém seus registros.

3. Verifique se a instância é aprovada em todas as verificações de integridade do ajuste de escala automático e do Application Load Balancer (se você tiver configurado alguma).

   Você pode ver informações sobre a integridade do ajuste de escala automático na nova guia Instâncias.

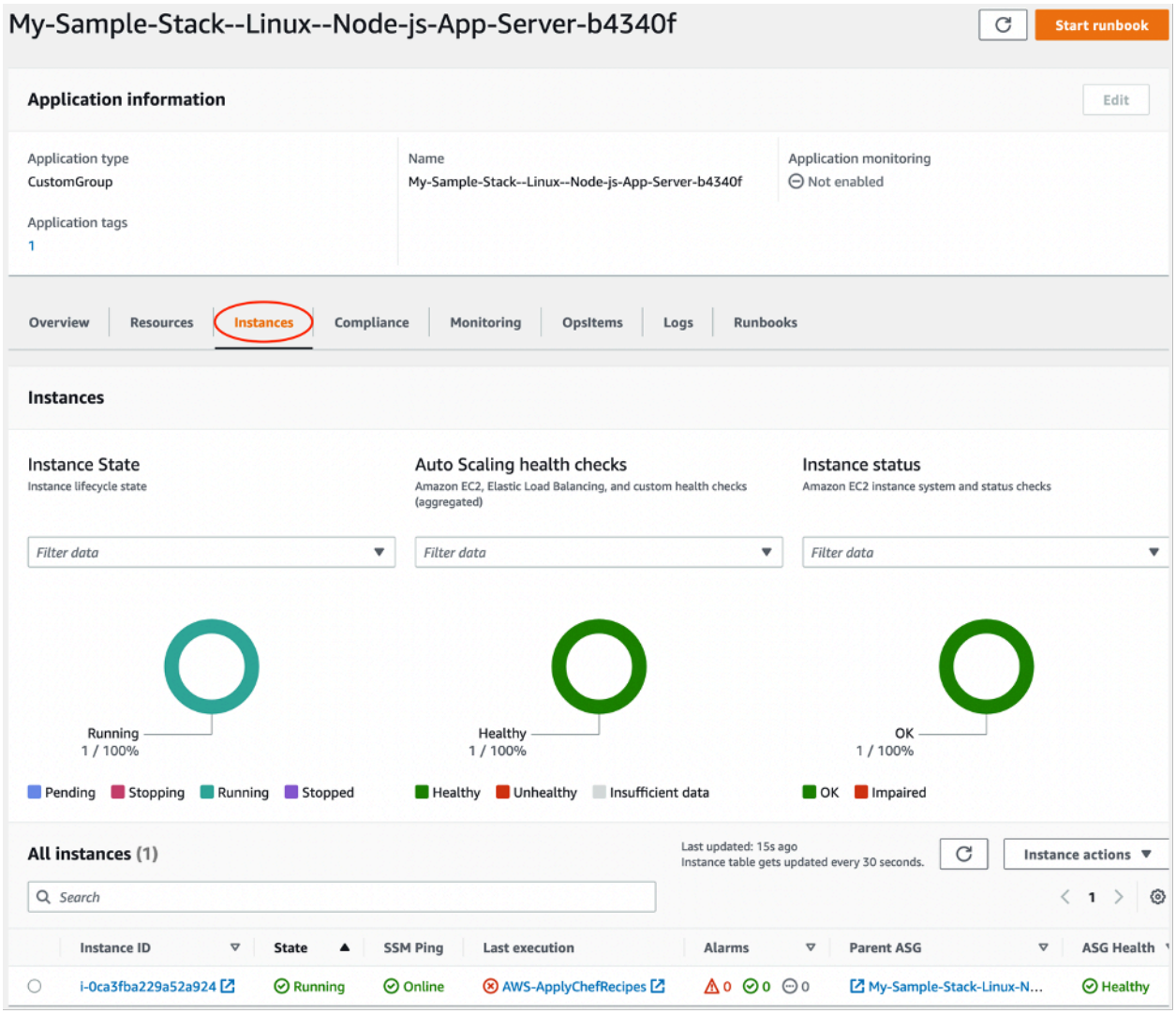
   a. Abra o console do Systems Manager em https://console.aws.amazon.com/systems-manager/.
   b. No painel de navegação, escolha Application Manager.
   c. Na seção Aplicativos, escolha Aplicativos personalizados.
   d. Selecione o aplicativo na lista. O Application Manager abre a guia Visão geral.
   e. Escolha a guia Instâncias para ver informações sobre a integridade do ajuste de escala automático.

Depois de verificar se as receitas do Chef são executadas com êxito, você pode diminuir a capacidade do grupo do Auto Scaling para encerrar a instância. Se você tiver alguma receita de terminação personalizada, verifique se as receitas funcionam conforme o esperado.

Etapa 9: monitore e execute operações em suas instâncias usando o Systems Manager Application Manager

Agora você pode monitorar e executar operações em suas instâncias usando uma nova guia Instâncias na página do Application Manager. Para obter mais informações sobre como trabalhar com a guia Instâncias, consulte Como trabalhar com as instâncias do seu aplicativo no Guia do usuário AWS Systems Manager .

Você pode usar a guia Instâncias para visualizar várias AWS instâncias em um só lugar. Usando essa guia, você pode ver informações sobre a integridade da instância e solucionar problemas.



Siga as etapas a seguir para ver a guia Instâncias.

1. Abra o console do Systems Manager em https://console.aws.amazon.com/systems-manager/.
2. No painel de navegação, escolha Application Manager.
3. Na seção Aplicativos, escolha Aplicativos personalizados.
4. Selecione o aplicativo na lista. O Application Manager abre a guia Visão geral.
5. Escolha a guia Instâncias para ver informações sobre o status e a EC2 integridade da sua instância.

Perguntas frequentes

Veja a seguir FAQs respostas para algumas perguntas comuns.

Tópicos

- Quais AWS OpsWorks Stacks versões posso migrar?
- Quais versões do Chef minhas instâncias migradas podem usar?
- Quais tipos de repositório posso migrar?
- Posso continuar usando um repositório Git privado?
- Quais chaves SSH posso usar para acessar minhas instâncias?
- Por que minhas instâncias estão aumentando e diminuindo automaticamente?
- Posso desativar o ajuste de escala automático?
- Posso realizar atualizações de kernel e pacote em EC2 instâncias lançadas?
- Por que os volumes do EBS em minhas instâncias não contêm dados?
- Por que os volumes do EBS descritos no meu modelo de lançamento não estão montados?
- Onde posso encontrar receitas do Chef e registros de volume do Mount EBS?
- Onde posso encontrar o log de depuração do script de migração?
- O script de migração oferece suporte ao controle CloudFormation de versão do modelo?
- Posso migrar várias camadas?
- Como crio um parâmetro SecureString?
- Como posso proteger as instâncias do novo grupo do Auto Scaling contra eventos de encerramento?
- Quais balanceadores de carga estão disponíveis com o script de migração?
- As receitas de configuração do livro de receitas personalizado foram migradas?
- Posso executar receitas de implantação e desimplantação em minhas instâncias recém-criadas?
- Posso alterar quais sub-redes meu grupo do Auto Scaling abrange?

Quais AWS OpsWorks Stacks versões posso migrar?

Você só pode migrar pilhas do Chef 11.10 e do Chef 12, do Amazon Linux, do Amazon Linux 2, do Ubuntu e do Red Hat Enterprise Linux 7.

Quais versões do Chef minhas instâncias migradas podem usar?

As instâncias migradas podem usar as versões 11 a 14 do Chef.

> **Note**
>
> Não há suporte para migração de pilhas do Windows.

Quais tipos de repositório posso migrar?

Você pode migrar os tipos de repositório S3, Git e HTTP.

Posso continuar usando um repositório Git privado?

Sim, você pode continuar usando um repositório Git privado.

Se você usar um GitHub repositório privado, deverá criar uma nova chave de `Ed25519` host para SSH. Isso ocorre porque GitHub alterou quais chaves são suportadas no SSH e removeu o protocolo Git não criptografado. Para obter mais informações sobre a chave do `Ed25519` host, consulte a postagem do GitHub blog Melhorando a segurança do protocolo Git em. GitHub Depois de gerar uma nova chave de host `Ed25519`, crie um parâmetro `SecureString` do Systems Manager para essa chave SSH e use o nome do parâmetro como o valor do parâmetro `--repo-private-key`. Para obter mais informações sobre como criar um `SecureString` parâmetro do Systems Manager, consulte Create a SecureString parameter (AWS CLI) no Guia AWS Systems Manager do usuário.

Para qualquer outro tipo de repositório Git, crie um parâmetro `SecureString` do Systems Manager para essa chave SSH e use o nome do parâmetro como o valor do parâmetro `--repo-private-key` do script.

Quais chaves SSH posso usar para acessar minhas instâncias?

Quando você executa o script, o script migra as chaves SSH e as instâncias configuradas na pilha. Você pode usar as chaves SSH para acessar sua instância. Se as chaves SSH forem fornecidas para a pilha e a instância, o script usará as chaves da pilha. Se você não tiver certeza de quais chaves SSH usar, visualize as instâncias no EC2 console (https://console.aws.amazon.com/ec2/). A página Detalhes no EC2 console mostra as chaves SSH da sua instância.

Por que minhas instâncias estão aumentando e diminuindo automaticamente?

O ajuste de escala automático dimensiona as instâncias com base nas regras de escalabilidade do grupo do Auto Scaling. Você pode definir os valores de capacidade mínima, máxima e desejada para seu grupo. O grupo do Auto Scaling dimensiona automaticamente sua capacidade de acordo com a atualização desses valores.

Posso desativar o ajuste de escala automático?

Você pode desativar o ajuste de escala automático definindo os valores de capacidade mínima, máxima e desejada do grupo do Auto Scaling para o mesmo número. Por exemplo, se você quiser sempre ter dez instâncias, defina os valores de capacidade mínima, máxima e desejada como dez.

Posso realizar atualizações de kernel e pacote em EC2 instâncias lançadas?

Por padrão, as atualizações do kernel e dos pacotes ocorrem quando a EC2 instância é inicializada. Use as etapas a seguir para realizar atualizações de kernel ou pacote em uma EC2 instância iniciada. Por exemplo, talvez você queira aplicar atualizações após executar o deploy ou o configure recipes.

1. Conecte-se à sua EC2 instância.
2. Crie a função `perform_upgrade` a seguir e execute-a na sua instância.

```
perform_upgrade() {
    #!/bin/bash
    if [ -e '/etc/system-release' ] || [ -e '/etc/redhat-release' ]; then
     sudo yum -y update
    elif [ -e '/etc/debian_version' ]; then
     sudo apt-get update
     sudo apt-get dist-upgrade -y
    fi
}
perform_upgrade
```

3. Depois das atualizações do kernel e do pacote, talvez seja necessário reinicializar sua EC2 instância. Para verificar se é necessário reinicializar, crie a `reboot_if_required` função a seguir e execute-a na sua EC2 instância.

```
reboot_if_required () {
 #!/bin/bash
 if [ -e '/etc/debian_version' ]; then
   if [ -f /var/run/reboot-required ]; then
     echo "reboot is required"
   else
     echo "reboot is not required"
   fi
 elif [ -e '/etc/system-release' ] || [ -e '/etc/redhat-release' ]; then
  export LC_CTYPE=en_US.UTF-8
```

```
  export LC_ALL=en_US.UTF-8
  LATEST_INSTALLED_KERNEL=`rpm -q --last kernel | perl -X -pe 's/^kernel-(\S+).*/
$1/' | head -1`
  CURRENTLY_USED_KERNEL=`uname -r`
  if [ "${LATEST_INSTALLED_KERNEL}" != "${CURRENTLY_USED_KERNEL}" ];then
     echo "reboot is required"
  else
     echo "reboot is not required"
  fi
 fi
}
reboot_if_required
```

4. Se estiver executando os `reboot_if_required` resultados em uma `reboot is required` mensagem, reinicie a EC2 instância. Se você receber uma `reboot is not required` mensagem, não precisará reinicializar a EC2 instância.

Por que os volumes do EBS em minhas instâncias não contêm dados?

Quando você executa o script, o script migra a configuração dos volumes do EBS, criando uma arquitetura substituta para sua OpsWorks pilha e sua camada. O script não migra instâncias reais nem os dados contidos nas instâncias. O script migra somente a configuração dos volumes do EBS no nível da camada e anexa os volumes vazios do EBS às instâncias iniciadas. EC2

Siga as etapas a seguir para extrair dados dos volumes do EBS de suas instâncias anteriores.

1. Crie um snapshot dos volumes do EBS de instâncias anteriores. Para obter mais informações sobre a criação de um snapshot, consulte Criar um snapshot do Amazon EBS no Guia do usuário da Amazon EC2 .
2. Criar um volume a partir de um snapshot. Para obter mais informações sobre a criação de um volume a partir de um snapshot, consulte Criar um volume a partir de um snapshot no Guia EC2 do usuário da Amazon.
3. Anexe o volume que você criou à instância. Para obter mais informações sobre anexar volumes, consulte Anexar um volume do Amazon EBS a uma instância no Guia EC2 do usuário da Amazon.

Por que os volumes do EBS descritos no meu modelo de lançamento não estão montados?

Se você fornecer uma ID de modelo de execução para o parâmetro `--launch-template` com volumes do EBS, o script anexará os volumes do EBS, mas não montará os volumes. Você pode montar os volumes do EBS anexados executando o `MountEBSVolumes` RunCommand documento que o script criou para a EC2 instância iniciada.

Se você não definir o `--launch-template` parâmetro, o script cria um modelo e, quando o grupo Auto Scaling inicia uma nova EC2 instância, o grupo Auto Scaling anexa automaticamente os volumes do EBS e, em seguida, executa `SetupAutomation` o comando para montar os volumes anexados nos pontos de montagem definidos nas configurações da camada.

Onde posso encontrar receitas do Chef e registros de volume do Mount EBS?

OpsWorks entrega os registros em um bucket do S3 que você pode especificar fornecendo um valor para o `--command-logs-bucket` parâmetro. O nome padrão do bucket do S3 tem o formato: `aws-opsworks-stacks-application-manager-logs-`*`account-id`*. Os registros de receitas do Chef são armazenados no prefixo `ApplyChefRecipes`. Os registros de volume do Mount EBS são armazenados no prefixo `MountEBSVolumes`. Todas as camadas que são migradas de uma pilha entregam registros para o mesmo bucket do S3.

> **Note**
>
> - A configuração do ciclo de vida do bucket S3 inclui uma regra para excluir os registros após 30 dias. Se quiser manter os registros por mais de 30 dias, você deve atualizar a regra na configuração do ciclo de vida do bucket S3.
> - Atualmente, registra OpsWorks apenas o Chef `setup` e `terminate` as receitas.

Onde posso encontrar o log de depuração do script de migração?

O script coloca os logs de depuração em um bucket chamado `aws-opsworks-stacks-transition-logs-`*`account-id`*. Você pode encontrar os logs de depuração pasta `migration_script` do bucket do S3 em pastas que correspondem ao nome da camada que você migrou.

O script de migração oferece suporte ao controle CloudFormation de versão do modelo?

O script gera documentos do tipo Systems Manager CloudFormation que criam uma substituição para a camada ou pilha que você deseja migrar. Executar o script novamente, mesmo com os mesmos parâmetros, exporta uma nova versão do modelo de camada exportado anteriormente. As versões do modelo são armazenadas no mesmo bucket do S3 que os logs do script.

Posso migrar várias camadas?

O parâmetro `--layer-id` do script passa em uma única camada. Para migrar várias camadas, execute novamente o script e passe uma diferente `--layer-id`.

As camadas que fazem parte da mesma OpsWorks pilha são listadas sob o mesmo aplicativo no Application Manager.

1. Abra o console do Systems Manager em https://console.aws.amazon.com/systems-manager/.
2. No painel de navegação, escolha Application Manager.
3. Na seção Aplicativos, escolha Aplicativos personalizados.
4. Escolha o aplicativo. O nome do aplicativo começa com `app-`*`stack-name`*`-`*`first-six-characters-stack-id`*.
5. O elemento de nível superior, começando com app, mostra todos os componentes que correspondem à sua OpsWorks pilha. Isso inclui componentes correspondentes à sua OpsWorks camada.
6. Escolha o componente correspondente à camada para visualizar os recursos da camada. Os componentes que representam OpsWorks camadas também são visíveis na seção Aplicativos personalizados como aplicativos individuais.

Como crio um parâmetro **`SecureString`**?

Você pode usar o Systems Manager para criar um parâmetro `SecureString`. Para obter mais informações sobre como criar um `SecureString` parâmetro do Systems Manager, consulte Create a SecureString parameter (AWS CLI) ou Create a Systems Manager parameter (console) no Guia do AWS Systems Manager Usuário.

Você deve fornecer um parâmetro `SecureString` como valor para os parâmetros `--http-username`, `--http-password`, ou `--repo-private-key`.

Como posso proteger as instâncias do novo grupo do Auto Scaling contra eventos de encerramento?

Você pode proteger as instâncias definindo o `--enable-instance-protection` parâmetro TRUE e adicionando uma chave de `protected_instance` tag a cada EC2 instância que você deseja proteger contra eventos de encerramento. Quando você define o parâmetro `--enable-instance-protection` para TRUE e adiciona uma chave de tag `protected_instance`, o script adiciona uma política de encerramento personalizada ao seu novo grupo do Auto Scaling e suspende o processo `ReplaceUnhealthy`. As instâncias com a chave de tag `protected_instance` são protegidas dos seguintes eventos de encerramento:

- Eventos de redução de escala horizontalmente
- Atualização de instância
- Rebalanceamento
- Vida útil máxima da instância
- Permitir o término de instâncias
- Rescisão e substituição de instâncias não íntegras

> **Note**
>
> Você deve definir a chave da tag `protected_instance` nas instâncias que deseja proteger. Observe que a chave da tag faz distinção entre maiúsculas e minúsculas. Qualquer instância com essa chave de tag é protegida, independentemente do valor da tag.
> Para reduzir o tempo de execução da política de encerramento personalizada, você pode aumentar o número padrão de instâncias que a função do Lambda usa para filtrar instâncias protegidas atualizando o valor da variável de código da função `default_sample_size`. O valor padrão é 15. Se você aumentar o `default_sample_size`, talvez seja necessário aumentar a memória alocada para a função do Lambda, o que aumentaria o custo da sua função do Lambda. Para obter mais informações sobre a definição de preço do AWS Lambda , consulte Definição de preço do AWS Lambda.

Quais balanceadores de carga estão disponíveis com o script de migração?

O script fornece três opções de balanceador de carga.

- (Recomendado) Crie um novo Application Load Balancer. Por padrão, o script cria um novo Application Load Balancer. Também é possível definir o parâmetro `--lb-type` para `ALB`. Para obter mais informações sobre os Application Load Balancers, consulte O que é Application Load Balancer? no guia do usuário do Elastic Load Balancing
- Se um Application Load Balancer não for uma opção, crie um Classic Load Balancer definindo o parâmetro `--lb-type` para `Classic`. Se você selecionar essa opção, seu Classic Load Balancer existente anexado à sua OpsWorks camada será mantido separado do seu aplicativo. Para obter mais informações sobre Application Load Balancers, consulte O que é um balanceador de carga clássico? no guia do usuário do Elastic Load Balancing: Classic Load Balancers.
- Você pode conectar um balanceador de carga existente definindo o parâmetro `--lb-type` para `None`.

> **Important**
>
> Recomendamos criar novos balanceadores de carga do Elastic Load Balancing para suas camadas do AWS OpsWorks Stacks. Se escolher usar um balanceador de carga do Elastic Load Balancing existente, você deve primeiro confirmar que não está sendo usado para outros fins e não tem instâncias anexadas. Depois que o balanceador de carga é anexado à camada, OpsWorks remove todas as instâncias existentes e configura o balanceador de carga para lidar somente com as instâncias da camada. Apesar de ser tecnicamente possível usar o console do Elastic Load Balancing ou API para modificar a configuração do balanceador de carga após anexá-lo a camada, você não deve fazê-lo; as mudanças não serão permanentes.

Para anexar seu balanceador OpsWorks de carga de camada existente ao seu grupo de Auto Scaling

1. Execute o script de migração com o parâmetro `--lb-type` definido como `None`. Quando o valor é definido como `None`, o script não clona nem cria um balanceador de carga.
2. Depois que o script implantar a CloudFormation pilha, atualize os `Min` `Max` grupos `Desired capacity` e valores do Auto Scaling e teste seu aplicativo.
3. Escolha `Link to the template` mostrada na saída do script. Se você fechou seu terminal, siga estas etapas para acessar o modelo.

   a. Abra o console do Systems Manager em https://console.aws.amazon.com/systems-manager/.

   b. No painel de navegação, escolha Application Manager.
   c. Escolha CloudFormation pilhas e, em seguida, escolha Biblioteca de modelos.
   d. Escolha Minha propriedade e localize seu modelo.
4. No CloudFormation modelo, escolha Editar no menu Ações.
5. Atualize a `LabelBalancerNames` propriedade na seção de `ApplicationAsg` recursos do CloudFormation modelo.

```
ApplicationAsg:
   DependsOn: CustomTerminationLambdaPermission
   Properties:
   #(other properties in ApplicationAsg to remain unchanged)
      LoadBalancerNames:
        - load-balancer-name
      HealthCheckType: ELB
```

6. Se você quiser que sua verificação de integridade de instâncias de grupo do Auto Scaling também use a verificação de integridade do balanceador de carga, remova a seção `HealthCheckType` abaixo e insira `ELB`. Se você precisar apenas de verificações de EC2 saúde, não precisará alterar o modelo.
7. Salve as alterações. Salvar cria uma nova versão padrão do modelo. Se esta é a primeira vez que você executa o script para a camada e a primeira vez que você salva as alterações no console, a versão mais recente é 2.
8. Em Ações, escolha Provisionar pilha.
9. Confirme que você deseja usar a versão padrão do modelo. Certifique-se de que a opção Selecionar uma pilha existente esteja selecionada e escolha a CloudFormation pilha a ser atualizada.
10. Escolha Próximo para cada uma das páginas subsequentes até ver a página Revisar e provisionar. Na página Revisar e provisionar, escolha Eu reconheço que AWS CloudFormation pode criar recursos do IAM com nomes personalizados e entendo que alterações no modelo selecionado podem AWS CloudFormation fazer com que os AWS recursos existentes sejam atualizados ou removidos.
11. Selecione Provision stack (Provisionar pilha).

Se você precisar reverter as atualizações, siga as seguintes etapas:

1. Escolha Ações e depois escolha Provisionar pilha.
2. Selecione Escolher uma das versões existentes e, em seguida, escolha a versão anterior do modelo.
3. Escolha Selecionar uma pilha existente e, em seguida, escolha a CloudFormation pilha a ser atualizada.

As receitas de configuração do livro de receitas personalizado foram migradas?

Não há suporte para execução de livros de receitas personalizados durante um evento de configuração. O script migra o livro de receitas personalizado, configura receitas e cria um runbook do Systems Manager Automation para você. No entanto, você deverá executar as receitas manualmente.

Siga as seguintes etapas para executar suas receitas de configuração.

1. Abra o console do Systems Manager em https://console.aws.amazon.com/systems-manager/.
2. No painel de navegação, escolha Application Manager.
3. Na seção Aplicativos, escolha Aplicativos personalizados.
4. Escolha o aplicativo. O nome do aplicativo começa com `app-`*`stack-name`*.
5. Escolha Recursos e, em seguida, escolha o runbook de configuração.
6. Escolha Executar automação.
7. Escolha a instância IDs para a qual você deseja executar as receitas de configuração e, em seguida, escolha Executar.

Posso executar receitas de implantação e desimplantação em minhas instâncias recém-criadas?

O script pode criar três possíveis runbooks de automação, dependendo da configuração da sua camada.

- Configuração
- Configurar
- Encerrar

O script também pode criar os seguintes parâmetros do Systems Manager que contêm valores de entrada para o documento `AWS-ApplyChefRecipes Run Command`.

- Configuração
- Implantar
- Configurar
- Desfazer a Implementação
- Encerrar

Quando ocorre um evento de expansão, o runbook de automação de configuração é executado automaticamente. Isso inclui a configuração e a implantação de receitas personalizadas de livros de receitas a partir de sua OpsWorks camada original. Quando um evento de aumento da escala na vertical acontece, o runbook de automação de término é executado automaticamente. O runbook terminate Automation contém as receitas de desligamento da sua camada original. OpsWorks

Se você deseja executar, desimplantar ou configurar receitas manualmente, siga as seguintes etapas.

1. Abra o console do Systems Manager em https://console.aws.amazon.com/systems-manager/.
2. No painel de navegação, escolha Application Manager.
3. Na seção Aplicativos, escolha Aplicativos personalizados.
4. Escolha o aplicativo. O nome do aplicativo começa com `app-`*`stack-name`*`-`*`first-six-characters-stack-id`*. O Application Manager abre a guia Visão geral.
5. Escolha Recursos e, em seguida, escolha o runbook de configuração de automação.
6. Escolha Executar automação.
7. Para o parâmetro de entrada do Automation runbook `applyChefRecipesPropertiesParameter`, consulte o parâmetro correto do Systems Manager. O nome do parâmetro Systems Manager segue o formato`/ApplyChefRecipes-Preset/`*`OpsWorks-stack-name`*`-`*`OpsWorks-layer-name`*`-`*`first-six-characters-stack-id`*`/`*`event`*, onde o valor para *`event`* está `ConfigureDeploy`, ou `Undeploy` dependendo das receitas que você deseja executar.
8. Escolha a instância IDs em que você deseja executar as receitas e escolha Executar.

Posso alterar quais sub-redes meu grupo do Auto Scaling abrange?

Por padrão, o grupo Auto Scaling abrange todas as sub-redes em sua VPC de pilha. OpsWorks Para atualizar as sub-redes a serem abrangidas, siga as seguintes etapas:

1. Escolha `Link to the template` mostrada na saída do script. Se você fechou seu terminal, siga estas etapas para acessar o modelo.
   a. Abra o console do Systems Manager em https://console.aws.amazon.com/systems-manager/.
   b. No painel de navegação, escolha Application Manager.
   c. Escolha CloudFormation pilhas e, em seguida, escolha Biblioteca de modelos.
   d. Escolha Minha propriedade e localize seu modelo.
2. Em Ações, escolha Provisionar pilha.
3. Confirme que você deseja usar o modelo padrão. Escolha Selecionar uma pilha existente e, em seguida, escolha a CloudFormation pilha a ser atualizada.

   > **Note**
   > Se você executou o script com o `--provision-application` parâmetro definido como`FALSE`, deverá criar uma nova CloudFormation pilha.

4. Para o `SubnetIDs` parâmetro, forneça uma lista separada por vírgulas da sub-rede IDs que você deseja que seu grupo de Auto Scaling abranja.
5. Escolha Avançar até ver a página Revisar e provisionar.
6. Na página Revisar e provisionar, escolha Eu reconheço que AWS CloudFormation pode criar recursos do IAM com nomes personalizados e entendo que alterações no modelo selecionado podem AWS CloudFormation fazer com que os AWS recursos existentes sejam atualizados ou removidos.
7. Selecione Provision stack (Provisionar pilha).

Solução de problemas

Esta seção contém alguns problemas comuns, e soluções sugeridas para esses problemas.

Tópicos

- A entidade principal fornecida não é valida
- Não é possível excluir a CloudFormation pilha quando as instâncias protegidas por grupos do Auto Scaling estão habilitadas
- Erro de acesso negado ao fornecer bucket e prefixo S3 existentes

A entidade principal fornecida não é valida

Problema: você recebe uma mensagem de erro informando que a entidade principal fornecida não é válida.

Causa: isso ocorre porque o grupo do Auto Scaling não tem um perfil de serviço.

Solução: crie um grupo do Auto Scaling na região em que o erro ocorreu. Criar um grupo do Auto Scaling cria o perfil vinculado ao serviço necessário para sua política de rescisão personalizada.

Não é possível excluir a CloudFormation pilha quando as instâncias protegidas por grupos do Auto Scaling estão habilitadas

Problema: o `--enable-instance-protection` parâmetro está definido como `TRUE` e algumas das EC2 instâncias do seu grupo do Auto Scaling são protegidas com a chave de `protected_instance` tag, o que impede que sua AWS CloudFormation pilha seja completamente excluída.

Causa: as EC2 instâncias têm uma chave de `protected_instance` tag que as protege de eventos de encerramento.

Solução: remova a chave de `protected_instance` tag das EC2 instâncias. Isso permite que o grupo do Auto Scaling reduza a escala verticalmente. Depois que o grupo Auto Scaling for reduzido, você poderá excluir a AWS CloudFormation pilha.

Erro de acesso negado ao fornecer bucket e prefixo S3 existentes

Problema: você recebe um erro `AccessDenied` ao fornecer um bucket e um prefixo do S3 existentes.

Causa: a política de bucket do S3 não fornece as permissões necessárias para entregar os logs do balanceador de carga ao bucket.

Solução: atualize a política de bucket do S3 para permitir que o script entregue os registros de acesso do balanceador de carga ao bucket. Para obter mais informações sobre como atualizar a

política de bucket, consulte Habilitar registros de acesso para seu Application Load Balancer no guia do usuário do Elastic Load Balancing: Application Load Balancers.

Usando a AWS OpsWorks Stacks ferramenta Detach in Place

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível.

Esta seção descreve como usar a ferramenta AWS OpsWorks Stacks Detach in Place para separar suas OpsWorks instâncias do serviço OpsWorks Stacks.

As instâncias que você desanexar permanecerão na sua Conta da AWS, mas você não poderá mais gerenciá-las usando OpsWorks. Em vez disso, você usará a Amazon EC2 ou qualquer abordagem EC2 compatível para configurar e gerenciar as instâncias. AWS Systems Manager

Em um alto nível, o processo de desapego envolve as seguintes etapas:

1. A ferramenta realiza verificações de validação para garantir que os recursos estejam prontos para serem separados.
2. A ferramenta exporta o JSON personalizado da sua OpsWorks pilha e o armazena como um objeto no Amazon S3.
3. A ferramenta cria documentos do Systems Manager Automation representando cada evento do ciclo de vida do OpsWorks Stacks.
4. A ferramenta cria um AWS Service Catalog AppRegistry catálogo para todas as instâncias que estão sendo desanexadas e separa quaisquer balanceadores de carga do Elastic Load Balancing (ELB) das camadas. OpsWorks
5. Por fim, a ferramenta separa e cancela o registro de outros recursos, incluindo instâncias do Amazon Relational Database Service (Amazon RDS).

Como o processo funciona

A ferramenta Detach In Place fornece os três comandos a seguir e uma experiência semelhante à de um assistente que orienta você por uma série de etapas para verificar e configurar suas instâncias antes de prosseguir com a separação da camada.

Comando	Descrição
`handle-prerequisites`	Esse comando analisa se todas as instâncias em uma camada são elegíveis para desanexaç ão e resolve os pré-requisitos. As instâncias devem estar em um estado íntegro OpsWorks, não podem ter escaladores automáticos baseados em tempo ou carga e devem ter a versão mais recente do OpsWorks Agente instalada. Além disso, o comando verifica se todas as instâncias têm as permissões necessárias para oferecer suporte ao SSM Agent e se a versão mais recente do SSM Agent está instalada. O comando instalará o SSM Agent se ele não estiver presente e atualizará o SSM Agent se ele não estiver usando a versão mais recente. O comando também adicionará todas as permissões necessárias.
`detach`	Esse comando separa todas as OpsWorks instâncias da camada especificada. Primeiro, o comando executará uma verificação de pré-requisitos para garantir que a camada se qualifique para desanexação. Se você não quiser resolver os pré-requisitos, você tem a opção de forçar a separação. Em seguida, o comando indicará que todas as tags adicionadas às suas instâncias por meio

Comando	Descrição
	de OpsWorks marcação APIs ou propagaçã o de tags de suas camadas e pilhas serão mantidas. Você pode remover qualquer uma dessas etiquetas usando as relevantes EC2 APIs após a conclusão da separação. Em seguida, o comando verificará se você deseja exportar a configuração relacionada ao Chef para os parâmetros SSM. Se você tiver um Classic Load Balancer anexado à camada, o comando perguntará se ele pode desconectar o balanceador de carga para evitar qualquer tempo de inatividade.
`cleanup`	Esse comando exclui todas as entidades OpsWorks da sua conta. Isso encerrará as instâncias e excluirá todas as pilhas. Isso deve ser usado para recursos que não são mais necessários como última etapa para limpar a conta. **Note** Recomendamos que você execute a nova configuração por alguns dias antes de executar o `cleanup` comando. Isso garante que todas as configurações necessárias da pilha estejam prontamente disponíveis, se necessário.

Limitações

O objetivo principal da ferramenta Detach In Place é separar com segurança as instâncias do OpsWorks Stacks. Esta seção resume as limitações da ferramenta.

- Agente SSM do Windows — Se o agente SSM não estiver instalado na instância, você precisará instalá-lo manualmente. O mesmo se aplica se o Agente não estiver atualizado para a versão mais recente.
- Instâncias do Auto Scaling de tempo/carga — A ferramenta de desvinculação não é compatível com instâncias com o Auto Scaling ativado. Você deve desativar o Auto Scaling nas instâncias que você deseja separar.
- Permissões — A ferramenta de desanexação não cria nem gera entidades do IAM especificadas na página de Permissões do OpsWorks console.
- Aplicativos — A ferramenta de separação não cria nem gera aplicativos fora do OpsWorks.

Conceitos básicos

Etapa 1: Verificar se os pré-requisitos foram atendidos

Todos os três comandos da ferramenta Detach In Place são scripts Python, que você pode executar localmente, EC2 por exemplo ou usando. AWS CloudShell

AWS CloudShell é um shell baseado em navegador que fornece acesso por linha de comando aos AWS recursos selecionados. Região da AWS AWS CloudShell vem pré-instalado com ferramentas populares (como AWS CLI Python). Ao usar AWS CloudShell, você usa as mesmas credenciais que usa para entrar no console.

Este passo a passo pressupõe que você esteja usando. AWS CloudShell

Etapa 2: Baixe o script

1. Faça o download do arquivo zip que contém o script de migração e todos os arquivos relevantes executando o seguinte comando:

   ```
   aws s3api get-object \
   --bucket detach-in-place-bucket-prod-us-east-1 \
   --key detach_in_place_script.zip detach_in_place_script.zip
   ```

2. Descompacte o arquivo executando o comando a seguir.

```
unzip detach_in_place_script.zip
```

Depois que o arquivo for descompactado, os seguintes arquivos estarão disponíveis:

- README.md
- LICENSE
- NOTICE
- requirements.txt
- TODO.py

3. Se necessário, instale `pipenv` executando o comando a seguir.

```
pip install pipenv
```

Etapa 3: executar o script

Primeiro, configure seu ambiente para que você possa executar o script executando os comandos a seguir.

```
pipenv install -r requirements.txt
pipenv shell
```

Em seguida, revise os parâmetros do script.

Command	Parameter	Descrição	Tipo	Obrigatório	Padrão
`handle-prerequisites`	`--layer-id`	O ID da camada que você deseja desanexar.	String	Sim	-
	`--region`	A região da OpsWorks pilha. Se a região da OpsWorks pilha e a região do endpoint da API forem diferentes, use a região da pilha. Essa é	String	Não	us-east-1

Command	Parameter	Descrição	Tipo	Obrigatório	Padrão
		a mesma região que os outros recursos que fazem parte da sua OpsWorks pilha (por exemplo, EC2 instâncias e sub-redes).			
`detach`	`--layer-id`	ID da camada que você deseja desanexar.	String	Sim	-
	`--batch-size`	Número de instâncias a serem separadas de uma camada (por exemplo, 5).	String	Não	-
	`--region`	A região da OpsWorks pilha. Se a região da OpsWorks pilha e a região do endpoint da API forem diferentes, use a região da pilha. Essa é a mesma região que os outros recursos que fazem parte da sua OpsWorks pilha (por exemplo, EC2 instâncias e sub-redes).	String	Não	us-east-1

Command	Parameter	Descrição	Tipo	Obrigatório	Padrão
`cleanup`	`--stack-id`	ID da pilha que você deseja excluir.	String	Não	Mutuamente exclusivo, você deve especificar um ID de camada ou um ID de pilha
	`--layer-id`	ID da camada que você deseja excluir	String	Não	
	`--region`	A região da OpsWorks pilha. Se a região da OpsWorks pilha e a região do endpoint da API forem diferentes, use a região da pilha. Essa é a mesma região que os outros recursos que fazem parte da sua OpsWorks pilha (por exemplo, EC2 instâncias e sub-redes).	String	Não	us-east-1

Você pode ver as opções disponíveis para os `cleanup` comandos`detach`, `handle-prerequisites` e executando os comandos com a `--help` opção da seguinte forma:

```
python3 layer_detacher.py detach --help
python3 layer_detacher.py handle-prerequisites --help
python3 layer_detacher.py cleanup --help
```

Agora você está pronto para começar. Os exemplos a seguir mostram como você pode executar os comandos para diferentes casos de uso.

Exemplos:

- Exemplo 1: Verifique se uma camada preenche todos os pré-requisitos e está qualificada para desanexação
- Exemplo 2: Desanexar todas as instâncias de uma camada
- Exemplo 3: Separar todas as instâncias de uma camada em lotes
- Exemplo 4: Limpe todos os recursos de uma camada e exclua a camada
- Exemplo 5: Limpe todos os recursos de uma pilha e exclua a pilha

Exemplo 1: Verifique se uma camada preenche todos os pré-requisitos e está qualificada para desanexação

O comando a seguir lê as informações sobre uma OpsWorks camada (e as instâncias que ela inclui) e verifica se os seguintes pré-requisitos foram atendidos:

- Todas as instâncias estão online.
- Não há instâncias do Auto Scaling de Carga/Tempo.
- Todas as instâncias têm o OpsWorks Agente mais recente.
- Todas as instâncias têm o agente SSM mais recente instalado e configurado.
- Todas as instâncias têm um par de chaves SSH.
- Cada instância pertence a exatamente uma camada.

```
python3 layer_detacher.py handle-prerequisites \
--layer-id opsworks-layer-id \
--region opsworks-stack-region
```

Exemplo 2: Desanexar todas as instâncias de uma camada

O comando a seguir repetirá todas as instâncias da camada, verificará se as instâncias atendem aos pré-requisitos e tentará separar paralelamente todas as instâncias que atendam aos pré-requisitos. Se um ou mais pré-requisitos não forem atendidos, o comando fornecerá uma opção de desconexão forçada para as demais instâncias não compatíveis.

Antes de desanexar qualquer instância, o comando irá:

1. Salve o JSON personalizado e faça o upload para o S3.

2. Crie documentos de automação SSM para cada evento OpsWorks do ciclo de vida da camada e carregue os registros de execução dos documentos de automação no S3.
3. Crie um AppRegistry aplicativo para todas as instâncias que serão desanexadas. O aplicativo tem um grupo de recursos associado que contém todas as instâncias e recursos separados. Os recursos incluem documentos de automação de SSM e parâmetros de SSM que contêm informações sobre eventos do ciclo de vida e receitas personalizadas do Chef.
4. Separa o Classic Load Balancer da camada, se houver.

Esse comando modificará somente OpsWorks os recursos. O status das EC2 instâncias permanecerá o mesmo.

```
python3 layer_detacher.py detach \
--layer-id opsworks-layer-id \
--region opsworks-stack-region
```

Exemplo 3: Separar todas as instâncias de uma camada em lotes

O comando a seguir faz o mesmo que o exemplo anterior. A única diferença é que ele separa as instâncias em lotes.

Esse comando modificará somente OpsWorks os recursos. O status das EC2 instâncias permanecerá o mesmo.

```
python3 layer_detacher.py detach \
--layer-id opsworks-layer-id \
--region opsworks-stack-region \
--batch-size 5
```

Exemplo 4: Limpe todos os recursos de uma camada e exclua a camada

O comando a seguir repetirá todos os recursos de uma camada e os excluirá. Mais detalhadamente, ele interromperá e excluirá todas as instâncias OpsWorks e desconectará o balanceador de carga e cancelará o registro de instâncias EC2, elásticas e volumes do Amazon RDS. IPs Depois de limpar os recursos, a camada será excluída.

Esse comando excluirá OpsWorks recursos e EC2 instâncias. Se você quiser que suas EC2 instâncias permaneçam intocadas, use o `detach` comando antes de usar o `cleanup` comando. Dessa forma, o `cleanup` comando excluirá todos os recursos restantes.

```
python3 layer_detacher.py cleanup \
--layer-id opsworks-layer-id \
--region opsworks-stack-region
```

Exemplo 5: Limpe todos os recursos de uma pilha e exclua a pilha

O comando a seguir iterará em todas as camadas e, em seguida, nos recursos de cada camada. Para cada camada, o comando interromperá e excluirá todas as instâncias em OpsWorks e EC2, desconectará os balanceadores de carga e cancelará o registro de instâncias, elásticas e volumes do Amazon RDS. IPs Em seguida, o comando excluirá a camada. O mesmo processo será executado em todas as camadas pertencentes a essa pilha. Finalmente, depois que todas as camadas forem excluídas, a pilha será removida.

Esse comando excluirá OpsWorks recursos e EC2 instâncias. Se você quiser que suas EC2 instâncias permaneçam intocadas, use o `detach` comando antes de usar o `cleanup` comando. Dessa forma, o `cleanup` comando excluirá todos os recursos restantes.

```
python3 layer_detacher.py cleanup \
--stack-id opsworks-stack-id \
--region opsworks-stack-region
```

Etapa 4: continue operando seus recursos após se desconectar do OpsWorks

Depois de executar o `detach` comando, a ferramenta cria um novo AWS Service Catalog AppRegistry aplicativo correspondente à camada separada. O nome do aplicativo segue o formato*layer-name*`---`*layer-id*. Ele também adiciona a `OpsWorksLayerId` tag para identificar de forma exclusiva o aplicativo correspondente à camada separada.

Para adicionar novos AWS recursos a esse aplicativo (por exemplo, novas EC2 instâncias), você pode fazer o seguinte:

1. Marque o recurso com a tag de aplicativo exclusiva do AppRegistry aplicativo:

   Chave da tag: `awsApplication`

   Valor: `arn:aws:resource-groups:`*region*`:`*account-id*`:group/`*application-name*`/`*application-id*`>`

2. Execute a associate-resourcecomando .

Além disso, para cada AppRegistry aplicativo, um grupo de recursos é criado. O Grupo de recursos contém as seguintes tags.

Chave de tag	Valor
`EnableAWSServiceCatalogAppRegistry`	`TRUE`
`aws:servicecatalog:applicationName`	*`application-name`*
`aws:servicecatalog:applicationId`	*`application-id`*
`aws:servicecatalog:applicationArn`	`arn:aws:servicecatalog:` *`region`*`:`*`account-id`*` :/applications/` *`application-id`*

Executando tarefas após o desapego

A tabela a seguir fornece informações sobre como realizar tarefas após a separação:

Tarefa	Descrição
Executando eventos do ciclo de vida	Depois de executar o `detach` comando e se você selecionou a opção, o script cria 5 documentos de automação que correspondem aos 5 eventos do OpsWorks ciclo de vida. O nome de cada documento de automação segue este formato:*`layer-id`*`_`*`lifecycle-event`*`_automation_document` . Para simular o OpsWorks comportamento no Systems Manager, você precisará acionar manualmente as execuções de automação ao provisionar, encerrar EC2 instâncias ou implantar/remover receitas.

Tarefa	Descrição
Atualizando o JSON personalizado	O JSON personalizado para a pilha e a camada é armazenado em um bucket do S3 especificado durante a separação ou, alternati vamente, em um novo bucket do S3 que é criado. Os nomes dos arquivos armazenados para os arquivos JSON são os seguintes: • `layercustomjson.json` • `stackcustomjson.json`
Alterando sua lista de execução para eventos do ciclo de vida	A lista de execução para cada evento do ciclo de vida é definida no documento de automação correspondente. Para alterar a lista de execução, encontre os documento s de automação no AppRegistry aplicativo e modifique o `RunList` parâmetro. O processo de atualização de receitas e livros de receitas permanece inalterado porque`AWS-ApplyChefRecipes` , acionado pelos documentos de automação, suporta a mesma fonte de. OpsWorks

Tarefa	Descrição
Gerenciando a recuperação automática/ escalonamento automático	Quando você desanexa uma instância, o OpsWorks Agente é desinstalado. Sem o agente, OpsWorks não é possível reparar ou substituir automaticamente instâncias não íntegras, nem escalar automaticamente sua frota. Para continuar com o escalonamento automático e a substituição de instâncias com falha, crie um grupo Amazon EC2 Auto Scaling. O grupo lançará novas instâncias para manter a capacidade desejada quando a Amazon EC2 detectar instâncias insalubres que precisam ser substituídas.
Gerenciando o Load Balancer	Se sua camada usa um Classic Load Balancer, o `detach` comando o desanexará antes de cancelar o registro das instâncias. Isso é feito para garantir que todas as associações de instâncias do ELB permaneçam preservad as na Amazon EC2 durante todo o processo de separação, resultando em zero tempo de inatividade. Depois que o processo for concluído, você poderá gerenciar seu ELB em EC2.

Tarefa	Descrição
Como estabelecer conexão com suas instâncias	Quando você executa o `detach` comando `handle-prerequisites` or, duas verificações ocorrem: • A versão do agente SSM e as permissões • Chaves SSH Os comandos também oferecem a opção de atualizar o Agente SSM e adicionar as permissões necessárias para que você possa se conectar às instâncias usando o Gerenciador de Sessões. Se existirem chaves SSH, você também tem a opção de usar SSH na instância.

Usando a guia Systems Manager Application Manager Instances

Após a separação, você poderá visualizar e gerenciar suas instâncias na guia Instâncias do Application Manager.

A guia Instâncias fornece informações agregadas sobre as EC2 instâncias de um aplicativo, como status, estado de saúde e status do último comando. Usando essa guia, você pode visualizar informações detalhadas sobre instâncias individuais, como histórico de comandos, estados de alarme, integridade do agente do Systems Manager e muito mais. A guia Instâncias também fornece uma variedade de ações, como a capacidade de aplicar receitas do Chef, iniciar ou interromper uma instância ou adicionar ou remover uma instância de um grupo do Auto Scaling.

Começando a usar o AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks fornece um rico conjunto de componentes personalizáveis que você pode misturar e combinar para criar uma pilha que satisfaça seus propósitos específicos. O desafio para os novos usuários é entender como montar esses componentes em uma pilha e como gerenciá-la com eficácia. Veja como você pode começar.

Se você deseja...	Conclua esta demonstração:
Criar uma pilha de exemplo o mais rápido possível	Conceitos básicos: exemplo
Fazer experiências com uma pilha baseada no Linux	Conceitos básicos: Linux
Fazer experiências com uma pilha baseada no Windows	Conceitos básicos: Windows
Saiba como criar seus próprios livros de receitas do Chef	Conceitos básicos: livros de receitas

Se você tem recursos de computação existentes, como EC2 instâncias da Amazon ou até mesmo instâncias locais que estão sendo executadas em seu próprio hardware, você pode incorporá-los em uma pilha, junto com as instâncias que você criou com o Stacks. AWS OpsWorks Em seguida, você pode usar o AWS OpsWorks Stacks para gerenciar todas as instâncias relacionadas como um grupo, independentemente de como elas foram criadas.

Suporte regional

Você pode acessar o AWS OpsWorks Stacks globalmente; você também pode criar e gerenciar instâncias globalmente. Os usuários podem configurar instâncias do AWS OpsWorks Stacks para serem lançadas em qualquer AWS região, exceto AWS GovCloud (Oeste dos EUA) e na região da China (Pequim). Para trabalhar com o AWS OpsWorks Stacks, as instâncias precisam ser capazes de se conectar a um dos seguintes endpoints da API do serviço de instâncias do AWS OpsWorks Stacks.

Os recursos podem ser gerenciados apenas na região em que são criados. Os recursos que são criados em um único endpoint regional não estão disponíveis, nem podem ser clonados para outro endpoint regional. Você pode iniciar instâncias em qualquer uma das regiões a seguir.

- Região Leste dos EUA (Ohio)
- Região Leste dos EUA (Norte da Virgínia)
- Região Oeste dos EUA (Oregon)
- Região Oeste dos EUA (Norte da Califórnia).
- Região do Canadá (Central) (somente API, não disponível para pilhas criadas no AWS Management Console.)
- Região Ásia-Pacífico (Mumbai)
- Região Ásia-Pacífico (Singapura)
- Região Ásia-Pacífico (Sydney)
- Região Ásia-Pacífico (Tóquio)
- Região Ásia-Pacífico (Seul)
- Região Europa (Frankfurt)
- Região Europa (Irlanda)
- Região Europa (Londres)
- Região Europa (Paris)
- Região América do Sul (São Paulo)

Conceitos básicos de uma pilha de exemplo

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este passo a passo mostra como usar o AWS OpsWorks Stacks para criar rapidamente um ambiente de aplicativo Node.js de amostra com apenas alguns cliques do mouse e sem escrever código. Ao

terminar, você tem uma instância do Amazon Elastic Compute Cloud (Amazon EC2) executando o Chef 12, um servidor HTTP Node.js e um aplicativo web que você pode usar para interagir com o Twitter e deixar comentários em uma página da web.

> **Note**
>
> Como a conclusão desta demonstração cria automaticamente uma instância com um tipo de c3.large, você não poderá usar essa demonstração ou a ferramenta de criação Pilha de exemplo no AWS OpsWorks Stacks no Nível gratuito da AWS. Embora o uso da ferramenta de criação Sample Stack em uma VPC crie uma instância t2.medium VPCs , elas não estão disponíveis atualmente no nível gratuito.AWS

Etapa 1: concluir os pré-requisitos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você deve concluir as seguintes etapas de configuração antes de iniciar a demonstração. Essas etapas de configuração incluem se inscrever em uma AWS conta, criar um usuário administrativo e atribuir permissões de acesso ao AWS OpsWorks Stacks.

Tópicos

- Inscreva-se para um Conta da AWS
- Criar um usuário com acesso administrativo
- Atribuir permissões de acesso ao serviço

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra a https://portal.aws.amazon.com/billing/inscrição.
2. Siga as instruções online.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, você pode visualizar a atividade atual da sua conta e gerenciar sua conta acessando https://aws.amazon.com/e escolhendo Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, insira a senha.

   Para obter ajuda ao fazer login usando o usuário-raiz, consulte Fazer login como usuário-raiz no Guia do usuário do Início de Sessão da AWS .
2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

   Para obter instruções, consulte Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz (console) no Guia do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilita o Centro de Identidade do IAM.

Para obter instruções, consulte Habilitar o AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center .

2. No Centro de Identidade do IAM, conceda o acesso administrativo a um usuário.

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com o seu usuário do Centro de Identidade do IAM, use o URL de login enviado ao seu endereço de e-mail quando o usuário do Centro de Identidade do IAM foi criado.

  Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

   Para obter instruções, consulte Criar um conjunto de permissões no Guia do usuário do AWS IAM Identity Center .

2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Adicionar grupos no Guia do usuário do AWS IAM Identity Center .

Atribuir permissões de acesso ao serviço

Ative o acesso ao serviço AWS OpsWorks Stacks (e aos serviços relacionados dos quais o AWS OpsWorks Stacks depende) adicionando as `AmazonS3FullAccess` permissões `AWSOpsWorks_FullAccess` e à sua função ou usuário.

Para obter mais informações sobre adicionar permissões, consulte Adicionar permissões de identidade do IAM (console).

Agora você concluiu todas as etapas de configuração e pode iniciar esta demonstração.

Etapa 2: criar uma stack

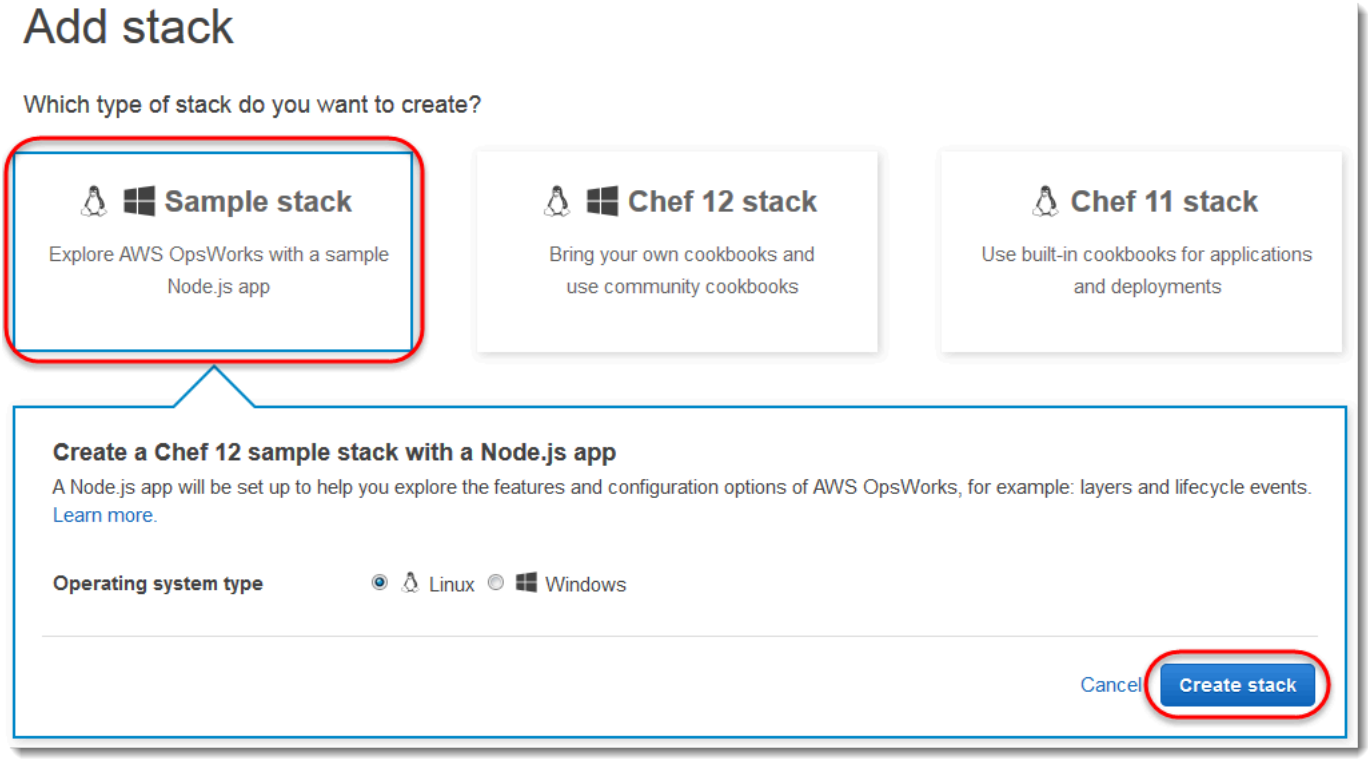
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Nesta etapa, você usa o console AWS OpsWorks Stacks para criar uma pilha. Uma pilha é uma coleção de instâncias (como EC2 instâncias da Amazon) e AWS recursos relacionados que têm um propósito comum e que você deseja gerenciar juntos. (Para ter mais informações, consulte Pilhas.) Há apenas uma instância para esta demonstração.

Antes de iniciar esta etapa, conclua os pré-requisitos.

Para criar a pilha

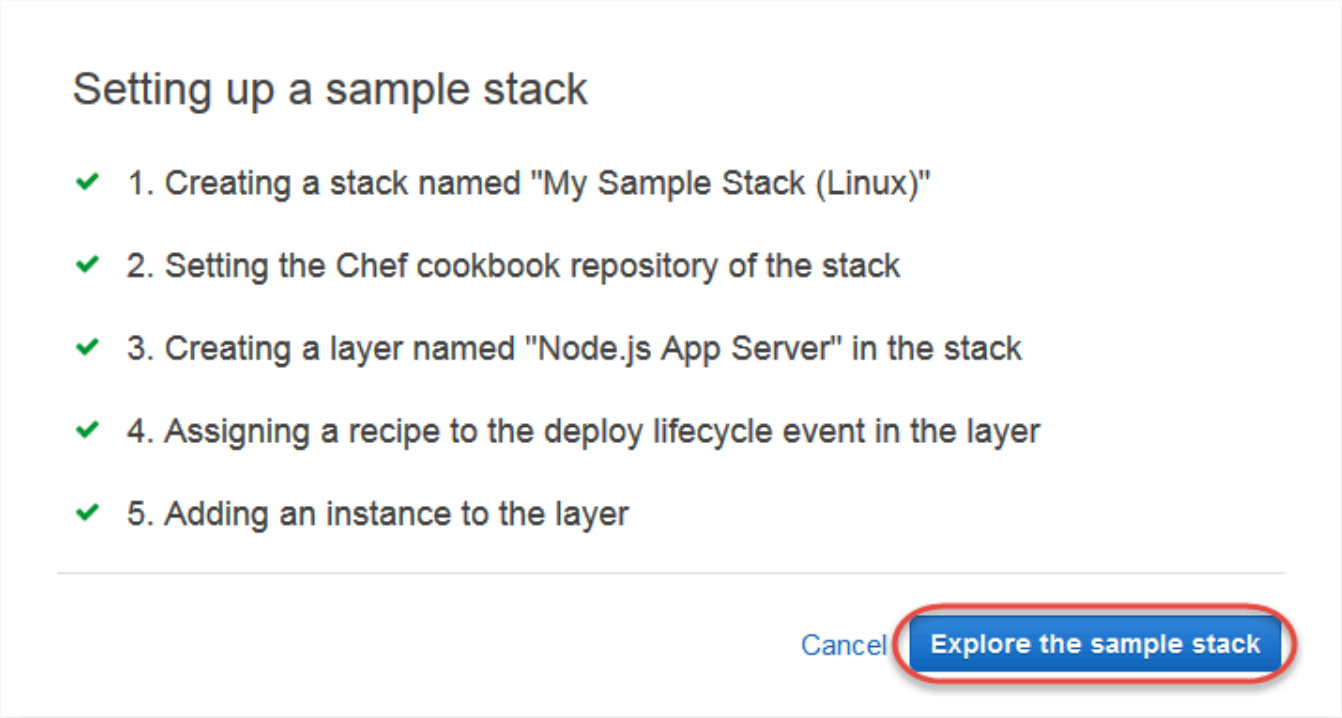
1. Faça login no AWS Management Console e abra o AWS OpsWorks console em https://console.aws.amazon.com/opsworks/.
2. Execute a ação que melhor se aplica:
   - Se a página Bem-vindo às AWS OpsWorks pilhas for exibida, escolha Adicionar sua primeira pilha ou Adicionar sua primeira pilha de AWS OpsWorks pilhas (ambas as opções fazem a mesma coisa). A página Add stack será exibida.
   - Se a página OpsWorks Painel for exibida, escolha Adicionar pilha. A página Add stack será exibida.
3. Com a página Add stack exibida, escolha Sample stack, caso ela ainda não tenha sido escolhida para você.
4. Com Linux selecionado para Operating system type, escolha a opção Create stack:



5. AWS OpsWorks Stacks cria uma pilha chamada My Sample Stack (Linux). AWS OpsWorks O Stacks também adiciona todos os componentes necessários para implantar o aplicativo na pilha:

   - Uma camada, que é um esquema para um conjunto de instâncias. Ela especifica as configurações, os recursos, os pacotes instalados e os grupos de segurança da instância. (Para ter mais informações, consulte Camadas.) A camada é chamada Node.js App Server.
   - Uma instância, que nesse caso é uma EC2 instância do Amazon Linux 2. (Para obter mais informações sobre as instâncias, consulte Instâncias). O nome de host da instância é nodejs-server1.
   - Um aplicativo, que é o código a ser executado na instância. (Para obter mais informações sobre aplicativos, consulte Apps). O aplicativo é chamado Node.js Sample App.

6. Depois que o AWS OpsWorks Stacks criar a pilha, escolha Explorar a pilha de amostra para exibir a página Minha pilha de amostras (Linux) (se você concluir essa explicação passo a passo várias vezes, Minha pilha de amostras (Linux) poderá ter um número sequencial depois dela, como 2 ou 3):



Na próxima etapa, você iniciará a instância e implantará o aplicativo na instância.

Etapa 3: Iniciar a instância e implantar o aplicativo

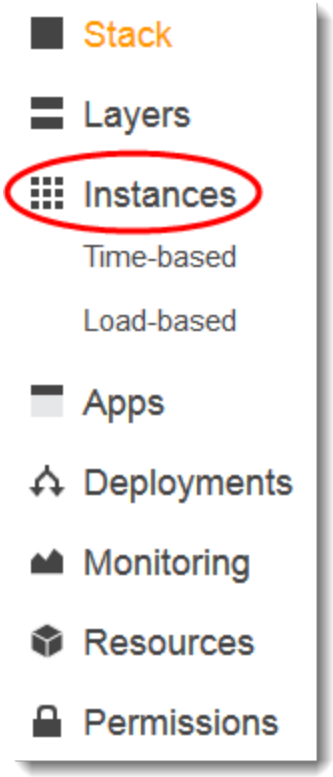
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Agora que você tem uma instância e um aplicativo, inicie a instância e implante o aplicativo na instância.

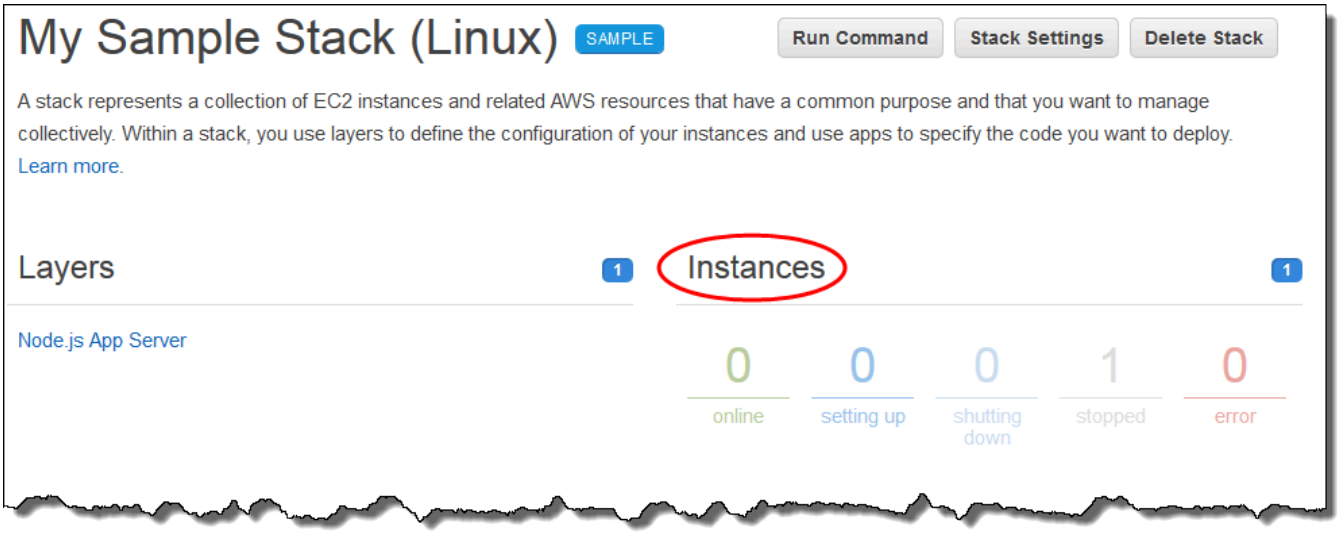
Para iniciar a instância e implantar o aplicativo

1. Execute um destes procedimentos:

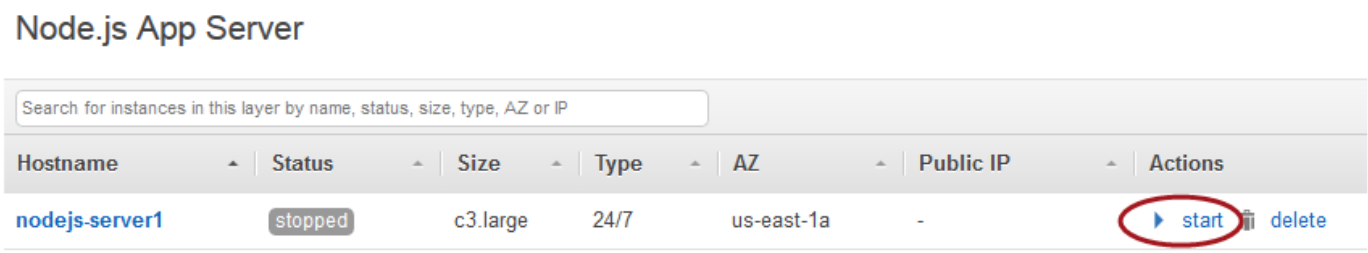
- No painel de navegação de serviços, escolha Instances:



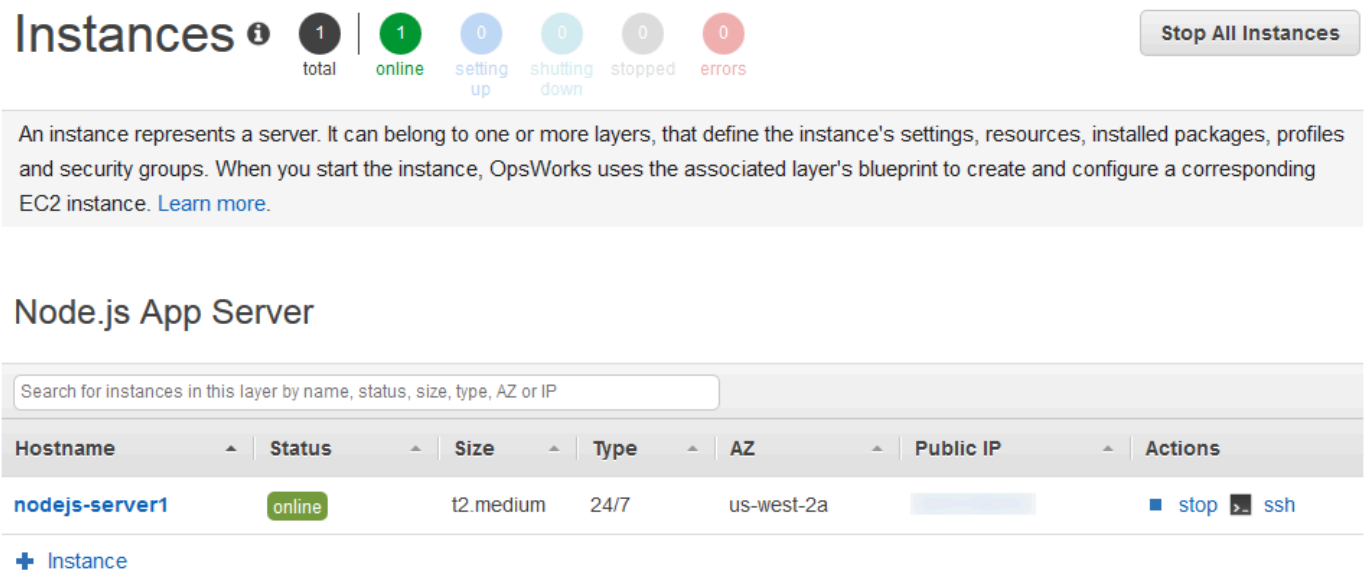
- Na página My Sample Stack (Linux), escolha Instances:



2. Na página Instances, para Node.js App Server, nodejs-server1, escolha start:



3. Não prossiga até que o círculo online fique verde brilhante. (Se você vir uma mensagem de falha, consulte Guia de depuração e solução de problemas).
4. À medida que a instância está sendo configurada, o AWS OpsWorks Stacks implanta o aplicativo na instância.
5. Os resultados devem se parecer com a captura de tela a seguir antes que você continue (se receber uma mensagem de falha, consulte Guia de depuração e solução de problemas.):



Você agora tem uma instância com um aplicativo que foi implantado na instância.

Na próxima etapa, você testará o aplicativo na instância.

Etapa 4: Testar o aplicativo implantado na instância

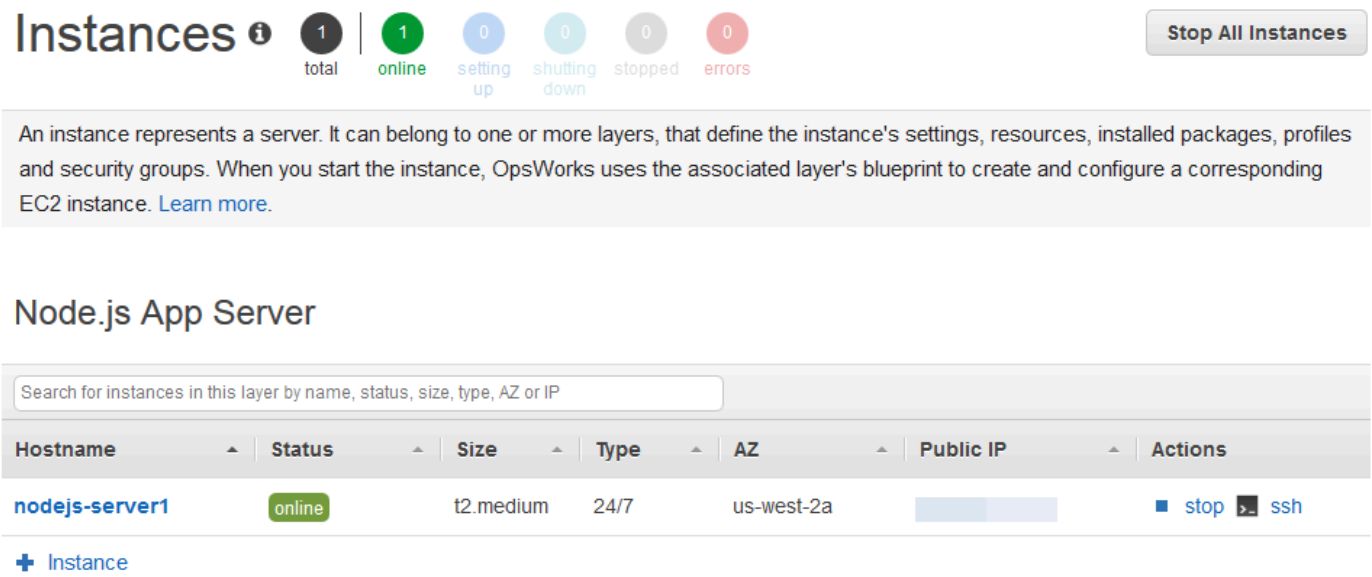
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Testar os resultados da implantação do aplicativo na instância.

Para testar a implantação na instância

1. Com a página Instances sendo exibida a partir da etapa anterior, para Node.js App Server, nodejs-server1, Public IP, escolha o endereço IP.



2. Na página da Web de congratulações, na caixa de texto Leave a comment, digite um comentário e escolha Send para testar o aplicativo. O aplicativo adiciona seu comentário à página da Web. Deixe comentários e escolha Send sempre que desejar.

3. Se você tiver uma conta do Twitter, escolha Tweetar ou Seguir @ AWSOps Works e siga as instruções na tela para tweetar sobre o aplicativo ou seguir @ AWSOps Works.

Agora, você testou com sucesso o aplicativo implantado na instância.

Nas etapas restantes, você pode usar o console AWS OpsWorks Stacks para explorar as configurações da pilha e seus componentes. Na próxima etapa, inicie sua exploração ao examinar as configurações da pilha.

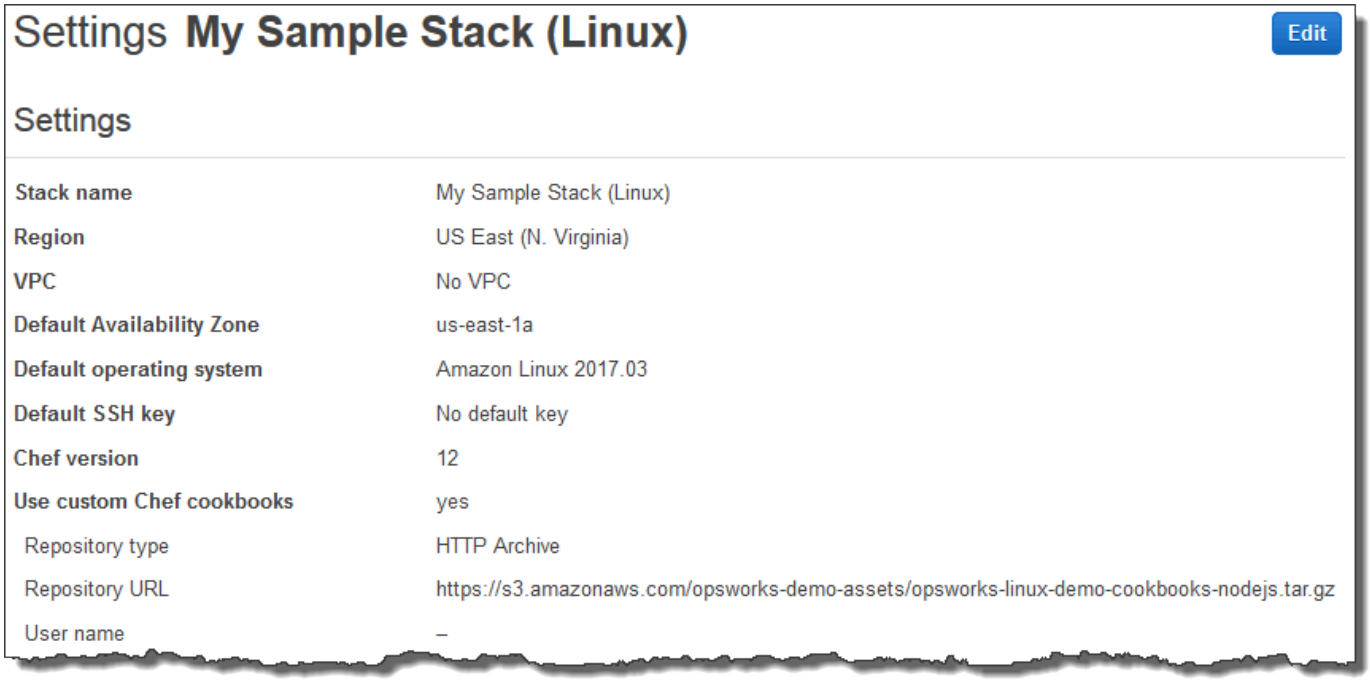
Etapa 5: Explorar as configurações da pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Examine como AWS OpsWorks as pilhas configuram a pilha.

Para exibir as configurações da pilha

1. Na barra de navegação de serviços, escolha Stack. A página My Sample Stack (Linux) será exibida.
2. Escolha Stack Settings. A página Settings My Sample Stack (Linux) será exibida:



Para saber mais sobre muitas das configurações, escolha Edit e passe o mouse sobre cada uma das configurações. (Nem todas as configurações têm descrições na tela). Para ter mais informações sobre essas configurações, consulte Criar uma nova pilha.

Para explorar o livro de receitas do Chef usado neste passo a passo, abra o repositório opsworks-linux-demo-cookbooks-nodejs em. GitHub

Na próxima etapa, você poderá explorar as configurações da camada.

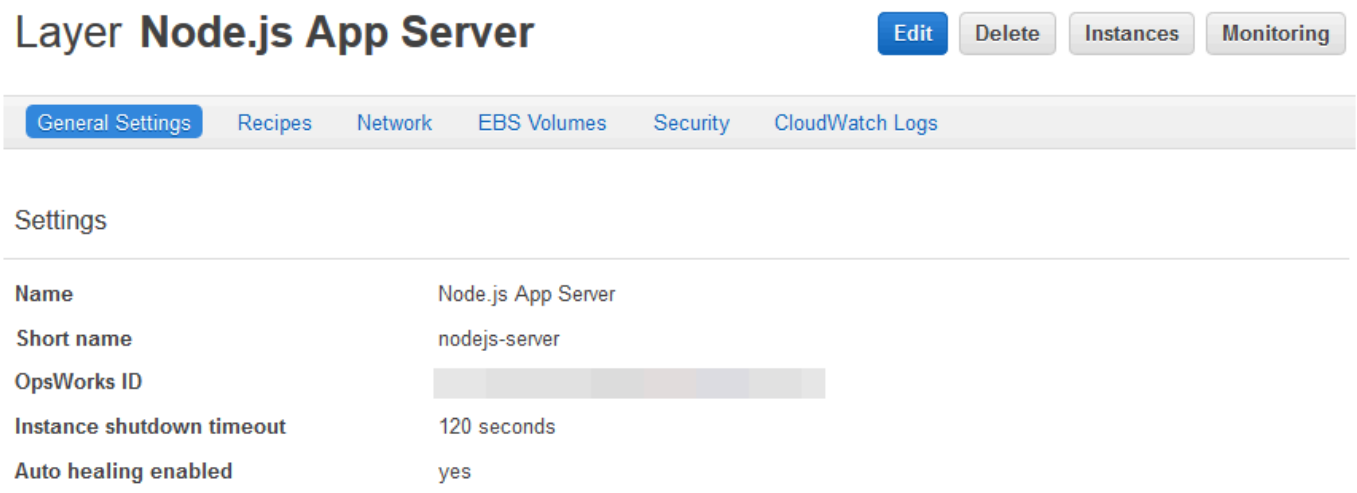
Etapa 6: Explorar as configurações da camada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Examine como AWS OpsWorks as pilhas configuram a camada.

Para exibir as configurações da camada

1. No painel de navegação de serviços, escolha Layers. A página Layers será exibida.
2. Escolha Node.js App Server. A página Layer Node.js App Server será exibida. Para ver as configurações da camada, escolha General Settings, Recipes, Network, EBS Volumes e Security:



Para saber mais sobre muitas das configurações, escolha Edit e passe o mouse sobre cada uma das configurações. (Nem todas as configurações têm descrições na tela). Para ter mais informações sobre essas configurações, consulte Como editar a configuração de uma OpsWorks camada.

Na próxima etapa, você poderá explorar as configurações e os logs da instância.

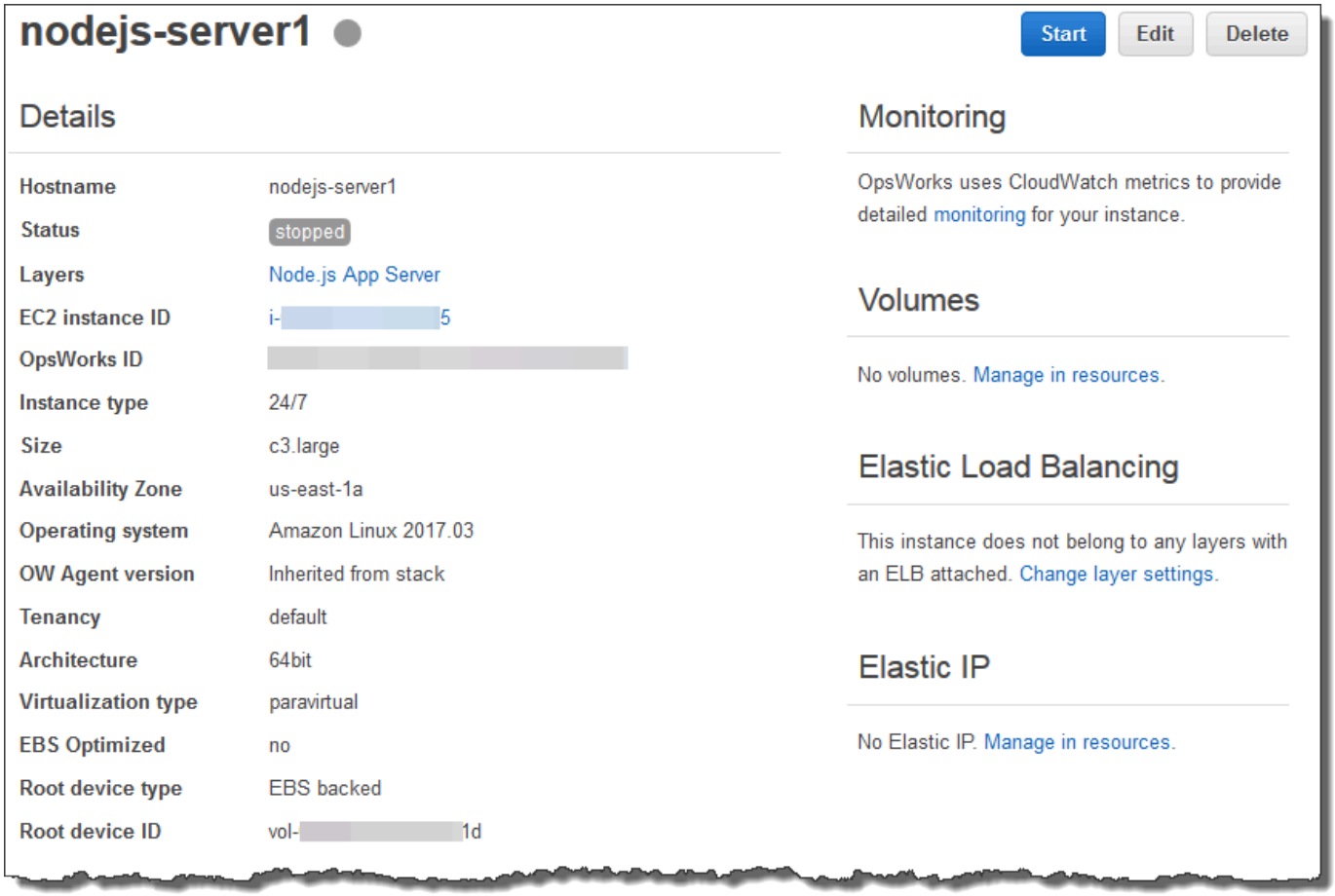
Etapa 7: Explorar as configurações e os logs da instância

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

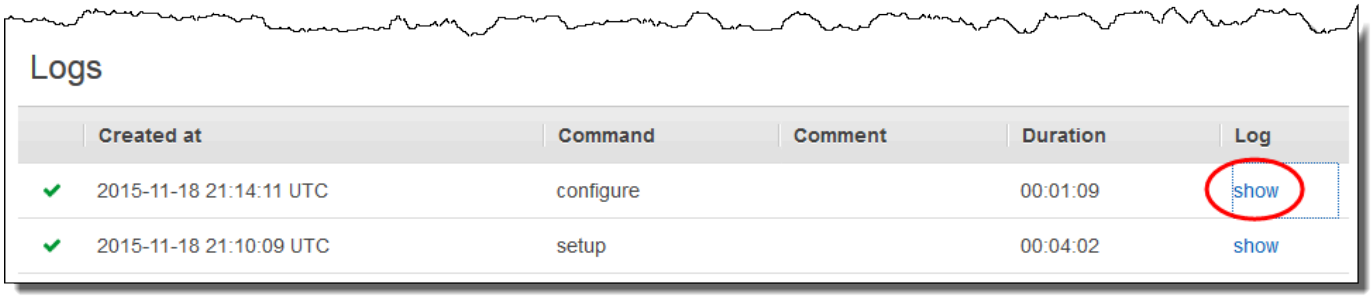
Examine as configurações que o AWS OpsWorks Stacks usou para iniciar a instância. Você também pode examinar os registros de instância criados pelo AWS OpsWorks Stacks.

Para exibir as configurações e os logs da instância

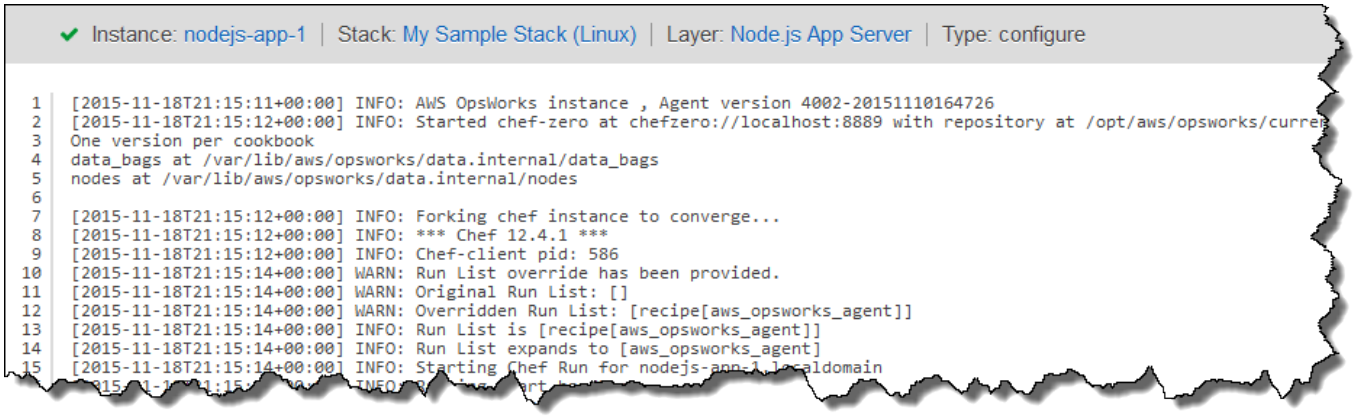
1. No painel de navegação de serviços, escolha Instances. A página Instances é exibida.
2. Para Node.js App Server, escolha nodejs-server1. A página de propriedades da instância é mostrada.



3. Para explorar os logs da instância, na seção Logs, para Log, escolha show.



4. AWS OpsWorks O Stacks exibe o registro em uma guia separada do navegador.



Para saber mais sobre o que algumas das configurações da instância representam, volte para a página nodejs-server1, escolha Stop e quando vir a mensagem de confirmação, escolha Stop. Escolha Edit depois que o Status mudar de stopping para stopped e passe o mouse sobre cada uma das configurações. (Nem todas as configurações têm descrições na tela). Para ter mais informações sobre essas configurações, consulte Adicionar uma instância a uma camada.

Quando terminar de examinar as configurações, escolha Start para reiniciar a instância e aguarde até o Status mudar para online. Caso contrário, você não poderá testar o aplicativo mais tarde, porque a instância permanecerá interrompida.

> **Note**
>
> Se quiser fazer login na instância para explorá-la ainda mais, primeiro forneça às AWS OpsWorks pilhas informações sobre sua chave SSH pública (que você pode criar com ferramentas como ssh-keygen ou PuTTYgen) e, em seguida, definir permissões na pilha My Sample Stack (Linux) para permitir que o usuário faça login na instância. Para obter instruções, consulte Registro de uma chave SSH pública de um usuário e Login com SSH.

Na próxima etapa, explore as configurações do aplicativo.

Etapa 8: Explorar os configurações do aplicativo

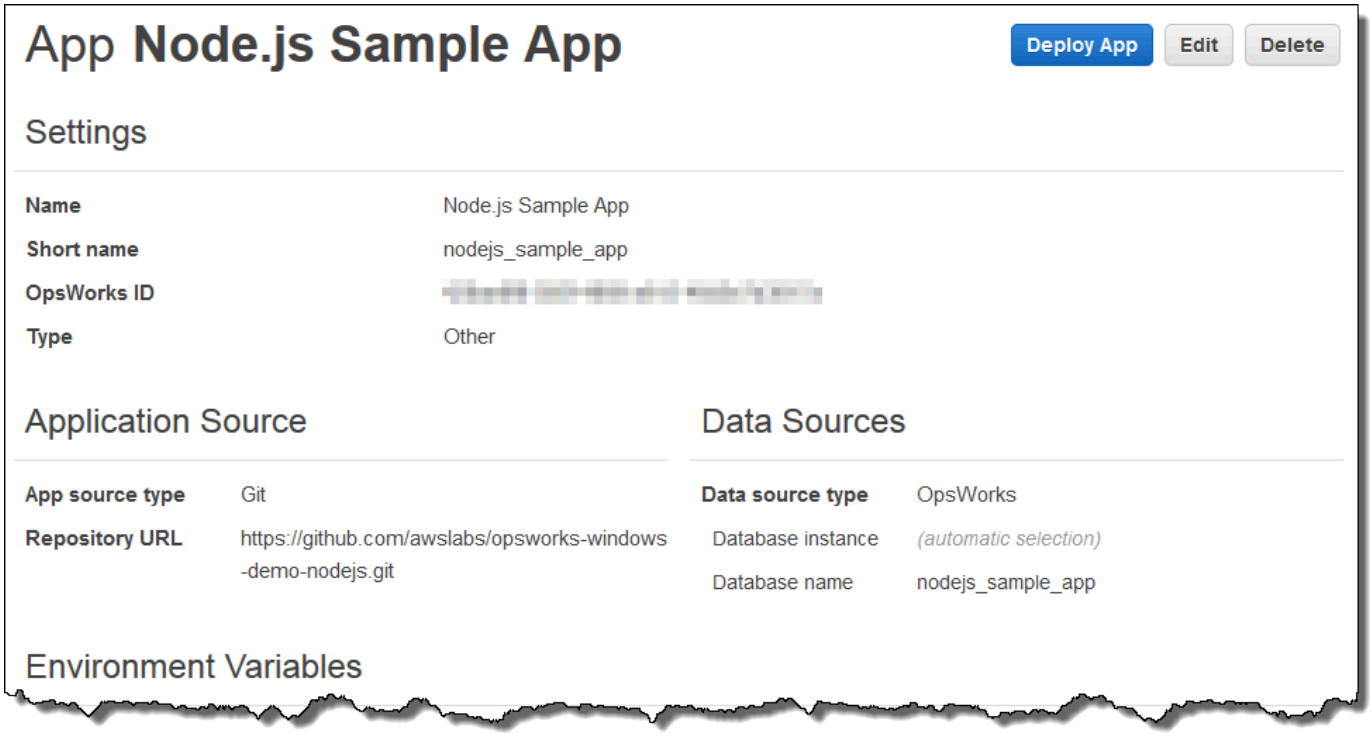
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Examine as configurações que o AWS OpsWorks Stacks usou para o aplicativo.

Para exibir as configurações do aplicativo

1. No painel de navegação do serviço, selecione Apps. A página Aplicativos será exibida.
2. Escolha Node.js Sample App. A página App Node.js Sample App será exibida:



Para saber mais sobre o que algumas das configurações representam, escolha Edit e passe o mouse sobre cada uma das configurações. (Nem todas as configurações têm descrições na tela). Para obter mais informações sobre essas configurações, consulte Adição de aplicativos.

Na próxima etapa, você poderá explorar os relatórios de monitoramento de camada.

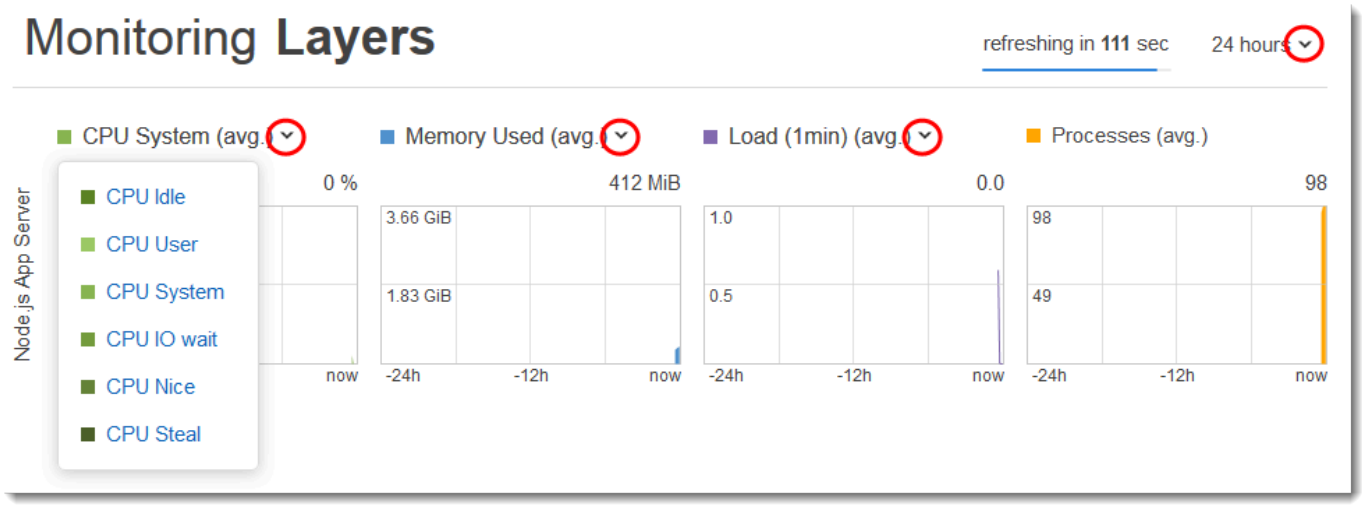
Etapa 9: Explorar os relatórios de monitoramento de camada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Examine os relatórios que o AWS OpsWorks Stacks gera sobre o desempenho computacional da camada.

Para exibir os relatórios de monitoramento de camada

1. No painel de navegação de serviços, escolha Monitoring. A página Monitoring Layers será exibida.
2. Para explorar exibições adicionais, escolha a seta ao lado de CPU, Memory, Load e o tempo:



Para obter mais informações sobre esses e outros relatórios, consulte Usando a Amazon CloudWatch e Monitoramento.

Na próxima etapa, você poderá explorar configurações de pilha adicionais.

Etapa 10: Explorar configurações adicionais de pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Nesta etapa, você pode examinar as configurações adicionais de pilha.

AWS OpsWorks As pilhas não executaram implantações separadas, não provisionaram recursos adicionais e não ajustaram permissões adicionais como parte dessa pilha, portanto, não há muito interesse nas páginas Implantações e Comandos, Recursos e Permissões. Se mesmo assim você quiser ver essas configurações, escolha Deployments, Resources e Permissions no painel de navegação de serviço, respectivamente. Se você quiser obter mais informações sobre o que essas páginas representam, consulte Implementação de aplicativos, Gerenciamento de recursos e Gerenciamento de permissões de usuário.

Na próxima etapa, você pode limpar os AWS recursos que você usou para este passo a passo. Essa próxima etapa é opcional. Talvez você queira continuar usando esses AWS recursos à medida que continua aprendendo mais sobre AWS OpsWorks as pilhas. No entanto, manter esses AWS recursos disponíveis pode resultar em algumas cobranças contínuas em sua AWS conta. Se quiser manter esses AWS recursos disponíveis para uso posterior, você concluiu este passo a passo e pode pular para. Próximas etapas

Etapa 11 (opcional): limpeza

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para evitar cobranças adicionais em sua AWS conta, você pode excluir o aplicativo e os AWS recursos que foram usados para esse passo a passo, incluindo a instância e a AWS OpsWorks pilha Stacks. (Para obter mais informações, consulte AWS OpsWorks Preços.) No entanto, talvez você queira continuar usando esses AWS recursos à medida que continua aprendendo mais sobre AWS OpsWorks as pilhas. Se quiser manter esses AWS recursos disponíveis, você concluiu este passo a passo e pode pular para. Próximas etapas

O conteúdo armazenado nos recursos que você criou para essa demonstração pode conter informações de identificação pessoal. Se você não quer mais que essas informações sejam armazenadas pela AWS, siga as etapas neste tópico.

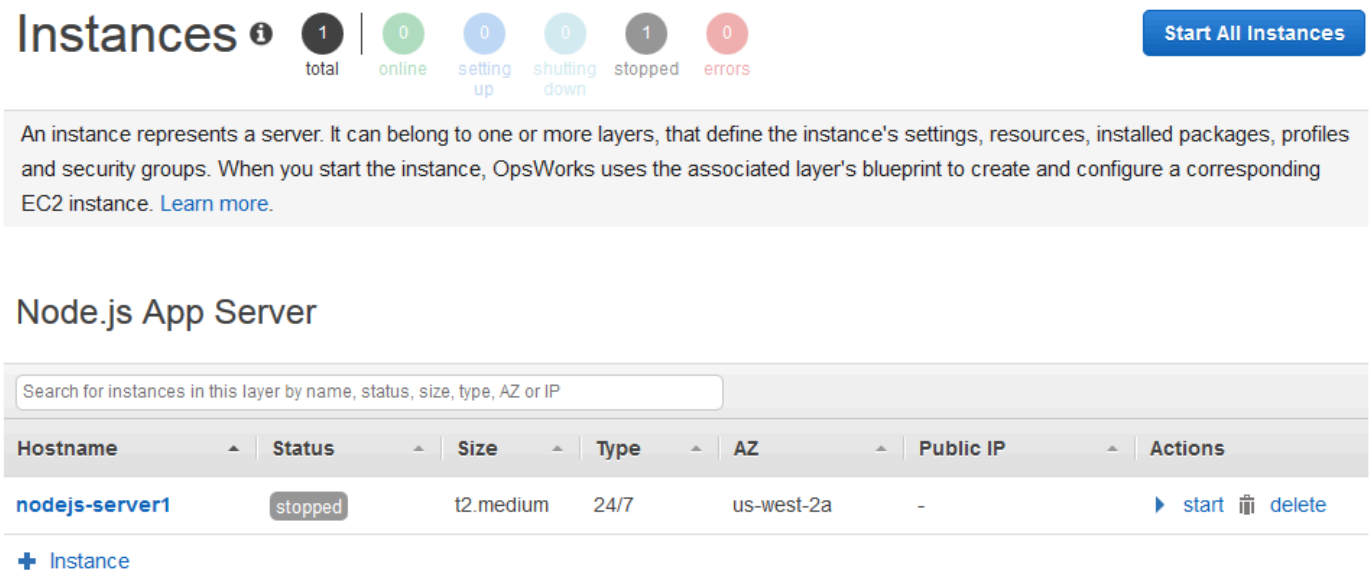
Para excluir o aplicativo da pilha

1. No painel de navegação do serviço, selecione Apps. A página Apps será exibida.
2. Para Node.js Sample App, Actions, escolha delete. Quando a mensagem de confirmação for exibida, escolha Delete. Quando o aplicativo for excluído, você verá a mensagem No apps.

Para deletar a instância da pilha

1. No painel de navegação de serviços, escolha Instances. A página Instances é exibida.
2. Para Node.js App Server, nodejs-server1, Actions, escolha stop. Quando a mensagem de confirmação for exibida, escolha Stop.

   Esse processo pode levar alguns minutos. Quando AWS OpsWorks as pilhas terminarem, os seguintes resultados serão mostrados.



3. Em Actions, selecione delete. Quando a mensagem de confirmação for exibida, escolha Delete. A instância é excluída e a mensagem No instances é exibida.

Para excluir a pilha

1. No painel de navegação de serviços, escolha Stack. A página My Sample Stack (Linux) é exibida.
2. Escolha Delete Stack. Quando a mensagem de confirmação for exibida, escolha Delete. A pilha é excluída e a página do OpsWorksPainel é exibida.

Opcionalmente, você pode excluir o usuário e o par de EC2 chaves da Amazon que você usou para este passo a passo, se não quiser reutilizá-los para acessar outros AWS serviços e instâncias. EC2 Para obter instruções, consulte Excluir um usuário do IAM e pares de EC2 chaves e instâncias Linux da Amazon.

Você acabou de concluir essa apresentação. Para obter mais informações, consulte Próximas etapas.

Próximas etapas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Agora que você concluiu este passo a passo, você pode aprender mais sobre como usar AWS OpsWorks o Stacks:

- Pratique recriar manualmente essa pilha sozinho usando AWS OpsWorks pilhas. Consulte Conceitos básicos: Linux.
- Explore o livro de receitas e o aplicativo que AWS OpsWorks Stacks usou para este passo a passo. Consulte Aprendendo mais: Explore o livro de receitas usado nessa apresentação e Aprendendo mais: Explore o aplicativo usado nessa apresentação na demonstração Conceitos básicos: Linux complementar.
- Pratique o uso AWS OpsWorks do Stacks com instâncias do Windows. Consulte Conceitos básicos: Windows.
- Saiba mais sobre as pilhas ao aprender a Criar uma nova pilha.
- Saiba mais sobre camadas ao Como editar a configuração de uma OpsWorks camada.
- Saiba mais sobre as instâncias ao Adicionar uma instância a uma camada.
- Saiba mais sobre aplicativos ao Implementação de aplicativos.
- Saiba mais sobre Livros de receitas e receitas.
- Crie seus próprios livros de receitas. Consulte Conceitos básicos: livros de receitas.
- Saiba mais sobre o controle de acesso às pilhas com Segurança e permissões.

Conceitos básicos de Linux Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Neste passo a passo, você aprenderá a usar o AWS OpsWorks Stacks para criar um ambiente de aplicativo Node.js. Ao terminar, você terá uma instância do Amazon Elastic Compute Cloud (Amazon EC2) executando o Chef 12, um servidor HTTP Node.js e um aplicativo web que você pode usar para interagir com o Twitter e deixar comentários em uma página da web.

O Chef é uma estrutura de terceiros para configurar e manter servidores, como EC2 instâncias, e como os aplicativos são implantados e mantidos nesses servidores. Se você não estiver familiarizado com o Chef, depois de concluir este passo a passo, recomendamos que você aprenda mais sobre o Chef para que possa aproveitar ao máximo tudo o que o AWS OpsWorks Stacks tem a oferecer. (Para obter mais informações, consulte o site Conheça o Chef.)

AWS OpsWorks O Stacks oferece suporte a quatro distribuições Linux: Amazon Linux, Ubuntu Server, CentOS e Red Hat Enterprise Linux. Para este passo a passo, usamos o Ubuntu Server. AWS OpsWorks O Stacks também funciona com o Windows Server. Embora tenhamos um passo a passo equivalente para pilhas do Windows Server, recomendamos que você conclua esse passo primeiro para aprender conceitos básicos sobre AWS OpsWorks Stacks e Chef que não se repetem lá. Depois de concluir essa explicação, consulte a explicação de Conceitos básicos: Windows.

Tópicos

- Etapa 1: concluir os pré-requisitos
- Etapa 2: criar uma stack
- Etapa 3: Adicionar uma camada à pilha
- Etapa 4: Especificar o aplicativo para implantar à instância
- Etapa 5: Executar uma instância
- Etapa 6: Implantar o aplicativo à instância
- Etapa 7: Testar o aplicativo implantado na instância
- Etapa 8 (opcional): limpeza
- Próximas etapas
- Aprendendo mais: Explore o livro de receitas usado nessa apresentação
- Aprendendo mais: Explore o aplicativo usado nessa apresentação

Etapa 1: concluir os pré-requisitos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Conclua as seguintes etapas de configuração antes de iniciar a demonstração. Essas etapas de configuração incluem se inscrever em uma AWS conta, criar um usuário administrativo e atribuir permissões de acesso ao AWS OpsWorks Stacks.

Se já tiver concluído a demonstração Conceitos básicos: Amostra, então você atendeu aos pré-requisitos dessa demonstração e pode pular diretamente para Etapa 2: criar uma stack.

Tópicos

- Inscreva-se para um Conta da AWS
- Criar um usuário com acesso administrativo
- Atribuir permissões de acesso ao serviço

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra a https://portal.aws.amazon.com/billing/inscrição.
2. Siga as instruções online.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática

recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, você pode visualizar a atividade atual da sua conta e gerenciar sua conta acessando https://aws.amazon.com/e escolhendo Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, insira a senha.

   Para obter ajuda ao fazer login usando o usuário-raiz, consulte Fazer login como usuário-raiz no Guia do usuário do Início de Sessão da AWS .

2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

   Para obter instruções, consulte Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz (console) no Guia do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilita o Centro de Identidade do IAM.

   Para obter instruções, consulte Habilitar o AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center .

2. No Centro de Identidade do IAM, conceda o acesso administrativo a um usuário.

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com o seu usuário do Centro de Identidade do IAM, use o URL de login enviado ao seu endereço de e-mail quando o usuário do Centro de Identidade do IAM foi criado.

  Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

   Para obter instruções, consulte Criar um conjunto de permissões no Guia do usuário do AWS IAM Identity Center .
2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Adicionar grupos no Guia do usuário do AWS IAM Identity Center .

Atribuir permissões de acesso ao serviço

Ative o acesso ao serviço AWS OpsWorks Stacks (e aos serviços relacionados dos quais o AWS OpsWorks Stacks depende) adicionando as `AmazonS3FullAccess` permissões `AWSOpsWorks_FullAccess` e à sua função ou usuário.

Para obter mais informações sobre adicionar permissões, consulte Adicionar permissões de identidade do IAM (console).

Agora você concluiu todas as etapas de configuração e pode iniciar esta demonstração.

Etapa 2: criar uma stack

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você usará o console AWS OpsWorks Stacks para criar uma pilha. Uma pilha é uma coleção de instâncias e AWS recursos relacionados que têm um propósito comum e que você deseja gerenciar juntos. (Para ter mais informações, consulte Pilhas.) Para essa explicação, há apenas uma instância.

Antes de iniciar, conclua os pré-requisitos, se ainda não o fez.

Para criar a pilha

1. Faça login no AWS Management Console e abra o AWS OpsWorks console em https://console.aws.amazon.com/opsworks/.
2. Execute a ação que melhor se aplica:
   - Se a página Bem-vindo às AWS OpsWorks pilhas for exibida, escolha Adicionar sua primeira pilha ou Adicionar sua primeira pilha de AWS OpsWorks pilhas (ambas as opções fazem a mesma coisa). A página Add stack será exibida.
   - Se a página OpsWorks Painel for exibida, escolha Adicionar pilha. A página Add stack será exibida.
3. Com a página Add stack exibida, escolha Chef 12 stack, caso ela ainda não tenha sido escolhida para você.
4. Na caixa Stack name (Nome da pilha), digite um nome, por exemplo **`MyLinuxDemoStack`**. (É possível inserir um nome diferente, mas substitua ele por `MyLinuxDemoStack` durante esta demonstração.)
5. Para Região, escolha Oeste dos EUA (Oregon).
6. Para VPC, realize um destes procedimentos:
   - Escolha um VPC se essa opção estiver disponível. (Para ter mais informações, consulte Execução de uma Stack em uma VPC.)
   - Caso contrário, escolha No VPC.
7. Para Default operating system (Sistema operacional padrão), selecione Linux e Ubuntu 18.04 LTS.
8. Para Use custom Chef cookbooks, escolha Yes.
9. Para Repository type, escolha Http Archive.

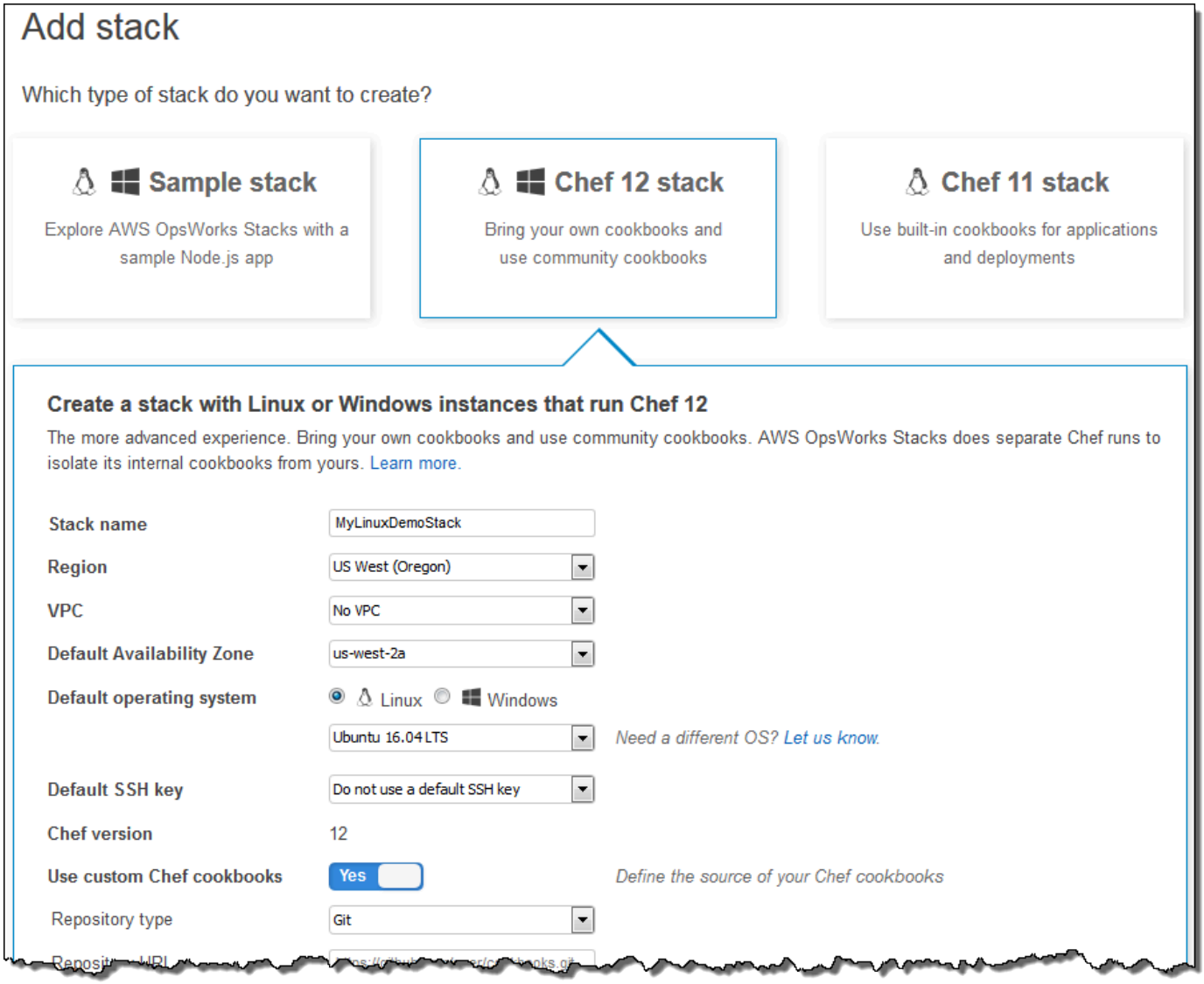
10. Para Repository URL (URL do repositório), digite **`https://s3.amazonaws.com/opsworks-demo-assets/opsworks-linux-demo-cookbooks-nodejs.tar.gz`**
11. Deixe os padrões para os seguintes:
    - Default Availability Zone (us-west-2a)
    - Default SSH key (Do not use a default SSH key)
    - User name (em branco)
    - Password (em branco)
    - Stack color (azul escuro)
12. Escolha Advanced (Avançado).
13. Para Perfil do IAM, realize uma das seguintes ações (para mais informações, consulte Permitindo que AWS OpsWorks as pilhas ajam em seu nome):
    - Escolha aws-opsworks-service-role se essa opção estiver disponível.
    - Se não aws-opsworks-service-roleestiver disponível, escolha Nova função do IAM.
14. Para Perfil de instância padrão do IAM, realize uma das seguintes ações (para mais informações, consulte Especificação de permissões para aplicativos executados em instâncias EC2):
    - Se aws-opsworks-ecduas funções estiverem disponíveis, escolha-as.
    - Se aws-opsworks-ec2 funções não estiverem disponíveis, escolha Novo perfil de instância do IAM.
15. Para API endpoint region, escolha o endpoint API regional com o qual deseja que pilha esteja associada. Se quiser que a pilha esteja na região Oeste dos EUA (Oregon) dentro da região de endpoint regional Leste dos EUA (N. da Virgínia), escolha us-east-1. Se quiser que a pilha esteja na região Oeste dos EUA (Oregon) e associada com o endpoint regional Oeste dos EUA (Oregon) , escolha us-west-2.

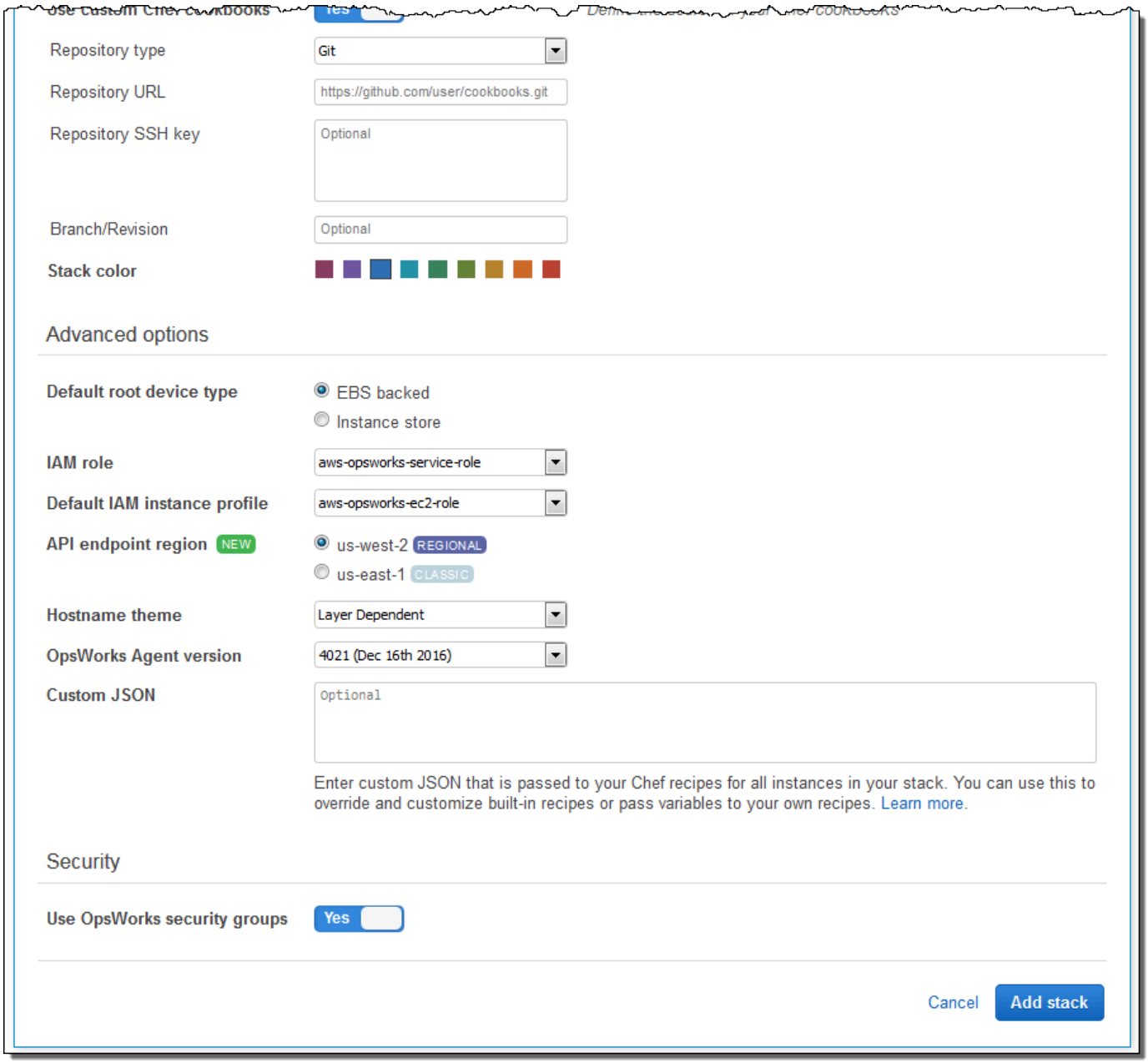
    > **Note**
    >
    > O endpoint regional do Leste dos EUA (Norte da Virgínia) inclui versões mais antigas Regiões da AWS para fins de compatibilidade com versões anteriores, mas é uma prática recomendada escolher o endpoint regional mais próximo de onde você gerencia. AWS Para obter mais informações, consulte Suporte regional.

16. Deixe os padrões para os seguintes:

- Tipo de dispositivo raiz padrão (Baseado em EBS)
- Tema do hostname (Depende do nível)
- OpsWorks Versão do agente (versão mais recente)
- Custom JSON (em branco)
- Use grupos OpsWorks de segurança (Sim)

17. Os resultados devem corresponder às seguintes capturas de tela, com exceção de VPC, perfil do IAM e perfil de instância padrão do IAM:





18. Escolha Adicionar pilha. AWS OpsWorks Stacks cria a pilha e exibe a MyLinuxDemoStackpágina.

Agora você tem uma pilha com as configurações corretas para essa explicação.

Na próxima etapa, você poderá adicionar uma camada à pilha.

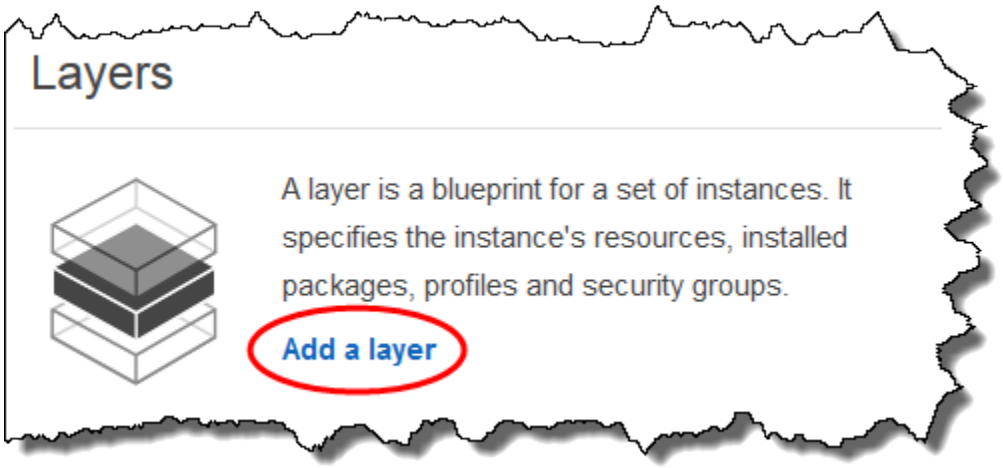
Etapa 3: Adicionar uma camada à pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

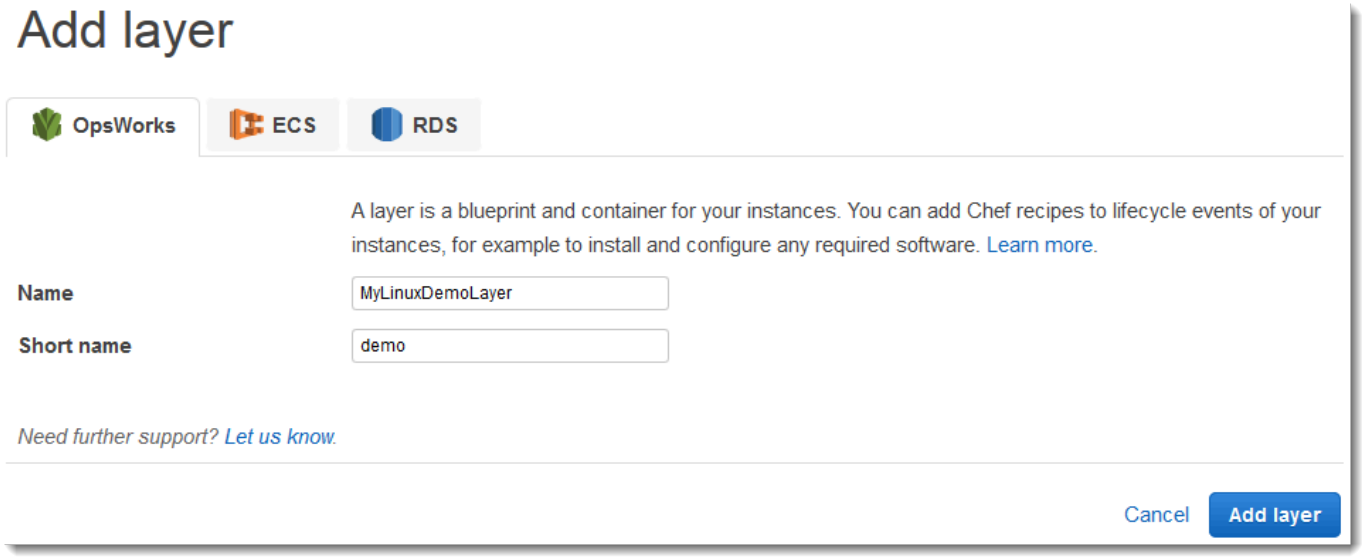
Uma camada é um modelo para um conjunto de instâncias, como EC2 instâncias da Amazon. Ela especifica informações como os recursos, os pacotes instalados e os grupos de segurança da instância. Depois, adicione uma camada à pilha. (Para obter mais informações sobre camadas, consulte Camadas.)

Para adicionar a camada à pilha

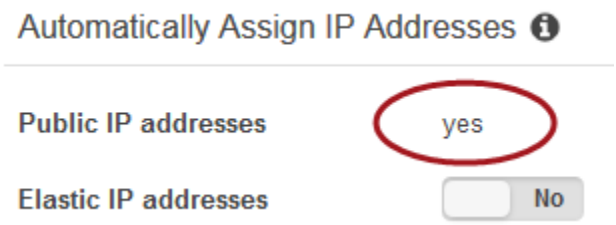
1. Com a MyLinuxDemoStackpágina exibida na etapa anterior, em Camadas, escolha Adicionar uma camada:



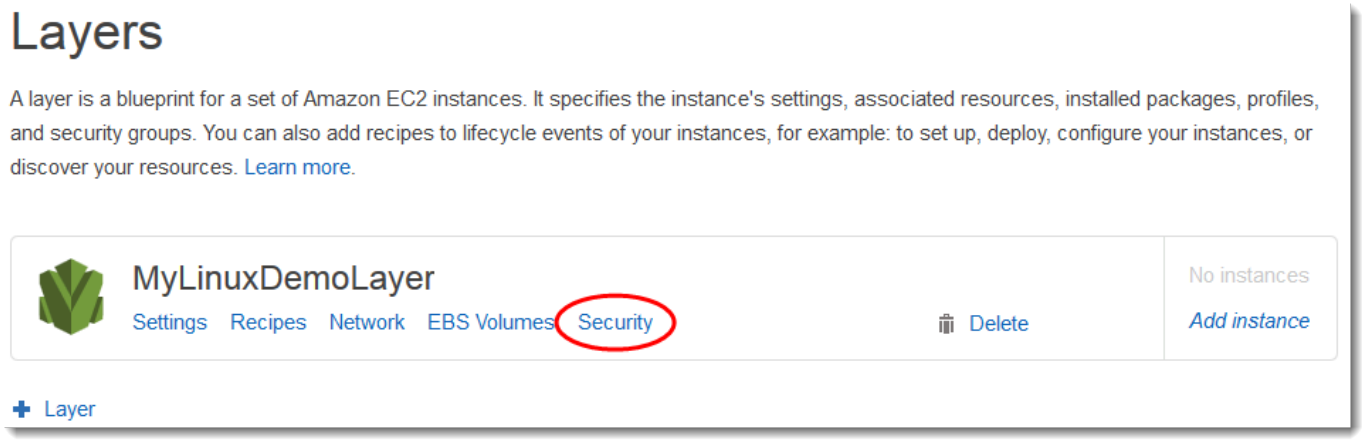
2. A página Add Layer será exibida. Na OpsWorksguia, em Nome, digite**`MyLinuxDemoLayer`**. (É possível inserir um nome diferente, mas substitua ele por `MyLinuxDemoLayer` durante esta demonstração.)
3. Para Short name (Nome curto), digite **`demo`**(é possível inserir um valor diferente, mas garanta que a substituição foi feita pela demonstração `demo`):



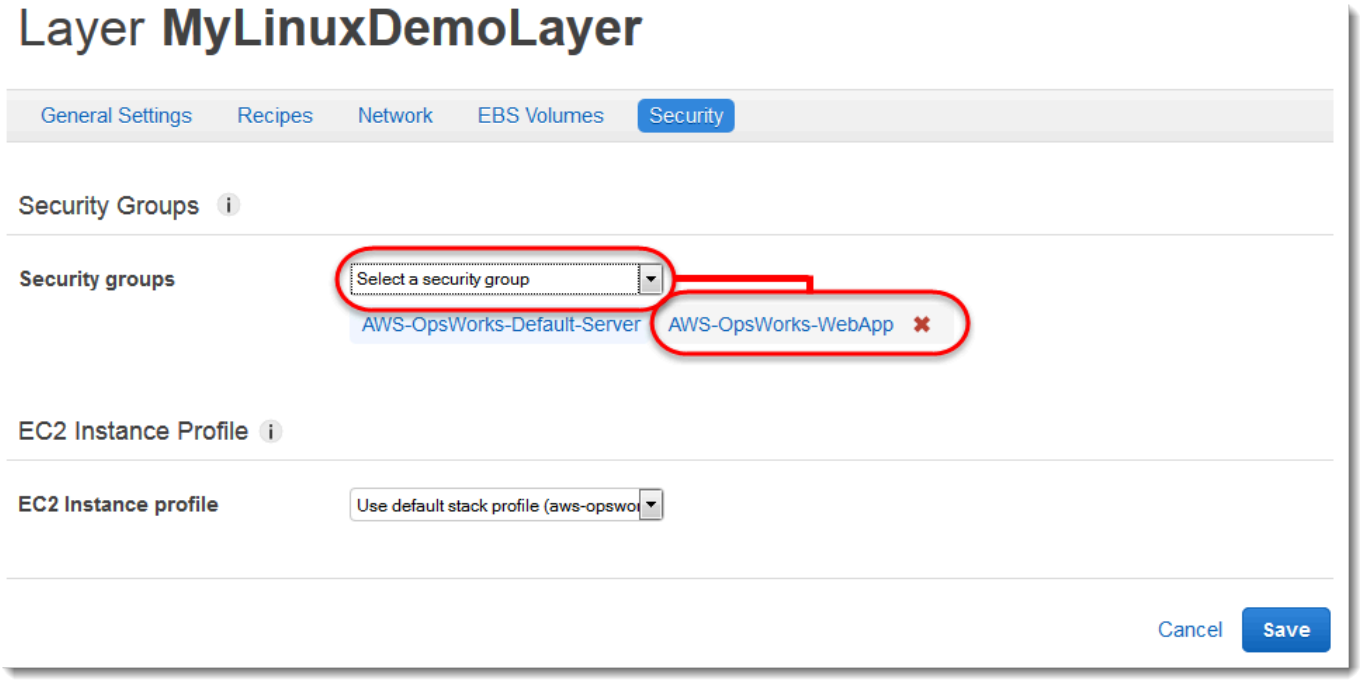
4. Escolha Adicionar camada. AWS OpsWorks Stacks cria a camada e exibe a página Camadas.
5. Na página Camadas, para MyLinuxDemoLayer, escolha Rede.
6. Na guia Network, em Automatically Assign IP Addresses, verifique se a opção Public IP addresses está definida como yes. Se tiver feito alterações, escolha Save.



7. Na página Layers, selecione Security:



8. A MyLinuxDemoLayer página Camada é exibida com a guia Segurança aberta. Para grupos de segurança, escolha AWS- OpsWorks - eWebApp, em seguida, selecione Salvar:



9. O grupo de segurança `AWS-OpsWorks-WebApp` é adicionado à camada. (Esse grupo de segurança permite que os usuários se conectem ao aplicativo na instância posteriormente neste passo a passo. Sem esse grupo de segurança, os usuários receberão uma mensagem no navegador da Web informando que não conseguem se conectar à instância.)

Agora você tem uma camada com as configurações corretas para essa explicação.

Na próxima etapa, você especificará o aplicativo para implantar à instância.

Etapa 4: Especificar o aplicativo para implantar à instância

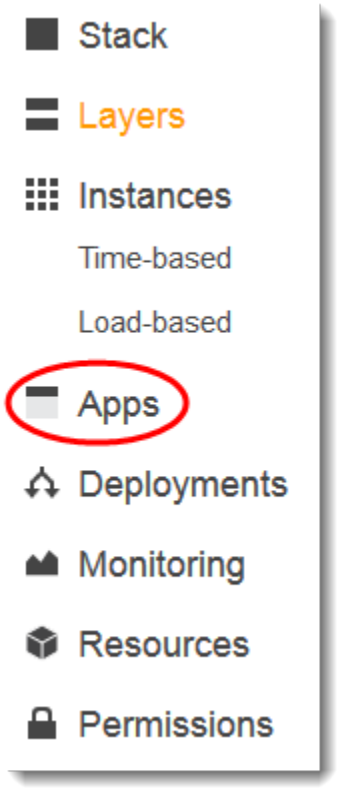
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Conte ao AWS OpsWorks Stacks sobre o aplicativo que você implantará na instância posteriormente neste passo a passo. Nesse contexto, o AWS OpsWorks Stacks define um aplicativo como código que você deseja executar em uma instância. (Para ter mais informações, consulte Apps.)

O procedimento descrito nesta seção se aplica a pilhas do Chef 12 e mais recentes. Para obter mais informações sobre como adicionar aplicativos a camadas nas pilhas do Chef 11, consulte Etapa 2.4: Criar e implantar um aplicativo - Chef 11.

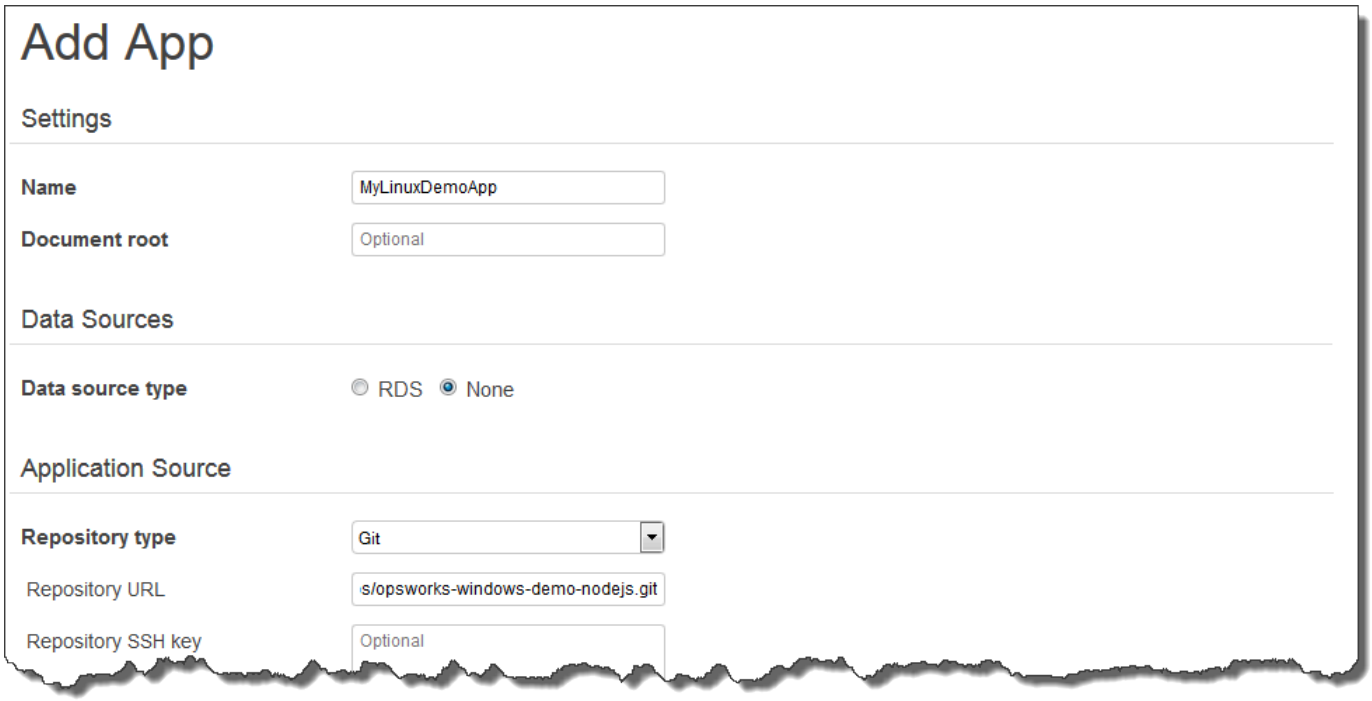
Para especificar o aplicativo para implantar

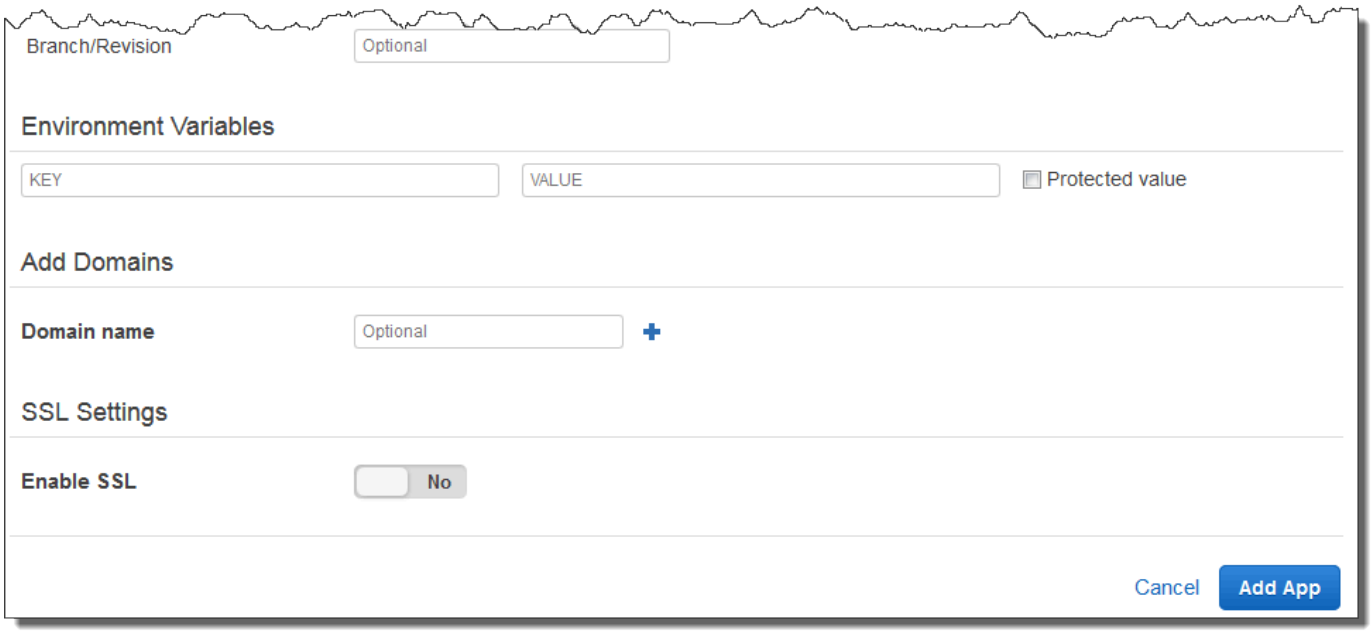
1. No painel de navegação do serviço, selecione Apps:



2. A página Aplicativos será exibida. Escolha Add an app. A página Add App será exibida.
3. Para Settings (Configurações), em Name (Nome), digite **`MyLinuxDemoApp`**. (É possível inserir um nome diferente, mas substitua ele por `MyLinuxDemoApp` durante esta demonstração.)
4. Para Application Source (Origem do aplicativo), em Repository URL (URL do repositório)., digite **`https://github.com/awslabs/opsworks-windows-demo-nodejs.git`**
5. Deixe os padrões para os seguintes:

- Settings, Document root (em branco)
- Fontes de dados, Tipo de fonte de dados (Nenhum)
- Tipo de repositório (Git)
- Repository SSH key (em branco)
- Branch/Revision (em branco)
- Environment Variables (opção KEY em branco, VALUE em branco e Protected Value desmarcada)
- Add Domains, Domain Name (em branco)
- Configurações de SSL, Habilitar SSL (No)





6. Escolha Adicionar aplicativo. AWS OpsWorks O Stacks adiciona o aplicativo e exibe a página Aplicativos.

Agora você tem um aplicativo com as configurações corretas para essa explicação.

Na próxima etapa, você iniciará a instância.

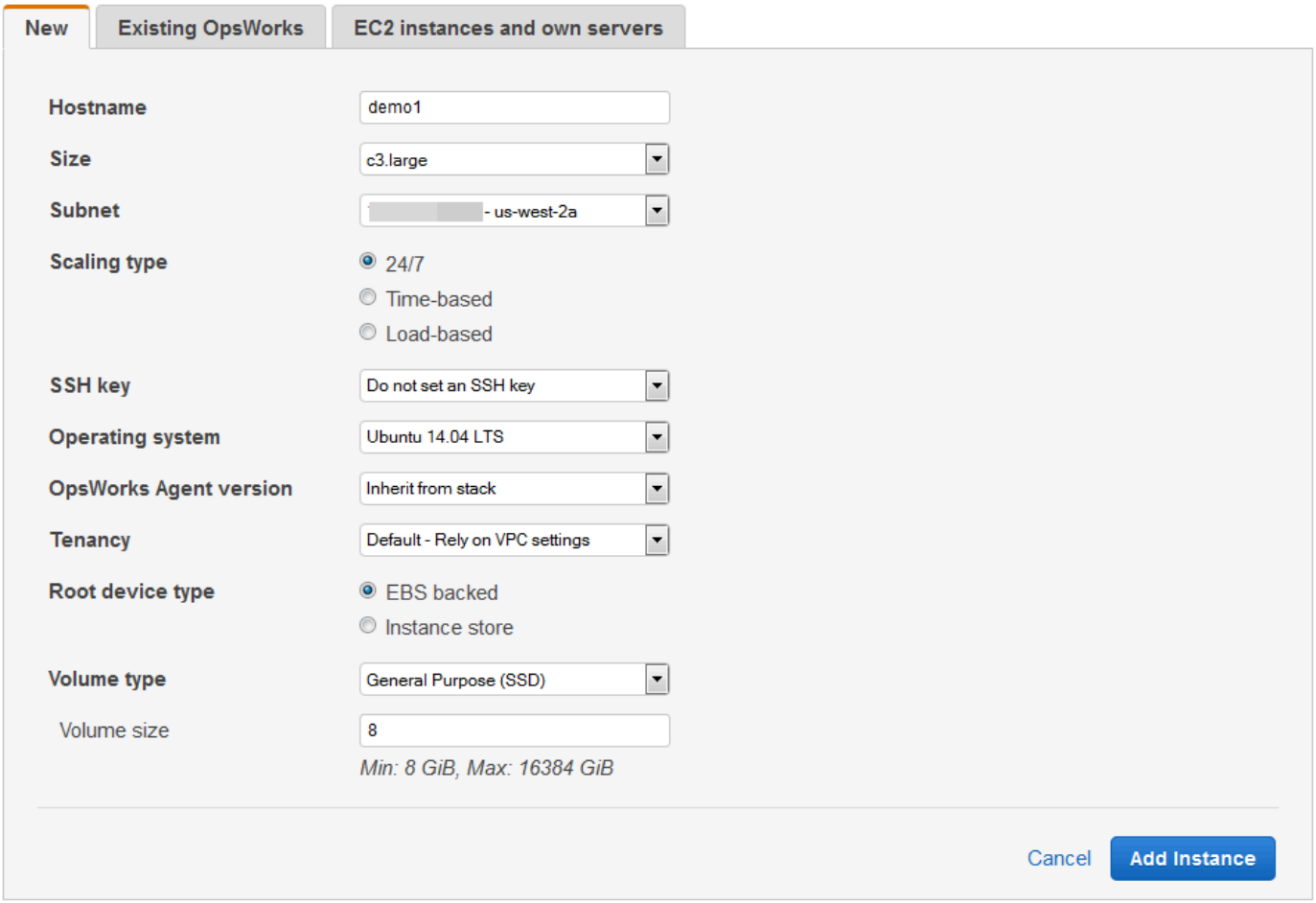
Etapa 5: Executar uma instância

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

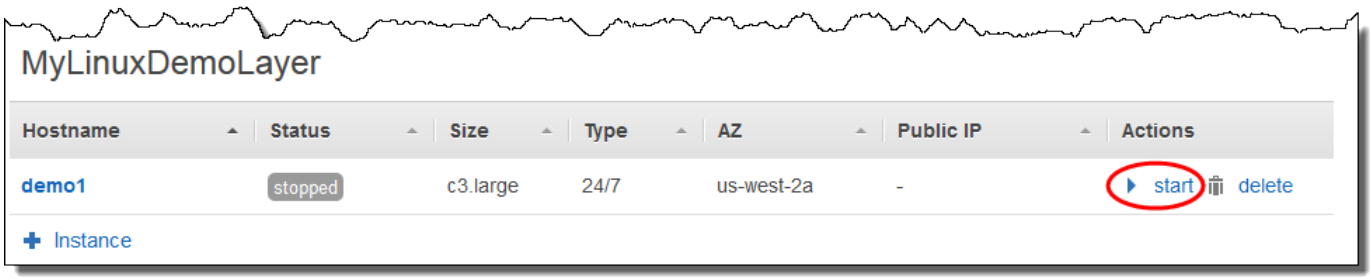
Use AWS OpsWorks Stacks para iniciar uma EC2 instância do Ubuntu Server Amazon. Essa instância usa as configurações que você definiu na camada criada anteriormente nessa apresentação. (Para ter mais informações, consulte Instâncias.)

Para iniciar a instância

1. No painel de navegação de serviços, escolha Instances. A página Instances será exibida.
2. Para MyLinuxDemoLayer, escolha Adicionar uma instância.
3. Na guia New, deixe os padrões da seguinte forma:
   - Hostname (demo1)
   - Tamanho (c3.large)
   - Sub-rede (*IP address*us-west-2a)
4. Escolha Advanced (Avançado).
5. Deixe os padrões para os seguintes:
   - Tipo de escalação (24/7)
   - Chave SSH (Não usar chave SSH padrão)
   - Operating system (Sistema operacional) (Ubuntu 18.04 LTS)
   - OpsWorks Versão do agente (herdar da pilha)
   - Locação (Padrão - usar as configurações de VPC)
   - Tipo de dispositivo raiz (Baseado em EBS)
   - Tipo de volume (Finalidade geral (SSD))
   - Tamanho de volume (8)
6. Seus resultados serão similares a captura de tela a seguir:



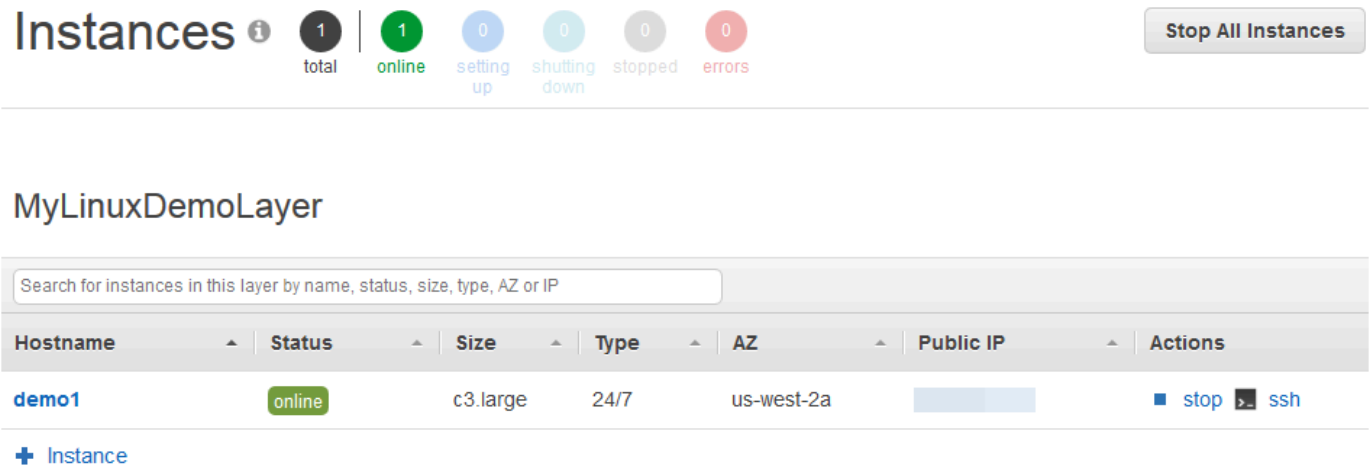
7. Escolha Adicionar instância. AWS OpsWorks Stacks adiciona a instância à camada e exibe a página Instâncias.
8. Para MyLinuxDemoLayer, para demo1, para Ações, escolha iniciar:



9. Durante vários minutos, o seguinte ocorre:
   - O círculo setting up muda de 0 para 1.

- Status passa de stopped para requested, depois pending, booting, running_setup e, finalmente, para online. Observe que esse processo pode levar vários minutos.
- Depois que Status muda para online, o indicador em círculo setting up muda de 1 para 0, e o círculo online muda de 0 para 1, em verde brilhante. Não prossiga até que o círculo online mude para um verde brilhante, e mostre a instância online 1.

10. Os resultados devem corresponder à captura de tela a seguir antes que você continue (se receber uma mensagem de falha, consulte Guia de depuração e solução de problemas):



Agora você tem uma instância pronta para receber implantação do aplicativo.

> **Note**
>
> Se quiser fazer login na instância para explorá-la ainda mais, primeiro forneça às AWS OpsWorks pilhas informações sobre sua chave SSH pública (que você pode criar com ferramentas como ssh-keygen ou PuTTYgen) e, em seguida, definir permissões na `MyLinuxDemoStack` pilha para permitir que o usuário faça login na instância. Para obter instruções, consulte Registro de uma chave SSH pública de um usuário e Login com SSH. Se você planeja usar SSH para se conectar a instâncias por meio do PuTTY, consulte Conectando-se à sua instância Linux a partir do Windows usando o PuTTY na documentação. AWS

Na próxima etapa, você implantará o aplicativo à instância.

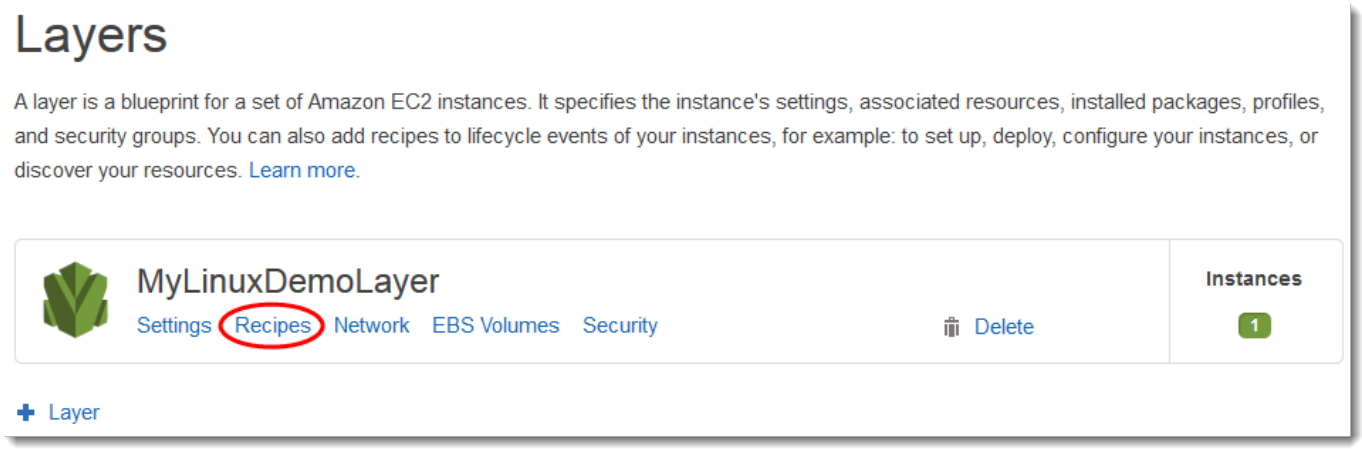
Etapa 6: Implantar o aplicativo à instância

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Nesta etapa, você implantará o aplicativo na instância em execução. GitHub (Para ter mais informações, consulte Implementação de aplicativos.) Antes de implantar o aplicativo, você deve especificar a receita a ser usada para coordenar a implantação. Uma receita é conceito do Chef. Receitas são instruções, escritas com sintaxe de linguagem Ruby, que especifica os recursos a serem usados e a ordem em que esses recursos são aplicados. (Para obter mais informações sobre, vá para Sobre receitas no site Aprender sobre Chef.)

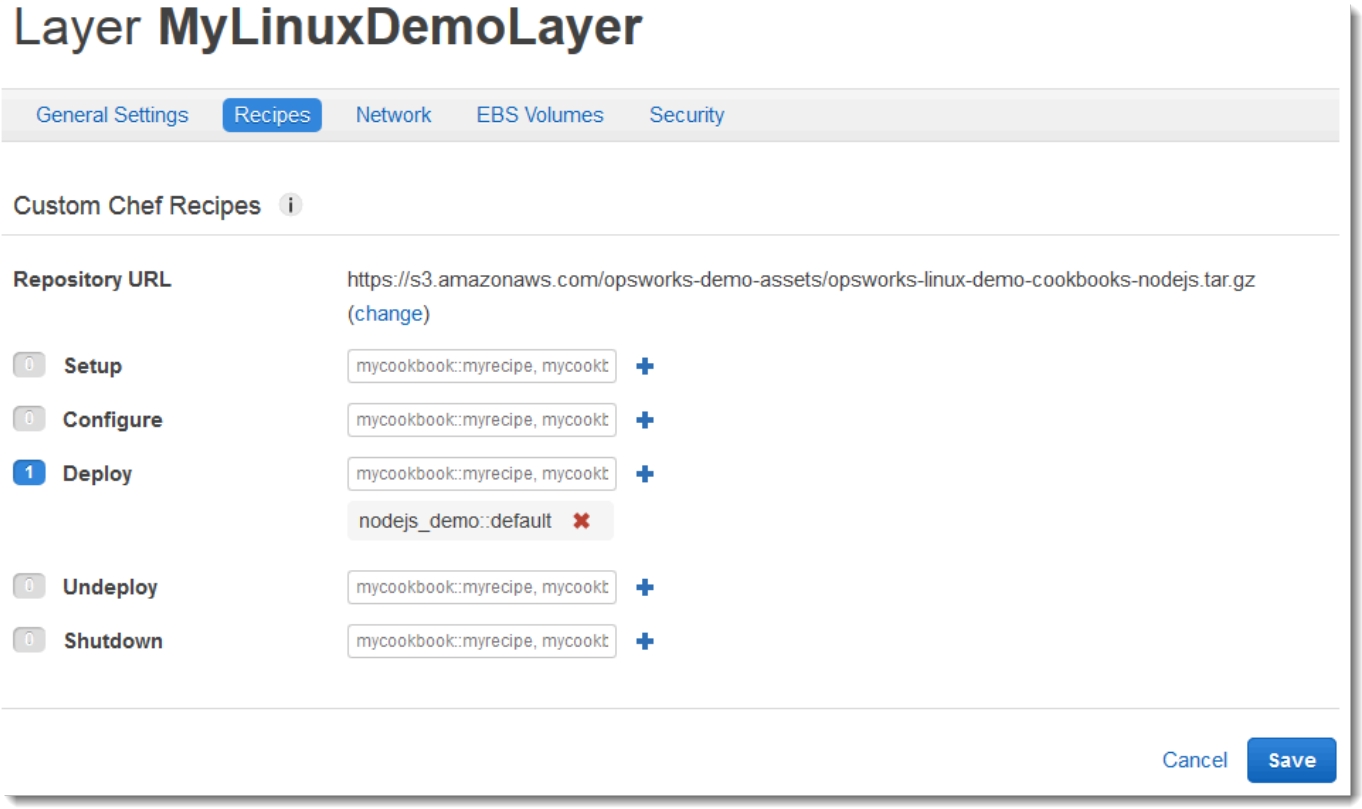
Para especificar a receita a ser usada para implantar o aplicativo à instância

1. No painel de navegação de serviços, escolha Layers. A página Layers será exibida.
2. Para MyLinuxDemoLayer, escolha Receitas:



   A MyLinuxDemoLayer página da camada é exibida com a guia Receitas aberta.
3. Para Custom Chef Recipes (Receitas do Chef personalizadas) e para Deploy (Implantar), digite **`nodejs_demo::default`** e pressione Enter. `nodejs_demo` é o nome do livro de receitas

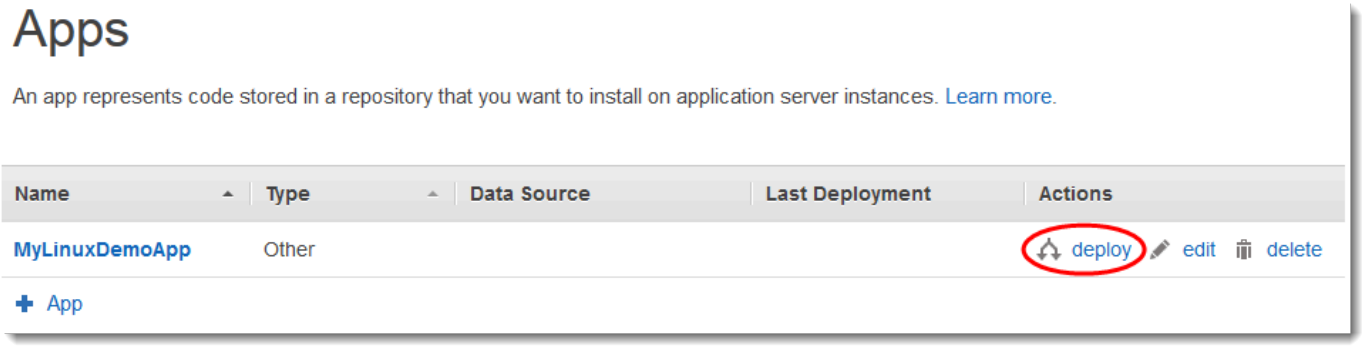
e `default` é o nome da receita de destino dentro dele. (Para explorar o código da receita, consulte Aprendendo mais: Explore o livro de receitas usado nessa apresentação.) Seus resultados devem ser compatíveis a captura de tela a seguir:



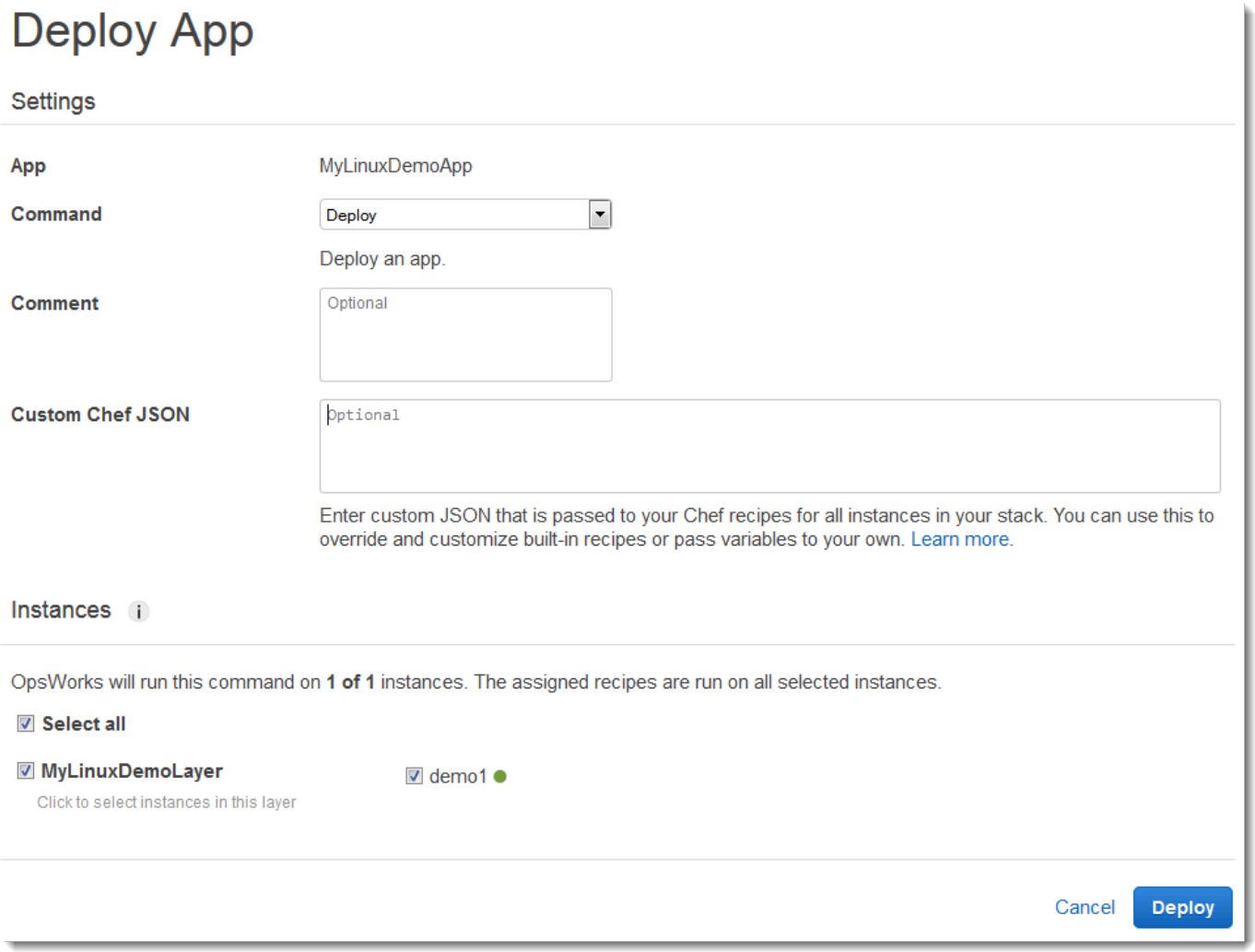
4. Escolha Salvar. AWS OpsWorks Stacks adiciona a receita ao evento Deploy lifecycle da camada.

Para implantar o aplicativo à instância

1. No painel de navegação do serviço, selecione Apps. A página Apps será exibida.
2. Para MyLinuxDemoApp, em Ações, escolha implantar, conforme exibido na captura de tela a seguir:



3. Na página Deploy App, deixe os padrões da seguinte forma:
   - Comando (Implantar)
   - Comment (em branco)
   - Settings, Advanced e Custom Chef JSON (em branco)
   - Instâncias, Avançado (marcado Selecionar tudo, marcado MyLinuxDemoLayer, marcado demo1)
4. Seus resultados devem ser compatíveis a captura de tela a seguir:



5. Escolha Implantar. A página Implantação MyLinuxDemoApp — implantação é exibida. Status muda de running para successful. Um círculo giratório é exibido próximo a demo1, que depois muda para uma marca de verificação verde. Observe que esse processo pode levar vários minutos. Não prossiga até que veja o Status de successful e o ícone da marcação em verde.
6. Os resultados precisam corresponder à captura de tela a seguir, exceto para Created at, Completed at, Duration e User. Se status for failed, para a solução de problemas, em Log, escolha show e veja os detalhes da falha:

Agora, você implantou com sucesso o aplicativo à instância.

Na próxima etapa, você testará o aplicativo implantado na instância.

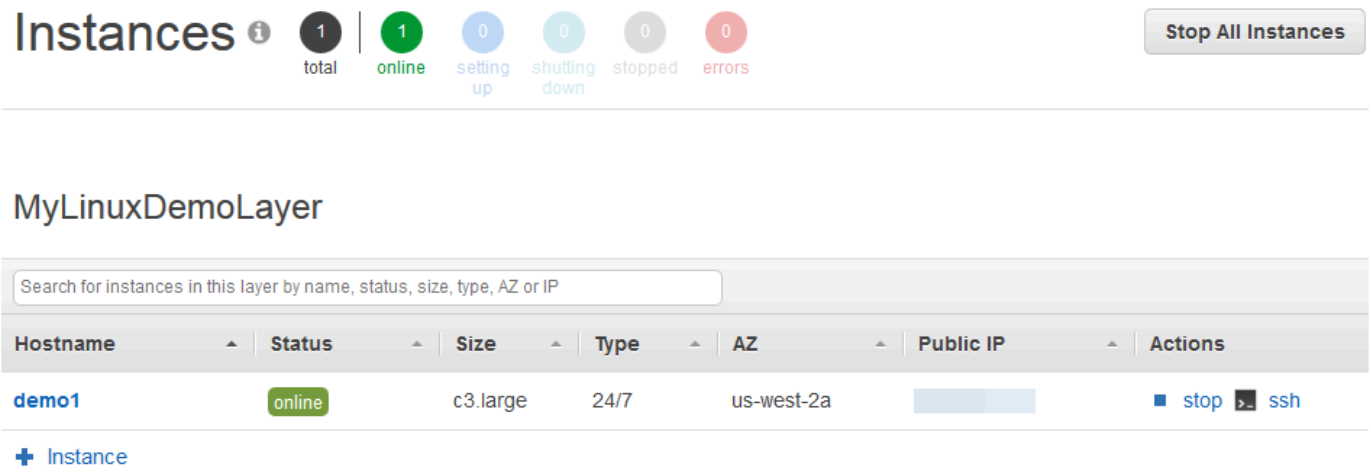
Etapa 7: Testar o aplicativo implantado na instância

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Agora, teste a implantação do aplicativo na instância.

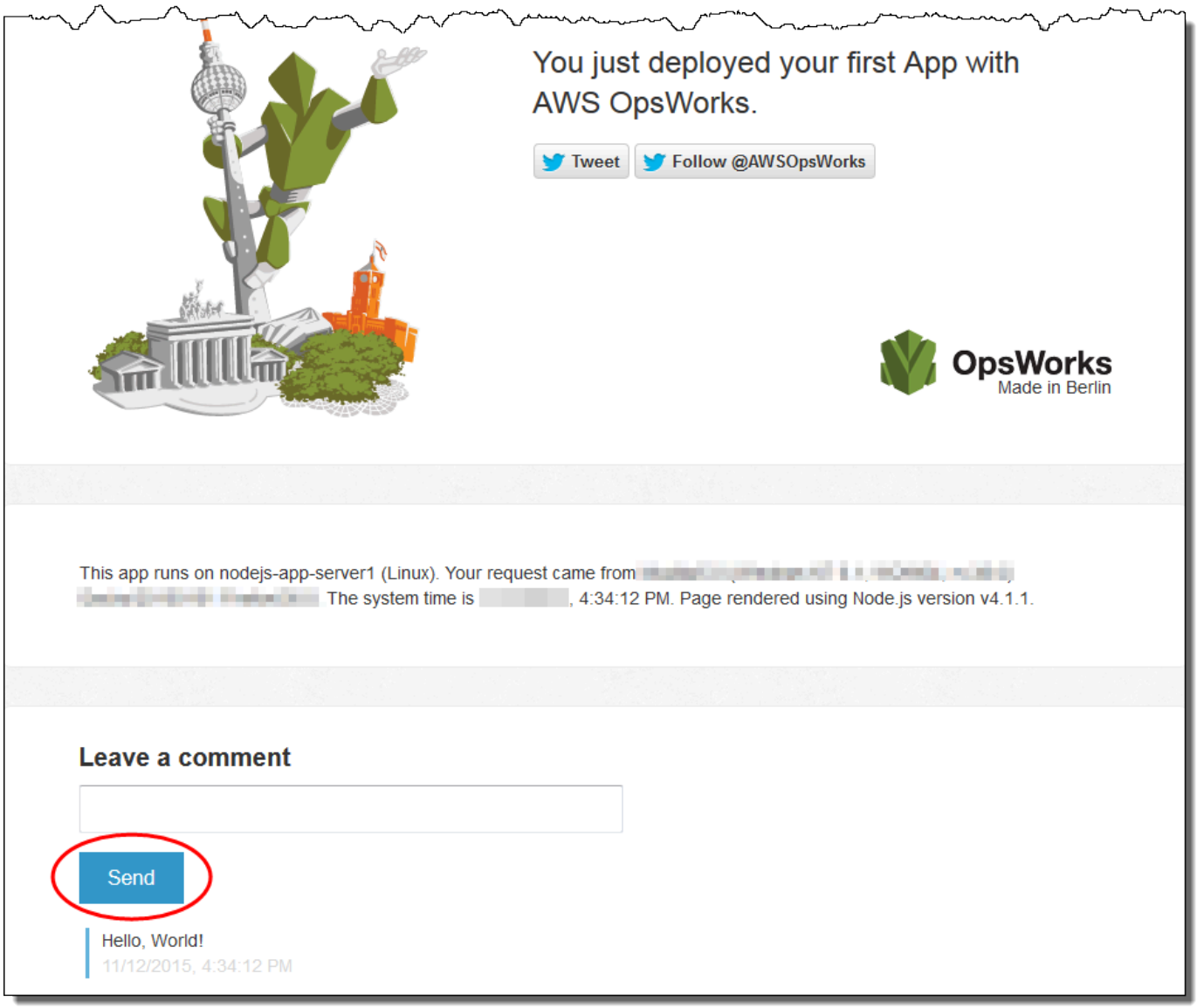
Para testar a implantação na instância

1. No painel de navegação de serviços, escolha Instances. A página Instances será exibida.
2. Para MyLinuxDemoLayerdemo1, para IP público, escolha o endereço IP:



Uma nova guia de navegador da web exibe o aplicativo.

3. Na página da Web de congratulações, na caixa de texto Leave a comment, digite um comentário e escolha Send para testar o aplicativo. O aplicativo adiciona seu comentário à página da Web. Deixe comentários e escolha Send sempre que desejar:



4. Se você tiver uma conta do Twitter, escolha Tweetar ou Seguir @ AWSOps Works e siga as instruções na tela para tweetar sobre o aplicativo ou seguir @ AWSOps Works.

Agora, você testou com sucesso o aplicativo implantado na instância.

Na próxima etapa, você pode limpar os AWS recursos que você usou para este passo a passo. Essa próxima etapa é opcional. Talvez você queira continuar usando esses AWS recursos à medida que continua aprendendo mais sobre AWS OpsWorks as pilhas. No entanto, manter esses AWS recursos disponíveis pode resultar em algumas cobranças contínuas em sua AWS conta. Se quiser manter esses AWS recursos disponíveis para uso posterior, você concluiu este passo a passo e pode pular para. Próximas etapas

Etapa 8 (opcional): limpeza

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para evitar cobranças adicionais em sua AWS conta, você pode excluir os AWS recursos que foram usados para este passo a passo. Esses AWS recursos incluem a pilha de AWS OpsWorks pilhas e os componentes da pilha. (Para obter mais informações, consulte AWS OpsWorks Preços.) No entanto, talvez você queira continuar usando esses AWS recursos à medida que continua aprendendo mais sobre AWS OpsWorks as pilhas. Se quiser manter esses AWS recursos disponíveis, você concluiu este passo a passo e pode pular para. Próximas etapas

O conteúdo armazenado nos recursos que você criou para essa demonstração pode conter informações de identificação pessoal. Se você não quer mais que essas informações sejam armazenadas pela AWS, siga as etapas neste tópico.

Para excluir o aplicativo da pilha

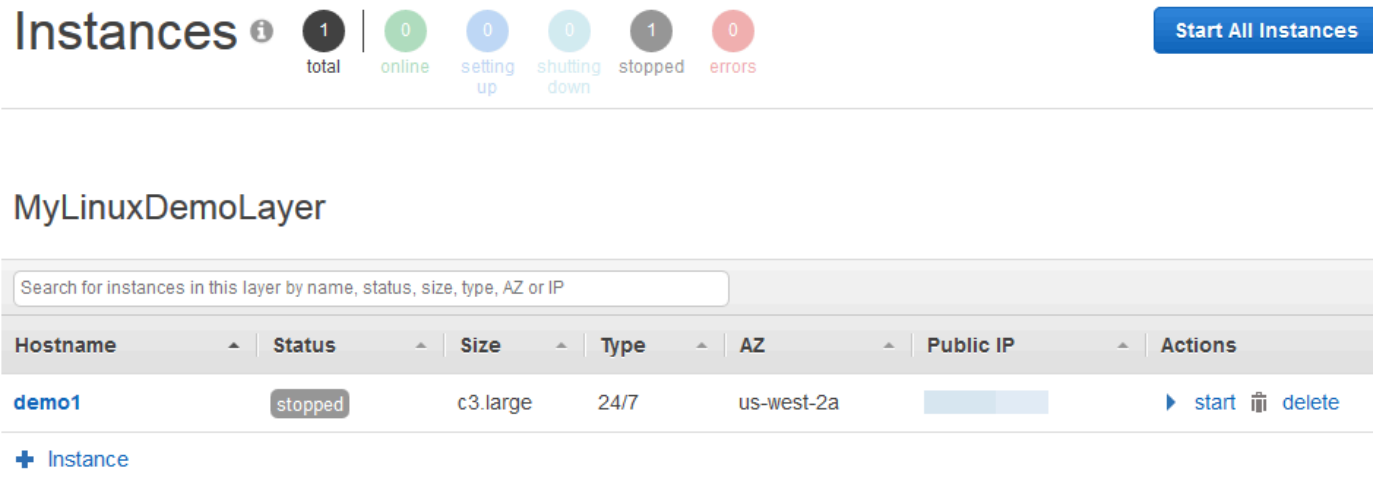
1. No console AWS OpsWorks Stacks, no painel de navegação do serviço, escolha Apps. A página Aplicativos será exibida.
2. Para MyLinuxDemoApp, em Ações, escolha excluir. Quando a mensagem de confirmação for exibida, escolha Excluir. AWS OpsWorks O Stacks exclui o aplicativo.

Para deletar a instância da pilha

1. No painel de navegação de serviços, escolha Instances. A página Instances será exibida.
2. Para MyLinuxDemoLayer, para demo1, para Ações, escolha parar. Quando a mensagem de confirmação for exibida, escolha Stop. O seguinte acontece.
   - Status muda de online para stopping e, em um determinado momento, para stopped.
   - online muda de 1 para 0.
   - shutting down muda de 0 para 1 e, em um determinado momento, de volta para 0.

- stopped muda, em determinado momento, de 0 para 1.

Esse processo pode levar alguns minutos. Quando AWS OpsWorks as pilhas terminarem, os seguintes resultados serão exibidos.



3. Em Actions, selecione delete. Ao ver a mensagem de confirmação, escolha Excluir. AWS OpsWorks Stacks exclui a instância e exibe a mensagem Sem instâncias.

Para excluir a pilha

1. No painel de navegação de serviços, escolha Stack. A página MyLinuxDemoStack será exibida.
2. Escolha Delete Stack. Ao ver a mensagem de confirmação, escolha Excluir. AWS OpsWorks Stacks exclui a pilha e exibe a página do OpsWorks Painel.

Opcionalmente, você pode excluir o usuário e o par de EC2 chaves da Amazon que você usou para este passo a passo, se não quiser reutilizá-los para acessar outros AWS serviços e instâncias. EC2 Para obter instruções, consulte Excluir um usuário do IAM e pares de EC2 chaves e instâncias Linux da Amazon.

Você acabou de concluir essa apresentação. Para obter mais informações, consulte Próximas etapas.

Próximas etapas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Agora que você concluiu este passo a passo, você pode aprender mais sobre como usar AWS OpsWorks o Stacks:

- Explore o livro de receitas e o aplicativo que você usou para esta demonstração. Consulte Aprendendo mais: Explore o livro de receitas usado nessa apresentação e Aprendendo mais: Explore o aplicativo usado nessa apresentação.
- Pratique o uso AWS OpsWorks do Stacks com instâncias do Windows. Consulte Conceitos básicos: Windows.
- Saiba mais sobre as pilhas ao aprender a Criar uma nova pilha.
- Saiba mais sobre camadas ao Como editar a configuração de uma OpsWorks camada.
- Saiba mais sobre as instâncias ao Adicionar uma instância a uma camada.
- Saiba mais sobre aplicativos ao Implementação de aplicativos.
- Saiba mais sobre Livros de receitas e receitas.
- Crie seus próprios livros de receitas. Consulte Conceitos básicos: livros de receitas.
- Saiba mais sobre o controle de acesso às pilhas com Segurança e permissões.

Aprendendo mais: Explore o livro de receitas usado nessa apresentação

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico descreve o livro de receitas que AWS OpsWorks Stacks usou para o passo a passo.

Um livro de receitas é um conceito do Chef. Livros de receitas são arquivos mortos que contêm informações de configuração, como receitas, valores de atributo, arquivos, modelos, bibliotecas, definições e recursos personalizados. Uma receita também e um conceito do Chef. Receitas são instruções, escritas com sintaxe de linguagem Ruby, que especifica os recursos a serem usados e a ordem em que esses recursos são aplicados. Para mais informações, vá para Sobre livros de receitas e Sobre receitas no site Saber sobre Chef.

Para ver o conteúdo do livro de receitas usado neste passo a passo, extraia o conteúdo do arquivo opsworks-linux-demo-cookbooks-nodejs.tar.gz em um diretório vazio em sua estação de trabalho local. (Você também pode entrar na instância na qual implantou o livro de receitas e explorar os conteúdos do diretório `/var/chef/cookbooks`.)

O arquivo `default.rb` no diretório `cookbooks/nodejs_demo/recipes` é onde o livro de receitas executa seu código:

```
app = search(:aws_opsworks_app).first
app_path = "/srv/#{app['shortname']}"

package "git" do
  options "--force-yes" if node["platform"] == "ubuntu" && node["platform_version"] ==
 "18.04"
end

application app_path do
  javascript "4"
  environment.update("PORT" => "80")

  git app_path do
    repository app["app_source"]["url"]
    revision app["app_source"]["revision"]
  end

  link "#{app_path}/server.js" do
    to "#{app_path}/index.js"
  end
```

```
  npm_install
  npm_start
end
```

Veja o que o arquivo faz:

- `search(:aws_opsworks_app).first` usa busca de Chef para procurar informações sobre o aplicativo que eventualmente serão implantados à instância. Essas informações incluem configurações como o nome curto do aplicativo e seus detalhes de repositório fonte. Como apenas um aplicativo foi implantado nessa demonstração, a busca do Chef obtém essas configurações do primeiro item de informações dentro do índice de busca `aws_opsworks_app` na instância. Sempre que uma instância é iniciada, o AWS OpsWorks Stacks armazena essas e outras informações relacionadas como um conjunto de pacotes de dados na própria instância, e você obtém o conteúdo do pacote de dados por meio da pesquisa do Chef. Apesar de você poder usar hardcoding nessas configurações nessa receita, usando recipientes de dados e busca de Chef é uma abordagem mais robusta. Para obter mais informações sobre recipientes de dados, consulte o AWS OpsWorks Referência do Stacks Data Bag. Consulte também Sobre recipientes de dados no site Aprender sobre Chef. Para obter mais informações sobre busca do Chef, vá para Sobre pesquisa no site Aprender sobre Chef.
- O recurso `package` instala Git na instância.
- O recurso `application` descreve e implanta aplicativos da web:
  - `javascripté` a versão do tempo de JavaScript execução a ser instalada.
  - `environment` define um ambiente variável.
  - `git` obtém o código fonte do repositório especificado e ramificação.
  - `app_path` é o caminho para onde clonar o repositório. Se o caminho não existir na instância, o AWS OpsWorks Stacks o cria.
  - `link` cria um link simbólico.
  - `npm_install` instala Gerente de pacote de nó, o gente de pacote de nó padrão de Node.js.
  - `npm_start` executa Node.js.

Embora AWS OpsWorks Stacks tenha criado o livro de receitas usado para este passo a passo, você pode criar seus próprios livros de receitas. Para saber como, consulte Conceitos básicos: livros de receitas. Além disso, vá para Sobre livro de receitas, Sobre receitas e Aprender o básico do Chef no Ubuntu no site Aprender sobre Chef, e a seção "Nosso primeiro livro de receitas do Chef" no Primeiras etapas com Chef no site Introdução ao Chef.

Aprendendo mais: Explore o aplicativo usado nessa apresentação

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico descreve o aplicativo que o AWS OpsWorks Stacks implanta na instância para este passo a passo.

Para ver o código-fonte do aplicativo, extraia o conteúdo do opsworks-windows-demo-nodejs GitHub repositório em um diretório vazio na sua estação de trabalho local. Você também pode entrar na instância na qual implantou o livro de receitas e explorar os conteúdos do diretório `/srv/mylinuxdemoapp`.

O arquivo `index.js` contém o código mais significativo para o aplicativo:

```
var express = require('express');
var app = express();
var path = require('path');
var os = require('os');
var bodyParser = require('body-parser');
var fs = require('fs');

var add_comment = function(comment) {
  var comments = get_comments();
  comments.push({"date": new Date(), "text": comment});
  fs.writeFileSync('./comments.json', JSON.stringify(comments));
};

var get_comments = function() {
  var comments;
  if (fs.existsSync('./comments.json')) {
    comments = fs.readFileSync('./comments.json');
    comments = JSON.parse(comments);
  } else {
    comments = [];
```

```
  }
  return comments;
};

app.use(function log (req, res, next) {
  console.log([req.method, req.url].join(' '));
  next();
});
app.use(express.static('public'));
app.use(bodyParser.urlencoded({ extended: false }))

app.set('view engine', 'jade');
app.get('/', function(req, res) {
  var comments = get_comments();
  res.render("index",
    { agent: req.headers['user-agent'],
      hostname: os.hostname(),
      nodeversion: process.version,
      time: new Date(),
      admin: (process.env.APP_ADMIN_EMAIL || "admin@unconfigured-value.com" ),
      comments: get_comments()
    });
});

app.post('/', function(req, res) {
  var comment = req.body.comment;
  if (comment) {
    add_comment(comment);
    console.log("Got comment: " + comment);
  }
  res.redirect("/#form-section");
});

var server = app.listen(process.env.PORT || 3000, function() {
  console.log('Listening on %s', process.env.PORT);
});
```

Veja o que o arquivo faz:

- `require` carrega módulos que contêm algum código dependente que esse aplicativo da web precisa para ser executado como esperado.
- As funções `add_comment` e `get_comments` escrevem informações para, e leem informações do arquivo `comments.json`.

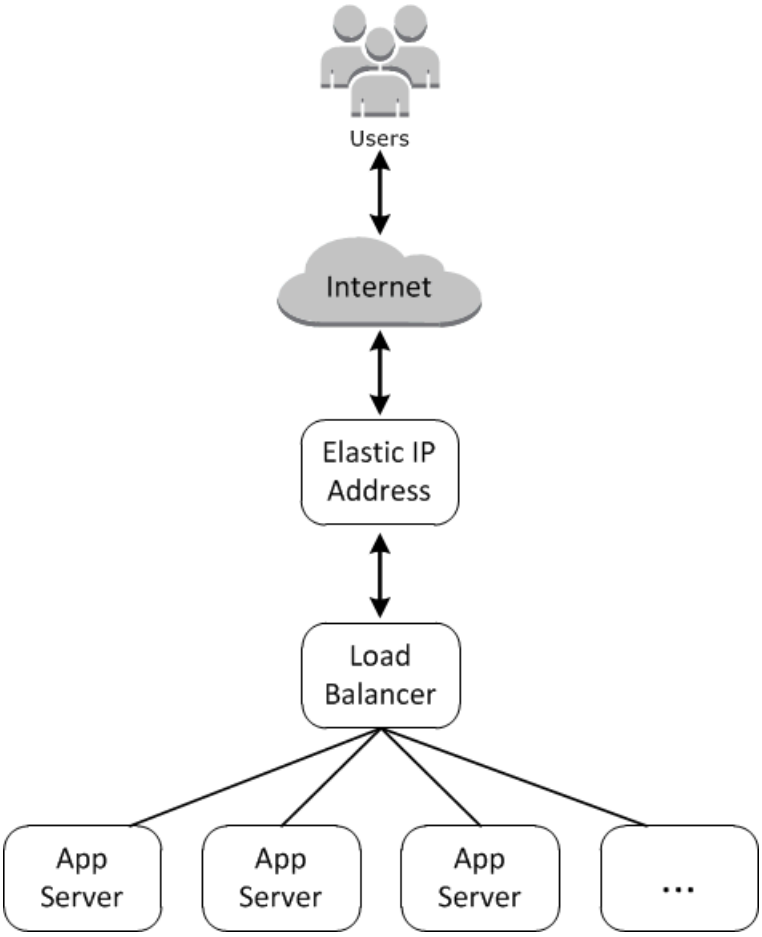
- Para obter mais informações sobre `app.get`, `app.listen`, `app.post`, `app.set` e `app.use`, consulte o Expressar referência da API.

Para aprender como criar pacote para aplicativo de implantação, consulte Origem do aplicativo.

Conceitos básicos de pilhas do Windows

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Aplicativos baseados na nuvem geralmente exigem um grupo de recursos relacionados (servidores de aplicativos, servidores de banco de dados e assim por diante) que devem ser criados e gerenciados coletivamente. Esta coleção de instâncias é chamada de pilha. Uma simples pilha de aplicativo pode ter a seguinte aparência.



A arquitetura básica contém o seguinte:

- Um Endereço IP elástico para receber as solicitações do usuário.
- Um load balancer para distribuir as solicitações de entrada igualmente entre os servidores de aplicativos.
- Um conjunto de instâncias do servidor de aplicativos, a quantidade necessária para lidar com o tráfego.

Além disso, normalmente você precisa de uma maneira de distribuir aplicativos para os servidores de aplicativos, gerenciar permissões de usuário e assim por diante.

AWS OpsWorks O Stacks fornece uma maneira simples e direta de criar e gerenciar pilhas e seus aplicativos e recursos associados. Este capítulo apresenta os conceitos básicos do AWS OpsWorks Stacks, e alguns de seus recursos mais sofisticados, orientando você durante o processo de criação da pilha de servidor de aplicativos no diagrama. Ele usa um modelo de desenvolvimento incremental

que o AWS OpsWorks Stacks facilita o acompanhamento: configure uma pilha básica e, depois de funcionar corretamente, adicione componentes até chegar a uma implementação completa.

- Etapa 1: concluir os pré-requisitos mostra como configurar a demonstração.
- Etapa 2: Criar uma pilha de servidor de aplicativos básico mostra como criar uma pilha básica para oferecer suporte a Internet Information Services (IIS) e implantar um aplicativo no servidor.
- Etapa 3: Dimensionar IISExample mostra como dimensionar uma pilha para lidar com o aumento de carga, adicionando mais servidores de aplicativos, um load balancer para distribuir o tráfego de entrada e um Endereço IP elástico para receber as solicitações de entrada.

Tópicos

- Etapa 1: concluir os pré-requisitos
- Etapa 2: Criar uma pilha de servidor de aplicativos básico
- Etapa 3: Dimensionar IISExample
- Próximas etapas

Etapa 1: concluir os pré-requisitos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Conclua as seguintes etapas de configuração antes de iniciar a demonstração. Essas etapas de configuração incluem se inscrever em uma AWS conta, criar um usuário administrativo e atribuir permissões de acesso ao AWS OpsWorks Stacks.

Se você tiver concluído as demonstrações Conceitos básicos: exemplo e Conceitos básicos: Linux, já cumpriu os pré-requisitos para esta demonstração e pode ir direto para Etapa 2: Criar uma pilha de servidor de aplicativos básico.

Tópicos

- Inscreva-se para um Conta da AWS
- Criar um usuário com acesso administrativo
- Atribuir permissões de acesso ao serviço
- Garanta que os usuários do AWS OpsWorks Stacks sejam adicionados ao seu domínio

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra a https://portal.aws.amazon.com/billing/inscrição.
2. Siga as instruções online.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, você pode visualizar a atividade atual da sua conta e gerenciar sua conta acessando https://aws.amazon.com/e escolhendo Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, insira a senha.

   Para obter ajuda ao fazer login usando o usuário-raiz, consulte Fazer login como usuário-raiz no Guia do usuário do Início de Sessão da AWS .

2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

   Para obter instruções, consulte Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz (console) no Guia do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilita o Centro de Identidade do IAM.

   Para obter instruções, consulte Habilitar o AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center .

2. No Centro de Identidade do IAM, conceda o acesso administrativo a um usuário.

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com o seu usuário do Centro de Identidade do IAM, use o URL de login enviado ao seu endereço de e-mail quando o usuário do Centro de Identidade do IAM foi criado.

  Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

   Para obter instruções, consulte Criar um conjunto de permissões no Guia do usuário do AWS IAM Identity Center .

2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Adicionar grupos no Guia do usuário do AWS IAM Identity Center .

Atribuir permissões de acesso ao serviço

Ative o acesso ao serviço AWS OpsWorks Stacks (e aos serviços relacionados dos quais o AWS OpsWorks Stacks depende) adicionando as `AmazonS3FullAccess` permissões `AWSOpsWorks_FullAccess` e à sua função ou usuário.

Para obter mais informações sobre adicionar permissões, consulte Adicionar permissões de identidade do IAM (console).

Garanta que os usuários do AWS OpsWorks Stacks sejam adicionados ao seu domínio

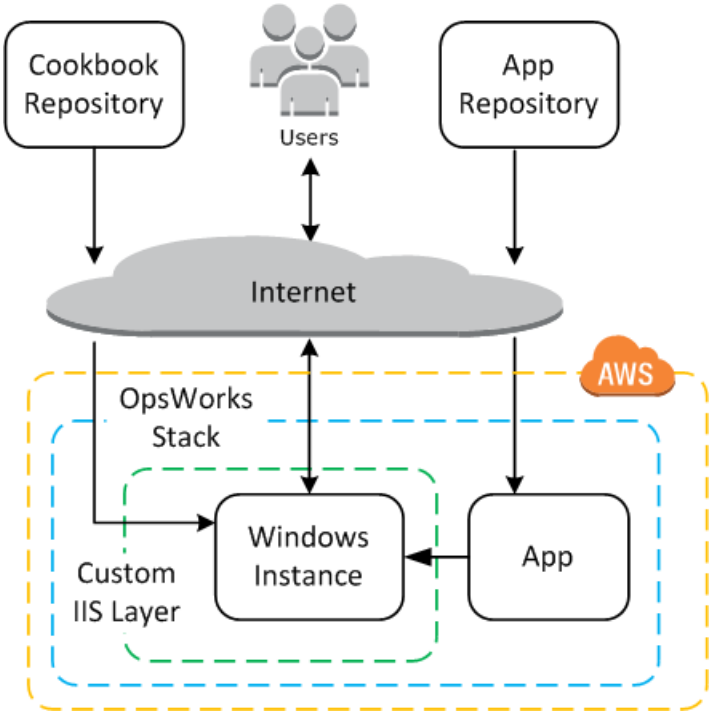
Em uma pilha do Chef 12.2, o livro de receitas `aws_opsworks_users` incluído cria usuários que têm acesso a SSH e a Remote Desktop Protocol (RDP) às instâncias baseadas no Windows. Quando você une instâncias do Windows em sua pilha a um domínio do Active Directory, a execução desse livro de receitas pode falhar se os usuários do AWS OpsWorks Stacks não existirem no Active Directory. Se os usuários não forem reconhecidos no Active Directory, as instâncias podem entrar em um estado `setup failed` quando você as reinicia depois de uni-las a um domínio. Para instâncias do Windows associadas a um domínio, não é suficiente conceder SSH/RDP acesso aos usuários do AWS OpsWorks Stacks na página de permissões do usuário.

Antes de unir instâncias do Windows em uma pilha do Chef 12.2 a um domínio do Active Directory, certifique-se de que todos os usuários do AWS OpsWorks Stacks da pilha baseada no Windows sejam membros do domínio. A melhor maneira de fazer isso é configurar a identidade federada com o IAM antes de criar sua pilha baseada em Windows e, em seguida, importar usuários federados para as AWS OpsWorks pilhas antes de unir instâncias em sua pilha a um domínio. Para obter mais informações sobre como fazer isso, consulte Habilitar a federação para AWS usando o Windows Active Directory, o ADFS, e o SAML 2.0 no Blog de segurança da AWS e Federação de usuários existentes no guia do usuário do IAM.

Etapa 2: Criar uma pilha de servidor de aplicativos básico

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Uma pilha de servidor de aplicativo básico consiste em uma instância de servidor de aplicativo única com um endereço IP público para receber solicitações de usuário. Código de aplicativo e quaisquer arquivos relacionados são armazenados em um repositório separado e implantados de lá para o servidor. O diagrama a seguir ilustra essa pilha.



A pilha tem os seguintes componentes:

- Uma camada, que representa um grupo de instâncias e especifica como elas são configuradas.

  A camada neste exemplo representa um grupo de instâncias do IIS.
- Uma instância, que representa uma EC2 instância da Amazon.

  Nesse caso, a layer configura uma única instância para executar o IIS, mas as camadas podem ter qualquer número de instâncias.
- Um aplicativo, que contém as informações necessárias para instalar um aplicativo na instância.
- Um livro de receitas, que contém receitas personalizadas do Chef que oferecem suporte à camada personalizada do IIS. O livro de receitas e o código do aplicativo são armazenados em repositórios remotos, como um arquivo morto em um bucket do Amazon S3 ou um repositório Git.

As seções a seguir descrevem como usar o console AWS OpsWorks Stacks para criar a pilha e implantar o aplicativo.

Tópicos

- Etapa 2.1: Criar a pilha
- Etapa 2.2: Autorizar o acesso RDP
- Etapa 2.3: Implementar um livro de receitas personalizado
- Etapa 2.4: Adicionar uma camada do IIS
- Etapa 2.5: Implantar um aplicativo

Etapa 2.1: Criar a pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você inicia um projeto AWS OpsWorks Stacks criando uma pilha, que atua como um contêiner para suas instâncias e outros recursos. A configuração da pilha especifica algumas configurações básicas, como a região da AWS e o sistema operacional padrão, que são compartilhados por todas as instâncias da pilha.

Para criar um nova pilha

1. Adicionar uma pilha

   Se você ainda não tiver feito isso, faça login no console do AWS OpsWorks Stacks.

   - Se a conta não tiver pilhas existentes, você verá a OpsWorks página Bem-vindo à AWS; escolha Adicionar sua primeira pilha.
   - Caso contrário, você verá o painel de AWS OpsWorks pilhas, que lista as pilhas da sua conta; escolha Adicionar pilha.
2. Configurar a pilha

   Na página Add Stack, escolha Chef 12 stack e, em seguida, especifique as seguintes configurações:

Nome da pilha

Insira um nome para sua pilha, que pode conter caracteres alfanuméricos (a–z, A–Z e 0–9) e hífens (-). A pilha de exemplo para esta demonstração se chama **`IISWalkthrough`**.

Região

Selecione Oeste dos EUA (Oregon) como a região da pilha.

Você pode criar uma pilha em qualquer região, mas recomendamos o Oeste dos EUA (Oregon) para os tutoriais.

Sistema operacional padrão

Escolha Windows e especifique Microsoft Windows Server 2022 Base, que é a configuração padrão.

Usar livros de receitas do Chef personalizados

Para esta demonstração, especifique No para esta opção.

3. Escolha Advanced para confirmar que você tem uma função do IAM e o perfil de instância do IAM padrão selecionado.

   Perfil do IAM

   Especifique a função IAM (AWS Identity and Access Management) da pilha. AWS OpsWorks O Stacks precisa acessar outros serviços da AWS para realizar tarefas como criar e gerenciar EC2 instâncias da Amazon. A função do IAM especifica a função que AWS OpsWorks Stacks assume para trabalhar com outros serviços da AWS em seu nome. Para obter mais informações, consulte Permitindo que AWS OpsWorks as pilhas ajam em seu nome.

   - Se sua conta tiver uma função existente do AWS OpsWorks Stacks IAM, você poderá selecioná-la na lista.

     Se a função foi criada pelo AWS OpsWorks Stacks, ela será nomeada`aws-opsworks-service-role`.

   - Caso contrário, selecione Nova função do IAM para direcionar AWS OpsWorks Stacks para criar uma nova função para você com as permissões corretas.

Observação: Se você tiver permissões de Acesso Total ao AWS OpsWorks Stacks, a criação de uma nova função exigirá diversas permissões adicionais do IAM. Para obter mais informações, consulte Exemplo de políticas.

4. Aceite os valores padrão das outras configurações e, em seguida, escolha Add Stack. Para obter mais informações sobre as diversas configurações de pilha, consulte Criar uma nova pilha.

Etapa 2.2: Autorizar o acesso RDP

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Agora que uma pilha foi criada, você criará uma camada e adicionará uma instância do Windows à camada. No entanto, antes de fazer isso, é preciso configurar a pilha para permitir que você use o RDP para se conectar às instâncias personalizadas da camada. Para isso, você deve fazer o seguinte:

- Adicione uma regra de entrada para o grupo de segurança que controla o acesso RDP.
- Defina suas permissões de AWS OpsWorks pilhas para essa pilha para permitir o acesso RDP.

Quando você cria a primeira pilha em uma região, o AWS OpsWorks Stacks cria um conjunto de grupos de segurança. Eles incluem um chamado algo como`AWS-OpsWorks-RDP-Server`, que o AWS OpsWorks Stacks anexa a todas as instâncias do Windows para permitir acesso ao RDP. No entanto, por padrão, esse grupo de segurança não tem regras. Assim, você deve adicionar uma regra de entrada para permitir o acesso RDP para suas instâncias.

Para permitir o acesso RDP

1. Abra o EC2 console da Amazon, configure-o para a região da pilha e escolha Security Groups no painel de navegação.
2. Escolha AWS- OpsWorks -RDP-Server, escolha a guia Entrada e escolha Editar.

3. Selecione Add Rule e especifique as seguintes configurações:

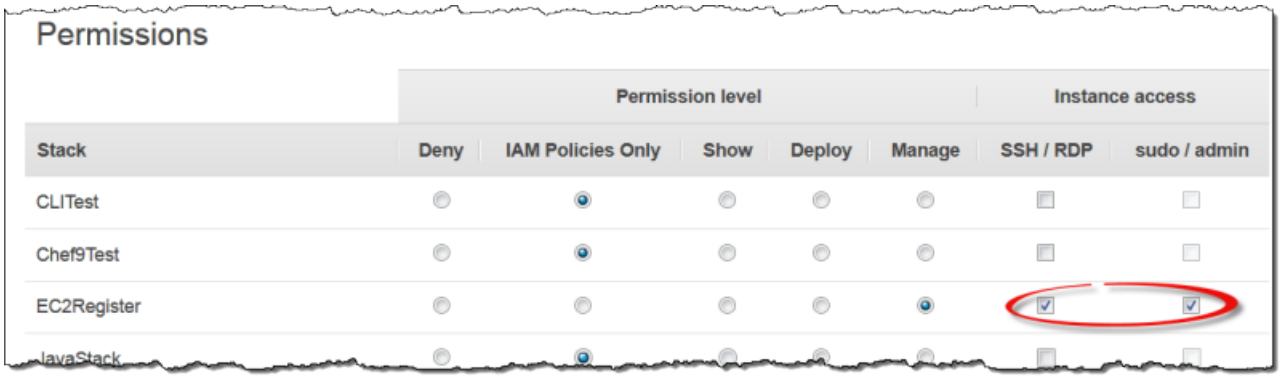
   - Tipo: RDP.
   - Origem: os endereços IP de origem permitidos.

     Normalmente, você permite solicitações RDP de entrada de seu endereço IP ou de uma faixa especificada de endereços IP (normalmente a faixa de endereços IP de sua empresa). Para fins de aprendizado, normalmente é suficiente especificar 0.0.0.0/0, que permite o acesso RDP de qualquer endereço IP.

O grupo de segurança permite que a instância receba solicitações de conexão RDP, mas essa é apenas metade da história. Um usuário comum fará login na instância usando uma senha fornecida pelo AWS OpsWorks Stacks. Para que o AWS OpsWorks Stacks gere essa senha, você deve autorizar explicitamente o acesso RDP para o usuário.

Para autorizar o RDP para um usuário

1. No painel de AWS OpsWorks pilhas, escolha a IISWalkthroughpilha.
2. No painel de navegação da pilha, escolha Permissions.
3. Na página Permissions, escolha Edit.
4. Na lista de usuários, marque a caixa de seleção SSH/RDP para o usuário do ao qual você deseja conceder as permissões necessárias. Se você deseja que o usuário também tenha permissões de administrador, selecione sudo/admin.



5. Escolha Salvar.

O usuário pode obter uma senha e usá-la para efetuar o login na instância, conforme descrito posteriormente.

> **Note**
>
> Você também pode fazer login como Administrador. Para obter mais informações, consulte Login como administrador.

Etapa 2.3: Implementar um livro de receitas personalizado

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Apesar de uma pilha ser basicamente um contêiner de instâncias, você não adiciona instâncias diretamente à pilha. Você pode adicionar uma ou mais camadas, cada uma representando um grupo de instâncias relacionadas, e então adicionar instâncias às camadas.

Uma camada é basicamente um modelo que o AWS OpsWorks Stacks usa para criar um conjunto de EC2 instâncias da Amazon com a mesma configuração. Uma instância começa com uma versão base do sistema operacional, e a camada da instância executa uma variedade de tarefas na instância para implementar o esquema, que podem incluir:

- Criação de diretórios e arquivos
- Gerenciamento de usuários
- Instalação e configuração de software
- Iniciar ou interromper servidores
- Implantação de código de aplicativo e arquivos relacionados.

Uma camada executa tarefas em instâncias ao executar receitas do Chef, receitas, para abreviar. Uma receita é um aplicativo Ruby que usa uma linguagem específica de domínio (DSL) do Chef para descrever o estado final da instância. Com o AWS OpsWorks Stacks, cada receita geralmente é atribuída a um dos eventos do ciclo de vida da camada: instalação, configuração, implantação, desimplantação e desativação. Quando ocorre um evento de ciclo de vida em uma instância, o AWS

OpsWorks Stacks executa as receitas do evento para realizar as tarefas apropriadas. Por exemplo, o evento Setup ocorre após a conclusão da inicialização de uma instância. AWS OpsWorks Em seguida, o Stacks executa as receitas de instalação, que normalmente executam tarefas como instalar e configurar o software do servidor e iniciar serviços relacionados.

AWS OpsWorks O Stacks fornece a cada camada um conjunto de receitas integradas que executam tarefas padrão. Você pode estender a funcionalidade de uma camada ao implementar receitas personalizadas para realizar tarefas adicionais e atribuí-las aos eventos de ciclo de vida da camada. As pilhas do Windows são compatíveis com camadas personalizadas, que têm um conjunto mínimo de receitas que executam apenas algumas tarefas básicas. Para adicionar funcionalidade às suas instâncias do Windows, você deve implementar as receitas personalizadas para instalar software, implantar aplicativos e assim por diante. Este tópico descreve como criar uma camada personalizada simples para dar suporte a instâncias do IIS.

Tópicos

- Uma rápida introdução aos livros de receitas e às receitas
- Implementar uma receita para instalar e iniciar o IIS
- Ativar o livro de receitas personalizado

Uma rápida introdução aos livros de receitas e às receitas

Uma receita define um ou mais aspectos do estado esperado de uma instância: quais diretórios ela deve ter, quais pacotes de software devem estar instalados, quais aplicativos devem ser implantados e assim por diante. As receitas são empacotadas em um livro de receitas, que normalmente contém uma ou mais receitas relacionadas, além de arquivos associados, como modelos para a criação de arquivos de configuração.

Este tópico é uma introdução básica às receitas, apenas o suficiente para mostrar como implementar um livro de receitas para dar suporte a uma camada do IIS personalizada simples. Para obter uma introdução mais geral aos livros de receitas, consulte Livros de receitas e receitas. Para um tutorial detalhado de introdução à implementação de livros de receitas, incluindo alguns tópicos específicos do Windows, consulte Introdução a livros de receitas.

Tecnicamente, as receitas do Chef são aplicativos Ruby, mas a maior parte do código está no Chef DSL. O DSL é composto principalmente de um conjunto de recursos, que você pode usar para especificar de forma declarativa um aspecto do estado das instâncias. Por exemplo, um `directory` recurso define um diretório a ser adicionado ao sistema. O exemplo a seguir define um diretório `C:`

`\data` com direitos de controle total que pertence ao usuário especificado e não herda direitos do diretório pai.

```
directory 'C:\data' do
  rights :full_control, 'WORKGROUP\username'
  inherits false
  action :create
end
```

Quando o Chef executa uma receita, ele executa cada recurso ao passar os dados para um provedor associado, um objeto Ruby que lida com os detalhes da modificação do estado da instância. Neste caso, o fornecedor cria um novo diretório com a configuração especificada.

O livro de receitas personalizado para a camada personalizada do IIS deve realizar as seguintes tarefas:

- Instalar o recurso IIS e iniciar o serviço.

  Você normalmente realiza essa tarefa durante a instalação, logo após o término da inicialização da instância.
- Implantar um aplicativo na instância, uma página HTML simples para este exemplo.

  Você normalmente realiza esta tarefa durante a instalação. No entanto, os aplicativos geralmente precisam ser atualizados regularmente e, portanto, você também precisa implantar as atualizações enquanto a instância está online.

Você poderia ter uma única receita para realizar todas essas tarefas. No entanto, a abordagem preferencial é ter receitas separadas para as tarefas de instalação e de implantação. Dessa maneira, você pode implantar as atualizações de aplicativo a qualquer momento, sem que seja necessário executar o código de instalação também. A seguir, como configurar um livro de receitas para dar suporte a uma camada personalizada do IIS. Os tópicos subsequentes mostrarão como implementar as receitas.

Para começar

1. Crie um diretório chamado `iis-cookbook` em um local conveniente na sua estação de trabalho.
2. Adicione um arquivo `metadata.rb` com o seguinte conteúdo a `iis-cookbook`.

```
name "iis-cookbook"
version "0.1.0"
```

Este exemplo usa um `metadata.rb` mínimo. Para obter mais informações sobre como você pode usar este arquivo, consulte metadata.rb.

3. Adicione um diretório `recipes` a `iis-cookbook`.

   Esse diretório, que deve ser nomeado como `recipes`, contém as receitas do livro de receitas.

Em geral, os livros de receitas podem conter uma variedade de outros diretórios. Por exemplo, se uma receita usar um modelo para criar um arquivo de configuração, o modelo geralmente ficará no diretório `templates\default`. O livro de receitas para este exemplo consiste inteiramente em receitas e, portanto, não precisará de outros diretórios. Além disso, este exemplo usa um único livro de receitas, mas você pode usar a quantidade necessária; com frequência, é preferível usar vários livros de receitas para projetos complexos. Por exemplo, você pode ter livros de receitas separados para as tarefas de instalação e de implementação. Para obter mais exemplos de livros de receitas, consulte Livros de receitas e receitas.

Implementar uma receita para instalar e iniciar o IIS

O IIS é um recurso do Windows, uma de um conjunto de componentes opcionais do sistema que você pode instalar no Windows Server. Você pode fazer uma receita instalar o IIS de uma das seguintes formas:

- Usando um recurso `powershell_script` para executar o cmdlet `Install-WindowsFeature`.
- Usando o recurso livro de receitas do Windows `windows_feature` do Chef.

  O livro de receitas do `windows` contém um conjunto de recursos cujos provedores usam o Deployment Image Servicing and Management (DISM) para realizar uma variedade de tarefas em instâncias do Windows, incluindo a instalação de recursos.

> **Note**
>
> O `powershell_script` está entre os recursos mais úteis para receitas do Windows. Você pode usá-lo para realizar várias tarefas em uma instância executando um PowerShell script

ou cmdlet. Ele é especialmente útil para as tarefas que não são compatíveis com recursos do Chef.

Este exemplo executa um PowerShell script para instalar e iniciar o Servidor Web (IIS). O livro de receitas do `windows` será descrito mais tarde. Para obter um exemplo de como usar o `windows_feature` para instalar o IIS, consulte Instalar um recurso do Windows: IIS.

Adicione uma receita chamada `install.rb` com o conteúdo a seguir ao diretório `recipes` do livro de receitas.

```
powershell_script 'Install IIS' do
  code 'Install-WindowsFeature Web-Server'
  not_if "(Get-WindowsFeature -Name Web-Server).Installed"
end

service 'w3svc' do
  action [:start, :enable]
end
```

A receita contém dois recursos.

powershell_script

`powershell_script`executa o PowerShell script ou cmdlet especificado. O exemplo tem as seguintes configurações de atributo:

- `code`— Os PowerShell cmdlets a serem executados.

  Este exemplo executa um cmdlet `Install-WindowsFeature`, que instala o servidor web (IIS). Em geral, o atributo `code` pode ter qualquer número de linhas para que você possa executar quantos cmdlets precisar.

- `not-if`: um atributo guard que garante que a receita só instalará o IIS se ele ainda não tiver sido instalado.

  Você geralmente deseja que as receitas sejam idempotentes, de forma que elas não percam tempo realizando a mesma tarefa mais de uma vez.

Cada recurso tem uma ação, que especifica a ação que o provedor deve realizar. Não há nenhuma ação explícita neste exemplo, então o provedor executa a `:run` ação padrão, que

executa o PowerShell script especificado. Para obter mais informações, consulte Executando um PowerShell script do Windows.

serviço

Um `service` gerencia um serviço, neste caso, o serviço Servidor web IIS (W3SVC). O exemplo usa atributos padrão e especifica duas ações, `:start` e `:enable`, as quais iniciam e ativam o IIS.

> **Note**
>
> Se você quiser instalar software que use um instalador de pacotes, como o MSI, use um recurso `windows_package`. Para obter mais informações, consulte Instalar um pacote.

Ativar o livro de receitas personalizado

AWS OpsWorks O Stacks executa receitas de um cache local em cada instância. Para executar suas receitas personalizadas, você deve fazer o seguinte:

- Armazene o livro de receitas em um repositório remoto.

  AWS OpsWorks O Stacks baixa os livros de receitas desse repositório para o cache local de cada instância.
- Edite a pilha para ativar os livros de receitas personalizados.

  Os livros de receitas personalizados estão desativados por padrão; portanto, você deve ativar os livros de receitas personalizados para a pilha e fornecer o URL do repositório e as informações relacionadas.

AWS OpsWorks O Stacks oferece suporte a arquivos S3 e repositórios Git para livros de receitas personalizados; este exemplo usa um arquivo S3. Para obter mais informações, consulte Repositórios de livro de receitas.

Para usar um arquivo do S3

1. Crie um arquivo `.zip` do diretório `iis-cookbook`.

AWS OpsWorks As pilhas também oferecem suporte a arquivos `.tgz` (tar compactado com gzip) para pilhas do Windows.

2. Faça upload do arquivo para um bucket do S3 na região Oeste dos EUA (N. da Califórnia) e torne o arquivo público. Você também pode usar arquivos privados do S3, mas os arquivos públicos são suficientes para este exemplo e um pouco mais simples de trabalhar.

   a. Faça login no AWS Management Console e abra o console do Amazon S3 em. https://console.aws.amazon.com/s3/

   b. Se você ainda não tiver um bucket na `us-west-1`, escolha Criar bucket e crie um bucket na região Oeste dos EUA (N. da Califórnia).

   c. Na lista de buckets, clique no nome do bucket para o qual você deseja fazer upload do arquivo e clique em Upload (Fazer upload).

   d. Escolha Adicionar arquivos.

   e. Selecione o arquivo para fazer upload e escolha Open (Abrir).

   f. Na parte inferior da caixa de diálogo Upload - Select Files and Folders (Fazer upload — Selecionar arquivos e pastas), escolha Set Details (Definir detalhes).

   g. Na parte inferior da caixa de diálogo Set Details, escolha Set Permissions.

   h. Na caia de diálogo Set Permissions, escolha Make everything public.

   i. Na parte inferior da caixa de diálogo Set Permissions (Definir permissões), escolha Start Upload (Iniciar upload). Quando o upload for concluído, o arquivo `iis-cookbook.zip` será exibido em seu bucket.

   j. Escolha o bucket e escolha a guia Properties (Propriedades) para o bucket. Ao lado de Link, registre o URL do arquivo para uso posterior.

   Para obter mais informações sobre o upload de arquivos para um bucket do Amazon S3, consulte Como fazer upload de arquivos e pastas para um bucket do S3? no Guia do usuário do console do Amazon S3.

> **Important**
>
> Até agora, a demonstração só custou a você um pouco de tempo, o serviço AWS OpsWorks Stacks é gratuito. No entanto, você deverá pagar por qualquer recurso da AWS que utilizar,

como o armazenamento no Amazon S3. Assim que você fizer o upload do arquivo, começará a ser cobrado. Para obter mais informações, consulte Definição de preço da AWS.

Para ativar os livros de receitas personalizados para a pilha

1. No console AWS OpsWorks Stacks, escolha Stack no painel de navegação e, em seguida, escolha Stack Settings no canto superior direito.
2. Na parte superior direita da página Settings, escolha Edit.
3. Na página Settings, defina Use custom Chef cookbooks como Yes e insira as seguintes informações:
   - Tipo de repositório: Arquivamento do S3.
   - URL do repositório: o URL do S3 do arquivo do livro de receitas registrado anteriormente.
4. Escolha Save (Salvar) para atualizar a configuração da pilha.

AWS OpsWorks O Stacks instala seu livro de receitas personalizado em todas as novas instâncias. Observe que o AWS OpsWorks Stacks não instala ou atualiza automaticamente os livros de receitas personalizadas em instâncias online. Você pode fazer isso manualmente, conforme descrito posteriormente.

Etapa 2.4: Adicionar uma camada do IIS

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Seu livro de receitas tem uma receita que simplesmente instala e inicia o IIS. Isso é o suficiente para criar a camada e verificar se você tem uma instância do IIS em funcionamento. Posteriormente, você adicionará a funcionalidade de implantação de aplicativos à camada.

Criar uma camada

Você começa pela adição de uma camada à pilha. Então, você pode adicionar funcionalidade à layer ao atribuir receitas personalizadas aos eventos de de ciclo de vida apropriados.

Para adicionar uma camada do IIS à pilha

1. No painel de navegação, escolha Layers e depois Add a layer.
2. Configure a camada da seguinte forma:
   - Nome: **`IISExample`**
   - Nome curto: **`iisexample`**

     AWS OpsWorks Stacks usa o nome curto para identificar a camada internamente. Você também pode usar o nome curto para identificar a camada em receitas, embora este exemplo não faça isso. Você pode especificar qualquer nome curto, mas ele só poderá conter caracteres alfanuméricos minúsculos e um pequeno número de sinais de pontuação. Para obter mais informações, consulte Camadas personalizadas.
3. Escolha Add Layer.

Se você adicionasse uma instância nesse IISWalkthrough momento e a iniciasse, o AWS OpsWorks Stacks instalaria automaticamente os livros de receitas, mas não funcionaria. `install.rb` Depois que uma instância estiver online, você poderá executar as receitas manualmente usando o comando de pilha Executar Receitas. No entanto, uma abordagem melhor é atribuir a receita a um dos eventos do ciclo de vida da camada. AWS OpsWorks Em seguida, o Stacks executa automaticamente a receita no ponto apropriado do ciclo de vida da instância.

Instale e inicie o IIS assim que a inicialização da instância for concluída. Para fazer isso, atribua `install.rb` ao evento `Setup` da camada.

Para atribuir a receita a um evento de ciclo de vida

1. No painel de navegação, escolha Layers
2. Na caixa da IISExamplecamada, escolha Receitas.
3. No canto superior direito, escolha Edit (Editar).
4. Em Custom Chef Recipes (Receitas do Chef personalizadas), na caixa de receitas Setup (instalação), digite **`iis-cookbook::install`**.

> **Note**
>
> Use *`cookbook-name`*`::`*`recipe-name`* para identificar receitas, em que o sufixo `.rb` do nome da receita é omitido.

5. Escolha + para adicionar a receita à camada. Um x vermelho aparece ao lado da receita para facilitar sua remoção posteriormente.
6. Escolha Save para salvar a nova configuração. O Configurar receitas personalizado agora deve incluir`iis-cookbook::install`.

Adicionar uma instância à camada e iniciá-la

Você pode testar a receita adicionando uma instância à camada e iniciando a instância. AWS OpsWorks O Stacks instala automaticamente os livros de receitas e é executado `install.rb` durante a configuração, assim que a instância termina de inicializar.

Para adicionar uma instância a uma camada e iniciá-la

1. No painel de navegação AWS OpsWorks Stacks, escolha Instances.
2. Em IISExamplecamada, escolha Adicionar uma instância.
3. Selecione o tamanho apropriado.. T2.micro (ou o menor tamanho disponível para você) deve ser suficiente para o exemplo.
4. Escolha Add Instance. Por padrão, o AWS OpsWorks Stacks gera nomes de instância anexando um inteiro ao nome curto da camada, então a instância deve ser chamada iisexample1.
5. Escolha iniciar na coluna Ações da instância para iniciar a instância. AWS OpsWorks Em seguida, as pilhas iniciarão uma EC2 instância e executarão as receitas de configuração para configurá-la. Se a camada tivesse alguma receita de implantação neste momento, AWS OpsWorks as pilhas as executariam após a conclusão das receitas de configuração.

   O processo pode levar vários minutos, durante o qual a coluna Status exibirá uma série de estados de status. Quando você chegar ao status online, o processo de instalação estará completo e a instância estará pronta para uso.

Confirmar se o IIS está instalado e em execução

Você pode usar o RDP para se conectar à instância e verificar se a sua receita de Instalação funcionou corretamente.

Para verificar se o IIS está instalado e em execução

1. Escolha Instâncias no painel de navegação e escolha rdp na coluna Ações da instância iisexample1. AWS OpsWorks O Stacks gera automaticamente uma senha RDP para você que expira após um período de tempo especificado.
2. Defina Session valid for como 2 horas e escolha Generate Password.
3. AWS OpsWorks O Stacks exibe a senha e também, para sua conveniência, o nome DNS público e o nome de usuário da instância. Copie os três e clique em Acknowledge and close.
4. Abra o cliente RDP e use os dados do Etapa 3 para se conectar à instância.
5. Na instância, abra o Windows Explorer e examine a unidade `C:`. Ela deve ter um diretório `C:\inetpub`, que foi criado pela instalação do IIS.
6. Abra o aplicativo Administrative Tools do Painel de Controle e abra Services. Você deve ver o serviço IIS na parte inferior da lista. Ele se chama World Wide Web Publishing Service e o status deve ser running.
7. Volte ao console do AWS OpsWorks Stacks e escolha o endereço IP público da instância iisexample1. Certifique-se de fazer isso no AWS OpsWorks Stacks, e não no EC2 console da Amazon. Isso envia automaticamente uma solicitação HTTP para o endereço, que deve abrir a página de boas-vindas padrão do IIS.

O próximo tópico discute como implantar um aplicativo na instância, uma simples página HTML estática para este exemplo. No entanto, se você quiser fazer uma pausa, escolha stop na coluna Actions da instância iisexample1 para interromper a instância e evitar incorrer em cobranças desnecessárias. Você poderá reiniciar a instância quando estiver pronto para continuar.

Etapa 2.5: Implantar um aplicativo

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A instalação do IIS cria um diretório `C:\inetpub\wwwroot` para o código do seu aplicativo e os arquivos relacionados. A próxima etapa é instalar um aplicativo nesse diretório. Para este exemplo, você instalará uma página inicial HTML estática, `default.html`, em `C:\inetpub\wwwroot`. Você pode facilmente estender a abordagem geral para lidar com cenários mais complexos, como aplicativos ASP.NET.

Você poderia incluir os arquivos do aplicativo em seu livro de receitas e fazer `install.rb` copiá-los para `C:\inetpub\wwwroot`. Para obter exemplos de como fazer isso, consulte Exemplo 6: Criação de arquivos. No entanto, essa abordagem não é muito flexível ou eficiente, e normalmente é melhor para separar o desenvolvimento do livro de receitas do desenvolvimento de aplicativos.

A melhor solução é implementar uma receita de implantação separada que recupere o código do aplicativo e os arquivos relacionados de um repositório, qualquer repositório que você preferir, e não apenas o repositório do livro de receitas; e o instale em cada instância de servidor IIS. Essa abordagem separa o desenvolvimento de receitas do desenvolvimento de aplicativos e, quando você precisar atualizar o aplicativo, ela permite que você simplesmente execute a receita de implantação novamente sem precisar atualizar seus livros de receitas.

Este tópico mostra como implementar uma receita de implantação simples que implanta `default.htm` em seu servidor IIS. Você pode facilmente estender este exemplo para aplicativos mais complexos.

Tópicos

- Criar o aplicativo e armazená-lo em um repositório
- Implemente uma receita para implantar o aplicativo
- Atualizar os livros de receitas da instância
- Adicionar a receita à camada personalizada do IIS
- Adicionar um aplicativo
- Implantar o aplicativo e abrir o aplicativo

Criar o aplicativo e armazená-lo em um repositório

Você pode usar qualquer repositório que preferir para seus aplicativos. Para simplificar, este exemplo armazena `default.htm` em um bucket público do S3.

Para criar o aplicativo

1. Crie um diretório chamado `iis-application` em um local conveniente na sua estação de trabalho.
2. Adicione um arquivo `default.htm` a `iis-application`, com o seguinte conteúdo.

```
<!DOCTYPE html>
<html>
  <head>
    <title>IIS Example</title>
  </head>
  <body>
    <h1>Hello World!</h1>
  </body>
</html>
```

3. Crie um bucket do S3, faça upload de `default.htm` para o bucket e registre o URL para uso posterior. Para simplificar, torne o arquivo público.

> **Note**
>
> Este é um aplicativo muito simples, mas você pode estender os princípios básicos para lidar com aplicativos de nível de produção.
>
> - Para aplicativos mais complexos com vários arquivos, geralmente é mais simples criar um arquivo .zip de `iis-application` e fazer o upload dele para seu bucket do S3.
>
> Você poderá então fazer download do arquivo .zip e extrair o conteúdo para o diretório apropriado. Não há necessidade de fazer download de vários arquivos, criar uma estrutura de diretório e assim por diante.
> - Para um aplicativo de produção, provavelmente será melhor manter seus arquivos privados. Para obter um exemplo de como fazer com que uma receita faça download dos arquivos de um bucket privado do S3, consulte Usando o SDK para Ruby em uma instância do Windows AWS OpsWorks Stacks.
> - Você pode armazenar seu aplicativo em qualquer repositório adequado.

> Você normalmente faz download do aplicativo usando uma API pública de um repositório. Este exemplo usa a API do Amazon S3. Se, por exemplo, você armazenar seu aplicativo em GitHub, poderá usar a GitHub API.

Implemente uma receita para implantar o aplicativo

Adicione uma receita chamada `deploy.rb` ao diretório `iis-cookbook` `recipes` com o seguinte conteúdo.

```
chef_gem "aws-sdk-s3" do
  compile_time false
  action :install
end

ruby_block "download-object" do
  block do
    require 'aws-sdk-s3'

    #1
    # Aws.config[:ssl_ca_bundle] = 'C:\ProgramData\Git\bin\curl-ca-bundle.crt'
    Aws.use_bundled_cert!

    #2
    query = Chef::Search::Query.new
    app = query.search(:aws_opsworks_app, "type:other").first
    s3region = app[0][:environment][:S3REGION]
    s3bucket = app[0][:environment][:BUCKET]
    s3filename = app[0][:environment][:FILENAME]

    #3
    s3_client = Aws::S3::Client.new(region: s3region)
    s3_client.get_object(bucket: s3bucket,
                         key: s3filename,
                         response_target: 'C:\inetpub\wwwroot\default.htm')
  end
  action :run
end
```

Este exemplo usa SDK para Ruby v2 para fazer download do arquivo. No entanto, o AWS OpsWorks Stacks não instala esse SDK em instâncias do Windows, então a receita começa com o `chef_gem`recurso, que processa essa tarefa.

> **Note**
>
> O recurso `chef_gem` instala gems na versão do Ruby dedicada do Chef, que é a versão usada pelas receitas. Se você quiser instalar um gem para uma versão do Ruby para todo o sistema, use o recurso gem_package.

O pacote da receita é um recurso `ruby_block`, que executa um bloco de código Ruby que usa o SDK para Ruby para fazer download de `default.htm`. O código no `ruby_block` pode ser dividido nas seguintes seções, que correspondem aos comentários numerados no exemplo de código.

1: Especificar um pacote de certificados

O Amazon S3 usa SSL, portanto, você precisa de um certificado apropriado para baixar os objetos de um bucket do S3. O SDK para Ruby v2 não inclui um pacote de certificados, então você deve fornecer um e configurar o SDK para Ruby para usá-lo. AWS OpsWorks O Stacks não instala um pacote de certificados diretamente, mas instala o Git, que inclui um pacote de certificados (). `curl-ca-bundle.crt` Para sua conveniência, este exemplo configura o SDK para Ruby para usar o pacote de certificados Git para SSL. Você também pode instalar seu próprio pacote e configurar o SDK apropriadamente.

2: Recuperar os dados do repositório

Para fazer download de um objeto do Amazon S3, você precisa da região da AWS, do nome do bucket e do nome da chave. Conforme descrito posteriormente, este exemplo fornece essas informações ao associar um conjunto de variáveis de ambiente ao aplicativo. Quando você implanta um aplicativo, o AWS OpsWorks Stacks adiciona um conjunto de atributos ao objeto node da instância. Esses atributos são basicamente uma tabela de hash que contém a configuração do aplicativo, incluindo as variáveis de ambiente. Os atributos de aplicativo para esse aplicativo serão semelhantes aos seguintes, no formato JSON.

```
{
  "app_id": "8f71a9b5-de7f-451c-8505-3f35086e5bb3",
  "app_source": {
      "password": null,
```

```
        "revision": null,
        "ssh_key": null,
        "type": "other",
        "url": null,
        "user": null
    },
    "attributes": {
        "auto_bundle_on_deploy": true,
        "aws_flow_ruby_settings": {},
        "document_root": null,
        "rails_env": null
    },
    "data_sources": [{"type": "None"}],
    "domains": ["iis_example_app"],
    "enable_ssl": false,
    "environment": {
        "S3REGION": "us-west-2",
        "BUCKET": "windows-example-app",
        "FILENAME": "default.htm"
    },
    "name": "IIS-Example-App",
    "shortname": "iis_example_app",
    "ssl_configuration": {
        "certificate": null,
        "private_key": null,
        "chain": null
    },
    "type": "other",
    "deploy": true
}
```

As variáveis de ambiente do aplicativo são armazenadas no atributo `[:environment]`. Para recuperá-las, use uma consulta de pesquisa do Chef para recuperar a tabela de hash do aplicativo, que está no nó `aws_opsworks_app`. Este aplicativo será definido como o tipo `other`, para que a consulta pesquise aplicativos desse tipo. A receita aproveita o fato de que há apenas um aplicativo nesta instância e, portanto, a tabela de hash de interesse é simplesmente `app[0]`. Para sua conveniência, a receita então atribui a região, o bucket e os nomes de arquivo às variáveis.

Para obter mais informações sobre como usar o Chef, consulte.Obter valores de atributo com a pesquisa do Chef

3: Fazer download do arquivo

A terceira parte da receita cria um objeto de cliente do S3 e usa seu método `get_object` para fazer download de `default.htm` para o diretório `C:\inetpub\wwwroot` da instância.

> **Note**
>
> Uma receita é um aplicativo Ruby e, portanto, o código Ruby não precisa estar necessariamente em um `ruby_block`. No entanto, o código no corpo da receita é executado primeiro, seguido pelos recursos, na ordem. Para este exemplo, se você colocar o código de download no corpo da receita, ele falhará porque o recurso `chef_gem` ainda não terá instalado o SDK para Ruby. O código no recurso `ruby_block` é executado quando o recurso é executado, depois que o recurso `chef_gem` tiver instalado o SDK para Ruby.

Atualizar os livros de receitas da instância

AWS OpsWorks O Stacks instala automaticamente livros de receitas personalizados em novas instâncias. No entanto, você está trabalhando com uma instância existente e, portanto, deve atualizar seu livro de receitas manualmente.

Para atualizar os livros de receitas da instância

1. Crie um arquivo `.zip` de `iis-cookbook` e faça upload dele em um bucket do S3.

   Isso substituirá o livro de receitas existente, mas o URL permanecerá o mesmo, portanto, não será necessário atualizar a configuração da pilha.
2. Se a sua instância não estiver online, reinicie-a.
3. Depois que a instância estiver online, escolha Stack no painel de navegação e, em seguida, Run Command.
4. Para Command, escolha Update Custom Cookbooks. Esse comando instala o livro de receitas atualizado na instância.
5. Escolha Update Custom Cookbooks. Esse comando pode levar alguns minutos para ser concluído.

Adicionar a receita à camada personalizada do IIS

Assim como ocorre com `install.rb`, a forma preferencial de lidar com a implantação é atribuir `deploy.rb` ao evento de ciclo de vida apropriado. Em geral, você atribui as receitas de implantação ao evento Implantar, e elas são chamadas coletivamente como Implantar receitas. A atribuição de uma receita ao evento implantar não aciona o evento. Em vez disso:

- Para novas instâncias, o AWS OpsWorks Stacks executa automaticamente as receitas de implantação após a conclusão das receitas de configuração, para que as novas instâncias tenham automaticamente a versão atual do aplicativo.
- Para as instâncias online, você usa um comando de implantação para instalar manualmente os aplicativos novos ou atualizados.

  Este comando aciona um evento de Implantação nas instâncias da pilha, que executa o Implantar receitas.

Para atribuir deploy.rb ao evento Implantar da camada

1. Escolha Camadas no painel de navegação e, em seguida, escolha Receitas em IISExampleCamada.
2. Em Custom Chef Recipes (Receitas do Chef personalizadas), adicione **`iis-cookbook::deploy`** à caixa de receitas Deploy (Implantação) e escolha + para adicionar a receita à camada.
3. Escolha Save para salvar a nova configuração. O Implantar receitas personalizado agora deve incluir `iis-cookbook::deploy`.

Adicionar um aplicativo

A tarefa final é adicionar um aplicativo à pilha para representar seu aplicativo no ambiente AWS OpsWorks Stacks. Um aplicativo inclui metadados, como o nome de exibição do aplicativo, e os dados necessários para fazer o download do aplicativo de seu repositório.

Para adicionar o aplicativo à pilha

1. No painel de navegação, escolha Apps (Aplicativos) e depois Add an app (Adicionar um aplicativo).
2. Configure o aplicativo com as definições a seguir.

- Nome: I**IIS-Example-App**
- Tipo de repositório: outro
- Variáveis de ambiente: adicione estas três variáveis de ambiente:
  - **S3REGION**: a região do bucket (neste caso, `us-west-1`).
  - **BUCKET**: o nome do bucket, como `windows-example-app`.
  - **FILENAME**: o nome do arquivo: **default.htm**.

3. Aceite os valores padrão para as demais configurações e escolha Add App (Adicionar aplicativo) para adicionar o aplicativo à pilha.

> **Note**
>
> Este exemplo usa variáveis de ambiente para fornecer os dados de download. Uma abordagem alternativa é usar um tipo de repositório do S3 Archive e fornecer a URL do arquivo. AWS OpsWorks O Stacks adiciona as informações, junto com dados opcionais, como suas credenciais da AWS, ao atributo do `app_source` aplicativo. Sua receita de implantação deve obter o URL dos atributos do aplicativo e analisá-lo para extrair a região, o nome do bucket e o nome do arquivo.

Implantar o aplicativo e abrir o aplicativo

AWS OpsWorks O Stacks implanta aplicativos automaticamente em novas instâncias, mas não em instâncias on-line. Como sua instância já está em execução, será necessário implantar o aplicativo manualmente.

Para implantar o aplicativo

1. Escolha Apps (Aplicativos) no painel de navegação e escolha deploy (implantar) na coluna Actions (Ações) do aplicativo.
2. Command (Comando) deve ser definido como Deploy (Implantar). Escolha Deploy (Implantar) no canto inferior direito da página Deploy App (Implantar aplicativo). Esse comando pode levar alguns minutos para ser concluído.

   Depois que a implantação for concluída, você retornará para a página Apps (Aplicativos). O indicador Status mostrará successful (bem-sucedido) em verde, e o nome do aplicativo terá uma marca de verificação verde próxima a ele para indicar uma implantação bem-sucedida.

> **Note**
>
> Os aplicativos do Windows são sempre o tipo de aplicativo Other (Outros) e, portanto, a implantação do aplicativo faz o seguinte:
>
> - Adiciona os dados do aplicativo aos atributos de configuração e de implantação da pilha, conforme descrito anteriormente.
> - Aciona um evento Implantar nas instâncias da pilha, que executa seu Implantar receitas personalizado.

> **Note**
>
> Para obter mais informações sobre como solucionar falhas nas implantações ou nos aplicativos, consulte Depurar receitas.

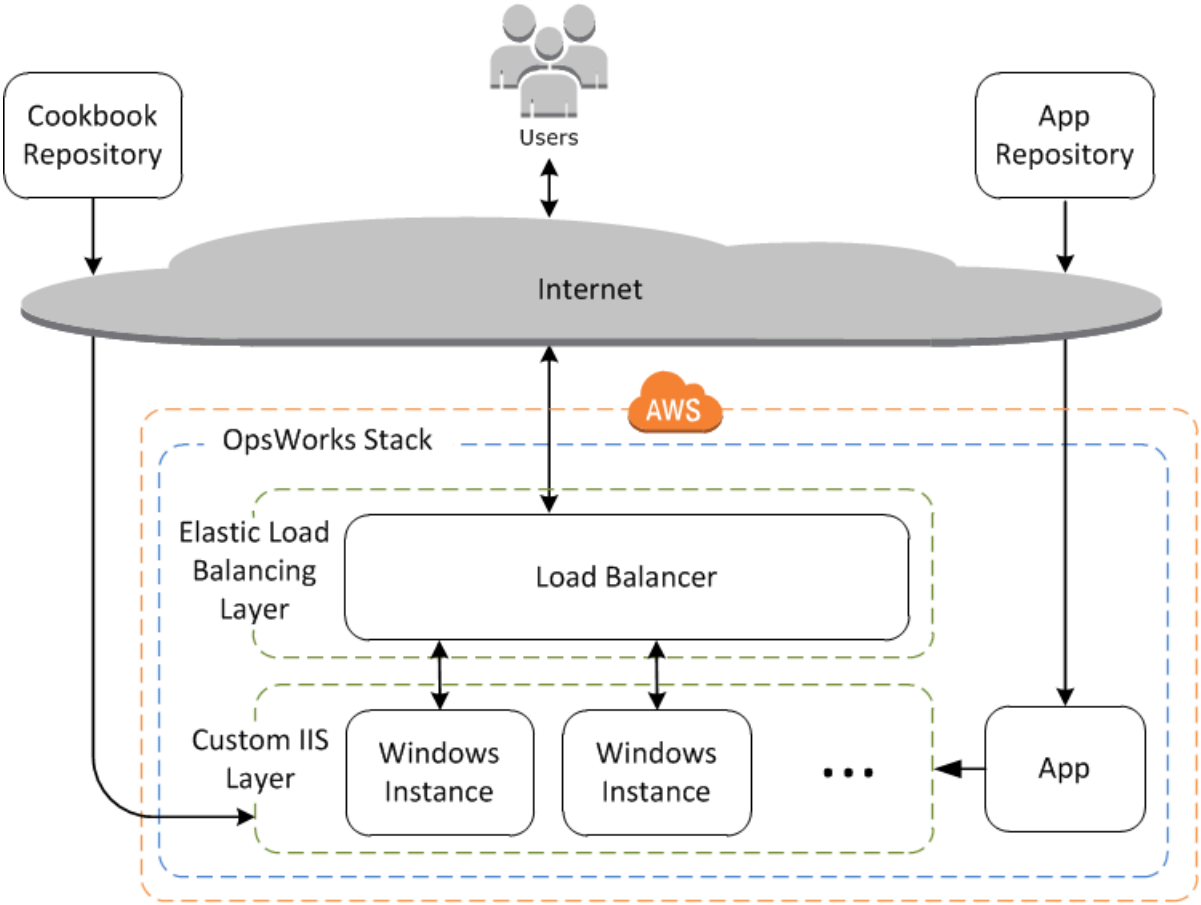
O aplicativo está instalado. Você pode abri-lo ao escolher Instances no painel Navigation e, em seguida, o endereço IP público da instância. Isso envia uma solicitação HTTP para a instância, e você verá em seu navegador algo como o mostrado a seguir.

Hello World!

Etapa 3: Dimensionar IISExample

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Se as suas solicitações de usuário de entrada começarem a se aproximar do limite do que você pode tratar com uma única instância t2.micro, será necessário aumentar sua capacidade de servidor. Você pode mudar a uma instância maior, mas isso tem limites. Uma abordagem mais flexível é adicionar instâncias ao seu pilha e colocá-las atrás de um load balancer. A arquitetura básica é semelhante à seguinte.



Entre outras vantagens, essa abordagem é muito mais robusta do que uma única instância grande.

- Se uma das suas instâncias falhar, o load balancer distribuirá as solicitações de entrada para as instâncias restantes e o aplicativo continuará a funcionar.
- Se você colocar as instâncias em diferentes Zonas de disponibilidade (a melhor prática), o aplicativo continuará a funcionar mesmo se uma Zona de disponibilidade tiver problemas.

AWS OpsWorks As pilhas facilitam a escalabilidade das pilhas. Esta seção descreve as noções básicas de como escalar uma pilha adicionando uma segunda instância do PHP App Server 24 horas por dia, 7 dias por semana, IISExample e colocando ambas as instâncias atrás de um load balancer do Elastic Load Balancing. Você pode estender facilmente o procedimento para adicionar

um número arbitrário de instâncias 24 horas por dia, 7 dias por semana, ou pode usar instâncias baseadas em tempo para que o AWS OpsWorks Stacks escale sua pilha automaticamente. Para obter mais informações, consulte Gerenciamento de carga com instâncias baseadas em tempo e carga.

Adicionar um load balancer

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O Elastic Load Balancing é um serviço da AWS que distribui automaticamente o tráfego de entrada do aplicativo em várias instâncias da Amazon. EC2 Um load balancer pode servir a dois propósitos. O óbvio é equalizar a carga em seus servidores de aplicativos. Muitos sites preferem isolar seus servidores de aplicativos e bancos de dados do acesso direto do usuário. Além de distribuir o tráfego, o Elastic Load Balancing faz o seguinte:

- Detecta instâncias insalubres da Amazon EC2 .

  Ele redireciona o tráfego para as instâncias íntegras restantes até que as instâncias com problemas sejam restauradas.
- Escala automaticamente solicitação de manuseio de capacidade em resposta a tráfego de entrada.

> **Note**
>
> AWS OpsWorks O Stacks não é compatível com o Application Load Balancer. Você só pode usar o Classic Load Balancer com o AWS OpsWorks Stacks.

Embora o Elastic Load Balancing normalmente seja mencionado como uma camada, ele funciona de uma forma um pouco diferente das outras camadas integradas. Em vez de criar uma camada e adicionar instâncias a ela, você cria um balanceador de carga do Elastic Load Balancing usando o EC2 console da Amazon e o anexa a uma de suas camadas existentes, geralmente uma camada de

servidor de aplicativos. AWS OpsWorks Em seguida, o Stacks registra as instâncias existentes da camada com o serviço e adiciona automaticamente quaisquer novas instâncias. O procedimento a seguir descreve como adicionar um load balancer.

Para anexar um load balancer à camada personalizada do IIS

1. Use o EC2 console da Amazon para criar um novo balanceador de carga para IISExample. Para obter mais informações, consulte Conceitos básicos do Elastic Load Balancing. Quando você executar o assistente do Create Load Balancer, configure o load balancer como a seguir:

   1: Define Load Balancer

   Atribua ao balanceador de carga um nome facilmente reconhecível, como IIS-LB, para facilitar a localização no console Stacks. AWS OpsWorks Aceite os valores padrão para as outras configurações e escolha Next: Assign Security Groups.

   2: Assign Security Groups

   Se a sua conta for compatível com a VPC padrão, o assistente exibirá esta página para determinar o grupo de segurança do load balancer. Ela não exibe esta página do EC2 Classic.

   Para este passo-a-passo, especifique default VPC security group e escolha Next: Configure Security Settings.

   3: Configure Security Settings

   Esta demonstração requer que o load balancer use um ouvinte seguro (ou seja, HTTPS ou SSL em sua conexão front-end); portanto, escolha Next: Configure Health Check para continuar.

   4: Configure Health Check

   Defina o caminho de ping como **/**. Aceite os padrões para as configurações restantes e escolha Avançar: Adicionar EC2 instâncias.

   5: Adicionar EC2 instâncias

   AWS OpsWorks O Stacks cuida automaticamente do registro de instâncias com o balanceador de carga. Escolha Next Add Tags para continuar.

   6: Add Tags

   Você não usará as tags para este exemplo. Selecione Review and Create.

7: Review

Revise suas escolhas e selecione Create e Close, que inicia o load balancer.

2. Se a sua conta for compatível com a VPC padrão, depois que você iniciar o load balancer, deverá garantir que seu grupo de segurança tenha as regras de entrada apropriadas. A regra padrão não aceita tráfego de entrada.

   1. Escolha Security Groups no painel de EC2 navegação da Amazon.
   2. Escolha default VPC security group
   3. Na guia Entrada, escolha Editar.
   4. Para esta demonstração, defina Source como Anywhere, que direciona o load balancer a aceitar tráfego de entrada de qualquer endereço IP.
   5. Clique em Salvar.
3. Volte para o console AWS OpsWorks Stacks. Na página Layers, selecione Network.
4. Na opção Elastic Load Balancing, selecione o load balancer IIS-LB criado na Etapa 1 e clique em Save.

   Depois de conectar o balanceador de carga à camada, o AWS OpsWorks Stacks registra automaticamente as instâncias atuais da camada e adiciona novas instâncias à medida que elas ficam on-line.
5. Na página Layers, clique no nome do load balancer para abrir sua página de detalhes. Uma marca de verificação verde ao lado da instância na página do load balancer indica que a instância passou por uma verificação de integridade.

Agora você pode executar IIS-Example-App enviando uma solicitação para o balanceador de carga.

Para executar IIS-Example-App o balanceador de carga

1. Escolha Layers. O load balancer IIS-ELB deve estar listado como uma camada e a coluna Integridade deve ter uma instância em verde, o que indica uma instância íntegra.
2. Escolha o nome DNS do balanceador de carga a ser executado. IIS-Example-App
   Ele deve estar listado no nome do load balancer e deve ser algo semelhante a `IIS-LB-1802910859.us-west-2.elb.amazonaws.com`. O load balancer encaminha a solicitação para a instância e retorna a resposta, que deve ser exatamente igual à resposta obtida quando você clica no endereço IP público da instância.

Você tem apenas uma instância neste ponto e, portanto, o load balancer não está realmente adicionando muito. No entanto, agora você pode adicionar outras instâncias à camada.

Para adicionar uma instância à camada

1. Escolha Instances e + instance para adicionar outra instância à camada e a inicie.
2. Inicie a instância.

Por serem novas instâncias, o AWS OpsWorks Stacks instala automaticamente os livros de receitas personalizados atuais e implanta a versão atual do aplicativo durante a configuração. Quando a instância fica on-line, o AWS OpsWorks Stacks a adiciona automaticamente ao balanceador de carga, para que sua instância comece imediatamente a lidar com as solicitações. Para verificar se o aplicativo ainda está funcionando, você pode escolher o nome DNS do load balancer novamente.

Próximas etapas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta demonstração apresentou as noções básicas da configuração de uma pilha simples de servidor de aplicativos do Windows. Aqui estão algumas sugestões sobre o que fazer a seguir.

- Se você quiser saber mais, Conceitos básicos: livros de receitas fornece um tutorial de introdução à implementação de livros de receitas e inclui vários exemplos específicos do AWS OpsWorks Stacks.
- É possível adicionar uma camada do Amazon Relational Database Service (Amazon RDS) à pilha para usar o servidor de banco de dados de back-end. Para obter informações sobre como conectar seu aplicativo ao banco de dados, consulte Usando uma receita predefinida.

Começando com livros de receitas em pilhas AWS OpsWorks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Uma pilha AWS OpsWorks Stacks em nível de produção normalmente requer alguma personalização, o que geralmente significa implementar um livro de receitas personalizado do Chef. Um livro de receitas é um arquivo de pacote que contém informações de configuração, incluindo instruções chamadas de receitas. Uma receita é um conjunto de uma ou mais instruções, escritas com sintaxe de linguagem Ruby, que especifica os recursos a serem usados e a ordem em que esses recursos são aplicados. Um recurso, como usado no Chef, é uma declaração de política de configuração. Este passo a passo fornece uma introdução básica à implementação dos livros de receitas do Chef para Stacks. AWS OpsWorks Para saber mais sobre Chef, livros de receitas, receitas e recursos, consulte os links em Próximas etapas.

Esta demonstração ensina como criar seus próprios livros de receitas. Você também pode usar receitas fornecidas pela comunidade disponível em sites como o Chef Supermarket. Para ajudar você a usar os livros de receitas da comunidade, incluímos instruções para usar um livro de receitas da comunidade do Chef Supermarket posteriormente nesta demonstração.

Antes de iniciar a demonstração, é necessário concluir algumas etapas de configuração. Se você já concluiu qualquer uma das outras demonstrações neste capítulo, como Conceitos básicos: exemplo, significa que já cumpriu os pré-requisitos para esta demonstração e poderá pular para a seção iniciar esta demonstração. Caso contrário, conclua os pré-requisitos e, em seguida, retorne a esta demonstração.

Tópicos

- Etapa 1: criar o livro de receitas
- Etapa 2: criar a pilha e seus componentes
- Etapa 3: executar e testar a receita
- Etapa 4: atualizar o livro de receitas para instalar um pacote

- Etapa 5: atualizar o livro de receitas na instância e executar a receita
- Etapa 6: atualizar o livro de receitas para adicionar um usuário
- Etapa 7: atualizar o livro de receita para criar um diretório
- Etapa 8: atualizar o livro de receitas para criar e copiar arquivos
- Etapa 9: atualizar o livro de receitas para executar um comando
- Etapa 10: atualizar o livro de receitas para executar um script
- Etapa 11: atualizar o livro de receitas para gerenciar um serviço
- Etapa 12: atualizar o livro de receitas para usar JSON personalizado
- Etapa 13: atualizar o livro de receitas para usar data bags
- Etapa 14: atualizar o livro de receitas para usar iteração
- Etapa 15: atualizar o livro de receitas para usar lógica condicional
- Etapa 16: atualizar o livro de receitas para usar livros de receitas da comunidade
- Etapa 17: limpar (opcional)
- Próximas etapas

Etapa 1: criar o livro de receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Comece criando um livro de receitas. Esse livro de receitas não fará muito para começar, mas servirá como base para o restante desta demonstração.

> **Note**
>
> Esta etapa mostra como criar um livro de receitas manualmente. Você pode criar um livro de receitas em menos tempo com o kit de desenvolvimento Chef (Chef DK), executando o

comando chef generate cookbook na sua estação de trabalho local. Contudo, este comando cria várias pastas e arquivos desnecessários para esta demonstração.

Para criar o livro de receitas

1. Na estação de trabalho local, crie um diretório chamado `opsworks_cookbook_demo`. É possível usar um nome diferente, mas substitua-o por `opsworks_cookbook_demo` durante esta demonstração.
2. No diretório `opsworks_cookbook_demo`, crie um arquivo chamado `metadata.rb` usando um editor de texto. Adicione o código a seguir para especificar o nome do livro de receitas. Para obter mais informações sobre `metadata.rb`, consulte metadata.rb no site do Chef.

```
name "opsworks_cookbook_demo"
```

3. No diretório `opsworks_cookbook_demo`, crie um subdiretório chamado `recipes`. Nesse subdiretório estarão todas as receitas que você criar para esse livro de receitas de demonstração.
4. No diretório `recipes`, crie um arquivo chamado `default.rb`. Esse arquivo contém uma receita com o mesmo nome do arquivo, mas sem a extensão de arquivo: `default`. Adicione a seguinte linha única de código para o arquivo `default.rb`. Esse código é uma receita de linha única que exibe uma mensagem simples no registro quando a receita executa:

```
Chef::Log.info("********** Hello, World! **********")
```

5. No terminal ou no prompt de comando, use o comando tar para criar um arquivo chamado `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo nele inserido. Por exemplo:

```
tar -czvf opsworks_cookbook_demo.tar.gz opsworks_cookbook_demo/
```

É possível usar um nome de arquivo diferente, mas substitua-o por `opsworks_cookbook_demo.tar.gz` durante esta demonstração.

> **Note**
>
> Ao criar o arquivo `tar` no Windows, o diretório principal deve ser o diretório pai do livro de receitas. Esta demonstração foi testada no Linux com o comando tar fornecido pelo

pacote `tar`, e no Windows com o comando tar fornecido pelo Git Bash. É possível que algo dê errado se outros comandos ou programas forem usados para criar um arquivo compactado TAR (.tar.gz).

6. Crie um bucket do S3 ou use um bucket existente. Para obter mais informações, acesse Criar um bucket.
7. Faça upload do arquivo `opsworks_cookbook_demo.tar.gz` no bucket do S3. Para obter mais informações, consulte Adicionar um objeto a um bucket.

Agora você tem um livro de receitas que será usado durante a demonstração.

Na próxima etapa, você cria uma pilha de AWS OpsWorks pilhas que usará posteriormente para carregar seu livro de receitas e executar as receitas do livro de receitas.

Etapa 2: criar a pilha e seus componentes

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Crie uma AWS OpsWorks pilha de pilhas e seus componentes, que incluem uma camada e uma instância. Em etapas posteriores, será necessário fazer o upload do livro de receitas para a instância e executar essas receitas nessa instância.

Para criar a pilha

1. Faça login no console do AWS OpsWorks Stacks em https://console.aws.amazon.com/opsworks.
2. Execute a ação que melhor se aplica:
   - Se a página Bem-vindo às AWS OpsWorks pilhas for exibida, escolha Adicionar sua primeira pilha ou Adicionar sua primeira pilha de AWS OpsWorks pilhas (ambas as opções fazem a mesma coisa). A página Add stack será exibida.

- Se a página OpsWorks Painel for exibida, escolha Adicionar pilha. A página Add Stack será exibida.

3. Escolha Chef 12 stack.
4. Na caixa Stack name (Nome da pilha), digite um nome para a stack (por exemplo, **`MyCookbooksDemoStack`**). É possível inserir um nome diferente, mas substitua ele por `MyCookbooksDemoStack` durante esta demonstração.
5. Para Região, escolha Oeste dos EUA (Oregon).
6. Para VPC, realize um destes procedimentos:
   - Escolha um VPC se essa opção estiver disponível. Para obter mais informações, consulte Execução de uma Stack em uma VPC.
   - Caso contrário, escolha No VPC.
7. Para Use custom Chef cookbooks, escolha Yes.
8. Para Repository type, escolha S3 Archive.

> **Note**
>
> No passo-a-passo Conceitos básicos: Linux, escolheu Http Archive. Ao invés disso, escolha S3 Archive aqui.

9. Para Repository URL, insira o caminho para o seu arquivo `opsworks_cookbook_demo.tar.gz` no S3. Para obter o caminho, selecione o arquivo `opsworks_cookbook_demo.tar.gz` no console do S3. No painel Properties, copie o valor do campo Link. (Ele deve ser semelhante a este: `https://s3.amazonaws.com/amzn-s3-demo-bucket/opsworks_cookbook_demo.tar.gz`.)
10. Se o bucket do S3 for privado, o que é o padrão, então para ID de chave de acesso e Chave de acesso secreta, digite o ID da chave de acesso e a chave de acesso secreta do usuário do IAM usado para esta demonstração. Para obter mais informações, consulte Editar permissões de objetos e Compartilhar um objeto com os outros.
11. Deixe os padrões para os seguintes:
    - Default Availability Zone (us-west-2a)
    - Sistema operacional padrão (Linux e Amazon Linux 2016.09)
    - Default SSH key (Do not use a default SSH key)
    - Stack color (azul escuro)

12. Escolha Advanced (Avançado).
13. Em Perfil do IAM, siga um destes procedimentos:
    - Escolha aws-opsworks-service-role se essa opção estiver disponível.
    - Se não aws-opsworks-service-roleestiver disponível, escolha Nova função do IAM.
14. Para Perfil de instância padrão do IAM, realize um dos seguintes procedimentos:
    - Se aws-opsworks-ecduas funções estiverem disponíveis, escolha-as.
    - Se aws-opsworks-ec2 funções não estiverem disponíveis, escolha Novo perfil de instância do IAM.
15. Deixe os padrões para os seguintes:
    - Tipo de dispositivo raiz padrão (Baseado em EBS)
    - Tema do hostname (Depende do nível)
    - OpsWorks Versão do agente (versão mais recente)
    - Custom Chef JSON (em branco)
    - Segurança, use grupos OpsWorks de segurança (Sim)
16. Escolha Adicionar pilha. AWS OpsWorks O Stacks cria a pilha e exibe a MyCookbooksDemoStackpágina.

Para criar a camada

1. No painel de navegação de serviços, escolha Layers. A página Layers será exibida.
2. Escolha Add a layer.
3. Na OpsWorksguia, em Nome, digite**`MyCookbooksDemoLayer`**. É possível inserir um nome diferente, mas substitua ele por `MyCookbooksDemoLayer` durante esta demonstração.
4. Para Short name (Nome abreviado), digite **`cookbooks-demo`**. É possível inserir um nome diferente, mas substitua ele por `cookbooks-demo` durante esta demonstração.
5. Escolha Adicionar camada. AWS OpsWorks O Stacks adiciona a camada e exibe a página Camadas.

Para criar e iniciar a instância

1. No painel de navegação de serviços, escolha Instances. A página Instances será exibida.

2. Escolha Add an instance.
3. Na guia New, selecione Advanced.
4. Deixe os padrões para os seguintes:
   - Hostname (cookbooks-demo1)
   - Tamanho (c3.large)
   - Sub-rede (*IP address*us-west-2a)
   - Tipo de escalação (24/7)
   - Chave SSH (Não usar chave SSH padrão)
   - Sistema operacional (Amazon Linux 2016.09)
   - OpsWorks Versão do agente (herdar da pilha)
   - Locação (Padrão - usar as configurações de VPC)
   - Tipo de dispositivo raiz (Baseado em EBS)
   - Tipo de volume (Finalidade geral (SSD))
   - Tamanho de volume (8)
5. Escolha Add instance.
6. Para MyCookbooksDemoLayer, para cookbooks-demo1, para Ações, escolha iniciar. Não continue até que o Status seja alterado para online. Esse processo pode levar alguns minutos.

Agora você tem uma pilha, uma camada e uma instância para a qual o livro de receitas foi copiado automaticamente do bucket do S3. Na próxima etapa, será necessário executar e testar a receita padrão do livro de receitas na instância.

Etapa 3: executar e testar a receita

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Execute e teste a `default` receita de dentro do livro de receitas que o AWS OpsWorks Stacks copiou para a instância. Como já foi mencionado, essa é a receita de linha única que exibe uma mensagem simples no registro quando a receita é executada.

Para executar a receita

1. No painel de navegação de serviços, escolha Stack. A página MyCookbooksDemoStack será exibida.
2. Escolha Run Command. A página Run Command é exibida.
3. Para Command, escolha Execute Recipes.
4. Para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::default`**.

   `opsworks_cookbook_demo` é o nome do livro de receitas definido no arquivo `metadata.rb`. **`default`** é o nome da receita a ser executada, ou seja, o nome do arquivo `default.rb` no subdiretório `recipes` do livro de receitas, sem a extensão do arquivo.
5. Deixe as seguintes configurações padrão:
   - Comment (em branco)
   - Advanced, Custom Chef JSON (em branco)
   - Instâncias (selecione todas as opções marcadas, MyCookbooksDemoLayerverificadas, cookbooks-demo1 verificadas)
6. Escolha Execute Recipes. A página Running command execute_recipes é exibida. Não continue até que o Status seja alterado para successful. Esse processo pode levar alguns minutos.

Para verificar os resultados da receita

1. Com a página Running command execute_recipes sendo exibida, para cookbooks-demo1 e Log, escolha show. A página de log execute_recipes é exibida.
2. Role para baixo no registro e encontre uma entrada semelhante à seguinte:

```
[2015-11-13T19:14:39+00:00] INFO: ********** Hello, World! **********
```

A primeira receita foi executada com sucesso! Na próxima etapa, você irá atualizar seu livro de receitas, adicionando uma receita que instala um pacote na instância.

Etapa 4: atualizar o livro de receitas para instalar um pacote

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize seu livro de receitas adicionando uma receita que instala na instância um pacote que contém o editor de texto GNU Emacs.

Embora você possa facilmente fazer login na instância e instalar o pacote uma vez, escrever uma receita permite que você execute a receita do AWS OpsWorks Stacks uma vez para instalar vários pacotes em várias instâncias em uma pilha simultaneamente.

Para atualizar o livro de receitas para instalar um pacote

1. De volta na sua estação de trabalho local, no subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `install_package.rb` com o seguinte código:

   ```
   package "Install Emacs" do
     package_name "emacs"
   end
   ```

   Essa receita instala o pacote `emacs` na instância. (Para obter mais informações, consulte pacote.)

   > **Note**
   >
   > Dê à receita qualquer nome de arquivo que desejar. Apenas certifique-se de especificar o nome correto da receita sempre que quiser que o AWS OpsWorks Stacks execute a receita.

2. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.

Essa nova receita é executada quando o livro de receitas for atualizado na instância e, em seguida, a nova receita do livro de receitas atualizado for executada. A próxima etapa mostra como fazer isso.

Depois de concluir a próxima etapa, será possível efetuar o login na instância e, em seguida, digitar emacs a partir do prompt de comando para iniciar o GNU Emacs. (Para obter mais informações, acesse Conectar à instância do Linux.) Para sair do GNU Emacs, pressione Ctrl+Xe, em seguida, Ctrl+C.

> **Important**
>
> Para fazer o login na instância, forneça ao AWS OpsWorks Stacks as informações sobre a chave SSH pública (que pode ser criada com ferramentas como ssh-keygen ou PuTTYgen) e, em seguida, configure as permissões na `MyCookbooksDemoStack` pilha para permitir que o usuário do faça login na instância. Para obter instruções, consulte Registro de uma chave SSH pública de um usuário e Login com SSH.

Etapa 5: atualizar o livro de receitas na instância e executar a receita

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize o livro de receitas na instância e, em seguida, execute a receita do livro de receitas atualizado na instância. Ao longo desta demonstração, repita esta etapa sempre que atualizar o livro de receitas adicionando uma nova receita.

Para atualizar o livro de receitas na instância

1. No painel de navegação de serviços, escolha Stack. A página MyCookbooksDemoStack será exibida.
2. Escolha Run Command. A página Run Command é exibida.
3. Para Command, escolha Update Custom Cookbooks.
4. Deixe as seguintes configurações padrão:
   - Comment (em branco)
   - Advanced, Custom Chef JSON (em branco)
   - Avançado, Instâncias (selecione todos os marcados, MyCookbooksDemoLayerverificados, cookbooks-demo1 verificados)
5. Escolha Update Custom Cookbooks. A página Running command update_custom_cookbooks é exibida. Não continue até que o Status seja alterado para successful. Esse processo pode levar alguns minutos.

Para executar a receita

1. No painel de navegação de serviços, escolha Stack. A página MyCookbooksDemoStack será exibida.
2. Escolha Run Command. A página Run Command é exibida.
3. Para Command, escolha Execute Recipes.
4. Para Recipes to execute, digite o nome da receita a ser executada. Na primeira vez que fizer isso, a receita é chamada **`opsworks_cookbook_demo::install_package`**.

   > **Note**
   >
   > Conforme esse procedimento for repetido posteriormente, digite o nome do livro de receitas (**`opsworks_cookbook_demo`**), seguido por dois (**`::`**), seguidos pelo nome da receita (o nome de arquivo da receita, sem a extensão de arquivo `.rb`).

5. Deixe as seguintes configurações padrão:
   - Comment (em branco)
   - Advanced, Custom Chef JSON (em branco)

- Instâncias: Selecione todos (verificados, MyCookbooksDemoLayerverificados, cookbooks-demo1, verificados)

6. Escolha Execute Recipes. A página Running command execute_recipes é exibida. Não continue até que o Status seja alterado para successful. Esse processo pode levar alguns minutos.

> **Note**
>
> Não é necessário executar as receitas manualmente. Você pode atribuir receitas aos eventos do ciclo de vida de uma camada, como os eventos Setup e Configure, e o AWS OpsWorks Stacks executará essas receitas automaticamente quando o evento ocorrer. Para obter mais informações, consulte AWS OpsWorks Eventos de ciclo de vida de pilhas.

Na próxima etapa, você irá atualizar o livro de receitas para adicionar um usuário para a instância.

Etapa 6: atualizar o livro de receitas para adicionar um usuário

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize o livro de receitas adicionando uma receita que inclui um usuário local à instância e define o diretório inicial e shell do usuário. Isso é semelhante a executar os comandos adduser ou useradd no Linux ou o comando net user no Windows. Adicione um usuário local a uma instância quando quiser controlar o acesso aos arquivos e diretórios da instância, por exemplo.

Além disso, é possível gerenciar usuários sem usar os livros de receitas. Para obter mais informações, consulte Gerenciamento de usuários.

Para atualizar o livro de receitas na instância e executar a nova receita

1. Na sua estação de trabalho local, no subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `add_user.rb` com o seguinte código (para mais informações, acesse usuário):

   ```
   user "Add a user" do
     home "/home/jdoe"
     shell "/bin/bash"
     username "jdoe"
   end
   ```

2. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.
4. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::add_user`**.

Para testar a receita

1. Faça login na instância, se ainda não tiver feito.
2. No prompt de comando, execute o comando a seguir para confirmar que o novo usuário foi adicionado:

   ```
   grep jdoe /etc/passwd
   ```

   Informações sobre o usuário semelhantes às seguintes são exibidas, incluindo detalhes, como o nome do usuário, o número de ID, número de ID do grupo, diretório inicial e shell:

   ```
   jdoe:x:501:502::/home/jdoe:/bin/bash
   ```

Na próxima etapa, você irá atualizar o livro de receitas para criar um diretório na instância.

Etapa 7: atualizar o livro de receita para criar um diretório

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize seu livro de receitas, adicionando uma receita que adiciona um diretório à instância. Isso é semelhante a executar o comando mkdir no Linux ou os comandos md ou mkdir no Windows.

Para atualizar o livro de receitas na instância e executar a nova receita

1. Na sua estação de trabalho local, no subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `create_directory.rb` com o seguinte código. Para obter mais informações, consulte diretório:

   ```
   directory "Create a directory" do
     group "root"
     mode "0755"
     owner "ec2-user"
     path "/tmp/create-directory-demo"
   end
   ```

2. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.
4. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::create_directory`**.

Para testar a receita

1. Faça login na instância, se ainda não tiver feito.
2. No prompt de comando, execute o comando a seguir para confirmar que o novo diretório foi adicionado:

```
ls -la /tmp/create-directory-demo
```

Informações sobre o diretório recém-adicionado são exibidas, incluindo informações como permissões, nome do proprietário e nome do grupo:

```
drwxr-xr-x 2 ec2-user root 4096 Nov 18 00:35 .
drwxrwxrwt 6 root     root 4096 Nov 24 18:17 ..
```

Na próxima etapa, você irá atualizar o livro de receitas para criar um arquivo na instância.

Etapa 8: atualizar o livro de receitas para criar e copiar arquivos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize seu livro de receitas, adicionando uma receita que adiciona dois arquivos à instância. O primeiro recurso na receita cria um arquivo completamente com código de receita. Isso é semelhante a executar os comandos cat, echo ou touch no Linux ou os comandos echo ou fsutil no Windows. Essa técnica é útil para alguns arquivos, pequenos ou simples. O segundo recurso na receita copia um arquivo no livro de receitas para outro diretório na instância. Isso é semelhante a executar o comando cp no Linux ou o comando copy no Windows. Essa técnica é útil para muitos arquivos, grandes ou complexos.

Antes de iniciar esta etapa, complete Etapa 7: atualizar o livro de receita para criar um diretório para certificar-se de que o diretório pai dos arquivos já existe.

Para atualizar o livro de receitas na instância e executar a nova receita

1. Na estação de trabalho local, no diretório `opsworks_cookbook_demo`, crie um subdiretório chamado `files`.
2. No subdiretório `files`, crie um arquivo chamado `hello.txt` com o seguinte texto: **`Hello, World!`**
3. No subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `create_files.rb` com o seguinte código. Para obter mais informações, acesse o arquivo e cookbook_file.

```
file "Create a file" do
  content "<html>This is a placeholder for the home page.</html>"
  group "root"
  mode "0755"
  owner "ec2-user"
  path "/tmp/create-directory-demo/index.html"
end

cookbook_file "Copy a file" do
  group "root"
  mode "0755"
  owner "ec2-user"
  path "/tmp/create-directory-demo/hello.txt"
  source "hello.txt"
end
```

   O recurso `file` cria um arquivo no caminho especificado. O recurso `cookbook_file` copia o arquivo do diretório `files` recém-criado no livro de receitas (o Chef espera encontrar um subdiretório de nome padrão chamado `files` de onde ele pode copiar arquivos) para outro diretório na instância.
4. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
5. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.
6. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::create_files`**.

Para testar a receita

1. Faça login na instância, se ainda não tiver feito.
2. No prompt de comando, execute os seguintes comandos, um de cada vez, para confirmar que os novos arquivos foram adicionados:

```
sudo cat /tmp/create-directory-demo/index.html

sudo cat /tmp/create-directory-demo/hello.txt
```

O conteúdo dos arquivos é exibido:

```
<html>This is a placeholder for the home page.</html>

Hello, World!
```

Na próxima etapa, você irá atualizar o livro de receitas para executar um comando na instância.

Etapa 9: atualizar o livro de receitas para executar um comando

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize seu livro de receitas, adicionando uma receita que executa um comando que cria uma chave SSH na instância.

Para atualizar o livro de receitas na instância e executar a nova receita

1. Na sua estação de trabalho local, no subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `run_command.rb` com o seguinte código. Para obter mais informações, consulte execute.

```
execute "Create an SSH key" do
  command "ssh-keygen -f /tmp/my-key -N fLyC3jbY"
end
```

2. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.
4. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::run_command`**.

Para testar a receita

1. Faça login na instância, se ainda não tiver feito.
2. No prompt de comando, execute os seguintes comandos, um de cada vez, para confirmar que a chave SSH foi criada:

```
sudo cat /tmp/my-key

sudo cat /tmp/my-key.pub
```

O conteúdo das chaves SSH privada e pública são exibidos:

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,DEF7A09C...541583FA
A5p9dCuo...wp0YYH1c
-----END RSA PRIVATE KEY-----

ssh-rsa AAAAB3N...KaNogZkT root@cookbooks-demo1
```

Na próxima etapa, você irá atualizar o livro de receitas para executar um script na instância.

Etapa 10: atualizar o livro de receitas para executar um script

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize seu livro de receitas, adicionando uma receita que executa um script na instância. Essa receita cria um diretório e, em seguida, cria um arquivo nesse diretório. Escrever uma receita para executar um script que contém vários comandos é mais fácil do que executar esses comandos um de cada vez.

Para atualizar o livro de receitas na instância e executar a nova receita

1. Na sua estação de trabalho local, no subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `run_script.rb` com o seguinte código. Para obter mais informações, consulte script.

   ```
   script "Run a script" do
     interpreter "bash"
     code <<-EOH
       mkdir -m 777 /tmp/run-script-demo
       touch /tmp/run-script-demo/helloworld.txt
       echo "Hello, World!" > /tmp/run-script-demo/helloworld.txt
     EOH
   end
   ```

2. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.
4. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::run_script`**.

Para testar a receita

1. Faça login na instância, se ainda não tiver feito.
2. No prompt de comando, execute o comando a seguir para confirmar que o novo arquivo foi adicionado:

   ```
   sudo cat /tmp/run-script-demo/helloworld.txt
   ```

   O conteúdo do arquivo é exibido:

   ```
   Hello, World!
   ```

Na próxima etapa, você irá atualizar o livro de receitas para gerenciar um serviço na instância.

Etapa 11: atualizar o livro de receitas para gerenciar um serviço

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize seu livro de receitas, adicionando uma receita que gerencia um serviço na instância. Isso é semelhante a executar o comando service no Linux ou os comandos net stop, net start e semelhantes do Windows. Essa receita interrompe o serviço crond na instância.

Para atualizar o livro de receitas na instância e executar a nova receita

1. Na sua estação de trabalho local, no subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `manage_service.rb` com o seguinte código. Para obter mais informações, consulte service.

   ```
   service "Manage a service" do
     action :stop
     service_name "crond"
   ```

```
end
```

2. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.
4. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::manage_service`**.

Para testar a receita

1. Faça login na instância, se ainda não tiver feito.
2. No prompt de comando, execute o comando a seguir para confirmar que o serviço crond está interrompido:

   ```
   service crond status
   ```

   A seguinte estrutura é exibida:

   ```
   crond is stopped
   ```

3. Para reiniciar o serviço crond, execute o seguinte comando:

   ```
   sudo service crond start
   ```

   A seguinte estrutura é exibida:

   ```
   Starting crond:  [  OK  ]
   ```

4. Para confirmar que o serviço crond foi iniciado, execute o comando a seguir novamente:

   ```
   service crond status
   ```

   Informações semelhante às seguintes serão exibidas:

```
crond (pid  3917) is running...
```

Na próxima etapa, você irá atualizar o livro de receitas para fazer referência a informações armazenadas como JSON personalizado na instância.

Etapa 12: atualizar o livro de receitas para usar JSON personalizado

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize seu livro de receitas adicionando uma receita que faz referência a JSON personalizado armazenado na instância.

É possível especificar informações em formato JSON personalizado sempre que criar, atualizar ou clonar uma pilha ou quando executar um comando de implantação ou pilha. Isso é útil, por exemplo, para disponibilizar uma porção pequena e inalterável de dados para as receitas na instância ao invés de receber esses dados de um banco de dados. Para obter mais informações, consulte Usar JSON personalizado.

Para esta demonstração, será necessário usar JSON personalizado para fornecer informações fictícias sobre uma fatura de cliente. O JSON personalizado será descrito posteriormente nesta etapa.

Para atualizar o livro de receitas na instância e executar a nova receita

1. Na sua estação de trabalho local, no subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `custom_json.rb` que contenha o seguinte código de receita:

   ```
   Chef::Log.info("********** For customer '#{node['customer-id']}' invoice
    '#{node['invoice-number']}' **********")
   ```

```
Chef::Log.info("********** Invoice line number 1 is a '#{node['line-items']
['line-1']}' **********")
Chef::Log.info("********** Invoice line number 2 is a '#{node['line-items']
['line-2']}' **********")
Chef::Log.info("********** Invoice line number 3 is a '#{node['line-items']
['line-3']}' **********")
```

Essa receita exibe mensagens no registro sobre os valores no JSON personalizado.

2. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.
4. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::custom_json`**. Para Advanced, Custom Chef JSON, digite o seguinte JSON personalizado:

```
{
  "customer-id": "0123",
  "invoice-number": "9876",
  "line-items": {
    "line-1": "tractor",
    "line-2": "passenger car",
    "line-3": "trailer"
  }
}
```

Para testar a receita

1. Com a página Running command execute_recipes sendo exibida a partir dos procedimentos anteriores, para cookbooks-demo1 e Log, escolha show. A página de log execute_recipes é exibida.
2. Role para baixo no registro para encontrar entradas semelhante ao seguinte:

```
[2015-11-14T14:18:30+00:00] INFO: ********** For customer '0123' invoice '9876'
 **********
```

```
[2015-11-14T14:18:30+00:00] INFO: ********** Invoice line number 1 is a 'tractor'
 **********
[2015-11-14T14:18:30+00:00] INFO: ********** Invoice line number 2 is a 'passenger
 car' **********
[2015-11-14T14:18:30+00:00] INFO: ********** Invoice line number 3 is a 'trailer'
 **********
```

Essas entradas exibem informações do JSON personalizado que foram digitadas na caixa Advanced, Custom Chef JSON.

Na próxima etapa, você atualizará o livro de receitas para obter informações dos pacotes de dados, que são coleções de configurações de pilha que o AWS OpsWorks Stacks armazena em cada instância.

Etapa 13: atualizar o livro de receitas para usar data bags

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize seu livro de receitas adicionando uma receita que faça referência às configurações da pilha que o AWS OpsWorks Stacks armazena na instância em um conjunto de pacotes de dados. Essa receita exibe mensagens no registro sobre as configurações específicas de pilha que são armazenadas na instância. Para obter mais informações, consulte o AWS OpsWorks Referência do Stacks Data Bag.

Para atualizar o livro de receitas na instância e executar a nova receita

1. Na sua estação de trabalho local, no subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `data_bags.rb` que contenha o seguinte código:

```
instance = search("aws_opsworks_instance").first
layer = search("aws_opsworks_layer").first
```

```
stack = search("aws_opsworks_stack").first

Chef::Log.info("********** This instance's instance ID is
 '#{instance['instance_id']}' **********")
Chef::Log.info("********** This instance's public IP address is
 '#{instance['public_ip']}' **********")
Chef::Log.info("********** This instance belongs to the layer '#{layer['name']}'
 **********")
Chef::Log.info("********** This instance belongs to the stack '#{stack['name']}'
 **********")
Chef::Log.info("********** This stack gets its cookbooks from
 '#{stack['custom_cookbooks_source']['url']}' **********")
```

Essa receita exibe mensagens no registro sobre as configurações específicas de pilha que são armazenadas na instância.

2. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.
4. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::data_bags`**.

Para testar a receita

1. Com a página Running command execute_recipes sendo exibida a partir do procedimento anterior, para cookbooks-demo1 e Log, escolha show. A página de log execute_recipes é exibida.
2. Role para baixo no registro e encontre entradas semelhantes ao seguinte:

```
[2015-11-14T14:39:06+00:00] INFO: ********** This instance's instance ID is
 'f80fa119-81ab-4c3c-883d-6028e52c89EX' **********
[2015-11-14T14:39:06+00:00] INFO: ********** This instance's public IP address is
 '192.0.2.0' **********
[2015-11-14T14:39:06+00:00] INFO: ********** This instance belongs to the layer
 'MyCookbooksDemoLayer' **********
[2015-11-14T14:39:06+00:00] INFO: ********** This instance belongs to the stack
 'MyCookbooksDemoStack' **********
```

```
[2015-11-14T14:39:06+00:00] INFO: ********** This stack gets its cookbooks from
 'https://s3.amazonaws.com/amzn-s3-demo-bucket/opsworks_cookbook_demo.tar.gz'
 **********
```

Essa receita exibe mensagens sobre as configurações específicas de pilha que são armazenadas na instância.

Na próxima etapa, você atualizará o livro de receitas para executar código de receita várias vezes.

Etapa 14: atualizar o livro de receitas para usar iteração

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualize seu livro de receitas adicionando um código que usa iteração, uma técnica que repete um código de receita várias vezes. Essa receita exibe mensagens no registro para um item do data bag que contém muitos conteúdos.

Para atualizar o livro de receitas na instância e executar a nova receita

1. Na sua estação de trabalho local, no subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `iteration_demo.rb` que contenha o seguinte código:

```
stack = search("aws_opsworks_stack").first
Chef::Log.info("********** Content of 'custom_cookbooks_source' **********")

stack["custom_cookbooks_source"].each do |content|
  Chef::Log.info("********** '#{content}' **********")
end
```

> **Note**
>
> Gravar o código de receita anterior é um procedimento mais curto, mais flexível e menos propenso a erros do que escrever o seguinte código de receita que não usa iteração:

```
stack = search("aws_opsworks_stack").first
Chef::Log.info("********** Content of 'custom_cookbooks_source' **********")

Chef::Log::info("********** '[\"type\", \"#{stack['custom_cookbooks_source']
['type']}\"]' **********")
Chef::Log::info("********** '[\"url\", \"#{stack['custom_cookbooks_source']
['url']}\"]' **********")
Chef::Log::info("********** '[\"username\",
 \"#{stack['custom_cookbooks_source']['username']}\"]' **********")
Chef::Log::info("********** '[\"password\",
 \"#{stack['custom_cookbooks_source']['password']}\"]' **********")
Chef::Log::info("********** '[\"ssh_key\",
 \"#{stack['custom_cookbooks_source']['ssh_key']}\"]' **********")
Chef::Log::info("********** '[\"revision\",
 \"#{stack['custom_cookbooks_source']['revision']}\"]' **********")
```

2. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.
4. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::iteration_demo`**.

Para testar a receita

1. Com a página Running command execute_recipes sendo exibida a partir dos procedimentos anteriores, para cookbooks-demo1 e Log, escolha show. A página de log execute_recipes é exibida.
2. Role para baixo no registro e encontre entradas semelhantes ao seguinte:

```
[2015-11-16T19:56:56+00:00] INFO: ********** Content of 'custom_cookbooks_source'
 **********
[2015-11-16T19:56:56+00:00] INFO: ********** '["type", "s3"]' **********
[2015-11-16T19:56:56+00:00] INFO: ********** '["url", "https://s3.amazonaws.com/
amzn-s3-demo-bucket/opsworks_cookbook_demo.tar.gz"]' **********
[2015-11-16T19:56:56+00:00] INFO: ********** '["username", "secret-key-value"]'
 **********
[2015-11-16T19:56:56+00:00] INFO: ********** '["password", "secret-access-key-
value"]' **********
[2015-11-16T19:56:56+00:00] INFO: ********** '["ssh_key", nil]' **********
[2015-11-16T19:56:56+00:00] INFO: ********** '["revision", nil]' **********
```

Essa receita exibe mensagens no registro para um item do data bag que contém muitos conteúdos. O item do data bag está no data bag `aws_opsworks_stack`. O item do data bag tem conteúdo chamado `custom_cookbooks_source`. Dentro desse conteúdo existem seis conteúdos chamados `type`, `url`, `username`, `password`, `ssh_key` e `revision`, e seus valores também são exibidos.

Na próxima etapa, você atualizará o livro de receitas para executar um código de receita somente se determinadas condições forem atendidas.

Etapa 15: atualizar o livro de receitas para usar lógica condicional

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Agora, atualize seu livro de receitas adicionando uma receita que usa a lógica condicional, uma técnica que executa o código somente se determinadas condições forem atendidas. Para obter mais informações, acesse Declarações com "if" e Declarações com "case".

Essa receita faz duas coisas com base no conteúdo do data bag: exibe uma mensagem no registro para identificar o sistema operacional em que a instância está sendo executada e, somente se o

sistema operacional for Linux, instala um pacote usando o gerenciador de pacote correto para a determinada distribuição do Linux. Esse pacote é chamado árvore; é um simples aplicativo para visualizar listas de diretórios.

Para atualizar o livro de receitas na instância e executar a nova receita

1. Na sua estação de trabalho local, no subdiretório `recipes`, em `opsworks_cookbook_demo directory`, crie um arquivo chamado `conditional_logic.rb` que contenha o seguinte código:

```
instance = search("aws_opsworks_instance").first
os = instance["os"]

if os == "Red Hat Enterprise Linux 7"
  Chef::Log.info("********** Operating system is Red Hat Enterprise Linux.
 **********")
elsif os == "Ubuntu 14.04 LTS" || os == "Ubuntu 16.04 LTS" || os == "Ubuntu 18.04
 LTS"
  Chef::Log.info("********** Operating system is Ubuntu. **********")
elsif os == "Microsoft Windows Server 2012 R2 Base"
  Chef::Log.info("********** Operating system is Windows. **********")
elsif os == "Amazon Linux 2015.03" || os == "Amazon Linux 2015.09" || os == "Amazon
 Linux 2016.03" || os == "Amazon Linux 2016.09" || os == "Amazon Linux 2017.03"
 || os == "Amazon Linux 2017.09" || os == "Amazon Linux 2018.03" || os == "Amazon
 Linux 2"
  Chef::Log.info("********** Operating system is Amazon Linux. **********")
elsif os == "CentOS Linux 7"
  Chef::Log.info("********** Operating system is CentOS 7. **********")
else
  Chef::Log.info("********** Cannot determine operating system. **********")
end

case os
when "Ubuntu 14.04 LTS", "Ubuntu 16.04 LTS", "Ubuntu 18.04 LTS"
  apt_package "Install a package with apt-get" do
    package_name "tree"
  end
when "Amazon Linux 2015.03", "Amazon Linux 2015.09", "Amazon Linux 2016.03",
 "Amazon Linux 2016.09", "Amazon Linux 2017.03", "Amazon Linux 2017.09", "Amazon
 Linux 2018.03", "Amazon Linux 2", "Red Hat Enterprise Linux 7", "CentOS Linux 7"
  yum_package "Install a package with yum" do
    package_name "tree"
  end
```

```
else
  Chef::Log.info("********** Cannot determine operating system type, or operating
 system is not Linux. Package not installed. **********")
end
```

2. No terminal ou no prompt de comando, use o comando tar para criar uma nova versão do arquivo `opsworks_cookbook_demo.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo atualizado.
3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` atualizado no bucket do S3.
4. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::conditional_logic`**.

Para testar a receita

1. Com a página Running command execute_recipes sendo exibida a partir dos procedimentos anteriores, para cookbooks-demo1 e Log, escolha show. A página de log execute_recipes é exibida.
2. Role para baixo ao longo do registro e encontre uma entrada semelhante à seguinte:

```
[2015-11-16T19:59:05+00:00] INFO: ********** Operating system is Amazon Linux.
 **********
```

   Como o sistema operacional da instância é Amazon Linux 2016.09, somente a entrada anterior (das cinco possíveis entradas no código de receitas) será exibida no registro.
3. Se o sistema operacional for Linux, a receita instalará o pacote árvore. Para visualizar o conteúdo de um diretório, digite **`tree`** no prompt de comando a partir do diretório desejado ou com o caminho do diretório desejado (por exemplo, `tree /var/chef/runs`).

Na próxima etapa, você atualizará o livro de receitas para usar a funcionalidade a partir de um livro de receitas externo fornecido pela comunidade Chef.

Etapa 16: atualizar o livro de receitas para usar livros de receitas da comunidade

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Finalmente, atualize o livro de receitas para usar a funcionalidade fornecida em um livro de receitas externo fornecido pela comunidade Chef. O livro de receitas externo que você usará para esta demonstração está disponível no Chef Supermarket, um local conhecido para acessar livros externos de receitas do Chef. Esse livro de receitas externo fornece um recurso personalizado que permite que você faça o download e instale aplicativos, similar ao que era feito em Etapa 4: atualizar o livro de receitas para instalar um pacote. No entanto, esse recurso pode instalar aplicativos da Web e outros tipos de aplicativo, além de pacotes.

Quando um livro de receitas depende de outro livro de receitas, é necessário especificar uma dependência no outro livro de receitas. Para declarar e gerenciar dependências do livro de receitas, recomendamos que você use uma ferramenta chamada Berkshelf. Para obter mais informações sobre como instalar o Berkshelf na sua estação de trabalho local, consulte Sobre o Berkshelf no site do Chef.

Depois de instalar o Berkshelf, siga estes procedimentos para declarar a dependência do livro de receitas e crie uma receita que chama o recurso no livro de receitas externo:

Para declarar a dependência de do livro de receitas

1. Na estação de trabalho local, no diretório `opsworks_cookbook_demo`, adicione a seguinte linha no final do arquivo `metadata.rb`:

   ```
   depends "application", "5.0.0"
   ```

   Isso declara uma dependência em um livro de receitas chamado `application`, versão 5.0.0.
2. Na raiz do diretório `opsworks_cookbook_demo`, execute o seguinte comando: O período no final do comando é intencional.

```
berks init .
```

O Berkshelf cria uma série de pastas e arquivos que você poderá usar posteriormente para cenários mais avançados. O único arquivo que precisamos para esta demonstração é o arquivo chamado `Berksfile`.

3. Adicione a seguinte linha no final do arquivo `Berksfile`:

```
cookbook "application", "5.0.0"
```

Isso informa ao Berkshelf que você deseja usar o livro de receitas do aplicativo, versão 5.0.0, obtido por download pelo Berkshelf no Chef Supermarket.

4. No terminal ou prompt de comando, execute o seguinte comando a partir da raiz do diretório `opsworks_cookbook_demo`:

```
berks install
```

O Berkshelf criará uma lista de dependências para o seu livro de receitas e para o livro de receitas do aplicativo. O Berkshelf usará essa lista de dependências no próximo procedimento.

Para atualizar o livro de receitas na instância e executar a nova receita

1. No subdiretório `recipes`, no diretório `opsworks_cookbook_demo`, crie um arquivo chamado `dependencies_demo.rb` que contenha o seguinte código:

```
application "Install NetHack" do
  package "nethack.x86_64"
end
```

Essa receita depende do recurso do aplicativo do livro de receitas do aplicativo para instalar o popular jogo de aventura baseado em texto NetHack na instância. (É possível substituir qualquer outro nome do pacote que quiser, desde que o pacote já está disponível para o gerenciador de pacote na instância.)

2. Execute o comando a seguir na raiz de `opsworks_cookbook_demo`:

```
berks package
```

Berkshelf usa a lista de dependências do procedimento anterior para criar um arquivo chamado `cookbooks-timestamp.tar.gz`, que contém o diretório `opsworks_cookbook_demo` e o conteúdo dele atualizado, incluindo os livros de receitas dependentes do livro de receitas. Renomeie esse arquivo para `opsworks_cookbook_demo.tar.gz`.

3. Carregue o arquivo `opsworks_cookbook_demo.tar.gz` renomeado e atualizado no bucket do S3.
4. Para atualizar o livro de receitas na instância e para executar a nova receita, siga as instruções em Etapa 5: atualizar o livro de receitas na instância e executar a receita. No processo "Para executar a receita", para Recipes to execute (Receitas para executar), digite **`opsworks_cookbook_demo::dependencies_demo`**.
5. Depois de executar a receita, faça o login na instância e, em seguida, digite **`nethack`** no prompt de comando para iniciar a reprodução. (Para obter mais informações sobre o jogo, consulte NetHacko NetHackGuia.)

Na próxima etapa, você pode limpar os AWS recursos que você usou para este passo a passo. Essa próxima etapa é opcional. Talvez seja necessário continuar usando os AWS recursos da para aprender mais sobre o AWS OpsWorks Stacks. No entanto, manter esses AWS recursos disponíveis pode resultar em algumas cobranças contínuas em sua AWS conta. Se quiser manter esses AWS recursos disponíveis para uso posterior, você concluiu este passo a passo e pode pular para. Próximas etapas

Etapa 17: limpar (opcional)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para evitar contrair cobranças adicionais na sua AWS conta, você poderá excluir os AWS recursos usados nesta demonstração. Esses AWS recursos incluem o bucket S3, a pilha AWS OpsWorks Stacks e os componentes da pilha. (Para obter mais informações, consulte Definição de OpsWorks preço da AWS.) No entanto, talvez seja necessário continuar usando os AWS recursos da para

aprender mais sobre o AWS OpsWorks Stacks. Se quiser manter esses AWS recursos disponíveis, você concluiu este passo a passo e pode pular para. Próximas etapas

O conteúdo armazenado nos recursos que você criou para essa demonstração pode conter informações de identificação pessoal. Se você não quer mais que essas informações sejam armazenadas pela AWS, siga as etapas neste tópico.

Para excluir o bucket do S3

- Consulte Excluir o bucket do Amazon S3.

Para deletar a instância da pilha

1. No console do AWS OpsWorks Stacks, no painel de navegação do serviço, escolha Instâncias. A página Instances será exibida.
2. Para MyCookbooksDemoLayer, para cookbooks-demo1, para Ações, escolha parar. Quando a mensagem de confirmação for exibida, escolha Stop.
3. As alterações a seguir levam alguns minutos. Aguarde e não prossiga até que todos os itens a seguir tenham sido concluídos.
   - Status muda de online para stopping e, em um determinado momento, para stopped.
   - online muda de 1 para 0.
   - shutting down muda de 0 para 1 e, em um determinado momento, de volta para 0.
   - stopped muda, em determinado momento, de 0 para 1.
4. Em Actions, selecione delete. Quando a mensagem de confirmação for exibida, escolha Excluir. AWS OpsWorks O Stacks excluirá a instância e exibe Nenhuma instância.

Para excluir a pilha

1. No painel de navegação de serviços, escolha Stack. A página MyCookbooksDemoStack será exibida.
2. Escolha Delete Stack. Quando a mensagem de confirmação for exibida, escolha Excluir. AWS OpsWorks O Stacks excluirá a pilha e exibirá a página Painel.

Opcionalmente, você pode excluir o usuário do IAM e o par de EC2 chaves da Amazon que você usou para essa demonstração, caso não queira reutilizá-los para acessar outros AWS serviços da e

instâncias. EC2 Para obter instruções, consulte Excluir um usuário do IAM e pares de EC2 chaves e instâncias Linux da Amazon.

Você acabou de concluir essa apresentação. Para obter mais informações, consulte Próximas etapas.

Próximas etapas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Agora que você concluiu este passo a passo, você pode aprender mais sobre o suporte do AWS OpsWorks Stacks para livros de receitas do Chef analisando os seguintes recursos:

- Livros de receitas e receitas— Descreve as versões do Chef e do Ruby que o AWS OpsWorks Stacks suporta atualmente. Também mostra como instalar e atualizar livros de receitas personalizados em instâncias e como executar receitas em instâncias.
- Saiba mais sobre Chef – Fornece links para tutoriais do Chef, uma biblioteca de habilidades do Chef, documentação completa sobe o Chef e cursos de treinamento sobre o Chef.
- Tudo sobre o Chef: fornece a documentação completa sobre o Chef. Tópicos específicos de interesse incluem:
  - Sobre livros de receitas – Descreve os principais componentes do livro de receitas, como características, receitas, arquivos, metadados e modelos.
  - Sobre receitas – Descreve os conceitos básicos de receitas, por exemplo, como trabalhar com data bags, incluem outras receitas e usa o código Ruby em receitas.
  - Recursos – Descreve como usar todos os recursos integrados do Chef, como `apt_package`, `cookbook_file`, `directory`, `execute`, `file` e `package`.
  - Sobre a receita DSL – Descreve como escrever código para receitas do Chef com instruções como `ifcase`, `data_bag`, `data_bag_item` e `search`.
- Sobre modelos – Descreve como usar modelos integrados Ruby (ERB) para gerar dinamicamente arquivos de texto estáticos, como arquivos de configuração.

- Trilhas de introdução: descreve como usar o Chef para gerenciar uma instância, gerenciar um aplicativo básico da Web, desenvolver e testar o código de infraestrutura de análise, usar o Chef e muito mais.
- Introdução ao Chef: uma introdução ao Chef. Publicado pela O'Reilly Media.
- Introdução ao Chef: exemplos de código – Fornece exemplos de código para acompanhar o livro Introdução ao Chef publicado por O'Reilly Media.

AWS OpsWorks Práticas recomendadas do Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As estratégias, técnicas e sugestões nesta seção podem ajudá-lo a obter o máximo benefício e os melhores resultados do AWS OpsWorks Stacks.

Tópicos

- Práticas recomendadas: armazenamento do dispositivo raiz para instâncias
- Práticas recomendadas: otimização do número de servidores de aplicativos
- Melhores práticas: Gerenciamento de permissões
- Melhores Práticas: Gerenciamento e implementação de aplicativos e guias de procedimentos
- Empacotar dependências do livro de receitas localmente

Práticas recomendadas: armazenamento do dispositivo raiz para instâncias

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este tópico não se aplica a instâncias do Windows, que devem ter o suporte do Amazon Elastic Block Store.

As instâncias Linux do Amazon Elastic Compute Cloud (Amazon EC2) têm as seguintes opções de armazenamento em dispositivos raiz.

- Instâncias com suporte do armazenamento de instâncias: o dispositivo raiz é temporário.

  Caso você pare a instância, os dados no dispositivo raiz desaparecem e não podem ser recuperados. Para obter mais informações, consulte Amazon EC2 Instance Store.
- Instâncias com suporte do Amazon EBS: o dispositivo raiz é um volume Amazon EBS.

  Caso você pare a instância, o volume do Amazon EBS persiste. Caso você reinicie a instância, o volume é remontado automaticamente, restaurando o estado da instância e todos os dados armazenados. Você também pode montar o volume em uma instância diferente. Para obter mais informações, consulte Amazon Elastic Block Store (Amazon EBS).

Considere o seguinte ao decidir qual opção de armazenamento de dispositivo raiz usar.

Tempo de inicialização

Após a inicialização, as instâncias do Amazon EBS normalmente reiniciam mais rapidamente.

O tempo de inicialização é aproximadamente o mesmo para qualquer um dos tipos de armazenamento. Ambos os tipos devem realizar uma configuração completa, o que inclui tarefas relativamente demoradas, como a instalação de pacotes de repositórios remotos. No entanto, observe estas distinções quando você reiniciar uma instância depois:

- As instâncias com suporte do armazenamento de instâncias realizam as mesmas tarefas de configuração que realizaram na inicialização, inclusive a instalação do pacote.

  A reinicialização demora aproximadamente o mesmo tempo que a inicialização.

- As instâncias com suporte do Amazon EBS remontam o volume raiz e executam as receitas de instalação.

  A reinicialização costuma ser bem mais rápida do que a inicialização, porque as receitas de instalação não precisas realizar tarefas como reinstalar pacotes já estão instalados no volume raiz.

Custo

As instâncias com suporte do Amazon EBS são mais caras:

- Com uma instância com suporte do armazenamento de instâncias, você só paga quando a instância está em execução.
- Com as instâncias com suporte do Amazon EBS, você paga pelo volume do Amazon EBS, independentemente da instância estar em execução ou não.

  Para obter mais informações, consulte a definição de preço do Amazon EBS.

Registro em log

As instâncias com suporte do Amazon EBS mantêm logs automaticamente:

- Com a instância com suporte do armazenamento de instâncias, os logs desaparecem quando a instância para.

  Você deve recuperar os registros antes de interromper a instância ou usar um serviço como o CloudWatch Logs para armazenar os registros selecionados remotamente.
- Com uma instância com suporte do Amazon EBS, os logs são armazenados no volume do Amazon EBS.

  Você pode visualizá-los reiniciando a instância ou montando o volume em outra instância.

Dependências

Os dois tipos de armazenamento têm dependências diferentes:

- As instâncias com suporte do armazenamento de instâncias dependem do Amazon S3.

  Quando você inicia a instância, ela deve fazer download do AMI do Amazon S3.
- As instâncias com suporte do Amazon EBS dependem do Amazon EBS.

  Quando você inicia a instância, ela deve montar o volume raiz do Amazon EBS.

Recomendação: caso você não tenha certeza de qual tipo de armazenamento seja o mais indicado para os requisitos, recomendamos começar por instâncias do Amazon EBS. Embora você incorrerá em uma despesa modesta para os volumes do Amazon EBS, há menos risco de perda de dados não intencional.

Práticas recomendadas: otimização do número de servidores de aplicativos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Uma pilha de produção normalmente inclui vários servidores de aplicativos distribuídos em várias zonas de disponibilidade. No entanto, o número de solicitações de entrada pode variar muito, dependendo do momento do dia ou de acordo com o dia da semana. Você pode simplesmente executar servidores suficientes para lidar com o máximo de carga prevista, mas, então, na maior parte do tempo, acabará pagando por mais capacidade de servidor do que o necessário. Para executar seu site de maneira eficiente, a prática recomendada é corresponder o número de servidores com o volume atual de solicitações.

AWS OpsWorks O Stacks fornece três maneiras de gerenciar o número de instâncias do servidor.

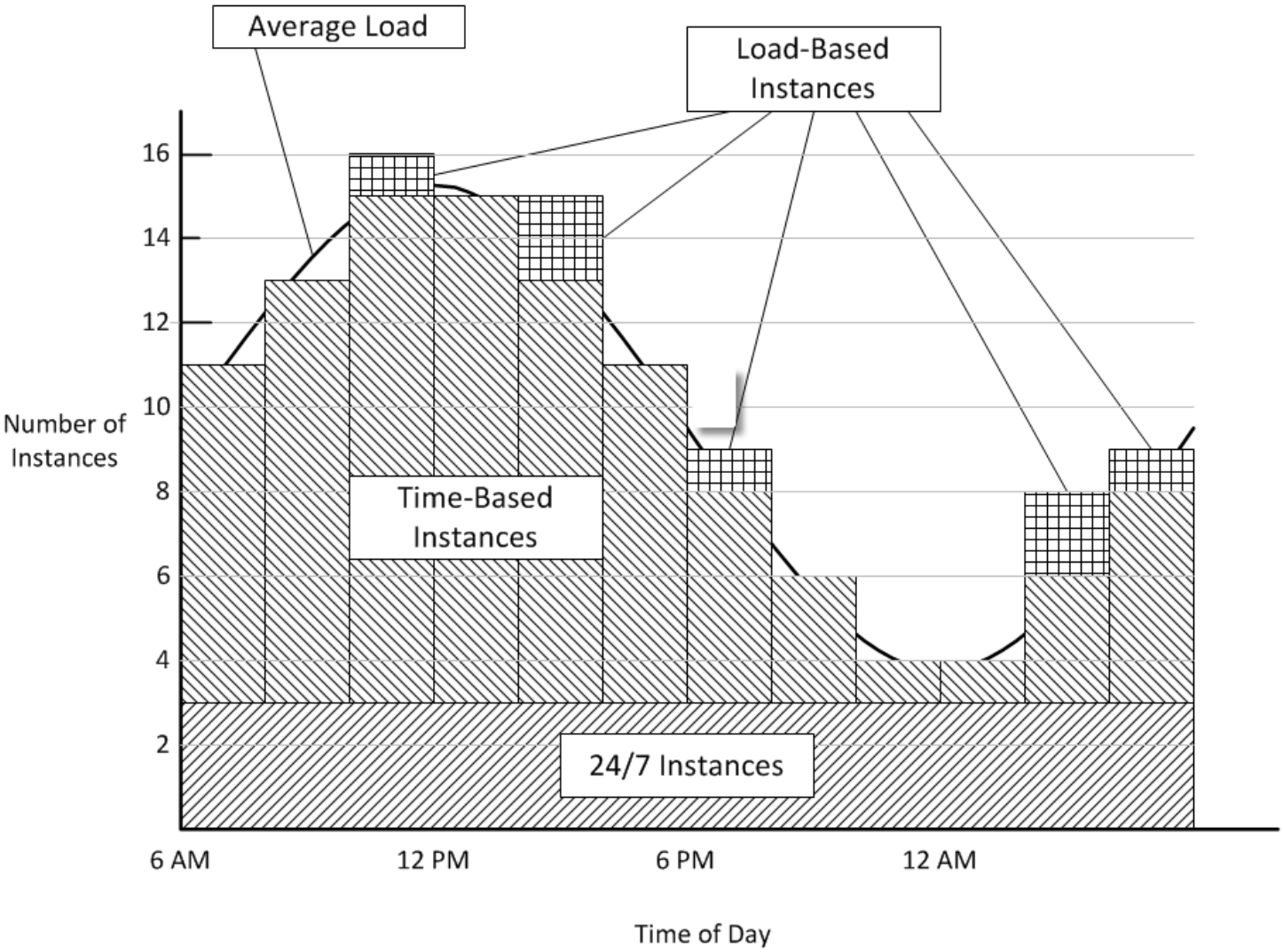
- As instâncias ininterruptas são iniciadas manualmente e executadas até que sejam manualmente interrompidas.
- As instâncias baseadas em tempo são iniciadas e interrompidas automaticamente pelo AWS OpsWorks Stacks em uma programação especificada pelo usuário.
- As instâncias baseadas em carga são iniciadas e interrompidas automaticamente pelas AWS OpsWorks pilhas quando ultrapassam o limite de uma métrica de carga especificada pelo usuário, como utilização de CPU ou memória.

> **Note**
>
> Depois que você criar e configurar as instâncias baseadas em tempo e em carga da pilha, o AWS OpsWorks Stacks as inicia e as interrompe automaticamente com base na configuração especificada. Você não precisa mexer nelas novamente, a menos que decida alterar a configuração ou o número de instâncias.

Recomendação: se você estiver gerenciando pilhas com mais de algumas instâncias do servidor de aplicativos, recomendamos o uso de uma combinação dos três tipos de instâncias. Veja a seguir um exemplo de como gerenciar uma capacidade de servidor da pilha para lidar com um volume de solicitações diárias variável com as seguintes características.

- A média de volume solicitado varia de maneira sinusoidal durante o dia.
- O volume de solicitação médio mínimo requer cinco instâncias do servidor de aplicativos.
- O volume de solicitação médio máximo requer dezesseis instâncias do servidor de aplicativos.
- Os picos de volume de solicitações geralmente podem ser processados por uma ou duas instâncias do servidor de aplicativos.

Esse é um modelo conveniente para fins de discussão, mas você pode adaptá-lo facilmente a qualquer variação no volume de solicitações e também estendê-lo para que lide com variações semanais. O diagrama a seguir mostra como usar os três tipos de instâncias para gerenciar esse volume de solicitações.



Esse exemplo tem as seguintes características:

- A pilha tem três instâncias ininterruptas, que estão sempre em execução e processam a carga de base.
- A pilha tem 12 instâncias baseadas no tempo, que são configurados para processar a variação média diária.

  Uma é executada das 22h às 2h, duas das 20h às 22h e das 2h às 4h, e assim por diante. Para simplificar, o diagrama modifica o número de instâncias baseadas no tempo a cada duas horas, mas você pode modificar o número a cada hora se quiser um controle refinado.
- A pilha tem instâncias baseadas em carga suficientes para lidar com os picos de tráfego que ultrapassarem o que pode ser processado pelas instâncias ininterruptas e baseadas no tempo.

AWS OpsWorks O Stacks inicia instâncias baseadas em carga somente quando a carga em todos os servidores atualmente em execução excede as métricas especificadas. O custo para instâncias que não estão em execução é mínimo (instâncias baseadas em Amazon EBS) ou nenhum (instâncias com base em armazenamento de instâncias), de modo que a prática recomendada é criar o suficiente a fim de processar confortavelmente os volumes de solicitações máximos previstos. Para este exemplo, a pilha deve ter no mínimo três instâncias baseadas em carga.

> **Note**
>
> Distribua os três tipos de instância em várias zonas de disponibilidade para minimizar o impacto das interrupções de serviço.

Melhores práticas: Gerenciamento de permissões

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você deve ter algum tipo de credencial da AWS para acessar os recursos da sua conta. Veja a seguir algumas diretrizes gerais para fornecer acesso a seus funcionários.

- Em primeiro lugar, recomendamos que você não use as credenciais raiz da sua conta para acessar os recursos da AWS.

  Em vez disso, crie identidades do IAM para seus funcionários e adicione permissões que forneçam acesso adequado. Assim, cada funcionário pode usar suas próprias credenciais para acessar os recursos.
- Os funcionários devem ter permissão para acessar apenas os recursos de que eles precisam para desempenhar suas funções.

Por exemplo, os desenvolvedores de aplicativos precisam acessar apenas as pilhas que executam seus aplicativos.

- Os funcionários devem ter permissão para executar apenas as ações de que eles precisam para desempenhar suas funções.

  Um desenvolvedor de aplicativos pode precisar de permissão completa para uma pilha de desenvolvimento e de permissão para implantar seus aplicativos na pilha de produção correspondente. Eles provavelmente não precisam de permissão para iniciar ou interromper instâncias na pilha de produção, criar ou excluir camadas, e assim por diante.

Para obter mais informações gerais sobre o gerenciamento de permissões, consulte Credenciais de segurança da AWS.

Você pode usar o AWS OpsWorks Stacks ou o IAM para gerenciar as permissões do usuário. Observe que as duas opções não são mutuamente exclusivas e, em algumas ocasiões, pode ser conveniente usar ambas.

AWS OpsWorks Gerenciamento de permissões de pilhas

Cada pilha tem uma página Permissions que você usa para conceder aos usuários permissões de acesso à pilha e para especificar as ações que eles podem executar. Você especifica as permissões de um usuário definindo um dos seguintes níveis de permissão. Cada nível representa uma política do IAM que concede permissões para um conjunto padrão de ações.

- Deny nega permissão para qualquer interação com a pilha.
- Show concede permissão para a visualização da configuração da pilha, mas não permite modificações no estado dela.
- Deploy inclui as permissões Show e também concede as permissões de usuário para a implantação de aplicativos.
- Manage inclui as permissões Deploy e também permite que o usuário execute diferentes ações de gerenciamento de pilha, como a criação ou exclusão de instâncias e camadas.

> **Note**
>
> O nível Gerenciar permissões não concede permissões para um pequeno número de ações de AWS OpsWorks pilhas de alto nível, incluindo criar ou clonar pilhas. Você deve usar uma política do IAM para conceder tais permissões.

Além de definir os níveis de permissões, você também pode usar a página de Permissões de uma pilha para especificar se os usuários têm SSH/RDP and sudo/admin privilégios nas instâncias da pilha. Para obter mais informações sobre o gerenciamento de permissões no AWS OpsWorks Stacks, consulte Concessão de permissões por pilha. Para obter mais informações sobre o gerenciamento de acesso a SSH, consulte Gerenciamento do acesso por SSH

Gerenciamento de permissões do IAM

Com o gerenciamento de permissões do IAM, você pode usar o console, API ou CLI do IAM para anexar uma política em formato JSON que especifique explicitamente as permissões de um usuário. Para obter mais informações sobre o gerenciamento de permissões do IAM, consulte O que é o IAM?

Recomendação: comece com o gerenciamento de permissões do AWS OpsWorks Stacks. Se você precisar ajustar as permissões de um usuário ou conceder permissões que não estão incluídas nos níveis de permissão Manage, pode combinar as duas abordagens. AWS OpsWorks Em seguida, o Stacks avalia as duas políticas para determinar as permissões do usuário.

> **Important**
>
> Se um usuário tiver várias políticas com permissões conflitantes, a negação sempre vence. Por exemplo, suponha que você anexa uma política do IAM ao usuário permitindo o acesso a uma determinada pilha, mas também usa a página Permissões dessa pilha para atribuir ao usuário o nível de permissão Negar. O nível de permissão Deny tem precedência, e o usuário não poderá acessar a pilha. Para obter mais informações, consulte Lógica de avaliação de políticas do IAM.

Por exemplo, suponha que você deseja permitir ao usuário a execução da maioria das operações em uma pilha, com exceção da adição ou exclusão de camadas.

- Especifique o nível de permissão Manage, que permite ao usuário executar a maioria das ações de gerenciamento de pilhas, incluindo a criação e a exclusão de camadas.
- Anexe a seguinte política gerenciada pelo cliente ao usuário, que nega permissões para usar as DeleteLayerações CreateLayere nessa pilha. Você identifica a pilha pelo Nome de recurso da Amazon (ARN), que pode ser encontrado na página Settings (Configurações) da pilha.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "opsworks:CreateLayer",
        "opsworks:DeleteLayer"
      ],
      "Resource": "arn:aws:opsworks:*:*:stack/2f18b4cb-4de5-4429-a149-ff7da9f0d8ee/"
    }
  ]
}
```

Para obter mais informações, incluindo exemplos de políticas, consulte Gerenciando permissões de AWS OpsWorks pilhas anexando uma política do IAM.

> **Note**
>
> Outra forma de usar as políticas do IAM é estabelecer uma condição que limita o acesso à pilha para funcionários com um endereço IP ou intervalo de endereços IP especificados. Por exemplo, para garantir que os funcionários tenham acesso às pilhas somente de dentro do seu firewall corporativo, defina uma condição que estabelece um limite de acesso ao intervalo de endereços IP corporativos. Para obter mais informações, consulte Condições.

Melhores Práticas: Gerenciamento e implementação de aplicativos e guias de procedimentos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks implanta aplicativos e livros de receitas em cada nova instância a partir de um repositório remoto. Durante o ciclo de vida de uma instância, você deve atualizar frequentemente os aplicativos ou guias de procedimentos nas instâncias online das pilhas para adicionar recursos, corrigir erros e assim por diante. Existe uma série de maneiras para gerenciar os aplicativos e guias de procedimentos de uma pilha, mas a abordagem usada deve cumprir os seguintes requisitos gerais:

- Todas as instâncias da pilha de produção devem ter o mesmo aplicativo e código do guia de procedimentos personalizado, com exceções limitadas para algumas finalidades, como o teste A/B.
- A implementação de uma atualização não deve interromper a operação do site, mesmo se algo der errado.

Esta seção descreve as práticas recomendadas para o gerenciamento e a implementação de aplicativos e guias de procedimentos personalizados.

Tópicos

- Manutenção da consistência
- Implementação do código em instâncias online

Manutenção da consistência

Em geral, você precisa manter total controle sobre o aplicativo ou código do guia de procedimentos que é executado em sua pilha de produção. Normalmente, todas as instâncias devem executar a versão atualmente aprovada do código. As exceções ocorrem ao atualizar seus aplicativos ou guias de procedimentos, conforme descrito posteriormente, e ao acomodar casos especiais, como a execução do teste A/B.

O código do aplicativo e guia de procedimentos é implementado a partir de um repositório de origem especificado para suas instâncias da pilha de duas formas:

- Quando você inicia uma instância, o AWS OpsWorks Stacks implanta automaticamente o código atual do aplicativo e do livro de receitas na instância.
- Para instâncias online, você deve implementar manualmente o código do aplicativo ou guia de procedimentos atual, executando um Comando de implementação (para aplicativos) ou um Comando de atualização de guias de procedimentos personalizados (para guias de procedimentos).

Como existem dois mecanismos de implementação, é essencial que você gerencie seu código-fonte cuidadosamente para evitar a execução involuntária de um código diferente em instâncias diferentes. Por exemplo, se você implanta aplicativos ou livros de receitas de uma ramificação mestre do Git AWS OpsWorks , o Stacks implanta o que está nessa ramificação no momento. Se você atualizar o código no branch mestre e, em seguida, iniciar uma nova instância, essa instância terá uma versão mais recente do código em relação às instâncias mais antigas. A versão mais recente pode nem estar aprovada para produção.

Recomendação: arquivos Amazon S3

Para garantir que todas as suas instâncias tenham a versão de código aprovada, recomendamos a implantação dos seus aplicativos e livros de receitas a partir de um arquivo do Amazon Simple Storage Service (Amazon S3). Isso garante que o código é um artefato estático, um .zip ou outro ficheiro de arquivo, que deve ser atualizado de forma explícita. Além disso, o Amazon S3 é altamente confiável, para que você raramente, se nunca, encontre dificuldades para acessar o arquivo. Para garantir ainda mais a consistência, atribua versões explicitamente a cada arquivo usando uma convenção de nomenclatura ou através do Versionamento do Amazon S3, que fornece uma trilha de auditoria e uma maneira fácil de reverter para uma versão anterior.

Por exemplo, você pode criar uma estrutura de implementação utilizando a ferramenta Jenkins. Assim que o código que você deseja implementar for comprometido e testado, crie um ficheiro de arquivos e faça o upload para o Amazon S3. Todas as implantações de aplicativos ou atualizações de livros de receita instalarão o código nesse arquivo compactado, e cada instância terá o mesmo código.

Recomendação: Repositórios Git ou Subversão

Se você preferir usar um repositório Git ou Subversão, não implemente a partir do branch mestre. Em vez disso, marque a versão aprovada e especifique essa versão como a fonte do aplicativo ou guia de procedimentos.

Implementação do código em instâncias online

AWS OpsWorks O Stacks não implanta automaticamente o código atualizado em instâncias on-line. Você deve executar essa operação manualmente, o que implica nos seguintes desafios:

- A implementação eficiente da atualização, sem comprometer a habilidade do site de lidar com solicitações de clientes durante o processo de implementação.

- O tratamento de uma implementação malsucedida, seja devido a problemas com o aplicativo ou guia de procedimentos implementado, ou devido a problemas com o processo de implementação em si.

A abordagem mais simples é executar um Comando de implementação (para aplicativos) ou Comando de atualização de guias de procedimentos personalizados (para guias de procedimentos) padrão, que implementa a atualização simultaneamente em todas as instâncias. Essa abordagem é simples e rápida, mas não há margem para erro. Se a implementação falhar ou o código atualizado possuir qualquer problema, cada instância da sua pilha de produção pode ser afetada, o que pode interromper ou desativar seu site até que o problema seja corrigido ou aconteça a reversão para a versão anterior.

Recomendação: Use uma estratégia de implementação robusta, o que permite que as instâncias que executam a versão antiga do código possam continuar lidando com as solicitações até que ocorra a verificação de que a implementação foi bem-sucedida e todo o tráfego de entrada pode ser transferido com confiança para a nova versão.

As seções a seguir oferecem dois exemplos de estratégias de implementação robustas, seguidos por uma discussão sobre como gerenciar um banco de dados de backend durante a implementação. Em resumo, eles descrevem as atualizações do aplicativo, mas você pode usar estratégias semelhantes para os livros de receitas.

Tópicos

- Utilização de uma implementação rolling
- Uso de pilhas separadas
- Gerenciamento de um banco de dados backend

Utilização de uma implementação rolling

Uma implementação rolling atualiza um aplicativo nas instâncias do servidor de aplicativos online da pilha em várias fases. Com cada fase, você atualiza um subconjunto de instâncias online e verifica se a atualização foi bem-sucedida antes de iniciar a próxima fase. Se encontrar problemas, as instâncias que ainda estão executando a versão antiga do aplicativo podem continuar lidando com o tráfego de entrada até que você resolva os problemas.

O exemplo a seguir considera que você está usando a prática recomendada de distribuir as instâncias do servidor de aplicativos da pilha em várias Zonas de Disponibilidade.

Para executar uma implementação rolling

1. Na página Deploy App, selecione Advanced, escolha uma única instância do servidor de aplicativos e implemente o aplicativo para essa instância.

   Se você quiser ser cauteloso, você pode remover a instância do load balancer antes de implementar o aplicativo. Isso garante que os usuários não irão encontrar o aplicativo atualizado até que seja verificado o seu funcionamento correto. Se você usar o Elastic Load Balancing, remova a instância do balanceador de carga usando o console, a CLI ou um SDK do Elastic Load Balancing.

2. Verifique se o aplicativo atualizado está funcionamento corretamente e se a instância possui índices de desempenho aceitáveis.

   Se você removeu a instância de um balanceador de carga do Elastic Load Balancing, use o console, CLI ou um SDK do Elastic Load Balancing para restaurar. A versão atualizada do aplicativo está, agora, lidando com as solicitações de usuários.

3. Implemente a atualização para o restante das instâncias na Zona de Disponibilidade e verifique se elas estão funcionando corretamente e se possuem os índices aceitáveis.

4. Repita a etapa 3 para as outras Zonas de Disponibilidade da pilha, uma zona por vez. Se você quiser ser especialmente cauteloso, repita as etapas 1 a 3.

> **Note**
>
> Caso use um balanceador de carga do Elastic Load Balancing, você pode usar sua verificação de integridade para verificar se a implantação foi bem-sucedida. No entanto, defina o caminho de ping para um aplicativo que confira as dependências e verifique se tudo está trabalhando corretamente, e não um arquivo estático que apenas confirma se o servidor do aplicativo está em execução.

Uso de pilhas separadas

Outra abordagem para o gerenciamento de aplicativos é usar uma pilha separada para cada fase do ciclo de vida do aplicativo. As diversas pilhas são, geralmente, chamadas de ambientes. Esse arranjo permite que você execute o desenvolvimento e os testes em pilhas que não estão acessíveis publicamente. Quando estiver pronto para implementar uma atualização, troque o tráfego

de usuários da pilha que hospeda a versão atual do aplicativo, para a pilha que hospeda a versão atualizada.

Tópicos

- Uso de pilhas de desenvolvimento, preparação e produção
- Uso de uma estratégia de implementação azul-verde

Uso de pilhas de desenvolvimento, preparação e produção

A abordagem mais comum usa as seguintes pilhas.

Pilha de desenvolvimento

Use uma pilha de desenvolvimento para tarefas como a implementação de novos recursos ou correção de erros. Uma pilha de desenvolvimento é, essencialmente, um protótipo da pilha de produção, com as mesmas camadas, aplicativos, recursos, etc., inclusos na sua pilha de produção. Como a pilha de desenvolvimento normalmente não precisa lidar com a mesma carga da pilha de produção, você normalmente pode usar menos instâncias, ou instâncias menores.

As pilhas de desenvolvimento não são abertas ao público; você controla o acesso da seguinte forma:

- Restrinja o acesso à rede, configurando as regras de entrada do grupo de segurança do load balancer ou do servidor do aplicativo para aceitar apenas solicitações de endereços IP, ou intervalos de endereços, específicos.

  Por exemplo, limite o acesso HTTP, HTTPS e SSH para endereços dentro do seu intervalo de endereços corporativo.
- Controle o acesso à funcionalidade de gerenciamento de AWS OpsWorks pilhas usando a página de permissões da pilha.

  Por exemplo, conceder o nível de permissões Gerenciar para a equipe de desenvolvimento, e o nível Mostrar para todos os outros funcionários.

Pilha de preparação

Use uma pilha de preparação para testar e finalizar candidatos para uma pilha de produção atualizada. Ao concluir o desenvolvimento, crie uma pilha de preparação através da clonagem da pilha de desenvolvimento. Em seguida, execute o pacote de testes na pilha de preparação e implemente as atualizações nessa pilha para corrigir problemas que surgirem.

As pilhas de preparação também não são abertas ao público; você controla o acesso à pilha e à rede da mesma forma que faria para a pilha de desenvolvimento. Observe que, ao clonar uma pilha de desenvolvimento para criar uma pilha de teste, você pode clonar as permissões concedidas pelo gerenciamento de permissões de pilhas. AWS OpsWorks No entanto, a clonagem não afeta as permissões concedidas por políticas IAM dos usuários. Você deve usar o console, CLI ou um SDK do IAM para modificar essas permissões. Para obter mais informações, consulte Gerenciamento de permissões de usuário.

Pilha de produção

A pilha de produção é a pilha aberta ao público que oferece suporte ao seu aplicativo atual. Quando a pilha de preparação passar nos testes, você promove-a para produção e retira a pilha de produção antiga. Para obter um exemplo de como fazer isso, consulte Uso de uma estratégia de implementação azul-verde.

> **Note**
>
> Em vez de usar o console AWS OpsWorks Stacks para criar pilhas manualmente, crie um AWS CloudFormation modelo para cada pilha. Essa abordagem tem as seguintes vantagens:
>
> - Velocidade e conveniência: ao iniciar o modelo, o AWS CloudFormation cria automaticamente a pilha, incluindo todas as instâncias necessárias.
> - Consistência: armazene o modelo de cada pilha em seu repositório de origem para garantir que os desenvolvedores usem a mesma pilha para a mesma finalidade.

Uso de uma estratégia de implementação azul-verde

Uma estratégia de implementação azul-verde é uma maneira comum de usar pilhas separadas de modo eficiente, para implementar uma atualização do aplicativo na produção.

- O ambiente azul é a pilha de produção, que hospeda o aplicativo atual.
- O ambiente verde é a pilha de preparação, que hospeda o aplicativo atualizado.

Quando você estiver pronto para implementar o aplicativo atualizado para a produção, basta alternar o tráfego de usuários da pilha azul para a pilha verde, que se torna a nova pilha de produção. Você então retira a antiga pilha azul.

O exemplo a seguir descreve como executar uma implementação azul-verde com pilhas do AWS OpsWorks Stacks em conjunto com o Route 53 e um grupo de balanceadores de carga do Elastic Load Balancing. Antes de fazer a inversão, você deve garantir o seguinte:

- A atualização do aplicativo na pilha verde passou nos testes e está pronta para a produção.
- A pilha verde é idêntica à azul, exceto pelo fato de incluir a aplicativo atualizado e não ser aberta ao público.

  Ambas as pilhas têm as mesmas permissões, o mesmo número e tipo de instâncias em cada camada, a mesma configuração com base no tempo e carga, e assim por diante.
- Todas as instâncias 24/7 e as instâncias com base no tempo agendadas da pilha verde estão online.
- Você tem um grupo de balanceadores de carga do Elastic Load Balancing que podem ser conectados dinamicamente a uma camada em ambas as pilhas e pode ser pré-aquecido para processar o volume de tráfego esperado.
- Você usou o atributo de roteamento ponderado Route 53 para criar um conjunto de registros em uma zona hospedada que inclui seu grupo de balanceadores de carga.
- Você atribuiu um peso diferente de zero para o load balancer que está conectado à camada do servidor de aplicativo da sua pilha azul, e peso zero para os load balancers não utilizados. Isso garante que o load balancer da pilha azul lida com todo o tráfego de entrada.

Para transferir os usuários para a pilha verde

1. Anexe um dos load balancers não utilizados do grupo na camada do servidor de aplicativo da pilha verde. Em alguns casos, como quando você espera tráfego instantâneo, ou se não for possível configurar um teste de carga para aumentar o tráfego gradualmente, prepare o load balancer para lidar com o tráfego esperado.
2. Após todas as instâncias da pilha verde terem passado pela verificação de integridade do Elastic Load Balancing, altere os pesos no conjunto de registros do Route 53, de modo que o balanceador de carga da pilha verde tenha peso zero e o balanceador de carga da pilha azul tenha um peso reduzido correspondente. Recomendamos que você comece deixando a pilha verde cuidar de uma pequena porcentagem das solicitações, talvez 5%, com a pilha azul cuidando do resto. Agora você tem duas pilhas de produção, com a pilha verde cuidando de algumas solicitações e a pilha azul cuidando do restante.
3. Monitore os índices de desempenho da pilha verde. Se eles estão aceitáveis, aumente o peso da pilha verde para que ela cuide de, talvez, 10% do tráfego de entrada.

4. Repita a etapa 3 até que a pilha verde esteja cuidando de aproximadamente metade do tráfego de entrada. Quaisquer possíveis problemas devem ter emergido até esse momento, portanto, se a pilha verde está funcionando de forma aceitável, você pode concluir o processo, reduzindo o peso da pilha azul para zero. A pilha verde é agora a nova pilha azul, e está cuidando de todo o tráfego de entrada.
5. Separe o load balancer da camada do servidor de aplicativo da antiga pilha azul e devolva-o para o grupo.
6. Embora a antiga pilha azul não esteja mais cuidando das solicitações de usuários, recomendamos mantê-la por um tempo caso ocorram problemas com a nova pilha azul. Neste caso, você pode reverter a atualização, revertendo o procedimento para direcionar o tráfego de entrada de volta para a antiga pilha azul. Quando estiver certo de que a nova pilha azul está operando de forma aceitável, encerre a antiga pilha azul.

Gerenciamento de um banco de dados backend

Se seu aplicativo depender de um banco de dados de back-end, você precisará fazer a transição do aplicativo antigo para o novo. AWS OpsWorks O Stacks oferece suporte às seguintes opções de banco de dados.

Camada Amazon RDS

Com uma camada do Amazon Relational Database Service (Amazon RDS), você cria a instância de banco de dados RDS separadamente e, em seguida, registra em sua pilha. Você pode registrar uma instância de banco de dados RDS em apenas uma pilha por vez, mas você pode alterar uma instância de banco de dados RDS de uma pilha para outra.

AWS OpsWorks O Stacks instala um arquivo com os dados de conexão em seus servidores de aplicativos em um formato que pode ser facilmente usado pelo seu aplicativo. AWS OpsWorks O Stacks também adiciona as informações de conexão do banco de dados aos atributos de configuração e implantação da pilha, que podem ser acessados por receitas. Você também pode usar JSON para fornecer dados de conexão para aplicativos. Para obter mais informações, consulte Conectar-se a um banco de dados.

Atualizar um aplicativo que depende de um banco de dados apresenta dois desafios básicos:

- Garantir que toda transação é devidamente registrada durante a transição e, ao mesmo tempo, evitar condições de corrida entre as versões nova e antiga do aplicativo.

- Executar a transição de uma forma que limite o impacto sobre o desempenho do seu site e minimize ou elimine o tempo de inatividade.

Ao usar as estratégias de implementação descritas neste tópico, não é possível simplesmente desconectar o banco de dados do aplicativo antigo e reconectá-lo ao novo. Ambas as versões do aplicativo são executadas em paralelo durante a transição e devem ter acesso aos mesmos dados. O conteúdo a seguir descreve duas abordagens para gerenciar a transição, assim como suas vantagens e desafios.

Abordagem 1: Tenha ambos os aplicativos conectados ao mesmo banco de dados

Vantagens

- Não há tempo de inatividade durante a transição.

  Um aplicativo para gradualmente de acessar o banco de dados, enquanto o outro assume gradualmente.
- Não é necessário sincronizar os dados entre dois banco de dados.

Desafios

- Ambos os aplicativos acessam o mesmo banco de dados, portanto é necessário gerenciar o acesso para evitar perda ou corrupção de dados.
- Se precisar migrar para um novo esquema de banco de dados, a versão antiga do aplicativo deve ser capaz de usar o novo esquema.

Se você estiver usando pilhas separadas, essa abordagem é provavelmente mais adequada para o Amazon RDS, pois a instância não está permanentemente vinculada a uma determinada pilha e pode ser acessada por aplicativos em execução em pilhas diferentes. No entanto, não é possível registrar uma instância de banco de dados RDS com mais de uma pilha por vez, de modo que é necessário fornecer dados de conexão para ambos os aplicativos, por exemplo usando JSON. Para obter mais informações, consulte Usando uma receita predefinida.

Se você realiza atualização frequente, as versões antiga e nova do aplicativo são hospedadas na mesma pilha, para que você possa usar uma camada do Amazon RDS ou MySQL.

Abordagem 2: Fornecer um banco de dados para cada versão do aplicativo

Vantagens

- Cada versão tem seu próprio banco de dados, portanto os esquemas não precisam ser compatíveis.

Desafios

- Sincronizar os dados entre os dois bancos de dados durante a transição sem perder ou danificar os dados.
- Garantir que o procedimento de sincronização não cause tempo de inatividade significativo ou diminua significativamente o desempenho do site.

Se você estiver usando pilhas separadas, cada uma tem seu próprio banco de dados. Se você estiver usando implementação frequente, você pode anexar dois bancos de dados à pilha, um para cada aplicativo. Se os aplicativos antigo e atualizado não possuem esquemas de banco de dados compatíveis, essa abordagem é melhor.

Recomendação: em geral, recomendamos o uso de uma camada do Amazon RDS como banco de dados de back-end de um aplicativo, pois ela é mais flexível e pode ser usada para qualquer cenário de transição. Para obter mais informações sobre como lidar com transições, consulte o Guia do usuário Amazon RDS.

Empacotar dependências do livro de receitas localmente

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode usar o Berkshelf para empacotar as dependências do livro de receitas localmente, fazer o upload do pacote para o Amazon S3 e modificar sua pilha para usar o pacote no Amazon S3 como fonte do livro de receitas. O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

As orientações a seguir descrevem como pré-empacotar seus livros de receitas e suas dependências em um arquivo .zip e, em seguida, usar o arquivo como sua fonte do livro de receitas das instâncias do Linux no Stacks. AWS OpsWorks A primeira demonstração descreve como

empacotar um livro de receitas. A segunda demonstração descreve como empacotar vários livros de receitas.

Antes de começar, instale o Kit de desenvolvimento do Chef (também conhecido como Chef DK), uma variedade de ferramentas construídas pela comunidade do Chef. Você precisará dele para usar a `chef` ferramenta da linha de comando.

Empacotar dependências do livro de receitas localmente no Chef 12

No Chef 12 Linux, o Berkshelf não está mais instalado por padrão nas instâncias de pilha. Recomendamos que você instale e use o Berkshelf em um computador de desenvolvimento local para empacotar as dependências do livro de receitas localmente. Carregue seu pacote, com as dependências incluídas, no Amazon S3. Por fim, modifique a pilha do Chef 12 Linux para usar o pacote carregado como uma origem de livro de receitas. Esteja ciente das seguintes diferenças ao criar pacotes de livros de receitas no Chef 12.

1. No computador local, crie um livro de receitas executando a ferramenta da linha de comando do `chef`.

   ```
   chef generate cookbook "server-app"
   ```

   Este comando cria um livro de receitas, um Berksfile, um arquivo `metadata.rb` e um diretório de receitas e os coloca em uma pasta com o mesmo nome do livro de receitas. O exemplo a seguir mostra a estrutura do que é criado.

   ```
   server-app <-- the cookbook you've just created
       ### Berksfile
       ### metadata.rb
       ### recipes
   ```

2. Em um editor de texto, edite o Berksfile para apontar para os livros de receitas dos quais o livro de receitas `server-app` dependerá. Em nosso exemplo, queremos que `server-app` dependa do livro de receitas `java` do Chef Supermarket. Estamos especificando a versão 1.50.0 ou a versão secundária mais recente, mas você pode especificar qualquer versão publicada em aspas simples. Salve as alterações e feche o arquivo.

   ```
   source 'https://supermarket.chef.io'
   cookbook 'java', '~> 1.50.0'
   ```

3. Edite o arquivo `metadata.rb` para adicionar a dependência. Salve as alterações e feche o arquivo.

```
depends 'java' , '~> 1.50.0'
```

4. Vá para o diretório de receitas `server-app` que o Chef criou para você e, em seguida, execute o comando `package` para criar um arquivo `tar` do livro de receitas. Se você estiver criando pacotes de vários livros de receita, convém executar esse comando no diretório raiz em que todos os livros de receita estão armazenados. Para criar pacote com um único livro de receitas, execute esse comando no nível do diretório do livro de receitas. Neste exemplo, executamos o comando no diretório `server-app`.

```
berks package cookbooks.tar.gz
```

A saída será semelhante à seguinte. O arquivo `tar.gz` é criado no seu diretório local.

```
Cookbook(s) packaged to /Users/username/tmp/berks/cookbooks.tar.gz
```

5. Na AWS CLI, faça upload do pacote recém-criado no Amazon S3. Anote o novo URL do pacote do livro de receitas. Depois carregá-lo no S3;, você precisará do URL para as configurações da pilha.

```
aws s3 cp cookbooks.tar.gz s3://bucket-name/
```

A saída será semelhante à seguinte.

```
upload: ./cookbooks.tar.gz to s3://bucket-name/cookbooks.tar.gz
```

6. Em AWS OpsWorks Stacks, modifique sua pilha para usar o pacote que você enviou como fonte do livro de receitas.

   a. Defina a configuração Use custom Chef cookbooks como Yes.

   b. Defina Repository type como S3 Archive.

   c. Em Repository URL, cole o URL do pacote do livro de receitas que você carregou na etapa 5.

Salve as alterações feitas na pilha.

Empacotar dependências do livro de receitas localmente para um livro de receitas

1. No computador local, crie um livro de receitas usando a ferramenta da linha de comando do Chef:

   ```
   chef generate cookbook "server-app"
   ```

   Este comando cria um livro de receitas e um Berksfile e os coloca em uma pasta com o mesmo nome do livro de receitas.
2. Vá para o diretório de receitas que o Chef criou para você e, em seguida, empacote tudo, executando o seguinte comando:

   ```
   berks package cookbooks.tar.gz
   ```

   O resultado se parece com:

   ```
   Cookbook(s) packaged to /Users/username/tmp/berks/cookbooks.tar.gz
   ```
3. Na AWS CLI, faça upload do pacote recém-criado no Amazon S3:

   ```
   aws s3 cp cookbooks.tar.gz s3://bucket-name/
   ```

   O resultado se parece com:

   ```
   upload: ./cookbooks.tar.gz to s3://bucket-name/cookbooks.tar.gz
   ```
4. Em AWS OpsWorks Stacks, modifique sua pilha para usar o pacote que você enviou como fonte do livro de receitas.

Empacotar dependências do livro de receitas localmente para vários livros de receitas

Este exemplo cria dois livros de receitas e empacota as dependências para eles.

1. No computador local, execute os seguintes `chef` comandos para gerar dois livros de receitas:

   ```
   chef generate cookbook "server-app"
   chef generate cookbook "server-utils"
   ```

Neste exemplo, o servidor de aplicativos do livro de receitas executa as configurações de Java, por isso, é preciso adicionar uma dependência de Java.

2. Edite `server-app/metadata.rb` para adicionar uma dependência no livro de receitas da comunidade do Java:

```
maintainer "The Authors"
maintainer_email "you@example.com"
license "all_rights"
description "Installs/Configures server-app"
long_description "Installs/Configures server-app"
version "0.1.0"
depends "java"
```

3. Informe ao Berkshelf o que empacotar editando o arquivo Berksfile no livro de receitas no diretório raiz da seguinte forma:

```
source "https://supermarket.chef.io"
cookbook "server-app", path: "./server-app"
cookbook "server-utils", path: "./server-utils"
```

A sua estrutura de arquivo agora tem a seguinte aparência:

```
..
   ### Berksfile
   ### server-app
   ### server-utils
```

4. Por fim, crie um pacote zip, faça seu upload no Amazon S3 e modifique sua pilha do AWS OpsWorks Stacks para utilizar a nova fonte do livro de receitas. Para fazer isso, siga as etapas 2 a 4 em Empacotar dependências do livro de receitas localmente para um livro de receitas.

Recursos adicionais

Para obter mais informações sobre a criação de pacotes das dependência do livro de receitas, consulte os tópicos a seguir.

- Como empacotar dependências de livros de receitas localmente com a Berkshelf no blog da AWS DevOps
- Linux Chef 12 com Berkshelf nos fóruns AWS OpsWorks

- Berkshelf no Chef 12 nos fóruns AWS OpsWorks
- Instalação de livros de receitas personalizados neste guia
- Repositórios de livro de receitas neste guia

Pilhas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A pilha é a entidade AWS OpsWorks Stacks de nível superior. Isso representa um conjunto de instâncias para gerenciar de maneira coletiva, normalmente porque elas têm uma finalidade em comum, tal como atender a aplicativos PHP. Uma pilha serve como um contêiner e lida com tarefas que se aplicam ao grupo de instâncias como um todo, tais como gerenciar aplicativos e livros de receitas.

Por exemplo, uma pilha cujo objetivo seja atender as aplicações web poderá se parecer com o seguinte:

- Um conjunto de instâncias do servidor de aplicações, sendo que cada uma delas lida com uma parte do tráfego de entrada.
- Uma instância de load balancer, que usa o tráfego de entrada e o distribui em todos os servidores de aplicativos.
- Uma instância de banco de dados, que serve como um armazenamento de dados de back-end para os servidores de aplicativos.

Uma prática comum é ter várias pilhas que representam diferentes ambientes. Um conjunto típico de pilhas consiste em:

- Uma pilha de desenvolvimento a ser usado pelos desenvolvedores para adicionar recursos, corrigir erros e executar outras tarefas de desenvolvimento e manutenção.

- Uma pilha de preparação para verificar atualizações ou correções antes de expô-las publicamente.
- Uma pilha de produção, a qual é a versão aberta ao público que lida com as solicitações de entrada dos usuários.

Esta seção descreve os conceitos básicos para trabalhar com pilhas.

Tópicos

- Migrar pilhas do Amazon EC2 Classic para uma VPC
- Criar uma nova pilha
- Execução de uma Stack em uma VPC
- Atualizar uma pilha
- Clonar uma pilha
- Executar comandos AWS OpsWorks do Stacks Stack
- Usar JSON personalizado
- Excluir uma pilha

Migrar pilhas do Amazon EC2 Classic para uma VPC

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico descreve como migrar uma AWS OpsWorks Stacks pilha da plataforma de rede do Amazon EC2 Classic para uma rede da Amazon Virtual Private Cloud (Amazon VPC).

Caso você tenha criado sua AWS conta da antes de 4/12/2013, talvez ela seja compatível com a plataforma EC2 -Classic em algumas regiões. AWS Alguns EC2 recursos e funções da Amazon, como redes aprimoradas e tipos de instância mais novos, precisam de uma virtual private cloud (VPC). Alguns recursos podem ser compartilhados entre o EC2 -Classic e uma VPC, e alguns não

podem. Para evitar interrupções no serviço, recomendamos que você migre suas AWS OpsWorks Stacks pilhas para uma VPC.

Tópicos

- Pré-requisitos
- Migre uma AWS OpsWorks Stacks pilha para uma VPC
- Consulte também

Pré-requisitos

Antes de começar, é necessário ter uma VPC que atenda aos requisitos de AWS OpsWorks Stacks configuração. Para configurar sub-redes privadas em sua VPC para AWS OpsWorks Stacks, consulte Execução de uma Stack em uma VPC neste guia. É possível criar uma VPC personalizada usando o gerenciamento de console da Amazon VPC. Para obter mais informações, consulte as configurações VPCs e sub-redes do assistente do console Amazon VPC no Guia do usuário da Amazon Virtual Private Cloud.

Para continuar com a migração, você precisará do ID da VPC e do ID da sub-rede que deseja usar.

Migre uma AWS OpsWorks Stacks pilha para uma VPC

Primeiro, clone uma pilha EC2 -Classic existente usando o AWS OpsWorks Stacks console ou a API. Em seguida, mova os recursos da pilha existente para a nova pilha. Inicie as novas instâncias na pilha clonada e implante aplicativos. Verifique se a nova pilha está funcionando. Por fim, exclua os recursos EC2 -Classic da pilha EC2 -Classic e, em seguida, exclua pilha antiga.

1. Clone sua pilha EC2 -Classic existente em sua VPC. A clonagem da pilha copia as configurações, camadas, aplicativos, usuários e permissões de usuário da pilha para a nova pilha. Para obter mais informações sobre como clonar uma pilha, consulte Clonar uma pilha neste guia.

   Você também pode clonar uma pilha usando a AWS OpsWorks Stacks API. Ao clonar uma pilha usando o AWS CLI ou AWS SDKs, defina o valor do `VpcId` parâmetro como o ID da VPC que você criou. Pré-requisitos Para obter mais informações, consulte CloneStack na Referência de APIs do AWS OpsWorks Stacks .
2. Crie novas instâncias nas camadas da pilha clonada. Não deixe de especificar a ID da sub-rede que você criou em Pré-requisitos. Para obter mais informações sobre como criar instâncias em uma pilha, consulte este Adicionar uma instância a uma camada guia.

3. Migre seus recursos clássicos, como grupos de EC2 segurança, balanceadores de carga do Elastic Load Balancing e endereços IP elásticos para sua VPC e, em seguida, associe-os à pilha clonada. Para obter mais informações, consulte Migrar seus recursos para uma VPC no Guia do usuário da EC2 Amazon.
4. Registre volumes do Amazon EBS e instâncias do Amazon RDS com a pilha clonada. Para obter mais informações sobre como registrar recursos com uma pilha, consulte Como registrar recursos com uma pilha neste guia.

   Os volumes do Amazon EBS não estão associados a uma VPC, e você pode usá-los em várias instâncias em pilhas do -Classic EC2 e em pilhas em uma VPC. Você pode registrar instâncias do Amazon RDS em EC2 -Classic com pilhas EC2 -Classic e pilhas em uma VPC.
5. Inicie instâncias na pilha clonada e, em seguida, mova uma pequena porcentagem de suas cargas de trabalho para a pilha clonada. Por exemplo, mova uma pequena porcentagem do tráfego para os balanceadores de carga do Elastic Load Balancing na pilha clonada. Se você estiver usando o Amazon Route 53, consulte Como rotear o tráfego para um balanceador de carga do ELB no Guia do desenvolvedor do Amazon Route 53.

   Direcione somente uma pequena porcentagem do tráfego até ter certeza de que a nova pilha está funcionando e dá suporte aos seus aplicativos. Deixe a nova pilha funcionar com uma pequena porcentagem de tráfego por um período de teste, como uma semana. Depois de verificar se a nova pilha está funcionando, direcione o tráfego restante para a pilha.
6. Depois de ter certeza de que a pilha clonada está funcionando, mova o restante do tráfego ou das cargas de trabalho de produção para a pilha clonada. Agora é possível interromper as instâncias na pilha EC2 -Classic. Recomendamos que você mantenha a pilha antiga disponível por várias semanas, para poder mover as cargas de trabalho de volta para a pilha antiga caso ocorra algum problema com a nova pilha nas semanas após a migração.
7. Quando a nova pilha estiver funcionando por várias semanas, exclua instâncias na pilha EC2 -Classic. Para obter mais informações sobre como excluir instâncias, consulte Excluindo instâncias do AWS OpsWorks Stacks neste guia.

   > **Important**
   >
   > Não use o EC2 console ou a API da Amazon ou a para interromper ou excluir AWS OpsWorks instâncias.
8. Exclua os aplicativos na pilha do EC2 -Classic. Para obter mais informações sobre como excluir aplicativos, consulte Excluir o aplicativo da pilha neste guia.

9. Exclua a a a a a a a pilha EC2 -Classic. Para obter mais informações sobre como excluir uma pilha, consulte Excluir uma pilha neste guia.

Consulte também

- Migrar do EC2 -Classic para uma VPC
- Guia de depuração e solução de problemas
- Execução de uma Stack em uma VPC

Criar uma nova pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para criar uma nova pilha, no painel Pilhas, clique em Adicionar AWS OpsWorks pilha. Em seguida, você pode usar a página Add Stack para configurar a pilha. Quando terminar, clique em Add Stack.

Tópicos

- Escolher o tipo de pilha para criar
- Opções básicas
- Opções avançadas

Escolher o tipo de pilha para criar

Antes de criar uma pilha, você deve decidir o tipo de pilha que deseja criar. Para obter ajuda, consulte a tabela a seguir.

Se você deseja criar...	Crie esse tipo de pilha se você deseja...	Para saber como, siga estas instruções:
Um exemplo de pilha	Explore os conceitos básicos da AWS OpsWorks com uma pilha Chef 12 baseada em Linux e uma amostra do aplicativo Node.js.	Conceitos básicos: exemplo
Uma pilha do Chef 12 baseada em Linux	Crie uma pilha baseada em Linux que use a versão mais recente do Chef compatíve l com a AWS. OpsWorks Escolha essa opção se você é um usuário avançado do Chef e deseja aproveitar a ampla seleção de livros de receitas da comunidade ou criar seus próprios livros de receitas personalizados. Para obter mais informações, consulte Chef 12 Linux.	Conceitos básicos: Linux
Uma pilha do Chef 12.2 baseada em Windows	Crie uma pilha baseada no Windows.	Conceitos básicos: Windows
Uma pilha do Chef 11.10 baseada em Linux	Crie essa pilha se sua organização exigir o uso do Chef 11.10 com Linux para compatibilidade com versões anteriores.	Conceitos básicos das pilhas Linux do Chef 11

Opções básicas

A página Add Stack tem as seguintes opções básicas.

Nome da pilha

(Obrigatório) Um nome usado para identificar a pilha no console AWS OpsWorks Stacks. O nome não precisa ser exclusivo. AWS OpsWorks O Stacks também gera um ID de pilha, que é uma GUID que identifica exclusivamente a pilha. Por exemplo, com comandos da CLI da AWS, como update-stack, você pode usar o ID da pilha para identificar a pilha particular. Depois de criar uma pilha, você pode localizar seu ID escolhendo Stack no painel de navegação e, em seguida, escolhendo Stack Settings. O ID é rotulado como OpsWorks ID.

Região

(Obrigatório) A região da AWS nas quais as instâncias serão executadas.

VPC

(Opcional) O ID da VPC no qual a pilha deve ser inicializada. Todas as instâncias serão executadas nessa VPC, e você não pode alterar o ID posteriormente.

- Se sua conta for compatível com EC2 Classic, você poderá especificar Sem VPC (o valor padrão) se não quiser usar uma VPC.

  Para obter mais informações sobre o EC2 Classic, consulte Plataformas suportadas.

- Se sua conta não for compatível com EC2 Classic, você precisa especificar uma VPC.

  A configuração padrão é VPC padrão, que combina a facilidade de uso do EC2 Classic com os benefícios dos recursos de rede VPC. Se você deseja executar a pilha em uma VPC normal, precisa criá-la usando o console da VPC, API ou CLI. Para obter mais informações sobre como criar uma VPC para uma pilha do AWS OpsWorks Stacks, consulte Execução de uma Stack em uma VPC. Para obter informações gerais, consulte Amazon Virtual Private Cloud.

Zona de disponibilidade padrão/sub-rede padrão

(Opcional) Esta configuração depende se você está criando a pilha em uma VPC:

- Se sua conta for compatível com EC2 Classic e você definir VPC como Sem VPC, essa configuração será chamada de Zona de disponibilidade padrão, que especifica a zona de disponibilidade padrão da AWS em que as instâncias serão lançadas.
- Se sua conta não oferecer suporte ao EC2 Classic ou se você optar por especificar uma VPC, esse campo será denominado Sub-rede padrão, que especifica a sub-rede padrão em que as instâncias serão executadas. Você pode executar uma instância em outras sub-redes anulando esse valor na criação da instância. Cada sub-rede é associada a uma zona de disponibilidade.

Você pode fazer com que o AWS OpsWorks Stacks execute uma instância em uma zona de disponibilidade ou sub-rede diferente substituindo essa configuração ao criar a instância.

Para obter mais informações sobre como executar uma pilha em uma VPC, consulte Execução de uma Stack em uma VPC.

Sistema operacional padrão

(Opcional) O sistema operacional que é instalado por padrão em cada instância. Você tem as seguintes opções:

- Um dos sistemas operacionais Linux integrados.
- Microsoft Windows Server 2012 R2.
- Uma AMI personalizada com base em um dos sistemas operacionais compatíveis.

  Se você seleciona Use custom AMI, o sistema operacional é determinado por uma AMI personalizada que você especifica ao criar instâncias. Para obter mais informações, consulte Usando o Custom AMIs.

Para obter mais informações sobre os sistemas operacionais disponíveis, consulte AWS OpsWorks Sistemas operacionais de pilha.

> **Note**
>
> Você pode substituir o sistema operacional padrão quando cria uma instância. No entanto, não pode substituir um sistema operacional Linux para especificar o Windows, ou o Windows para especificar um sistema operacional Linux.

Chave SSH padrão

(Opcional) Um par de EC2 chaves da Amazon na região da pilha. O valor padrão é none (nenhum). Se você especificar um par de chaves, o AWS OpsWorks Stacks instala a chave a pública na instância.

- Com instâncias do Linux, você pode usar a chave privada com um cliente SSH para fazer login nas instâncias da pilha.

  Para obter mais informações, consulte Login com SSH.
- Com instâncias do Windows, você pode usar a chave privada com o EC2 console Amazon ou CLI para recuperar a senha de administrador de uma instância.

Em seguida, você pode usar essa senha com um cliente RDP para fazer login na instância como administrador. Para obter mais informações, consulte Login com RDP.

Para obter mais informações sobre como gerenciar chaves SSH, consulte Gerenciamento do acesso por SSH.

> **Note**
>
> Você pode substituir essa configuração especificando um par de chaves diferente, ou nenhum par de chaves, ao criar uma instância.

Versão do Chef

Mostra a versão do Chef que você escolheu.

Para obter mais informações sobre as versões do Chef, consulte Versões do Chef.

Usar livros de receitas do Chef personalizados

Seja para instalar seus livros de receitas personalizados do Chef nas instâncias da pilha.

Para o Chef 12, a configuração padrão é Yes. Para o Chef 11, a configuração padrão é Não. A opção Sim exibe várias configurações adicionais que oferecem ao AWS OpsWorks Stacks as informações necessárias para implementar o livro de receitas personalizado do repositório para as instâncias da pilha, como o URL do repositório. Os detalhes dependem de qual repositório você usa para o livro de receitas. Para obter mais informações, consulte Instalação de livros de receitas personalizados.

Cor das pilhas

(Opcional) A tonalidade usada para representar a pilha no console AWS OpsWorks Stacks. Você pode usar cores diferentes para pilhas diferentes para ajudar a diferenciar, por exemplo, pilhas de desenvolvimento, preparação e produção.

Tags para pilhas

Você pode aplicar tags no nível de camada e pilha. Quando você cria uma tag, está aplicando-a para cada recurso dentro da estrutura com a tag. Por exemplo, se você aplica uma tag a uma pilha, está aplicando-a para cada camada e, dentro de cada camada, para cada instância, volume do Amazon EBS ou balanceador de carga do Elastic Load Balancing na camada. Para obter mais informações sobre como ativar as tags e usá-las para monitorar e gerenciar os custos dos

recursos do AWS OpsWorks Stacks, consulte Como usar tags de alocação de custos e Como ativar tags de alocação de custos definidas pelo usuário no Guia do usuário do Faturamento e gerenciamento de custos. Para obter mais informações sobre marcação no AWS OpsWorks Stacks, consulte. Tags

Opções avançadas

Para configurações avançadas, clique em Advanced **>>** para exibir as seções Advanced options e Security.

A seção Advanced options tem as seguintes opções:

Tipo de dispositivo raiz padrão

Determina o tipo de armazenamento a ser usado para o volume raiz da instância. Para obter mais informações, consulte Armazenamento.

- As pilhas do Linux usam um volume raiz baseado em Amazon EBS por padrão, mas você também pode especificar um volume raiz baseado em armazenamento de instâncias.
- As pilhas do Windows precisam usar um volume raiz baseado em Amazon EBS.

Perfil do IAM

(Opcional) O perfil do AWS Identity and Access Management (IAM) da pilha, que o AWS OpsWorks Stacks usa para interagir com a AWS em seu nome.

Perfil da instância do IAM padrão

(Opcional) O perfil do IAM padrão para ser associado às EC2 instâncias Amazon da pilha. Essa função concede permissões aos aplicativos em execução nas instâncias da pilha para acessar recursos da AWS, como buckets do S3.

- Para conceder permissões específicas a aplicativos, escolha um perfil da instância existente (função) que tenha as políticas adequadas.
- Inicialmente, a função do perfil não concede permissões, mas você pode usar o console IAM, API ou CLI para anexar as políticas adequadas. Para obter mais informações, consulte Especificação de permissões para aplicativos executados em instâncias EC2 .

Região de endpoint de API

Essa configuração usa o valor da região que você escolhe nas configurações básicas da pilha. Você pode escolher entre os endpoints regionais a seguir.

- Região Leste dos EUA (N. da Virgínia)
- Região Leste dos EUA (Ohio)
- US West (Oregon) Region
- Região Oeste dos EUA (Norte da Califórnia).
- Região do Canadá (Central) (somente API); não disponível para pilhas criadas no AWS Management Console
- Região Ásia-Pacífico (Mumbai)
- Região Ásia-Pacífico (Singapura)
- Região Ásia-Pacífico (Sydney)
- Região Ásia-Pacífico (Tóquio)
- Região Ásia-Pacífico (Seul)
- Região Europa (Frankfurt)
- Região Europa (Irlanda)
- Região Europa (Londres)
- Região Europa (Paris)
- Região América do Sul (São Paulo)

As pilhas que são criadas em um endpoint de API não estão disponíveis em outro endpoint de API. Como os usuários do AWS OpsWorks Stacks também são específicos da região, se você quiser que os usuários do AWS OpsWorks Stacks em uma dessas regiões de endpoint gerenciem pilhas em outra região de endpoint, você deve importar os usuários para o endpoint ao qual as pilhas estão associadas. Para obter mais informações sobre a importação de usuários, consulte como importar usuários para AWS OpsWorks pilhas.

Tema do hostname

(Opcional) Uma string usada para gerar um hostname padrão para cada instância. O valor padrão é Layer Dependent, que usa o nome curto da camada da instância e anexa um número exclusivo para cada instância. Por exemplo, a raiz do tema Load Balancer dependente de função é "lb". A primeira instância que você adiciona à camada é chamada de "lb1", a segunda é "lb2", e assim por diante.

OpsWorks Versão do agente

(Opcional) Se deve atualizar automaticamente o agente AWS OpsWorks Stacks quando uma nova versão estiver disponível ou usar uma versão específica do agente e atualizá-

la manualmente. Este recurso está disponível nas pilhas do Chef 11.10 e do Chef 12. A configuração padrão é Manual update, defina para a versão mais recente.

AWS OpsWorks O Stacks instala um agente em cada instância que se comunica com o serviço e executa tarefas como iniciar execuções do Chef em resposta a eventos do ciclo de vida. Esse agente é atualizado regularmente. Você tem duas opções para especificar a versão do agente para a pilha.

- Atualização automática: o AWS OpsWorks Stacks instala automaticamente cada nova versão do agente nas instâncias da pilha assim que a atualização fica disponível.
- Atualização manual: o AWS OpsWorks Stacks instala a versão especificada do agente nas instâncias da pilha.

  AWS OpsWorks O Stacks publica uma mensagem na página da pilha quando uma nova versão do agente está disponível, mas não atualiza as instâncias da pilha. Para atualizar o agente, você deve atualizar manualmente as configurações da pilha para especificar uma nova versão do agente e, em seguida, AWS OpsWorks as pilhas atualizarão as instâncias da pilha.

Você pode substituir a configuração padrão da versão do OpsWorks agente para uma instância específica atualizando sua configuração. Neste caso, a configuração da instância tem preferência. Por exemplo, suponha que a configuração padrão seja Auto-update, mas você especifica Manual update para uma instância específica. Quando o AWS OpsWorks Stacks lança uma nova versão do agente, ele atualiza automaticamente todas as instâncias da pilha, exceto aquela que está definida como Atualização manual. Para instalar uma nova versão do agente nessa instância, você precisa atualizar manualmente a configuração dela e especificar uma nova versão.

> **Note**
>
> O console exibe números abreviados da versão do agente. Para ver os números completos da versão, chame o describe-agent-versionscomando da AWS CLI ou os métodos equivalentes de API ou SDK. Eles retornam todos os números da versão para as versões de agente disponíveis.

JSON personalizado

(Opcional) Um ou mais atributos personalizados, formatados como uma estrutura JSON. Esses atributos são mesclados em atributos de configuração e implantação da pilha que são instalados

em cada instância e podem ser usados por receitas. Você pode usar o JSON personalizado, por exemplo, para personalizar as definições de configuração ao sobrescrever atributos incorporados que especificam as configurações padrão. Para obter mais informações, consulte Usar JSON personalizado.

A segurança tem uma opção, Usar grupos de OpsWorks segurança, que permite especificar se os grupos de segurança integrados do AWS OpsWorks Stacks devem ser associados às camadas da pilha.

AWS OpsWorks O Stacks fornece um conjunto padrão integrado de grupos de segurança, um para cada camada, que estão associados a camadas por padrão. Usar grupos OpsWorks de segurança permite que você forneça seus próprios grupos de segurança personalizados. Para obter mais informações, consulte Usar grupos de segurança.

Usar grupos OpsWorks de segurança tem as seguintes configurações:

- Sim: o AWS OpsWorks Stacks associa automaticamente o grupo de segurança incorporado a cada camada (configuração padrão).

  Você pode associar grupos de segurança adicionais a uma camada após criá-la, mas não pode excluir o grupo de segurança incorporado.
- Não: o AWS OpsWorks Stacks não associa grupos de segurança integrados a camadas.

  Você precisa criar EC2 security groups apropriados e associar um security group a cada layer que criar. Ainda é possível, porém, associar um grupo de segurança predefinido a uma layer ao ser criado. Os grupos de segurança personalizados são necessários apenas para as layers que precisam de configurações personalizadas.

Observe o seguinte:

- Se Usar grupos OpsWorks de segurança estiver definido como Sim, você não poderá restringir as configurações de acesso à porta de um grupo de segurança padrão adicionando um grupo de segurança mais restritivo a uma camada. Com vários grupos de segurança, a Amazon EC2 usa as configurações mais permissivas. Além disso, você não pode criar configurações mais restritivas modificando a configuração do grupo de segurança incorporado. Quando você cria uma pilha, o AWS OpsWorks Stacks substitui as configurações dos grupos de segurança integrados pelas configurações padrão, para que todas as alterações feitas sejam perdidas na próxima vez que você criar uma pilha. Se uma camada exigir configurações de grupo de segurança mais restritivas

do que o grupo de segurança incorporado, defina Usar grupos de OpsWorks segurança como Não, crie grupos de segurança personalizados com suas configurações preferidas e atribua-os às camadas na criação.

- Se você excluir acidentalmente um grupo de segurança do AWS OpsWorks Stacks e quiser recriá-lo, ele deverá ser uma cópia exata do original, incluindo a capitalização do nome do grupo. Em vez de recriar o grupo manualmente, recomendamos deixar o AWS OpsWorks Stacks realizar a tarefa para você. Basta criar uma nova pilha na mesma região da AWS, e da VPC, se estiver AWS OpsWorks presente; e o Stacks automaticamente recria todos os grupos de segurança incorporados, incluindo aquele que foi excluído. Você pode então excluir a pilha se não tiver mais uso para ela; os security groups permanecerão.

Execução de uma Stack em uma VPC

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode controlar o acesso de usuário a instâncias de uma stack ao criá-la em uma nuvem privada virtual (VPC). Por exemplo, talvez não seja necessário que os usuários tenham acesso direto aos servidores ou bancos de dados do aplicativo do seu stack e você queira que todo o tráfego público seja canalizado por meio de um elastic load balancer.

O procedimento básico para a execução de uma stack em uma VPC é:

1. Crie uma VPC configurada adequadamente usando o console ou a API da ou um modelo do AWS CloudFormation .
2. Especifique o ID da VPC quando você criar a stack.
3. Inicie as instâncias da stack na sub-rede apropriada.

A seguir, descrevemos resumidamente como VPCs funcionam no AWS OpsWorks Stacks.

Important

Se você usar o atributo Endpoint da VPC, esteja ciente de que cada instância da pilha deverá ser capaz de concluir as ações a seguir no Amazon Simple Storage Service (Amazon S3):

- Instale o agente da instância.
- Instale ativos, como o Ruby.
- Carregue os logs de execução do Chef.
- Recupere comandos da stack.

Para permitir estas ações, você deve garantir que as instâncias da stack tenham acesso aos seguintes buckets, que correspondem à região da stack. Caso contrário, as ações anteriores falharão.

Para o Chef 12 Linux e o Chefe 12.2 Windows, os buckets são os seguintes.

Buckets de agente	Buckets de ativo	Buckets de log	Buckets do DNA
• opsworks-instance-agent-sa-east-1 • opsworks-instance-agent-ap-south-1 • opsworks-instance-agent-ap-northeast-1 • opsworks-instance-agent-ap-northeast-2 • opsworks-instance-agent-ap-southeast-1 • opsworks-instance-agent-ap-southeast-2 • opsworks-instance-agent-ca-centr1 • opsworks-instance-agent-eu-centr1 • opsworks-instance-agent-eu-west-1 • opsworks-instance-agent-eu-oeste-2	• opsworks-instance-assets-us-leste-2 • opsworks-instance-assets-us-east-1 • opsworks-instance-assets-ap-south-1 • opsworks-instance-assets-ap-northeast-1 • opsworks-instance-assets-ap-northeast-2 • opsworks-instance-assets-ap-southeast-1 • opsworks-instance-assets-ap-southeast-2 • opsworks-instance-assets-ca-centr1 • opsworks-instance-assets-eu-centr1 • opsworks-instance-assets-eu-west-1	• opsworks-us-east-2-log • opsworks-us-east-1 log • opsworks-ap-south-1 log • opsworks-ap-northeast-1 log • opsworks-ap-northeast-2-log • opsworks-ap-southeast-1 log • opsworks-ap-southeast-2-log • opsworks-ca-central-1 log • opsworks-eu-central-1 log • opsworks-eu-west-1 log • opsworks-eu-west-2-log • opsworks-eu-west-3 troncos • opsworks-sa-east-1 log • opsworks-us-west-1 log • opsworks-us-west-2-log	• opsworks-us-east2-DNA • opsworks-us-east1-dna • opsworks-ap-south1-dna • opsworks-ap-northeast1-dna • opsworks-ap-northeast2-DNA • opsworks-ap-southeast1-dna • opsworks-ap-southeast2-DNA • opsworks-ca-central1-dna • opsworks-eu-central1-dna • opsworks-eu-west1-dna • opsworks-eu-west2-DNA • opsworks-eu-west-3-DNA • opsworks-sa-east1-dna • opsworks-us-west1-dna • opsworks-us-west2-DNA

Buckets de agente	Buckets de ativo	Buckets de log	Buckets do DNA
• opsworks-instance-agent-eu-west-3 • opsworks-instance-agent-us-east-1 • opsworks-instance-agent-us-leste-2 • opsworks-instance-agent-us-west-1 • opsworks-instance-agent-us-oeste-2	• opsworks-instance-assets-eu-oeste-2 • opsworks-instance-assets-eu-west-3 • opsworks-instance-assets-sa-east-1 • opsworks-instance-assets-us-west-1 • opsworks-instance-assets-us-oeste-2		

Para o Chef 11.10 e versões anteriores para Linux, os buckets são os seguintes. As pilhas do Chef 11.4 não são compatíveis com endpoints regionais fora do Leste dos EUA (N. da Virgínia).

Buckets de agente	Buckets de ativo	Buckets de log	Buckets do DNA
• opsworks-instance-agent-us-leste-2 • opsworks-instance-agent-us-east-1 • opsworks-instance-agent-ap-south-1 • opsworks-instance-agent-ap-northeast-1 • opsworks-instance-agent-ap-northeast-2 • opsworks-instance-agent-ap-southeast-1 • opsworks-instance-agent-ap-southeast-2 • opsworks-instance-agent-ca-centr1 • opsworks-instance-agent-eu-centr1 • opsworks-instance-agent-eu-west-1	• opsworks-instance-assets-us-leste-2 • opsworks-instance-assets-us-east-1 • opsworks-instance-assets-ap-south-1 • opsworks-instance-assets-ap-northeast-1 • opsworks-instance-assets-ap-northeast-2 • opsworks-instance-assets-ap-southeast-1 • opsworks-instance-assets-ap-southeast-2 • opsworks-instance-assets-ca-centr1 • opsworks-instance-assets-eu-centr1 • opsworks-instance-assets-eu-west-1	• prod_stage-log	• prod_stage-dna

Buckets de agente	Buckets de ativo	Buckets de log	Buckets do DNA
• opsworks-instance-agent-eu-oeste-2 • opsworks-instance-agent-eu-west-3 • opsworks-instance-agent-us-east-1 • opsworks-instance-agent-us-west-1 • opsworks-instance-agent-us-oeste-2	• opsworks-instance-assets-eu-oeste-2 • opsworks-instance-assets-eu-west-3 • opsworks-instance-assets-sa-east-1 • opsworks-instance-assets-us-west-1 • opsworks-instance-assets-us-oeste-2		

Para obter mais informações, consulte VPC Endpoints.

> **Note**
>
> Para que o AWS OpsWorks Stacks se conecte aos VPC endpoints que você habilita, você também deve configurar o roteamento para seu NAT ou IP público, pois AWS OpsWorks o agente Stacks ainda exige acesso ao endpoint público.

Tópicos

- Conceitos básicos da VPC
- Crie uma VPC para uma pilha de pilhas AWS OpsWorks

Conceitos básicos da VPC

Para uma discussão detalhada sobre VPCs, consulte Amazon Virtual Private Cloud. Em resumo, uma VPC consiste em uma ou mais sub-redes, sendo que cada uma delas contém uma ou mais instâncias. Cada sub-rede tem uma tabela de roteamento associada que direciona o tráfego de saída com base em seu endereço IP de destino.

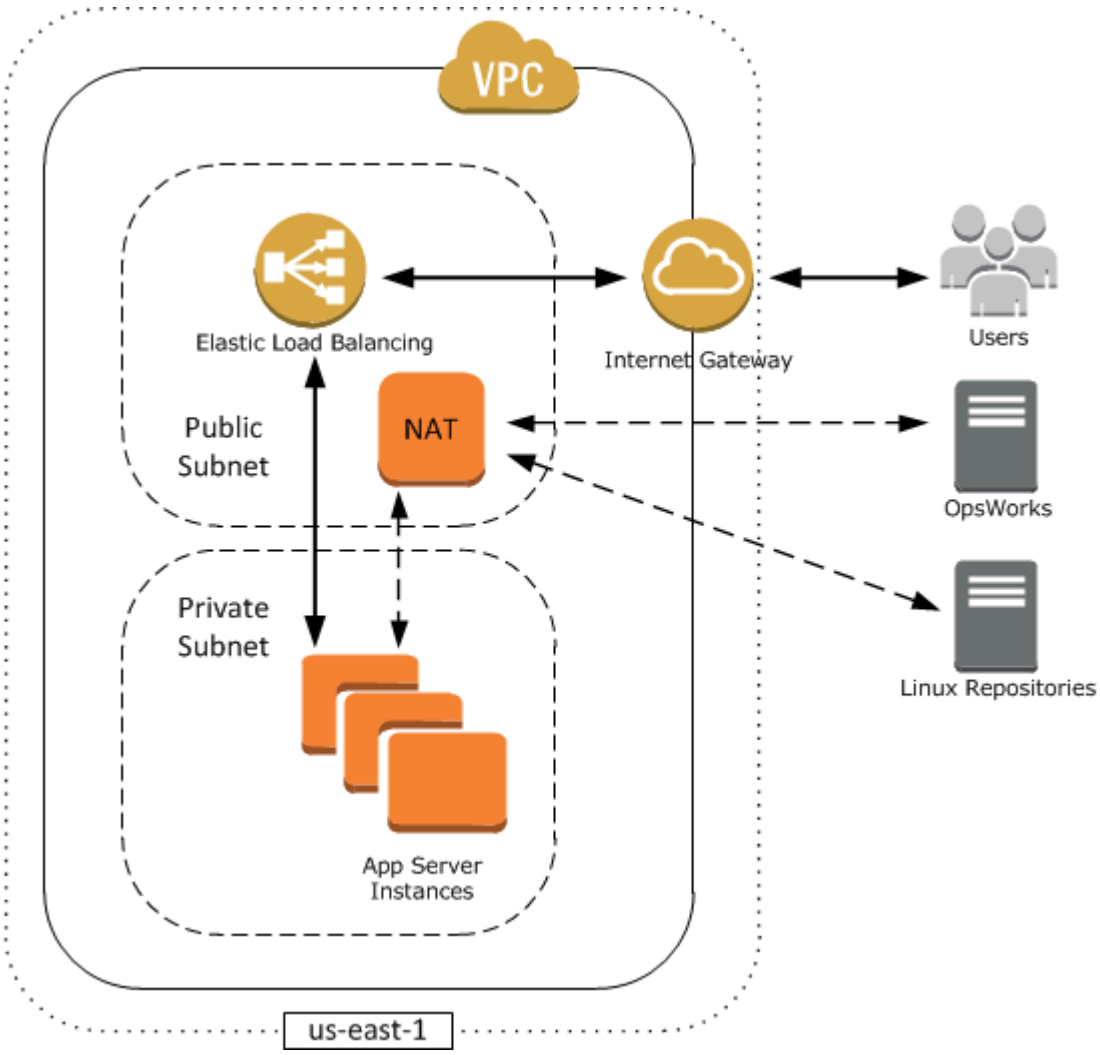
- As instâncias em uma VPC geralmente podem se comunicar entre si por padrão, independentemente da sub-rede. No entanto, alterações nas listas de controle de acesso à rede (ACLs), nas políticas de grupos de segurança ou no uso de endereços IP estáticos podem interromper essa comunicação.
- As sub-redes cujas instâncias podem se comunicar com a Internet são conhecidas como sub-redes públicas.
- As sub-redes cujas instâncias podem se comunicar apenas com outras instâncias na VPC e não podem se comunicar diretamente com a Internet são conhecidas como sub-redes privadas.

AWS OpsWorks As pilhas exigem que a VPC seja configurada para que cada instância na pilha, incluindo instâncias em sub-redes privadas, tenha acesso aos seguintes endpoints:

- Um dos endpoints do serviço AWS OpsWorks Stacks listados na seção “Region Support” do. Começando a usar o AWS OpsWorks Stacks
- Um dos seguintes endpoints de serviço de instância, usado pelo agente AWS OpsWorks Stacks. O agente é executado em instâncias gerenciadas pelo cliente para trocar dados com o serviço.
  - opsworks-instance-service.us-east-2.amazonaws.com
  - opsworks-instance-service.us-east-1.amazonaws.com
  - opsworks-instance-service.us-west-1.amazonaws.com
  - opsworks-instance-service.us-west-2.amazonaws.com
  - opsworks-instance-service.ap-south-1.amazonaws.com
  - opsworks-instance-service.ap-north-1.amazonaws.com
  - opsworks-instance-service.ap-north-2.amazonaws.com
  - opsworks-instance-service.ap-south-1.amazonaws.com
  - opsworks-instance-service.ap-south-2.amazonaws.com
  - opsworks-instance-service.ca-central-1.amazonaws.com
  - opsworks-instance-service.eu-central-1.amazonaws.com

  - opsworks-instance-service.eu-west-1.amazonaws.com
  - opsworks-instance-service.eu-west-2.amazonaws.com
  - opsworks-instance-service.eu-west-3.amazonaws.com
- Amazon S3
- Todos os repositórios de pacote dos quais seu sistema operacional depende, como os repositórios do Amazon Linux ou do Ubuntu Linux.
- Seus repositórios de aplicativos e de livros de receitas personalizados.

Há uma série de maneiras de configurar uma VPC para fornecer essa conectividade. Veja a seguir um exemplo simples de como você pode configurar uma VPC para uma pilha de servidores de aplicativos AWS OpsWorks Stacks.



Esta VPC tem vários componentes:

Sub-redes

A VPC tem duas sub-redes, uma pública e uma privada.

- A sub-rede pública contém um load balancer e um dispositivo de conversão de endereços de rede (NAT), que podem se comunicar com os endereços externos e com as instâncias na sub-rede privada.
- A sub-rede privada contém os servidores de aplicativos, que podem se comunicar com a NAT e o load balancer na sub-rede pública, mas não podem se comunicar diretamente com os endereços externos.

gateway da Internet

O Internet gateway permite que as instâncias com endereços IP públicos, como o load balancer, se comuniquem com endereços fora da VPC.

Load balancer

O load balancer do Elastic Load Balancing leva o tráfego de entrada de usuários, o distribui para os servidores de aplicativos na sub-rede privada e retorna as respostas aos usuários.

NAT

O dispositivo (NAT) fornece os servidores de aplicativos com acesso limitado à Internet, que normalmente é usado para fins como o download de atualizações de software de um repositório externo. Todas as instâncias do AWS OpsWorks Stacks devem ser capazes de se comunicar com o AWS OpsWorks Stacks e com os repositórios Linux apropriados. Uma forma de lidar com esse problema é colocar um dispositivo NAT com um Endereço IP elástico associado em uma sub-rede pública. Você pode então rotear o tráfego de saída de instâncias na sub-rede privada pro meio da NAT.

> **Note**
>
> Uma única instância NAT cria um ponto único de falha no tráfego de saída em sua sub-rede privada. Você pode melhorar a confiabilidade ao configurar a VPC com um par de instâncias NAT que assuma o lugar da outra em caso de falha. Para obter mais informações, consulte Alta disponibilidade para instâncias NAT da Amazon VPC. Você também pode usar um gateway NAT. Para obter mais informações, consulte NAT no Guia do usuário da Amazon VPC.

A configuração ideal de VPC depende da sua AWS OpsWorks pilha de pilhas. A seguir, alguns exemplos de quando você pode usar determinadas configurações da VPC. Para obter exemplos de outros cenários da VPC, consulte Cenários de uso da Amazon VPC.

Working with one instance in a public subnet (Trabalhar com uma instância em uma sub-rede pública)

Se você tiver uma pilha com uma única instância sem recursos privados associados, como uma instância do Amazon RDS que não deve ficar acessível publicamente, poderá criar uma VPC com uma sub-rede pública e colocar a instância na sub-rede. Se você não estiver usando uma VPC padrão, deverá fazer com que a layer da instância atribua um Endereço IP elástico à instância. Para obter mais informações, consulte OpsWorks Noções básicas sobre camadas.

Working with private resources (Trabalhar com recursos privados)

Se você tiver recursos que não devem ficar acessíveis publicamente, poderá criar uma VPC com uma sub-rede pública e uma sub-rede privada. Por exemplo, em um ambiente de escalabilidade automática com balanceamento de carga, você pode colocar todas as EC2 instâncias da Amazon na sub-rede privada e o balanceador de carga em uma sub-rede pública. Dessa forma, as EC2 instâncias da Amazon não podem ser diretamente acessadas da Internet, todo o tráfego de entrada deve ser roteado por meio do balanceador de carga.

A sub-rede privada isola as instâncias do acesso de usuário EC2 direto da Amazon, mas elas ainda devem enviar solicitações de saída para o AWS e os repositórios de pacotes do Linux apropriados. Para permitir essas solicitações, você pode, por exemplo, usar um dispositivo de conversão de endereços de rede (NAT) com seu próprio Endereço IP elástico e então rotear o tráfego de saída das instâncias por meio da NAT. Você pode colocar a NAT na mesma sub-rede pública do load balancer, como mostrado no exemplo anterior.

- Se você estiver usando um banco de dados de back-end, como uma instância do Amazon RDS, você pode colocar essas instâncias na sub-rede privada. Para instâncias do Amazon RDS, você deve especificar pelo menos duas sub-redes diferentes em Zonas de disponibilidade diferentes.
- Se você precisar de acesso direto para instâncias em uma sub-rede privada, por exemplo, você deseja usar o SSH para fazer login em uma instância, você pode colocar um bastion host na sub-rede pública que faz solicitações de proxy da Internet.

Extending your own network into AWS (Como estender sua própria rede na AWS)

Se você quiser estender sua própria rede para a nuvem e também acessar diretamente a Internet de sua VPC, poderá criar um gateway de VPN. Para obter mais informações, consulte Cenário 3: VPC com sub-redes pública e privada e acesso a VPN de hardware.

Crie uma VPC para uma pilha de pilhas AWS OpsWorks

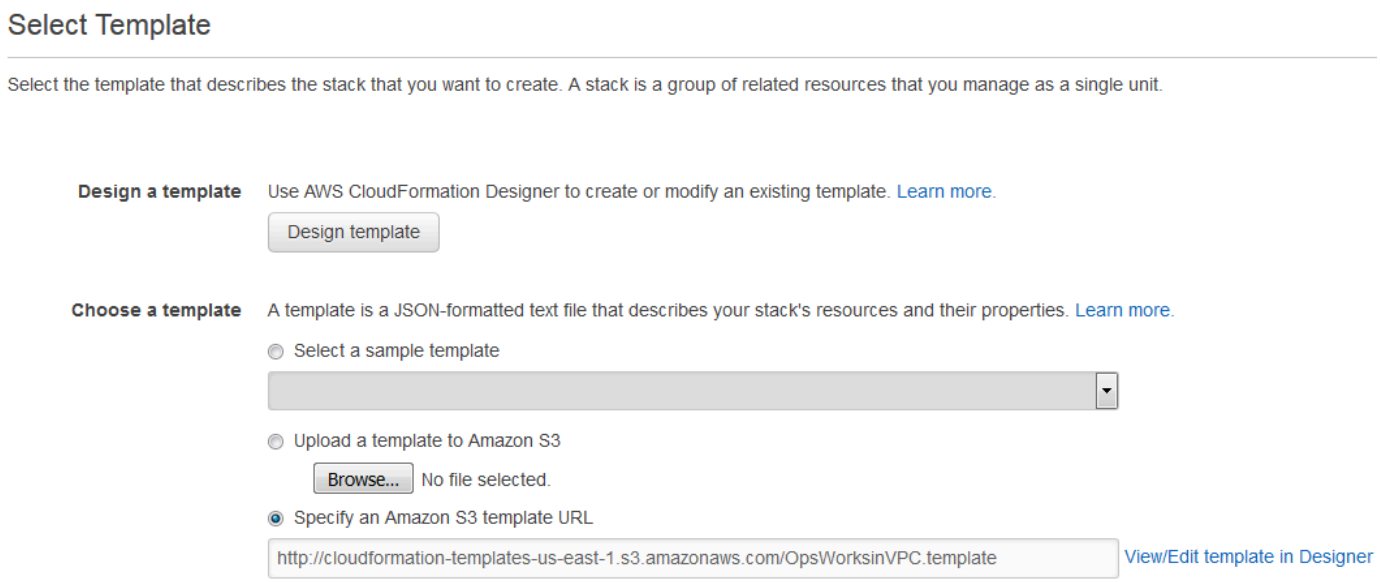
Esta seção mostra como criar uma VPC para uma pilha de AWS OpsWorks pilhas usando um exemplo de modelo da AWS. CloudFormation Você pode baixar o modelo no OpsWorksVPCtemplatesarquivo.zip. Para obter mais informações sobre como criar manualmente uma VPC como a discutida neste tópico, consulte Cenário 2: VPC com sub-redes pública e privada. Para obter detalhes sobre como configurar tabelas de roteamento, security groups e assim por diante, consulte o modelo de exemplo.

> **Note**
>
> Por padrão, o AWS OpsWorks Stacks exibe os nomes das sub-redes concatenando o intervalo CIDR e a zona de disponibilidade, como. `10.0.0.1/24 - us-east-1b` Para tornar os nomes mais legíveis, crie uma tag para cada sub-rede com Chave definida como **`Name`** e Valor definido como o nome da sub-rede. AWS OpsWorks Então, o Stacks acrescenta o nome da sub-rede ao nome padrão. Por exemplo, a sub-rede privada no exemplo a seguir tem uma tag com Nome definido como**`Private`**, que é OpsWorks exibida como`10.0.0.1/24 us-east - 1b - Private`.

Você pode iniciar um modelo de VPC usando o AWS CloudFormation console com apenas algumas etapas. O procedimento a seguir usa o modelo de exemplo para criar uma VPC no Leste dos EUA (N. da Virgínia). Para obter instruções sobre como usar o modelo para criar uma VPC em outras regiões, consulte a observação que segue o procedimento.

Como criar a VPC

1. Abra o console do AWS CloudFormation, selecione a região US East (N. Virginia) (Leste dos EUA (Norte da Virgínia)) e a opção Create Stack (Criar pilha).
2. Na página Select Template (Escolher modelo), selecione Upload a template (Fazer upload de um modelo). Procure o **`OpsWorksinVPC.template`** arquivo que você baixou no OpsWorksVPCtemplatesarquivo.zip. Escolha Continuar.



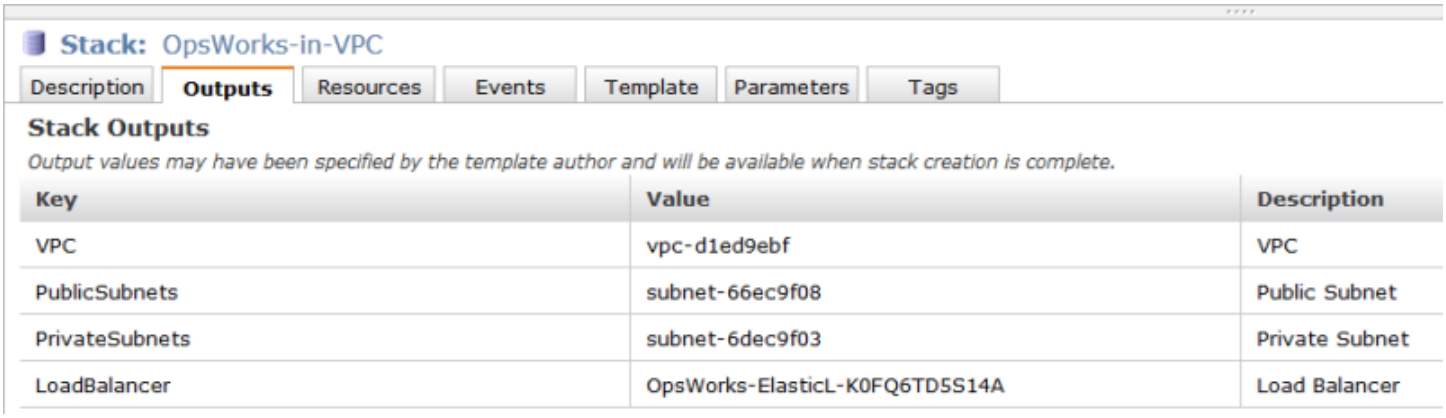
Você também pode iniciar essa pilha abrindo modelos de CloudFormation amostra da AWS, localizando o modelo de VPC do AWS OpsWorks Stacks e escolhendo Launch Stack.

3. Na página Specify Parameters (Especificar parâmetros), aceite os valores padrão e escolha Continue (Continuar).
4. Na página Add Tags (Adicionar tags), crie uma tag com Key (Chave) definida como **`Name`** e Value (Valor) definido como o nome da VPC. Essa tag facilitará a identificação de sua VPC ao criar uma AWS OpsWorks pilha de pilhas.
5. Escolha Continue (Continuar) e depois Close (Fechar) para iniciar a pilha.

Observação: você pode criar a VPC em outras regiões usando uma das seguintes abordagens.

- Acesse Usando modelos em diferentes regiões, escolha a região apropriada, localize o modelo de VPC do AWS OpsWorks Stacks e escolha Launch Stack.
- Copie o arquivo de modelo para seu sistema, selecione a região apropriada no console do AWS CloudFormation e use a opção Upload a template to Amazon S3 do assistente Create Stack para fazer o upload do modelo do seu sistema.

O modelo de exemplo inclui saídas que fornecem a VPC, a sub-rede e o IDs balanceador de carga necessários para criar a pilha Stacks. AWS OpsWorks Você pode vê-las escolhendo a guia Saídas na parte inferior da janela do AWS CloudFormation console.



Atualizar uma pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Depois de criar uma pilha, você poderá atualizar a configuração a qualquer momento. Na página Stack, clique em Stack Settings e Edit para exibir a página Settings. Faça as alterações desejadas e clique em Save.

As configurações são iguais às discutidas em Criar uma nova pilha. Consulte esse tópico para obter mais detalhes. Entretanto, observe o seguinte:

- Você não pode modificar a região ou o ID da VPC.
- Se a pilha for executada em uma VPC, as configurações incluirão uma configuração Default subnet, que lista as sub-redes da VPC. Se a pilha não estiver em execução em uma VPC, a configuração será identificada como Default Availability Zones e listará as Zonas de disponibilidade da região.
- Você pode alterar o sistema operacional padrão, mas não pode especificar um sistema operacional Linux para uma stack do Windows ou o Windows para uma pilha do Linux.

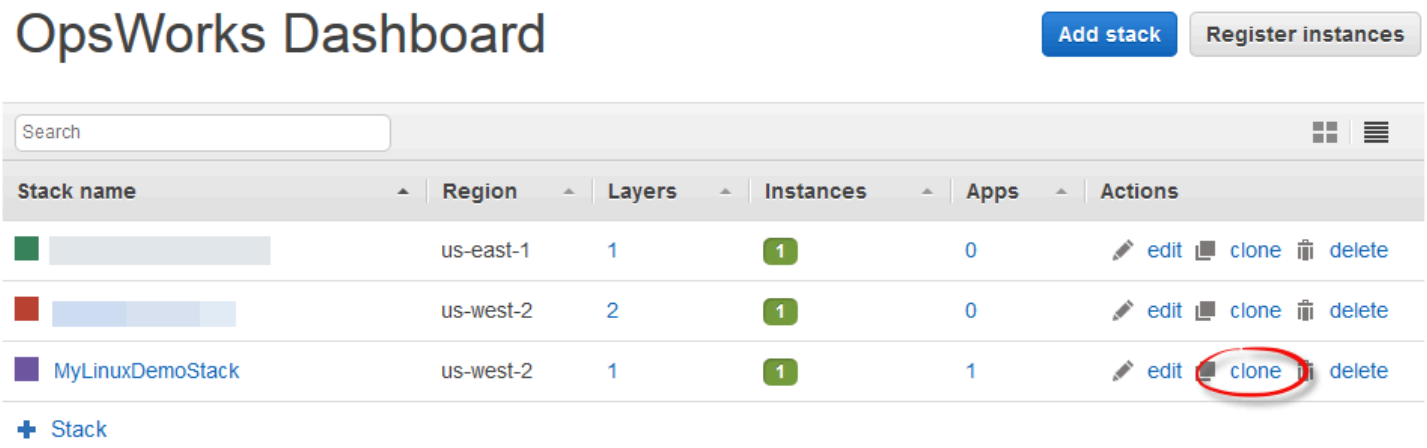
- Se você alterar qualquer uma das configurações de instância padrão, como Hostname theme ou Default SSH key, os novos valores se aplicarão somente a quaisquer novas instâncias criadas, e não para as instâncias existentes.
- A alteração do nome altera o nome exibido pelo console; não altera o nome abreviado subjacente que AWS OpsWorks Stacks usa para identificar a pilha.
- Antes de alterar Usar grupos de OpsWorks segurança de Sim para Não, cada camada deve ter pelo menos um grupo de segurança além do grupo de segurança incorporado da camada. Para obter mais informações, consulte Como editar a configuração de uma OpsWorks camada.

  AWS OpsWorks Em seguida, o Stacks exclui os grupos de segurança predefinidos de cada camada.
- Se você alterar Usar grupos de OpsWorks segurança de Não para Sim, o AWS OpsWorks Stacks adiciona o grupo de segurança incorporado apropriado a cada camada, mas não exclui os grupos de segurança existentes.

Clonar uma pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Às vezes, é útil criar várias cópias de uma pilha. Por exemplo, você pode querer adicionar redundância como uma recuperação de desastres ou medida de prevenção, ou pode usar uma pilha existente como um ponto de partida para um nova pilha. A abordagem mais simples é clonar a pilha de origem. No painel AWS OpsWorks Pilhas, na coluna Ações da linha da pilha que você deseja clonar, escolha clonar, o que abre a página Clonar pilha.



Inicialmente, as configurações da pilha clonada são idênticas às da pilha de origem, exceto que a palavra "copy" é acrescentada ao nome da pilha. Para obter informações sobre essas configurações, consulte Criar uma nova pilha. Há também duas configurações opcionais adicionais:

Permissões

Se all permissions for selecionado (o padrão), as permissões da pilha de origem são aplicadas à pilha clonada.

Apps

Lista aplicativos implantados na pilha de origem. Para cada aplicativo listado, se a caixa de seleção correspondente estiver selecionada (o padrão), o aplicativo será implantado na pilha clonada.

> **Note**
>
> Você não pode clonar uma pilha de um endpoint regional para outro; por exemplo, não é possível clonar uma pilha de Oeste dos EUA (Oregon) (us-west-2) para a região da Ásia-Pacífico (Mumbai) (ap-south-1).

Quando você tiver finalizado as configurações, escolha Clonar pilha. AWS OpsWorks O Stacks cria uma nova pilha que consiste das camadas da pilha de origem e, opcionalmente, seus aplicativos e permissões. As camadas têm a mesma configuração das originais, sujeitas a todas as modificações que você fez. No entanto, a clonagem não cria quaisquer instâncias. Você deve adicionar um conjunto adequado de instâncias a cada camada da pilha clonada e então iniciá-las. Assim como

acontece com qualquer pilha, você pode executar tarefas normais de gerenciamento em uma pilha clonada, como adição, exclusão ou modificação de camadas ou adição e implantação de aplicativos.

Para tornar a pilha clonada operacional, inicie as instâncias. AWS OpsWorks O Stacks instala e configura cada instância de acordo com a camada de associação. Ele também implanta todos os aplicativos, da mesma forma que faz com um nova pilha.

Executar comandos AWS OpsWorks do Stacks Stack

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks fornece um conjunto de comandos de pilha, que você pode usar para realizar várias operações nas instâncias de uma pilha. Para executar um comando da pilha, clique em Run Command na página Stack. Em seguida, você escolhe o comando apropriado, especifica quaisquer opções, e pressiona o botão no canto inferior direito, que estará rotulado com o nome do comando.

> **Note**
>
> AWS OpsWorks O Stacks também oferece suporte a um conjunto de comandos de implantação, que você usa para gerenciar a implantação do aplicativo. Para obter mais informações, consulte Implementação de aplicativos.

Você pode executar os seguintes comandos de pilha em qualquer pilha.

Atualize os livros de receita personalizados

Atualiza os livros de receitas personalizados das instâncias com a versão atual do repositório. Este comando não executa quaisquer receitas. Para executar as receitas atualizadas, você pode usar um `Execute Recipes`, `Setup`, ou `Configure` comando de pilha, ou implantar novamente

seu aplicativo para executar Implantar receitas. Para obter mais informações sobre livros de receitas personalizados, consulte Livros de receitas e receitas.

Executar receitas

Executa um grupo especificado de receitas nas instâncias. Para obter mais informações, consulte Execução manual de receitas.

Configuração

Executa o Configurar receitas na instância.

Configurar

Executa o Configurar receitas na instância.

> **Note**
>
> Para usar Setup ou Configure para executar receitas em uma instância, as receitas devem estar atribuas ao evento de ciclo de vida correspondente para a camada da instância. Para obter mais informações, consulte Execução de receitas.

Você pode executar os comandos de pilha a seguir apenas em pilhas baseadas em Linux.

Instalar dependências

Instala os pacotes da instância. A partir do Chef 12, esse comando não está disponível.

Atualizar dependências

(Linux apenas. A partir do Chef 12, esse comando não está disponível.) Instala atualizações de sistema operacional regulares e de pacote. Os detalhes dependem do sistema operacional da instância. Para obter mais informações, consulte Gerenciamento de atualizações de segurança.

Use o comando Upgrade Operating System (Atualizar sistema operacional) para atualizar instâncias para uma nova versão do Amazon Linux.

Atualizar sistema operacional

(Apenas Linux) Atualiza os sistemas operacionais Amazon das instâncias para a versão mais recente. Para obter mais informações, consulte AWS OpsWorks Sistemas operacionais de pilha.

> **Important**
>
> Depois de executar Upgrade Operating System (Atualizar sistema operacional), recomendamos que você execute também Setup (Configuração). Isso garante que os serviços são reiniciados corretamente.

Comandos de pilha têm as opções a seguir, algumas das quais aparecem apenas para determinados comandos.

Comentário

(Opcional) Insira quaisquer marcações personalizadas que deseja adicionar.

Receitas a serem executadas

(Obrigatório) Essa configuração aparece apenas se você selecionar o comando Execute Recipes. Insira as receitas a serem executadas usando o *recipe_name* formato padrão*cookbook_name*::, separadas por vírgulas. Se você especificar várias receitas, o AWS OpsWorks Stacks as executará na ordem listada.

Permitir reinicialização

(Opcional) Essa configuração aparece apenas se você selecionar o comando Upgrade Operating System. O valor padrão é Sim, que direciona o AWS OpsWorks Stacks a reinicializar as instâncias após a instalação da atualização.

JSON Chef personalizado

(Opcional) Escolha Advanced para exibir essa opção, que permite especificar atributos JSON personalizados a serem incorporados na configuração de pilha e nos atributos de implantação.

Instâncias

(Opcional) Especifique as instâncias nas quais executar o comando. Todas instâncias online são selecionadas por padrão. Para executar o comando em um subconjunto de instâncias, selecione as camadas apropriadas ou instâncias.

> **Note**
>
> Você pode ver execuções de execute_recipes que você não executou listadas nas páginas Deployment e Commands. Isso geralmente é o resultado da mudança de permissões, como

dar ou remover permissões do SSH a um usuário. Quando você faz essa alteração, o AWS OpsWorks Stacks usa execute_recipes para atualizar as permissões nas instâncias.

Usar JSON personalizado

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Várias ações do AWS OpsWorks Stacks permitem que você especifique um JSON personalizado, que o AWS OpsWorks Stacks instala em instâncias e pode ser usado por receitas.

Você pode especificar JSON personalizado nas seguintes situações:

- Quando você cria, atualiza ou clona um stack.

  O AWS OpsWorks Stacks instala JSON personalizado em todas as instâncias para todos os eventos de ciclo de vida subsequentes.
- Quando você executa um comando de implantação ou stack.

  O AWS OpsWorks Stacks passa o JSON personalizado às instâncias apenas para esse evento.

O JSON personalizado deve ser representado por e formatado como um objeto JSON válido. Por exemplo:

```
{
  "att1": "value1",
  "att2": "value2"
  ...
}
```

AWS OpsWorks O Stacks armazena JSON personalizado nos seguintes locais:

Em instâncias do Linux:

- `/var/chef/runs/run-ID/attribs.json`
- `/var/chef/runs/run-ID/nodes/hostname.json`

Em instâncias do Windows:

- `drive:\chef\runs\run-ID\attribs.json`
- `drive:\chef\runs\run-ID\nodes\hostname.json`

> **Note**
>
> No Chef 11.10 e em versões anteriores para o Linux, o JSON personalizado está localizado no seguinte caminho em instâncias do Linux; instâncias do Windows não estão disponíveis e não há um arquivo `attribs.json`. Os logs são armazenados na mesma pasta ou diretório que o JSON. Para obter mais informações sobre JSON personalizado no Chef 11.10 e em versões anteriores para o Linux, consulte Overriding Attributes with Custom JSON e Chef Logs.
> `/var/lib/aws/opsworks/chef/hostname.json`

Nos caminhos anteriores, *run-ID* é um ID exclusivo que o AWS OpsWorks Stacks atribui a cada execução do Chef em uma instância e *hostname* é o nome do host da instância.

Para acessar o JSON personalizado a partir das receitas do Chef, use a sintaxe `node` padrão do Chef.

Por exemplo, suponha que você deseje definir configurações simples para uma aplicação que quer implantar, como se a aplicação fica inicialmente visível e as cores de primeiro e segundo plano iniciais. Suponha que você defina essas configurações da aplicação com um objeto JSON da seguinte forma:

```
{
  "state": "visible",
  "colors": {
    "foreground": "light-blue",
    "background": "dark-gray"
  }
```

```
}
```

Para declarar o JSON personalizado para um stack:

1. Na página de pilhas, escolha Stack Settings e Edit.
2. Para Custom Chef JSON, digite o objeto JSON e, em seguida, escolha Save.

> **Note**
>
> Você pode declarar um JSON personalizado nos níveis de implantação, layer e pilha. Você pode querer fazer isso se quiser que alguns JSON personalizados fiquem visíveis apenas para uma implantação ou um layer individual. Ou, por exemplo, você pode querer substituir temporariamente JSON personalizados declarados no nível da stack por JSON personalizados declarados no nível do layer. Se você declarar JSON personalizados em vários níveis, os JSON personalizados declarados no nível da implantação substituirão os JSON personalizados declarados nos níveis de layer e pilha. Os JSON personalizados declarados no nível da camada substituem qualquer os JSON personalizados declarados apenas no nível da pilha.
>
> Para usar o console AWS OpsWorks Stacks para especificar o JSON personalizado para uma implantação, na página Implantar aplicativo, escolha Avançado. Insira o JSON personalizado na caixa Custom Chef JSON e escolha a opção Save.
>
> Para usar o console AWS OpsWorks Stacks para especificar JSON personalizado para uma camada, na página Camadas, escolha Configurações para a camada desejada. Digite o JSON personalizado na caixa Custom JSON e escolha Save.
>
> Para obter mais informações, consulte Como editar a configuração de uma OpsWorks camada e Implementação de aplicativos.

Quando você executa um comando de implantação ou stack, as receitas podem recuperar esses valores personalizados usando a sintaxe `node` padrão do Chef, que mapeia diretamente para a hierarquia no objeto JSON personalizado. Por exemplo, o seguinte código de receita grava mensagens para o log do Chef sobre os valores anteriores do JSON personalizado:

```
Chef::Log.info("********** The app's initial state is '#{node['state']}' **********")
Chef::Log.info("********** The app's initial foreground color is '#{node['colors']
['foreground']}' **********")
```

```
Chef::Log.info("********** The app's initial background color is '#{node['colors']
['background']}' **********")
```

Essa abordagem pode ser útil para passar dados para receitas. AWS OpsWorks O Stacks adiciona esses dados à instância e as receitas podem recuperá-los usando uma sintaxe padrão do Chefnode.

> **Note**
>
> JSON personalizado é limitado a 120 KB. Caso precise de mais capacidade, recomendamos armazenar alguns dados no Amazon Simple Storage Service (Amazon S3). Suas receitas personalizadas podem usar a CLI da AWS ou o AWS SDK para Ruby para fazer download dos dados do bucket do Amazon S3 para sua instância.

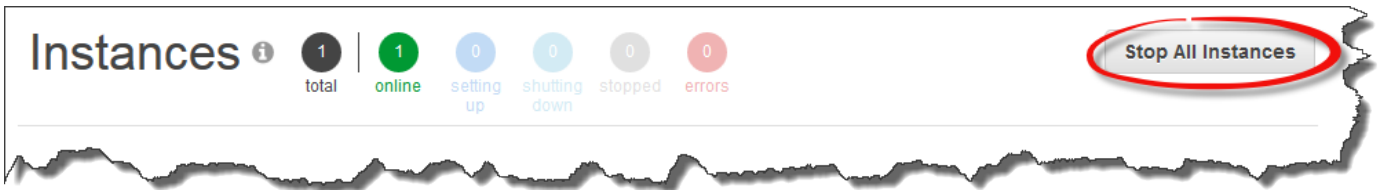
Excluir uma pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Se você não precisa mais da pilha, você pode excluí-la. Somente as pilhas vazias podem ser excluídas. Você deve primeiro excluir todas as instâncias, os aplicativos e as camadas da pilha.

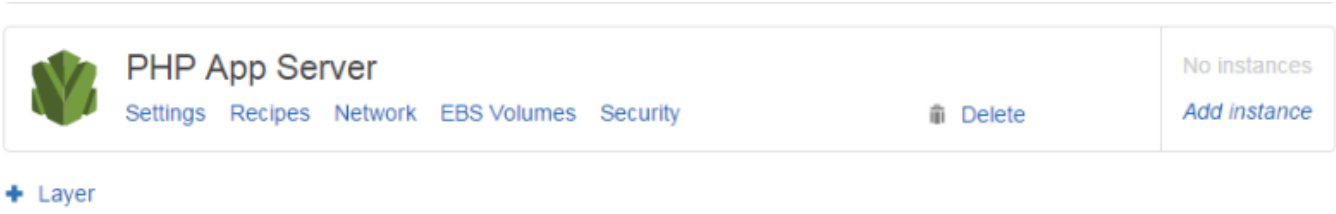
Para excluir uma pilha

1. No painel AWS OpsWorks Pilhas, escolha a pilha que você deseja fechar e excluir.
2. No painel de navegação, escolha Instâncias.
3. Na página Instances (Instâncias), selecione Stop all Instances (Interromper todas as instâncias).

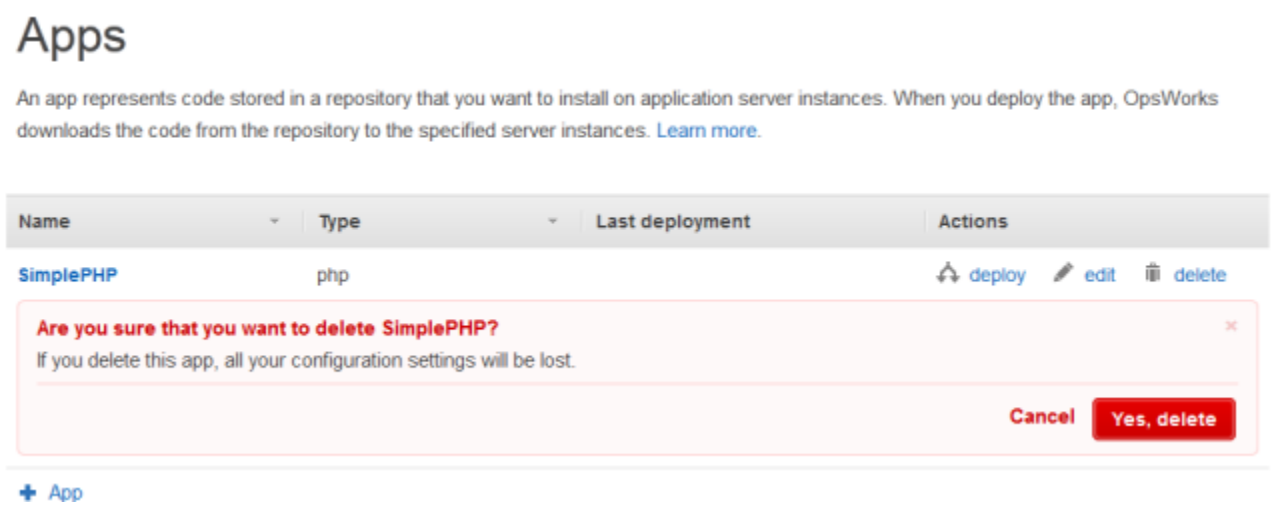


4. Depois que as instâncias forem interrompidas, em cada instância na camada, selecione delete (excluir) na coluna Actions (Ações). Quando a confirmação for solicitada, selecione Yes, Delete (Sim, excluir).

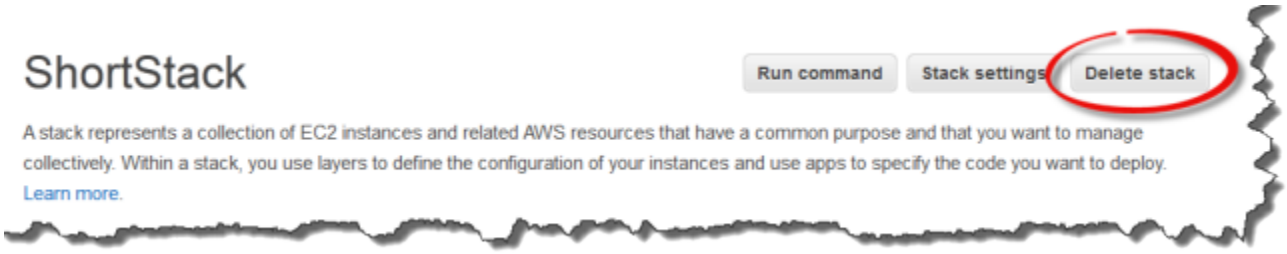
5. Quando todas as instâncias forem excluídas, no painel de navegação, selecione Layers (Camadas).
6. Na página Layers (Camadas), em cada camada na pilha, selecione delete (excluir). Na solicitação de confirmação, selecione Yes, Delete (Sim, excluir).



7. Quando todas as camadas forem excluídas, no painel de navegação, selecione Apps (Aplicativos).
8. Na página Apps (Aplicativos), em cada aplicativo na pilha, selecione delete (excluir) na coluna Actions (Ações). Na solicitação de confirmação, selecione Yes, Delete (Sim, excluir).



9. Quando todos os aplicativos forem excluídos, no painel de navegação, selecione Stack (Pilha).
10. Na página da pilha, selecione Delete stack (Excluir pilha). Na solicitação de confirmação, selecione Yes, Delete (Sim, excluir).



Excluindo outros AWS recursos usados por uma pilha

Você usa outros AWS recursos com AWS OpsWorks as Pilhas para criar e gerenciar suas pilhas. Ao excluir uma pilha, considere também excluir recursos que funcionaram com a pilha, se outra pilha não os estiver usando e os recursos fora das AWS OpsWorks Pilhas não os estiverem usando. A seguir estão os motivos sugeridos para limpar AWS os recursos externos que você usou com uma pilha.

- AWS Recursos externos podem continuar acumulando cobranças em sua AWS conta.
- Os recursos, como os buckets do Amazon S3, podem conter informações de identificação pessoal ou confidenciais.

> **Important**
>
> Não exclua esses recursos se eles estão sendo usados por outras pilhas. Observe que os perfis e os grupos de segurança do IAM são globais, por isso, pilhas em outras regiões podem estar usando esses mesmos recursos.

A seguir estão outros AWS recursos que as pilhas usam e links para informações sobre como excluí-las.

Funções de serviço e perfis de instância

Ao criar uma pilha, especifique uma perfil do IAM e um perfil de instância do que são usados pelo AWS OpsWorks Stacks para criar os recursos permitidos em seu nome. AWS OpsWorks cria a função e o perfil da instância para você se você não escolher os existentes. A função e o perfil de instância AWS OpsWorks criados para você são nomeados `aws-opsworks-service-role` e`aws-opsworks-ec2-role`, respectivamente. Se não há outras pilhas em sua conta usando o perfil do IAM e o perfil de instância, esses recursos podem ser excluídos com segurança. Para obter mais informações sobre como excluir perfis do IAM e perfis de instância, consulte Excluir funções ou perfis de instância no Guia do usuário do IAM.

Grupos de segurança

Em AWS OpsWorks Stacks, você pode especificar grupos de segurança definidos pelo usuário no nível da camada. Você pode criar os grupos de segurança usando o EC2 console ou a API da Amazon. As pilhas e camadas em outras regiões podem usar os mesmos grupos de segurança, pois eles são globais. Você pode excluir um grupo de segurança se ele não está sendo usado por outros AWS recursos da. Para obter mais informações sobre como excluir um grupo de segurança, consulte Excluir um grupo de segurança no Guia EC2 do usuário da Amazon.

Volumes do Amazon EBS

Em AWS OpsWorks Stacks, você adiciona volumes do EBS no nível da camada e eles são anexados às instâncias na camada. Você pode criar os volumes do EBS usando o console ou a API do EC2 serviço da Amazon e, em seguida, anexá-los às instâncias do AWS OpsWorks Stacks no nível da camada. Os volumes do EBS são específicos para uma zona de disponibilidade. Se você não está mais usando um volume do EBS em nenhuma pilha de uma determinada região e zona de disponibilidade, você pode excluir o volume. Para obter mais informações sobre como excluir um volume do Amazon EBS, consulte Excluir um volume do Amazon EBS no Guia do usuário da Amazon EC2 .

Buckets do Amazon Simple Storage Service (Amazon S3)

No AWS OpsWorks Stacks, você pode usar os buckets do Amazon S3 para o seguinte. O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

- Armazenar código dos aplicativos
- Armazenar livros de receitas e receitas
- CloudTrail logs, se você tiver ativado o CloudTrail login no AWS OpsWorks Stacks
- Streams do Amazon CloudWatch Logs, se você os tiver habilitado no AWS OpsWorks Stacks

Endereços IP elásticos

Se você registrou endereços IP elásticos no AWS OpsWorks Stacks e não precisa mais dos endereços IP elásticos, você pode liberar o endereço IP elástico.

Load balancers Elastic Load Balancing

Se você não precisa mais de um Classic Load Balancer do Elastic Load Balancing usado com as camadas na pilha, você pode excluí-lo. Para obter mais informações, consulte Excluir o load balancer no Guia do usuário para Classic Load Balancers.

Instâncias Amazon Relational Database Service (Amazon RDS)

Se você registrou as instâncias de banco de dados do Amazon RDS no AWS OpsWorks Stacks e não precisa mais delas, é possível excluí-las. Para obter mais informações sobre como excluir instâncias de banco de dados, consulte Excluir uma instância de banco de dados no Guia do usuário do Amazon RDS.

Clusters Amazon Elastic Container Service (Amazon ECS)

Se a pilha incluiu camadas de cluster do ECS, e você não estiver mais usando o cluster do ECS que foi registrado com uma camada, você pode excluir o cluster do ECS. Para obter mais informações sobre como excluir um cluster do ECS, consulte Excluir um cluster no Guia do desenvolvedor do Amazon ECS .

Camadas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Toda pilha contém uma ou mais camadas, cada uma representando um componente de pilha, como um balanceador de carga ou um conjunto de servidores de aplicações.

Ao trabalhar com AWS OpsWorks as camadas do Stacks, lembre-se do seguinte:

- Cada camada em uma pilha deve ter pelo menos uma instância e pode ter várias instâncias.
- Cada instância em uma pilha deve ser membro de pelo menos uma camada, exceto para instâncias registradas.

  Você não pode configurar uma instância diretamente, exceto para algumas configurações básicas, como a chave SSH e o nome de host. Você deve criar e configurar uma layer apropriada, além de adicionar a instância à camada.

EC2 As instâncias da Amazon podem fazer parte de várias camadas. Nesse caso, o AWS OpsWorks Stacks executa as receitas para instalar e configurar pacotes, implantar aplicativos e assim por diante em cada uma das camadas da instância.

Ao atribuir uma instância a várias camadas, você pode, por exemplo, fazer o seguinte:

- Reduza despesas hospedando o servidor do banco de dados e o balanceador de carga em uma única instância.
- Use um dos servidores de aplicações para administração.

  Crie uma camada administrativa personalizada e adicione uma das instâncias do servidor de aplicações a essa camada. As receitas da camada administrativa configuram essa instância do servidor de aplicações para realizar tarefas administrativas e instalar qualquer software adicional obrigatório. As outras instâncias do servidor de aplicações são apenas servidores de aplicações.

Esta seção descreve como trabalhar com camadas.

Tópicos

- OpsWorks Noções básicas sobre camadas
- Camada Elastic Load Balancing
- Camada de serviços do Amazon RDS
- Camadas de cluster do ECS
- Camadas AWS OpsWorks de pilhas personalizadas
- Instalações de pacotes de sistema operacional por camada

OpsWorks Noções básicas sobre camadas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como executar operações que são comuns a todas as camadas do AWS OpsWorks Stacks.

Tópicos

- Criando uma OpsWorks camada
- Como editar a configuração de uma OpsWorks camada
- Como usar a correção automática para substituir instâncias com falha
- Excluindo uma camada OpsWorks

Criando uma OpsWorks camada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Ao criar uma nova pilha, consulte a página a seguir:

Para adicionar a primeira OpsWorks camada

1. Clique em Add a Layer.
2. Na página Add Layer selecione a camada apropriada, que exibe as opções de configuração da camada.
3. Configure adequadamente a camada e clique em Add Layer para adicioná-la à pilha. As seções a seguir descrevem como configurar as várias camadas.

   > **Note**
   >
   > A página Add Layer exibe apenas as configurações mais utilizadas para cada camada. Você pode especificar configurações adicionais editando a camada.

4. Adicione as instâncias à layer e as inicie.

> **Note**
>
> Se uma instância é membro de várias camadas, você precisa adicioná-la a todas elas antes de iniciá-la. Você não pode adicionar uma instância online a uma camada.

Para adicionar mais camadas, abra a página Layers e clique em + Layer para abrir a página Add Layer.

Quando você inicia uma instância, o AWS OpsWorks Stacks executa automaticamente as receitas de instalação e implantação de cada uma das camadas da instância para instalar e configurar os pacotes apropriados e implantar os aplicativos apropriados. Você pode personalizar um processo de instalação e configuração da camada de diversas maneiras, por exemplo, atribuindo receitas personalizadas a eventos de ciclo de vida apropriados. AWS OpsWorks O Stacks executa receitas personalizadas depois das receitas padrão para cada evento. Para obter mais informações, consulte Livros de receitas e receitas.

As seções específicas da camada a seguir descrevem como lidar com as etapas 2 e 3 para as várias camadas de AWS OpsWorks pilhas. Para obter mais informações sobre como adicionar instâncias, consulte Adicionar uma instância a uma camada.

Como editar a configuração de uma OpsWorks camada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Depois que você criar uma camada, algumas propriedades (como região da AWS) ficam imutáveis, mas é possível alterar a maioria das configurações da camada a qualquer momento. Editar a camada também fornece acesso às configurações não disponíveis na página Add Layer. As definições terão efeito assim que você salvar a nova configuração.

Para editar uma OpsWorks camada

1. No painel de navegação, clique em Layers.
2. Na página Layers, escolha um nome de camada para abrir a página de detalhes, que mostra a configuração atual.

> **Note**
>
> Escolher um dos nomes no nome da camada leva você diretamente para a guia associada na página de detalhes.

3. Clique em Edit e selecione a guia apropriada: General Settings, Recipes, Network, EBS Volumes ou Security.

As seções a seguir descrevem as configurações em várias guias que estão disponíveis para todas as camadas. Algumas camadas têm configurações adicionais específicas de camada, que aparecem na parte superior da página. Além disso, algumas configurações estão disponíveis apenas para pilhas baseadas em Linux, como observado.

Tópicos

- Configurações gerais
- Fórmulas
- Rede
- Volumes do EBS
- Segurança
- CloudWatch Registros
- Tags

Configurações gerais

Todas as camadas tem as seguintes configurações:

Correção automática ativada

Se a correção automática está ativada para as instâncias da camada. A configuração padrão é Sim.

JSON personalizado

Os dados no formato JSON que são passados para as receitas do Chef para todas as instâncias nessa camada. Você pode usar isso, por exemplo, para passar dados para suas próprias receitas. Para obter mais informações, consulte Usar JSON personalizado.

> **Note**
>
> Você pode declarar um JSON personalizado nos níveis de implantação, layer e pilha. É possível fazer isso se você quiser que alguns JSON personalizados sejam exibidos em toda a pilha ou apenas para uma implantação individual. Ou, por exemplo, você pode querer substituir temporariamente JSON personalizados declarados no nível da camada por JSON personalizados declarados no nível da implantação. Se você declarar JSON personalizados em vários níveis, os JSON personalizados declarados no nível da implantação substituirão os JSON personalizados declarados nos níveis de layer e pilha. Os JSON personalizados declarados no nível da camada substituem qualquer os JSON personalizados declarados apenas no nível da pilha.
>
> Para usar o console AWS OpsWorks Stacks para especificar o JSON personalizado para uma implantação, na página Implantar aplicativo, escolha Avançado. Insira o JSON personalizado na caixa Custom Chef JSON e escolha a opção Save.
>
> Para usar o console AWS OpsWorks Stacks para especificar JSON personalizado para uma pilha, na página de configurações da pilha, digite o JSON personalizado na caixa JSON personalizado e escolha Salvar.
>
> Para obter mais informações, consulte Usar JSON personalizado e Implementação de aplicativos.

Tempo limite de desligamento da instância

Especifica o tempo (em segundos) que o AWS OpsWorks Stacks aguarda depois de acionar um evento de ciclo de vida de Desligamento antes de parar ou encerrar a instância da Amazon. EC2 A configuração padrão é de 120 segundos. O propósito da configuração é oferecer as receitas de Desligamento da instância em tempo suficiente para concluir as tarefas antes de encerrar a instância. Se as receitas de Desligamento personalizadas exigirem mais tempo, modifique a configuração de acordo. Para obter mais informações sobre o desligamento de instância, consulte Interromper uma instância.

As demais configurações dessa guia variam de acordo com o tipo de camada e são idênticas às configurações na página Add Layer da camada.

Fórmulas

A guia Recipes inclui as configurações a seguir.

Custom Chef recipes

Você pode atribuir receitas personalizadas do Chef aos eventos de ciclo de vida da camada. Para obter mais informações, consulte Execução de receitas.

Rede

A guia Network inclui as configurações a seguir.

Elastic Load Balancing

Você pode anexar um load balancer do Elastic Load Balancing a qualquer camada. AWS OpsWorks Em seguida, o Stacks registra automaticamente as instâncias on-line da camada com o balanceador de carga e remove o registro delas quando elas ficam off-line. Se você ativou o recurso de drenagem de conexão do balanceador de carga, pode especificar se o AWS OpsWorks Stacks é compatível com ele. Para obter mais informações, consulte Camada Elastic Load Balancing.

Automatically Assign IP Addresses

Você pode controlar se o AWS OpsWorks Stacks atribui automaticamente endereços IP públicos ou elásticos às instâncias da camada. Veja o que acontece quando você ativa essa opção:

- Por exemplo, instâncias com base em lojas, o AWS OpsWorks Stacks atribui automaticamente um endereço sempre que a instância é iniciada.
- Para instâncias com base no Amazon EBS, o AWS OpsWorks Stacks atribui automaticamente um endereço quando a instância é iniciada pela primeira vez.
- Se uma instância pertencer a mais de uma camada, o AWS OpsWorks Stacks atribuirá automaticamente um endereço se você tiver ativado a atribuição automática para pelo menos uma das camadas,

> **Note**
>
> Se você ativa a atribuição automática de endereços IP públicos, ela se aplica apenas a instâncias novas. AWS OpsWorks O Stacks não pode atualizar o endereço IP público para instâncias existentes.

Se a pilha está sendo executada em uma VPC, você tem configurações separadas para endereços IP elásticos e públicos. A tabela a seguir explica como elas interagem:

		Public IP addresses	
		Yes	No
Elastic IP addresses	Yes	Instances receive an Elastic IP address when they are started for the first time, or a public IP address if an Elastic IP cannot be assigned.	Instances receive an Elastic IP address when they are started for the first time.
	No	Instances receive a public IP address each time they are started.	Instances receive only a private IP address, which is not accessible from outside the VPC.

> **Note**
>
> As instâncias devem ter uma forma de se comunicar com o serviço AWS OpsWorks Stacks, os repositórios de pacotes Linux e os repositórios de livros de receitas. Se você não especifica nenhum endereço IP elástico ou público, a VPC precisa incluir um componente, como uma camada NAT, que permita que as instâncias se comuniquem com sites externos. Para obter mais informações, consulte Execução de uma Stack em uma VPC.

Se sua pilha não estiver em execução em uma VPC, Elastic IP addresses será a única configuração:

- Yes: as instâncias recebem um endereço IP elástico quando são iniciadas pela primeira vez ou um endereço IP público quando não é possível atribuir um endereço IP elástico.
- No: as instâncias recebem um endereço IP público cada vez que são iniciadas.

Volumes do EBS

A guia EBS Volumes inclui as configurações a seguir.

Instâncias otimizadas para EBS

Se as instâncias da camada devem ser otimizadas para o Amazon Elastic Block Store (Amazon EBS). Para obter mais informações, consulte Instâncias otimizadas para o Amazon EBS.

Additional EBS Volumes

(Somente Linux) Você pode adicionar volumes do Amazon EBS às instâncias da camada ou removê-los de lá. Quando você inicia uma instância, o AWS OpsWorks Stacks cria automaticamente os volumes e os anexa às instâncias. Você pode usar a página Resources para gerenciar os volumes do EBS de uma pilha. Para obter mais informações, consulte Gerenciamento de recursos.

- Ponto de montagem : (obrigatório) especifique o ponto de montagem ou diretório em que o volume do EBS será montado.
- N° discos: (opcional) o número de discos na matriz, se você especificou uma matriz RAID.

  Cada nível de RAID tem um número padrão de discos, mas você pode selecionar um número maior na lista.
- Tamanho total (GiB) : (obrigatório) o tamanho do volume, em GiB.

  Para obter uma matriz RAID, essa configuração especifica o tamanho total da matriz, não o tamanho de cada disco.

  A tabela a seguir mostra os tamanhos mínimo e máximo permitidos para cada tipo de volume.

Tipo de volume	Tamanho mínimo (GiB)	Tamanho máximo (GiB)
Magnético	1	1024
IOPS provisionado (SSD)	4	16384
Finalidade geral (SSD)	1	16384
Disco rígido com throughput otimizado	500	16384
Disco rígido frio	500	16384

- Tipo de volume : (opcional) especifique se deseja criar um volume magnético, de SSD de uso geral, de disco rígido com throughput otimizado, de disco rígido frio ou de PIOPS.

O valor padrão é Magnetic.

- Criptografado: (opcional) especifique se deseja criptografar o conteúdo do volume do EBS.
- IOPS por disco: (obrigatório para volumes SSD de finalidade geral e SSD IOPS provisionados) se você especificar um volume SSD de finalidade geral ou SSD IOPS provisionado, também precisa especificar o IOPS por disco.

  Para os volumes de IOPS provisionados, você pode especificar a taxa de IOPS ao criar o volume. O coeficiente de IOPS provisionado e o tamanho do volume solicitado podem ser um máximo de 30 (em outras palavras, um volume com 3.000 IOPS deve ser de pelo menos 100 GB). Tipos de volume (SSD) de finalidade geral têm uma linha de base do tamanho do volume x 3 com um máximo de 10.000 IOPS e com a possibilidade de intermitência até 3.000 IOPS por 30 minutos.

Quando você adiciona volumes a uma camada ou os remove dela, observe o seguinte:

- Se você adiciona um volume, cada nova instância é o novo volume, mas o AWS OpsWorks Stacks não atualiza as instâncias existentes.
- Se você remove um volume, ele se aplica apenas a novas instâncias; as instâncias existentes mantêm os volumes.

Como especificar um ponto de montagem

Você pode especificar o ponto de montagem que desejar. No entanto, lembre-se de que alguns pontos de montagem são reservados para uso pelo AWS OpsWorks Stacks ou Amazon EC2 e não devem ser usadas para volumes do Amazon EBS. Não use pastas típicas do sistema Linux, como `/home` ou `/etc`.

Os seguintes pontos de montagem são reservados para uso por AWS OpsWorks Stacks.

- `/srv/www`
- `/var/log/apache2` (Ubuntu)
- `/var/log/httpd` (Amazon Linux)
- `/var/log/mysql`
- `/var/www`

Quando uma instância é inicializada ou reinicializada, autofs (uma montagem automática do daemon) usa pontos de montagem de dispositivos temporários como `/media/ephemeral0` para vincular montagens. Essa operação ocorre antes que os volumes do Amazon EBS sejam montados. Para garantir que o ponto de montagem do volume do Amazon EBS não entre em conflito com autofs, não especifique um ponto de montagem do dispositivo temporário. Os pontos de montagem do dispositivo temporário possível dependem do tipo de instância específico, e se ela é baseada em armazenamento de instâncias ou em Amazon EBS. Para evitar um conflito com autofs, faça o seguinte:

- Verifique os pontos de montagem do dispositivo temporário para o tipo de instância particular e o armazenamento de backup que você deseja usar.
- Lembre-se de que um ponto de montagem que funciona para uma instância baseada em armazenamento de instâncias pode entrar em conflito com autofs se você alterna para uma instância baseada em Amazon EBS ou vice-versa.

> **Note**
>
> Se você deseja alterar o mapeamento de dispositivos de blocos de armazenamento de instâncias, pode criar uma AMI personalizada. Para obter mais informações, consulte Amazon EC2 Instance Store. Para obter mais informações sobre como criar uma AMI personalizada para AWS OpsWorks pilhas, consulteUsando o Custom AMIs.

Veja a seguir um exemplo de como usar uma receita personalizada para garantir que um ponto de montagem do volume não entre em conflito com autofs. Você pode adaptá-la conforme o necessário para o caso de uso específico.

Para evitar um ponto de montagem conflitante

1. Atribua um volume do Amazon EBS à camada desejada, mas use um ponto de montagem como `/mnt/workspace` que nunca entre em conflito com autofs.
2. Implemente a seguinte receita personalizada, que cria um diretório de aplicativo no volume do Amazon EBS e se vincula a ele a partir de `/srv/www/`. Para obter mais informações sobre como implementar receitas personalizadas, consulte Livros de receitas e receitas e Personalizando pilhas AWS OpsWorks.

```
mount_point = node['ebs']['raids']['/dev/md0']['mount_point'] rescue nil

if mount_point
  node[:deploy].each do |application, deploy|
    directory "#{mount_point}/#{application}" do
      owner deploy[:user]
      group deploy[:group]
      mode 0770
      recursive true
    end

    link "/srv/www/#{application}" do
      to "#{mount_point}/#{application}"
    end
  end
end
```

3. Adicione uma linha `depends 'deploy'` ao arquivo `metadata.rb` do livro de receitas personalizado.
4. Atribua essa receita ao evento de Instalação da camada.

Segurança

A guia Security inclui as configurações a seguir.

Grupos de segurança

Uma camada deve ter pelo menos um grupo de segurança associado. Você especifica como associar grupos de segurança ao criar ou atualizar uma pilha. AWS OpsWorks O Stacks oferece um conjunto padronizado de grupos de segurança predefinidos.

- A opção padrão é fazer com que o AWS OpsWorks Stacks associe automaticamente o grupo de segurança incorporado a cada camada.
- Você também pode optar por não associar automaticamente os grupos de segurança incorporados e, em vez disso, associar um grupo de segurança personalizado a cada camada ao criá-la.

Para obter mais informações sobre grupos de segurança, consulte Usar grupos de segurança.

Após a camada ser criada, você pode usar Security Groups para adicionar mais security groups à camada selecionando-os na lista Custom security groups. Depois de adicionar um grupo de

segurança a uma camada, o AWS OpsWorks Stacks o adiciona a todas as novas instâncias. (Observe que as instâncias de armazenamento de instância que forem reiniciadas serão levadas às novas instâncias, de modo que também terão os novos grupos de segurança.) AWS OpsWorks O Stacks não adiciona grupos de segurança a instâncias on-line.

Você pode excluir os grupos de segurança existentes clicando no x, da seguinte forma:

- Se você optar por fazer com que o AWS OpsWorks Stacks associe automaticamente grupos de segurança integrados, você pode excluir grupos de segurança personalizados que você adicionou anteriormente clicando no x, mas não pode excluir o grupo integrado.
- Se você opta por não associar automaticamente os grupos de segurança incorporados, pode excluir todos os grupos de segurança existentes, incluindo o original, desde que a camada retenha pelo menos um grupo.

Depois que você remove um grupo de segurança de uma camada, o AWS OpsWorks Stacks não o adiciona a nenhuma instância nova ou reiniciada. AWS OpsWorks O Stacks não remove grupos de segurança de instâncias on-line.

> **Note**
>
> Se a pilha for executada em uma VPC, você poderá adicionar ou remover um grupo de segurança para uma instância on-line usando o console EC2 Amazon, a API ou a CLI. No entanto, esse grupo de segurança não estará visível no console do AWS OpsWorks Stacks. Se você deseja remover o grupo de segurança, precisa usar também a Amazon EC2. Para obter mais informações, consulte Grupos de segurança.

Observe o seguinte:

- Você não pode restringir as configurações de acesso à porta de um grupo de segurança incorporado adicionando um grupo de segurança mais restritivo. Quando há vários grupos de segurança, a Amazon EC2 usa as configurações mais permissivas.
- Você não deve modificar a configuração de um grupo de segurança incorporado. Quando você cria uma pilha, o AWS OpsWorks Stacks substitui as configurações dos grupos de segurança integrados, para que todas as alterações feitas sejam perdidas na próxima vez que você criar uma pilha.

Se você descobrir que precisa de configurações mais restritivas para grupos de segurança para uma ou mais camadas, siga estas etapas:

1. Crie grupos de segurança personalizados com configurações apropriadas e os adicione às camadas apropriadas.

   Cada camada na pilha precisa ter pelo menos um grupo de segurança além do grupo incorporado, mesmo se apenas uma camada exigir configurações personalizadas.

2. Edite a configuração da pilha e alterne a configuração Usar grupos OpsWorks de segurança para Não.

   AWS OpsWorks O Stacks remove automaticamente o grupo de segurança integrado de cada camada.

Para obter mais informações sobre grupos de segurança, consulte Grupos EC2 de segurança da Amazon.

EC2 Perfil da instância

Você pode alterar o EC2 perfil das instâncias da camada. Para obter mais informações, consulte Especificação de permissões para aplicativos executados em instâncias EC2 .

CloudWatch Registros

A guia CloudWatch Registros permite ativar ou desativar o Amazon CloudWatch Logs. CloudWatch A integração do Logs funciona com as pilhas baseadas em Linux do Chef 12 e do Chef 12. Para obter mais informações sobre como ativar CloudWatch a integração do Logs e especificar os logs que você deseja gerenciar no console do CloudWatch Logs, consulteUsando Amazon CloudWatch Logs com AWS OpsWorks pilhas.

Tags

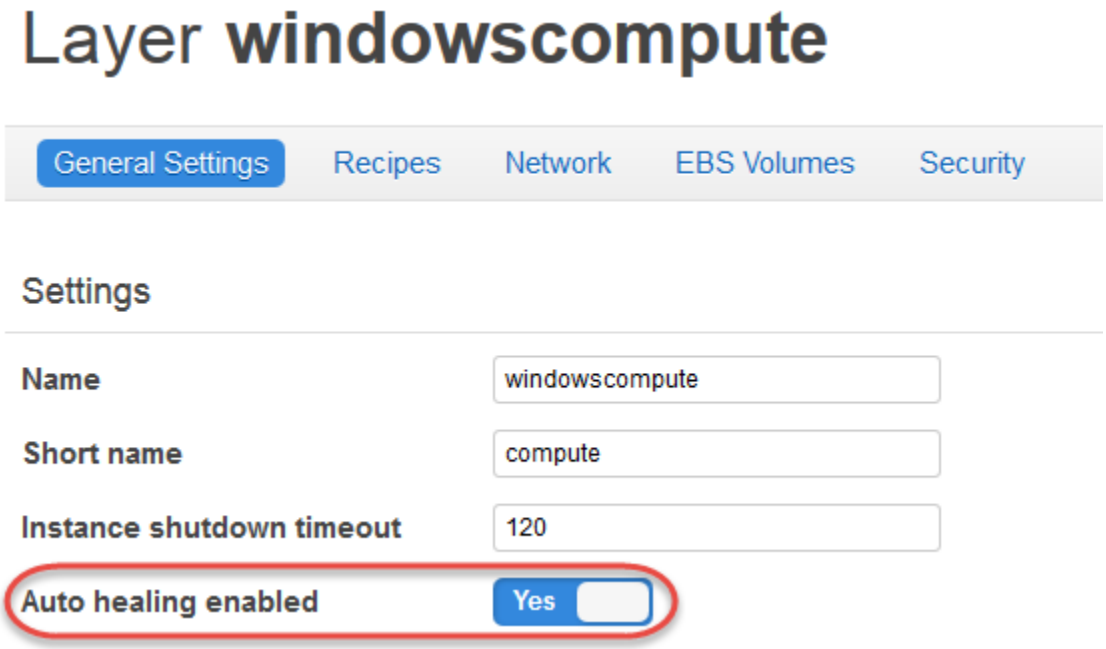
A guia Tags permite aplicar tags de alocação de custos à camada. Depois de adicionar tags, você pode ativá-las no Gerenciamento de Faturamento e Custos da AWS console. Quando você cria uma tag, está aplicando-a para cada recurso dentro da estrutura com a tag. Por exemplo, se você aplica uma tag a uma camada, está aplicando a tag a cada instância, volume do Amazon EBS ou balanceador de carga do Elastic Load Balancing na camada. Para obter mais informações sobre como ativar as tags e usá-las para monitorar e gerenciar os custos dos recursos do AWS OpsWorks Stacks, consulte Como usar tags de alocação de custos e Como ativar tags de alocação de custos definidas pelo usuário no Guia do usuário do Faturamento e gerenciamento de custos. Para obter mais informações sobre tags no AWS OpsWorks Stacks, consulte Tags.

Como usar a correção automática para substituir instâncias com falha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Cada instância tem um agente do AWS OpsWorks Stacks que se comunica regularmente com o serviço. AWS OpsWorks O Stacks usa essa comunicação para monitorar a integridade da instância. Se um agente não se comunicar com o serviço por mais de aproximadamente cinco minutos, o AWS OpsWorks Stacks considera que a instância falhou.

A correção automática é definida no nível da camada; você pode alterar a configuração da correção automática ao editar as configurações da camada, como mostrado na captura de tela a seguir.



> **Note**
>
> Uma instância pode ser um membro de várias camadas. Se alguma dessas camadas tiver a cura automática desativada, o AWS OpsWorks Stacks não recuperará a instância se ela falhar.

Se uma camada está com a correção automática ativada, a configuração padrão, o AWS OpsWorks Stacks substitui automaticamente as instâncias com falha da camada da seguinte forma:

Instâncias baseadas em armazenamento de instâncias

1. Interrompe a EC2 instância da Amazon e verifica se ela foi desativada.
2. Exclui os dados no volume raiz.
3. Cria uma nova EC2 instância da Amazon com os mesmos nomes de host, configuração e associação de camada.
4. Anexa novamente todos os volumes do Amazon EBS, incluindo aqueles que foram anexados após o início original da antiga instância.
5. Atribui um novo endereço IP público e privado.
6. Se a instância antiga foi associada a um endereço IP elástico, associa a instância nova ao mesmo endereço IP.

Instância baseada no Amazon EBS

1. Interrompe a EC2 instância da Amazon e verifica se ela foi interrompida.
2. Inicia a EC2 instância.

Depois que a instância com recuperação automática estiver on-line novamente, o AWS OpsWorks Stacks aciona um evento Configure lifecycle em todas as instâncias da pilha. Os atributos de configuração e implantação da pilha associados incluem endereços IP privados e públicos da instância. As receitas Configurar personalizadas podem obter os novos endereços IP do objeto do nó.

Se você especifica um volume do Amazon EBS para as instâncias de uma camada, o AWS OpsWorks Stacks cria um volume novo e o anexa a cada instância iniciada. Se depois você quiser desanexar o volume de uma instância, use a página Recursos.

Quando o AWS OpsWorks Stacks cura automaticamente uma das instâncias de uma camada, ele manipula os volumes da seguinte maneira:

- Se o volume foi anexado à instância quando a instância falhou, o volume e seus dados serão salvos e o AWS OpsWorks Stacks os anexará à nova instância.
- Se o volume não estava anexado à instância quando ela apresentou falha, o AWS OpsWorks Stacks cria um novo volume vazio com a configuração especificada pela camada e anexa esse volume à nova instância.

A correção automática é ativada por padrão para todas as camadas, mas você pode editar as configurações gerais da layer para desativá-la.

> **Important**
>
> Se a correção automática estiver ativada, faça o seguinte:
>
> - Use somente o console, a CLI ou a API do AWS OpsWorks Stacks para interromper as instâncias.
>
> Se você interrompe uma instância de qualquer outra maneira, como ao usar o EC2 console Amazon, por exemplo, o AWS OpsWorks Stacks trata a instância como se estivesse com falha e realiza a recuperação automática dela.
> - Use os volumes do Amazon EBS para armazenar os dados que você não deseja perder se a instância for corrigida automaticamente.
>
> A correção automática interrompe a antiga EC2 instância Amazon, e isso destrói os dados que não estão armazenados em um volume do Amazon EBS. Os volumes do Amazon EBS são reanexados à instância nova, o que preserva os dados armazenados.

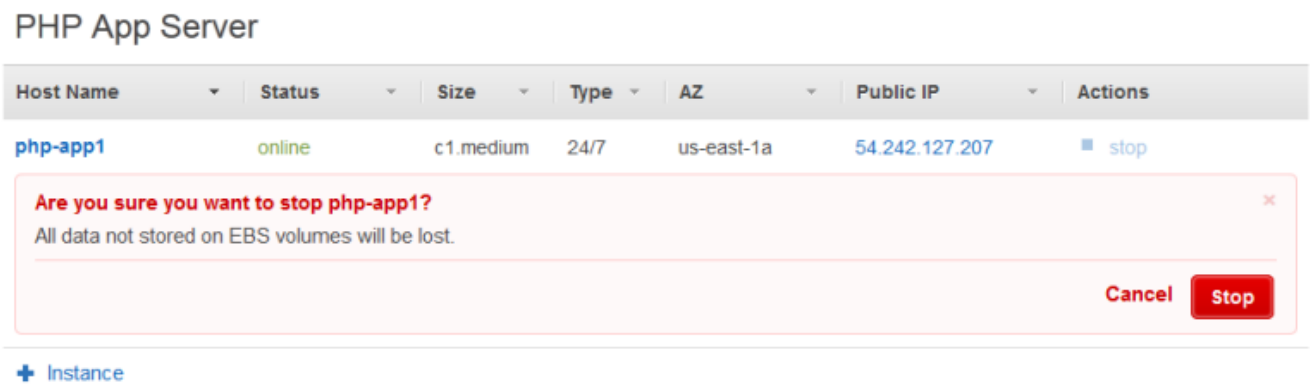
Excluindo uma camada OpsWorks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

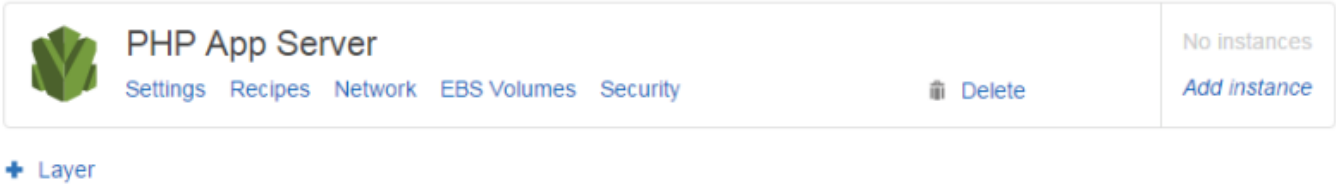
Se você não precisar mais de uma camada de AWS OpsWorks pilhas, poderá excluí-la da pilha.

Para excluir uma OpsWorks camada

1. No painel de navegação, clique em Instances.
2. Na página Instances, sob o nome da camada que você deseja excluir, clique em stop na coluna Actions de cada instância.



3. Depois que a instância for interrompida, clique em delete para removê-la da camada.
4. No painel de navegação, clique em Layers.
5. Na página Layers, selecione Delete.



Camada Elastic Load Balancing

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O Elastic Load Balancing funciona de maneira um pouco diferente de uma camada do AWS OpsWorks Stacks. Em vez de criar uma camada e adicionar instâncias a ela, use o console do Elastic Load Balancing ou API para criar um balanceador de carga e depois anexá-lo a uma camada existente. Além de distribuir tráfego às instâncias da camada, o Elastic Load Balancing faz o seguinte:

- Detecta EC2 instâncias Amazon não íntegras e redireciona tráfego para as instâncias íntegras restantes até que as instâncias com problemas sejam restauradas.
- Escala automaticamente solicitação de manuseio de capacidade em resposta a tráfego de entrada.
- Se você habilitar a diminuição de conexão, o load balancer interromperá o envio de novas solicitações que não são íntegras ou que estão prestes a terem seu registro cancelado, mas mantém a conexão ativa, até um valor de tempo limite específico, para permitir que a instância conclua qualquer solicitação em trânsito.

Depois de anexar um balanceador de carga a uma camada, o AWS OpsWorks Stacks faz o seguinte:

- Cancela o registro de quaisquer instâncias atualmente registradas.
- Registra automaticamente as instâncias da camada quando ficam online e cancela o registro das instâncias quando elas ficam offline, incluindo instâncias baseadas em carga e tempo.
- Inicia automaticamente as solicitações de roteamento para instâncias registradas nas zonas de disponibilidade.

Se você ativou o recurso de drenagem de conexão do balanceador de carga, pode especificar se o AWS OpsWorks Stacks é compatível com ele. Se você ativar o suporte à drenagem de conexão (a configuração padrão), depois que uma instância for encerrada, o AWS OpsWorks Stacks fará o seguinte:

- Cancela o registro da instância do load balancer.

  O load balancer interrompe o envio de novas solicitações e inicia a drenagem de conexão.
- Atrasa o lançamento de um Desligamento de evento de ciclo de vida até que load balancer tenha concluído a drenagem de conexão.

Se você não ativar o suporte à drenagem de conexão, o AWS OpsWorks Stacks acionará o evento Shutdown assim que a instância for encerrada, mesmo que ela ainda esteja conectada ao balanceador de carga.

Para usar o Elastic Load Balancing com uma pilha, você deve primeiro criar um ou mais load balancers na mesma região usando o console, a CLI ou a API do Elastic Load Balancing. Você deve estar ciente do seguinte:

- Você pode anexar apenas um load balancer a uma layer.
- Cada load balancer pode lidar com apenas uma layer.
- AWS OpsWorks O Stacks não é compatível com o Application Load Balancer. Você só pode usar o Classic Load Balancer com o AWS OpsWorks Stacks.

Isso significa que você deve criar um balanceador de carga do Elastic Load Balancing separado para cada camada em cada pilha que você deseja balancear e usá-lo apenas para esse fim. Uma prática recomendada é atribuir um nome distinto a cada balanceador de carga do Elastic Load Balancing que você planeja usar com o AWS OpsWorks Stacks, como MyStack 1- RailsLayer -ELB, para evitar o uso de um balanceador de carga para mais de uma finalidade.

> **Important**
>
> Recomendamos criar novos balanceadores de carga do Elastic Load Balancing para suas AWS OpsWorks camadas do Stacks. Se escolher usar um balanceador de carga do Elastic Load Balancing existente, você deve primeiro confirmar que não está sendo usado para outros fins e não tem instâncias anexadas. Depois que o balanceador de carga é anexado à camada, OpsWorks remove todas as instâncias existentes e configura o balanceador de carga para lidar somente com as instâncias da camada. Apesar de ser tecnicamente possível usar o console do Elastic Load Balancing ou API para modificar a configuração do balanceador de carga após anexá-lo a camada, você não deve fazê-lo; as mudanças não serão permanentes.

Para anexar um balanceador de carga do Elastic Load Balancing a uma camada

1. Caso ainda não tenha feito isso, use o console, a API, ou a CLI do Elastic Load Balancing para criar um balanceador de carga na região da pilha. Quando você cria o load balancer, faça o seguinte:
   - Não se esqueça de especificar uma integridade de um caminho de ping que seja apropriado para o aplicativo.

O caminho de ping padrão `/index.html`, para que a raiz de seu aplicativo não inclua `index.html`, você deve especificar um caminho de ping apropriado ou a verificação de integridade falhará.

- Se quiser usar a diminuição de conexão, verifique se o recurso está habilitado e tem um valor de tempo limite apropriado.

Para obter mais informações, consulte Elastic Load Balancing.

2. Criar a layer que quer ter balanceada ou edite uma configuração de rede da layer existente.

> **Note**
>
> Você não pode anexar um load balancer quando criar uma layer personalizada. Você deve editar as configurações da layer.

3. Em Elastic Load Balancing, selecione o balanceador de carga que você deseja anexar à camada e especifique se deseja que as AWS OpsWorks pilhas suportem a drenagem da conexão.

Depois que você anexa um balanceador de carga a uma camada, o AWS OpsWorks Stacks aciona um evento de ciclo de vida de configuração nas instâncias da pilha para notificá-las sobre a alteração. AWS OpsWorks O Stacks também aciona um evento de configuração quando você desanexa um balanceador de carga.

> **Note**
>
> Depois que uma instância é inicializada, o AWS OpsWorks Stacks executa as receitas de configuração e implantação, que instalam pacotes e implantam aplicativos. Depois que essas receitas terminarem, a instância estará no estado on-line e o AWS OpsWorks Stacks registrará a instância no Elastic Load Balancing. AWS OpsWorks O Stacks também aciona um evento de configuração após a instância ficar on-line. Isso significa que o registro de Elastic Load Balancing e o Configurar receitas pode rodar de forma conjunta, e a instância pode ser registrada antes do Configura receitas ter terminado. Para garantir que a receita termina antes de uma instância ser registrada com Elastic Load Balancing, você deve adicionar a receita aos eventos Configurar ou Implantar ciclo de vida da camada. Para obter mais informações, consulte Execução de receitas.

Às vezes é útil remover uma instância de um load balancer. Por exemplo, quando você atualiza um aplicativo, recomendamos que você implante o aplicativo à única instância e verifique se o aplicativo está funcionando corretamente antes de implanta-lo a todas as instâncias. Você geralmente remove essa instância do load balancer, então ele não recebe solicitações de usuário até que você verifique a atualização.

Você deve usar o console do Elastic Load Balancing ou API para remover temporariamente uma instância on-line de um balanceador de carga. A seguir, veja a descrição de como usar o console.

Para remover temporariamente uma instância de um load balancer

1. Abra o EC2 console da Amazon e escolha Load Balancers.
2. Escolha o load balancer apropriado e abra a guia Instances.
3. Escolha Remove from Load Balancer na coluna Actions da instância.
4. Quando tiver concluído, escolha Edit Instances e volte à instância ao load balancer.

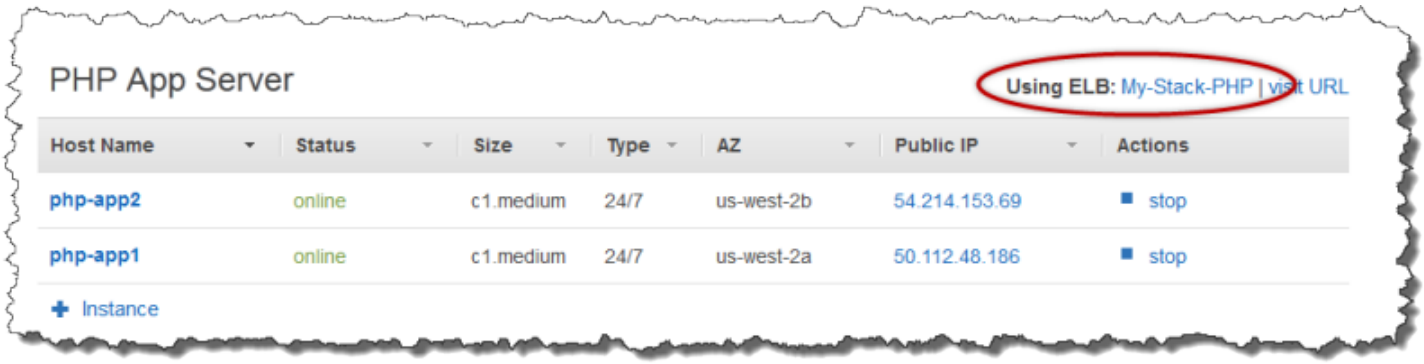
> **Important**
>
> Se você usar o console ou a API do Elastic Load Balancing para remover uma instância de um balanceador de carga, também deverá usar o Elastic Load Balancing para recuperá-la. AWS OpsWorks O Stacks não sabe das operações que você realiza com outros consoles de serviço e não retornará a instância para o balanceador de carga para você. APIs Às vezes, o AWS OpsWorks Stacks pode adicionar a instância de volta ao ELB, mas esse comportamento não é garantido e não ocorre em todos os casos.

Você pode anexar vários load balancers a um grupo particular de instância, como a seguir:

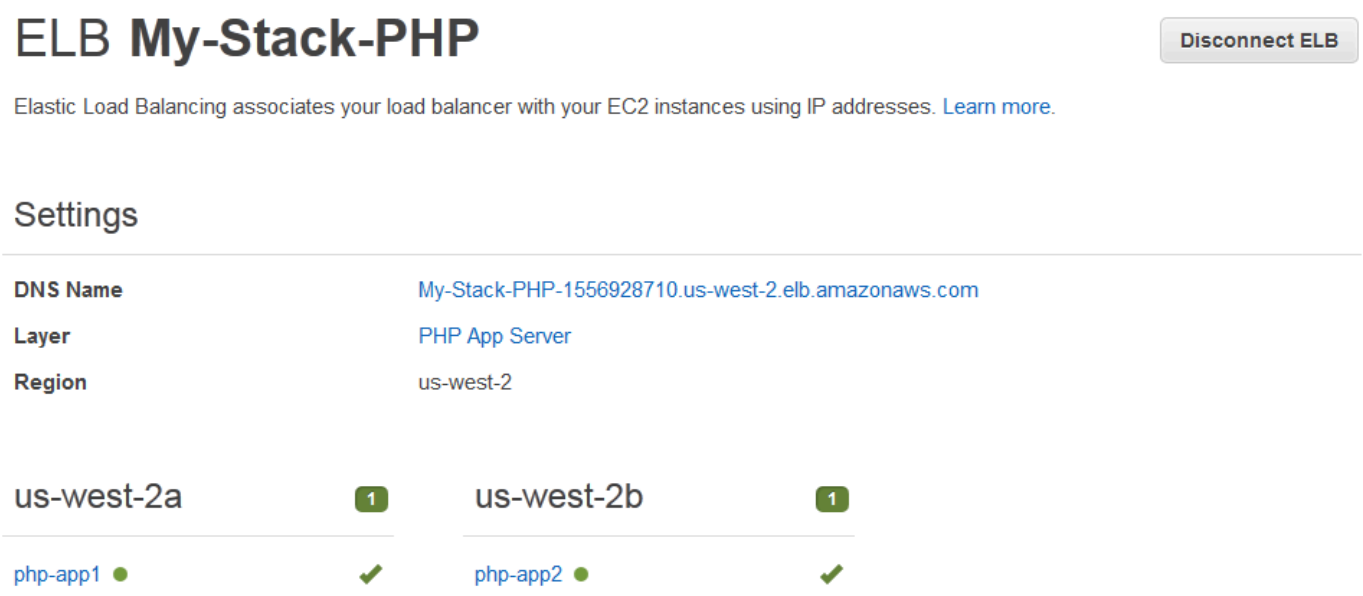
Para anexar vários load balancers

1. Use a API, a CLI ou o console do Elastic Load Balancing para criar um conjunto de balanceadores de carga.
2. Criar uma layer personalizada para cada load balancer e anexar um do load balancers a ele. Você não precisa implementar quaisquer receitas personalizadas para essas layers; como padrão, layer personalizada é suficiente.
3. Adicione o conjunto de instâncias a cada layer personalizada.

Você pode examinar as propriedades de um load balancer indo para a página Instâncias e clicando no nome do load balancer apropriado.



A página ELB mostra as propriedades básicas do load balancer, incluindo seu nome DNS e o status saudável das instâncias associadas. Se o stack estiver executando em uma VPC, a página mostra sub-redes em vez das Zonas de disponibilidade. Uma marcação verde indica uma instância saudável. Você pode clicar no nome para conectar a um servidor, por meio do load balancer.



Camada de serviços do Amazon RDS

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Uma camada de serviço Amazon RDS representa uma instância do Amazon RDS. A camada pode representar apenas instâncias do Amazon RDS existentes, as quais você deve criar separadamente usando o console do Amazon RDS ou API.

O procedimento básico para incorporar uma camada de serviço Amazon RDS em sua pilha é apresentado a seguir:

1. Use o console, API ou CLI do Amazon RDS para criar uma instância.

   Certifique-se de registrar o ID, o nome de usuário mestre, a senha mestre e o nome do banco de dados da instância.
2. Para adicionar uma camada Amazon RDS em sua pilha, registre a instância do Amazon RDS na pilha.
3. Anexe a camada a um aplicativo, o que irá adicionar as informações de conexão da instância do Amazon RDS aos `deploy` atributos do aplicativo.
4. Use os arquivos para o idioma específico ou as informações nos atributos `deploy` para conectar o aplicativo à instância do Amazon RDS.

   Para obter mais informações sobre como conectar um aplicativo a um servidor de banco de dados, consulte the section called “Conectar-se a um banco de dados”

> **Warning**
>
> Certifique-se de que os caracteres da senha mestre e do nome de usuário da instância são compatíveis com o seu servidor de aplicativo. Por exemplo, em uma camada Java App Server, incluir & em uma das strings gera um erro de análise XML que impede a inicialização do servidor Tomcat.

Tópicos

- Como especificar grupos de segurança
- Como registrar uma instância do Amazon RDS em uma pilha
- Como associar camadas de serviço Amazon RDS aos aplicativos

- Como remover uma camada de serviço Amazon RDS de uma pilha

Como especificar grupos de segurança

Para usar uma instância do Amazon RDS no AWS OpsWorks Stacks, o banco de dados ou os grupos de segurança da VPC devem permitir o acesso aos endereços de IP apropriados. Para uso da produção, um grupo de segurança normalmente limita o acesso a apenas os endereços de IP que precisam acessar o banco de dados. Ele geralmente inclui os endereços dos sistemas usados para gerenciar o banco de dados e as instâncias do AWS OpsWorks Stacks que precisam acessar o banco de dados. AWS OpsWorks O Stacks cria automaticamente um grupo EC2 de segurança da Amazon para cada tipo de camada ao criar sua primeira pilha em uma região. Uma maneira simples de fornecer acesso às instâncias do AWS OpsWorks Stacks é atribuir os grupos de segurança do AWS OpsWorks Stacks apropriados à instância do Amazon RDS ou à VPC.

Para especificar grupos de segurança em uma instância do Amazon RDS existente

1. Abra o console do Amazon RDS em https://console.aws.amazon.com/rds/.
2. Clique em Instâncias no painel de navegação e selecione a instância do Amazon RDS apropriada. Clique em Instance Actions, Modify.
3. Selecione os seguintes grupos de segurança na lista Security Group e, em seguida, clique em Continue e Modify DB Instance para atualizar a instância.
   - O grupo de segurança AWS- OpsWorks -DB-Master-Server (). `security_group_id`
   - O grupo de segurança para a camada do servidor de aplicativos cujas instâncias serão conectadas ao banco de dados. O nome do grupo inclui o nome da camada. Por exemplo, para fornecer acesso ao banco de dados para instâncias do PHP App Server, especifique o grupo AWS- OpsWorks PHP-App-Server.

Se estiver criando uma nova instância do Amazon RDS, você pode especificar os grupos de segurança do AWS OpsWorks Stacks apropriados na página Definir configurações avançadas do assistente Iniciar uma instância de banco de dados. Para obter uma descrição de como usar este assistente, consulte Criar uma instância de banco de dados MySQL e conectar-se a um banco de dados em uma instância de banco de dados MySQL.

Para obter informações sobre como especificar os grupos de segurança da VPC, consulte Grupos de segurança para sua VPC.

Como registrar uma instância do Amazon RDS em uma pilha

Para adicionar uma camada de serviço Amazon RDS em uma pilha, você deve registrar uma instância em uma pilha.

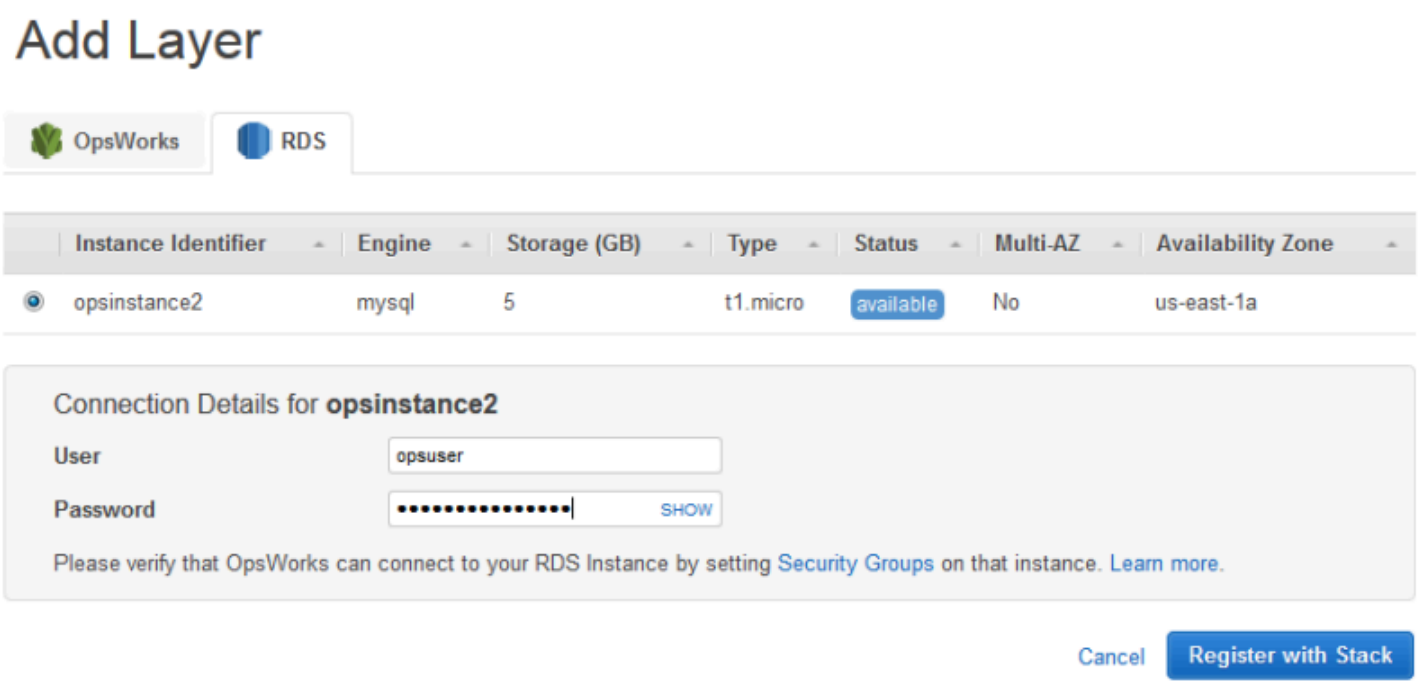
Para registrar uma instância do Amazon RDS em uma pilha

1. No console AWS OpsWorks Stacks, clique em Camada no painel de navegação, clique em + Camada ou Adicionar uma camada para abrir a página Adicionar camada e, em seguida, clique na guia RDS.
2. Se necessário, atualize a função de serviço da pilha, conforme descrito em Atualização da função de serviço da pilha.
3. Clique na guia RDS para listar as instâncias do Amazon RDS disponíveis.

> **Note**
>
> Se sua conta não possui instâncias do Amazon RDS, você poderá criar uma clicando em Adicionar uma instância do RDS na aba RDS, que o levará para o console do e iniciará o assistente Iniciar uma instância de banco de dados. Também é possível ir diretamente para o console do Amazon RDS e clicar em Iniciar uma instância de banco de dados, ou usar o API ou CLI do Amazon RDS. Para obter mais informações sobre como criar uma instância do Amazon RDS, consulte Conceitos básicos do Amazon RDS.

4. Selecione a instância apropriada, defina User e Password como os valores apropriados de usuário e senha e clique em Register to Stack.

> **Important**
>
> É necessário garantir que o usuário e a senha usados para registrar a instância Amazon RDS correspondem a um usuário e senha válidos. Caso não correspondam, seus aplicativos não serão capazes de se conectar à instância. No entanto, você pode editar a camada para fornecer valores de usuário e senha válidos e, em seguida, implementar novamente o aplicativo.



Quando você adiciona uma camada de serviço Amazon RDS a uma pilha, o AWS OpsWorks Stacks atribui a ela um ID e adiciona a configuração do Amazon RDS associada ao atributo atributos de implantação e configuração da pilha. `[:opsworks][:stack]`

> **Note**
>
> Se você alterar a senha de uma instância do Amazon RDS registrada, você deve atualizar a senha manualmente no AWS OpsWorks Stacks e, em seguida, implementar novamente seus aplicativos para atualizar os atributos de implantação e configuração da pilha nas instâncias da pilha.

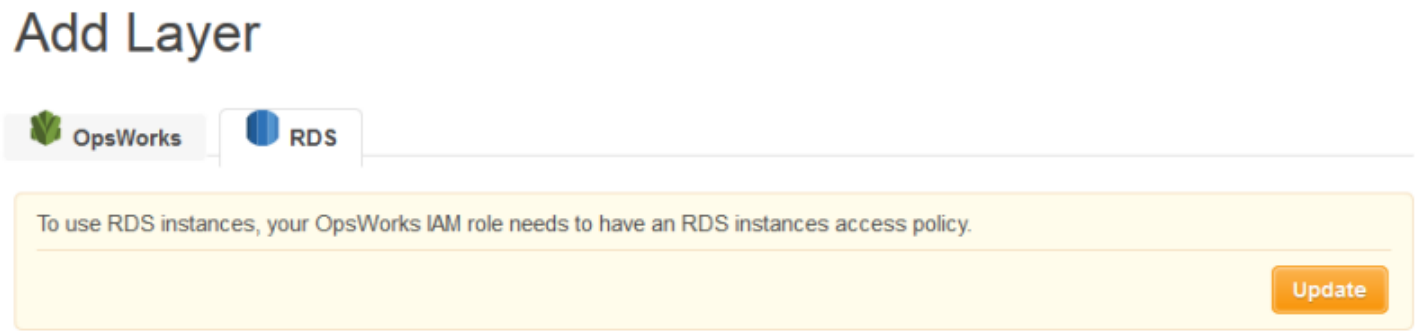
Tópicos

- Atualização da função de serviço da pilha

Atualização da função de serviço da pilha

Toda pilha tem uma função de serviço do IAM que especifica quais ações o AWS OpsWorks Stacks pode executar em seu nome, com outros serviços da AWS. Para registrar uma instância do Amazon RDS em uma pilha, seu perfil de serviço deve conceder ao AWS OpsWorks Stacks permissões de acesso ao Amazon RDS.

Na primeira vez que você adicionar uma camada de serviço Amazon RDS em uma de suas pilhas, o perfil de serviço pode não ter as permissões necessárias. Se esse for o caso, ao clicar na guia RDS, na página Add Layer, você verá o seguinte.



Clique em Atualizar para que o AWS OpsWorks Stacks atualize a política da função de serviço para o seguinte.

```
{"Statement": [{"Action": ["ec2:*", "iam:PassRole",
                       "cloudwatch:GetMetricStatistics",
                       "elasticloadbalancing:*",
                       "rds:*"],
            "Effect": "Allow",
            "Resource": ["*"] }]
}
```

> **Note**
>
> Só é necessário realizar a atualização uma vez. A função atualizada será então usada automaticamente por todas as suas pilhas.

Como associar camadas de serviço Amazon RDS aos aplicativos

Após adicionar uma camada de serviço Amazon RDS, você pode associá-la a um aplicativo.

- Você pode associar uma camada do Amazon RDS a um aplicativo ao criar o aplicativo, ou, posteriormente, editando a configuração do aplicativo.
- Para desassociar uma camada do Amazon RDS de um aplicativo, edite a configuração do aplicativo para especificar um outro servidor de banco de dados, ou deixar sem servidor.

A camada do Amazon RDS permanece como parte da pilha, e pode ser associada a outros aplicativos.

Depois que você associa uma instância do Amazon RDS a um aplicativo, o AWS OpsWorks Stacks coloca as informações de conexão do banco de dados no servidor do aplicativo. O aplicativo em cada instância de servidor pode então usar essas informações para se conectar ao banco de dados. Para obter mais informações sobre como se conectar a uma instância do Amazon RDS, consulte the section called "Conectar-se a um banco de dados".

Como remover uma camada de serviço Amazon RDS de uma pilha

Para remover uma camada de serviço Amazon RDS de uma pilha, basta cancelar seu registro.

Para cancelar o registro de uma camada de serviços do Amazon RDS

1. Clique em Camadas no painel de navegação e clique no nome da camada de serviço Amazon RDS.
2. Clique em Deregister e confirme que você deseja cancelar o registro da camada.

Este procedimento remove a camada da pilha, mas não exclui a instância do Amazon RDS subjacente. A instância e quaisquer bancos de dados permanecerão em sua conta e podem ser registrados em outras pilhas. Você deve usar o console, API ou CLI do Amazon RDS para excluir a instância. Para obter mais informações, consulte Exclusão de uma instância de banco de dados.

Camadas de cluster do ECS

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O Amazon Elastic Container Service (Amazon ECS) gerencia contêineres do Docker em um cluster de instâncias do Amazon Elastic Compute Cloud (Amazon EC2), conhecidas como instâncias de

contêineres. Uma camada do Cluster ECS representa um cluster do Amazon ECS, e simplifica o gerenciamento de clusters fornecendo recursos que incluem:

- Provisionamento e gerenciamento simplificado de instâncias de contêiner
- Atualizações do sistema operacional e do pacote de instâncias de contêiner
- Gerenciamento de permissões de usuários
- Monitoramento do desempenho de instâncias de contêiner
- Gerenciamento de volume do Amazon Elastic Block Store (Amazon EBS)
- Gerenciamento de endereços IP público e elástico
- Gerenciamento de grupos de segurança

A camada do Cluster ECS apresenta as seguintes restrições e requisitos:

- A camada está disponível apenas para pilhas do Chef 11.10 ou Chef 12 do Linux em execução em uma VPC, incluindo uma VPC padrão.
- As instâncias da camada devem ter um dos seguintes sistemas operacionais em execução.
  - Amazon Linux 2
  - Amazon Linux 2018.03
  - Amazon Linux 2017.09
  - Amazon Linux 2017.03
  - Amazon Linux 2016.09
  - Amazon Linux 2016.03
  - Amazon Linux 2015.09
  - Amazon Linux 2015.03
  - Ubuntu 18.04 LTS
  - Ubuntu 16.04 LTS
  - Ubuntu 14.04 LTS
  - Personalizado
- A versão do agente do AWS OpsWorks Stacks nas instâncias da camada deve ser `3425-20150727112318` ou posterior.

Tópicos

- Adição de uma camada do Cluster ECS a uma pilha
- Gerenciamento do cluster do ECS
- Exclusão de uma camada do ECS Cluster de uma pilha

Adição de uma camada do Cluster ECS a uma pilha

AWS OpsWorks O Stacks simplifica o processo de execução e manutenção de instâncias de contêiner para os clusters existentes do Amazon ECS. Para criar ou iniciar outras entidades do Amazon ECS, como clusters e tarefas, use o console do Amazon ECS, a interface da linha de comando (CLI – command line interface) ou a API. (Para obter mais informações, consulte o Guia do desenvolvedor do Amazon Elastic Container Service.) Em seguida, você pode associar um cluster a uma pilha criando uma camada do ECS Cluster que pode ser usada para gerenciar o cluster no AWS OpsWorks Stacks.

Você pode associar clusters a pilhas da seguinte forma:

- Cada pilha pode ter uma camada do Amazon ECS, que representa um único cluster.
- Um cluster pode ser associado a apenas uma pilha.

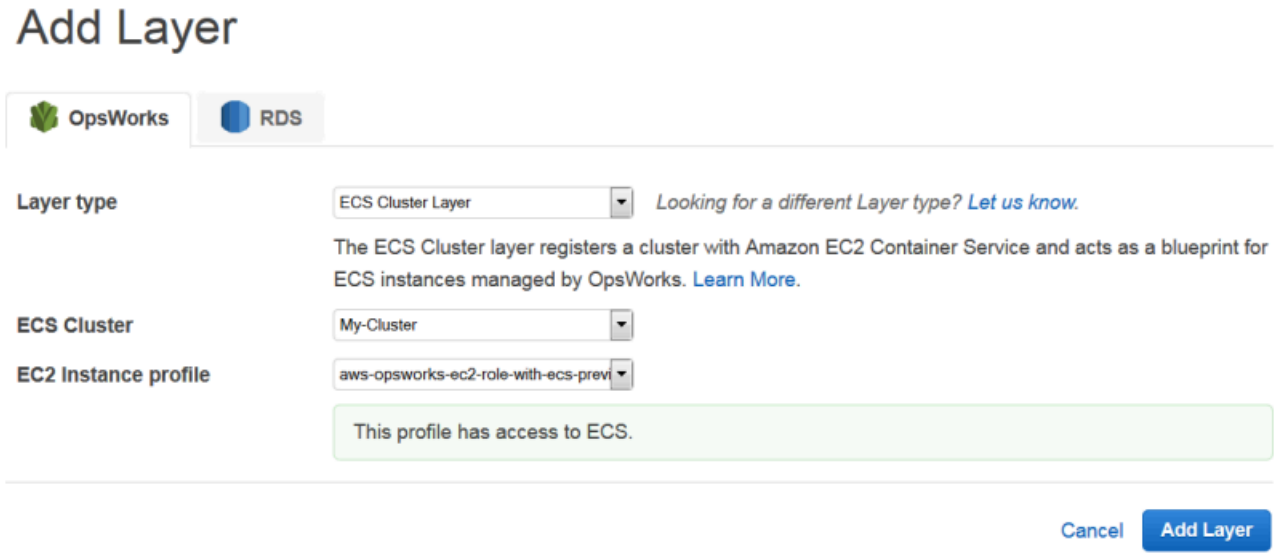
Antes de adicionar as camadas ECS Cluster às suas pilhas, você deve atualizar o perfil de serviço do AWS OpsWorks Stacks AWS Identity and Access Management (IAM), que é geralmente chamado de`aws-opsworks-service-role`, para permitir que o AWS OpsWorks Stacks interaja com o Amazon ECS em seu lugar. Para obter mais informações sobre as funções de serviço, consulte Permitindo que AWS OpsWorks as pilhas ajam em seu nome.

Na primeira vez que você cria uma camada do ECS Cluster, o console fornece um botão Atualizar, que você pode usar para fazer com que o AWS OpsWorks Stacks atualize o perfil para você. AWS OpsWorks O Stacks, em seguida, exibe a página Adicionar camada para que você possa adicionar a camada à pilha. Você só precisa atualizar a função de serviço uma vez. Em seguida, você pode usar o perfil atualizado para adicionar uma camada do ECS Cluster a qualquer pilha.

> **Note**
>
> Se você preferir, pode atualizar manualmente a política da função de serviço adicionando a permissão `ecs:*` à política existente da seguinte forma:

```
{
  "Statement": [
    {
      "Action": [
        "ec2:*",
        "iam:PassRole",
        "cloudwatch:GetMetricStatistics",
        "elasticloadbalancing:*",
        "rds:*",
        "ecs:*"
      ],
      "Effect": "Allow",
      "Resource": ["*"]
    }
  ]
}
```

A associação de um cluster a uma pilha exige duas operações: o registro do cluster na pilha e, em seguida, a criação da camada associada. O console AWS OpsWorks Stacks combina essas etapas; a criação da camada registra automaticamente o cluster especificado. Se você usar a API, a CLI ou o SDK do AWS OpsWorks Stacks, deverá usar operações separadas para registrar o cluster e criar a camada associada. Para usar o console para adicionar uma camada do ECS Cluster a sua pilha, escolha Camadas e, em seguida, + Camada ou Adicionar uma camada. Depois, escolha o tipo da camada do ECS Cluster.



A página Add Layer inclui as seguintes opções de configuração:

Cluster do ECS

O cluster do Amazon ECS que você deseja registrar na pilha.

EC2 Perfil da instância

O perfil de instância do Amazon Elastic Compute Cloud (Amazon EC2) do cluster. Esse perfil concede a permissão a aplicativos em execução nas instâncias de contêiner do cluster para acessar outros serviços da AWS, incluindo o Amazon ECS. Ao criar sua primeira camada do ECS Cluster, escolha Novo perfil com acesso ao ECS para fazer com que o AWS OpsWorks Stacks crie o perfil necessário, que é chamado de. `aws-opsworks-ec2-role-with-ecs` Em seguida, você pode usar esse perfil para todas as camadas subsequentes do ECS Cluster. Para obter mais informações sobre os perfis de instâncias, consulte Especificação de permissões para aplicativos executados em instâncias EC2 .

Você pode especificar outras configurações editando a configuração da layer, incluindo:

- Anexar um balanceador de carga do Elastic Load Balancing à camada.

  Essa abordagem pode ser adequada em alguns casos de uso, mas o Amazon ECS oferece opções mais sofisticadas. Para obter mais informações, consulte Balanceamento de carga do serviço.
- Especificar se deseja atribuir endereços IP públicos ou elásticos automaticamente às instâncias de contêiner.

  Se você desativar a atribuição automática para os dois tipos de endereços, a instância não entrará online, a menos que a sub-rede tenha um NAT configurado corretamente. Para obter mais informações, consulte Execução de uma Stack em uma VPC.

Gerenciamento do cluster do ECS

Depois de criar uma camada do ECS Cluster, você pode usar o AWS OpsWorks Stacks para gerenciar o cluster da seguinte forma:

Provisionamento e gerenciamento de instâncias de contêiner

Inicialmente, uma camada do ECS Cluster não contém nenhuma instância de contêiner, mesmo que o cluster original tivesse uma. Uma opção é gerenciar as instâncias da camada usando uma combinação apropriada das seguintes ações:

- Adicionar manualmente instâncias 24/7 à camada e excluí-las quando não forem mais necessárias.
- Adicionar ou excluir instâncias segundo uma programação, adicionando instâncias com base no tempo à camada.
- Adicionar ou excluir instâncias com base em CloudWatch alarmes ou métricas de host do AWS OpsWorks Stacks, adicionando instâncias com base em carga à camada.

> **Note**
>
> Se o Amazon ECS não for compatível com o sistema operacional padrão da pilha, você deverá especificar explicitamente um sistema operacional compatível (Amazon Linux 2, Amazon Linux 2018.03, Amazon Linux 2017.09, Amazon Linux 2017.03, Amazon Linux 2016.09, Amazon Linux 2016.03, Amazon Linux 2015.09, Amazon Linux 2015.03, Ubuntu 18.04 LTS, Ubuntu 16.04 LTS, Ubuntu 14.04 LTS ou personalizado) ao criar as instâncias de contêiner. Não use a AMI otimizada para ECS para criar instâncias em uma camada do ECS, pois essa AMI já inclui o agente do ECS. AWS OpsWorks O Stacks também tenta instalar o agente do ECS durante o processo de configuração da instância, e o conflito pode fazer com que a configuração falhe.

Para obter mais informações, consulteOtimização do número de servidores. AWS OpsWorks O Stacks atribui o grupo de segurança do OpsWorkscluster do AWS- -ECS a cada uma das instâncias. Terminada a inicialização de cada nova instância, o AWS OpsWorks Stacks converte a instância em uma instância de contêiner instalando o Docker e o agente do Amazon ECS, e, em seguida, registrando a instância no cluster.

Se você preferir usar instâncias de contêiner existentes, pode registrá-las na pilha e atribuí-las à camada do ECS Cluster. Observe que as instâncias devem executar um sistema operacional compatível, como o Amazon Linux 2015.03 ou posterior ou o Ubuntu 14.04 LTS ou posterior.

> **Note**
>
> Uma instância de contêiner não pode pertencer a ambas as camadas do ECS Cluster e a camada integrada. No entanto, uma instância de contêiner pode pertencer a uma camada do ECS Cluster e a uma ou mais camadas personalizadas.

Executar atualizações do sistema operacional e do pacote

Depois que a inicialização de uma nova instância é concluída, o AWS OpsWorks Stacks instala as atualizações mais recentes. Em seguida, você pode usar o AWS OpsWorks Stacks para manter as instâncias do contêiner atualizadas. Para obter mais informações, consulte Gerenciamento de atualizações de segurança.

Gerenciamento de permissões de usuário

AWS OpsWorks O Stacks fornece uma maneira simples de gerenciar permissões nas instâncias do contêiner, incluindo o gerenciamento das chaves SSH dos usuários. Para obter mais informações, consulte Gerenciamento de permissões de usuário e Gerenciamento do acesso por SSH.

Monitoração de métricas de desempenho

AWS OpsWorks O Stacks fornece várias maneiras de monitorar as métricas de desempenho da pilha, da camada ou das instâncias individuais. Para obter mais informações, consulte Monitoramento.

Você pode executar as outras tarefas de gerenciamento, como a criação de tarefas ou serviços, usando o Amazon ECS. Para obter mais informações, consulte o Guia do desenvolvedor do serviço Elastic Container da Amazon.

> **Note**
>
> Para acessar diretamente a página do cluster no console do Amazon ECS, escolha Instâncias e, em seguida, escolha ECS Cluster, que é exibido no canto superior direito da seção da camada do ECS Cluster.

Exclusão de uma camada do ECS Cluster de uma pilha

Quando você não precisar mais do cluster, exclua a camada do ECS Cluster e cancele o registro do cluster associado. A remoção de um cluster de uma pilha exige duas operações: o cancelamento do registro do cluster e, em seguida, a exclusão da camada associada. O console AWS OpsWorks Stacks combina essas etapas; a exclusão de camadas cancela automaticamente o registro do cluster especificado. Se você usa a API, a CLI ou o SDK do AWS OpsWorks Stacks, deve usar operações separadas para cancelar o registro do cluster e excluir a camada associada.

Para usar o console para excluir uma camada do ECS Cluster

1. Se você quiser controlar como as tarefas são desativadas, use o console, a API ou a CLI do Amazon ECS para reduzir a escala verticalmente e excluir os serviços do cluster. Para obter mais informações, consulte Liberação dos recursos do seu Amazon ECS.
2. Interrompa as instâncias da camada e, em seguida, exclua as instâncias. Quando você interrompe uma instância de contêiner, o AWS OpsWorks Stacks interrompe automaticamente qualquer tarefa em execução, cancela o registro da instância do cluster e encerra a instância.

   > **Note**
   >
   > Se você tiver registrado instâncias de contêiner existentes na pilha, pode cancelar a atribuição das instâncias na camada e, em seguida, cancelar seus registros, o que transfere o controle das instâncias de volta ao ECS.

3. Exclua a camada. AWS OpsWorks O Stacks cancela o registro do cluster associado, mas não exclui o mesmo. O cluster permanece no Amazon ECS.

Camadas AWS OpsWorks de pilhas personalizadas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Uma camada personalizada possui apenas um conjunto mínimo de receitas. Você pode então adicionar as funcionalidades apropriadas à camada, implementando as receitas personalizadas e atribuindo-as aos eventos de ciclo de vida da camada.

A camada personalizada possui as seguintes definições de configuração.

> **Note**
>
> AWS OpsWorks O Stacks instala automaticamente o Ruby nas instâncias da camada. Se você deseja executar o código Ruby na instância, mas não quer usar a versão padrão do Ruby, é possível usar um JSON personalizado ou um arquivo de atributos personalizado para especificar a versão desejada. Para obter mais informações, consulte Versões do Ruby.

O procedimento básico para a criação de uma camada personalizada é composto das seguintes etapas:

1. Implemente um livro de receitas que contém as receitas e os arquivos associados necessários para instalar e configurar pacotes, lidar com alterações de configuração, implementar aplicativos, e assim por diante.

   Dependendo das suas necessidades, você talvez precise de receitas responsáveis por desfazer implementações e de desligamento. Para obter mais informações, consulte Livros de receitas e receitas.
2. Crie uma camada personalizada.
3. Atribua suas receitas aos eventos de ciclo de vida apropriados.

Você pode então adicionar instâncias à camada, iniciá-las e implementar aplicativos nelas.

> **Important**
>
> Para implantar aplicativos nas instâncias de uma camada personalizada, você precisa implementar receitas para lidar com a operação de implantação e atribuí-las ao evento "Deploy" da camada.

Instalações de pacotes de sistema operacional por camada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Começando com o Chef 12, você deve usar receitas personalizadas para instalar pacotes em camadas que estão executando diferentes sistemas operacionais. Essa abordagem fornece a você o máximo de flexibilidade e controle sobre instalações de pacotes.

Por exemplo, suponha que você queira instalar o Apache em camadas que estão executando RedHat as versões Ubuntu e Amazon do sistema operacional Linux. O pacote Apache para RedHat o Amazon Linux é chamado`httpd`, mas no Ubuntu, ele é chamado`apache2`.

Para lidar com essa diferença na nomenclatura de pacotes, você pode usar uma sintaxe semelhante à receita de exemplo a seguir. A receita instala o pacote Apache apropriado para cada sistema operacional. Esse exemplo é baseado na documentação do Chef.

```
package "Install Apache" do
   case node[:platform]
      when "redhat", "amazon"
         package_name "httpd"
      when "ubuntu"
         package_name "apache2"
   end
end
```

Para informações detalhadas sobre como usar o recurso `package` para gerenciar pacotes, acesse a página package na documentação do Chef.

Como alternativa, você pode usar o método auxiliar `value_for_platform` a partir do Chef Recipe DSL (linguagem específica do domínio), que faz a mesma coisa de maneira mais sucinta:

```
package "Install Apache" do
   package_name value_for_platform(
      ["redhat", "amazon"] => { "default" => "httpd" },
      ["ubuntu"] => { "default" => "apache2" }
   )
end
```

Para informações sobre como usar o método auxiliar `value_for_platform`, consulte Sobre o Recipe DSL.

Instâncias

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Uma instância representa um recurso de computação, como uma EC2 instância da Amazon, que lida com o trabalho de servir aplicativos, equilibrar o tráfego e assim por diante. O sistema operacional de uma instância pode ter qualquer uma das diversas distribuições do Linux ou o Windows Server 2012 R2.

Você pode adicionar instâncias a uma pilha de uma destas maneiras:

- Use AWS OpsWorks pilhas para adicionar instâncias a uma pilha. As instâncias que você adiciona representam as EC2 instâncias da Amazon.
- Para pilhas baseadas em Linux, você pode registrar instâncias que foram criadas em outro lugar, incluindo instâncias que você criou com a Amazon EC2 e instâncias locais que estão sendo executadas em seu próprio hardware.

  Em seguida, você pode usar o AWS OpsWorks Stacks para gerenciar essas instâncias da mesma forma que as instâncias criadas com AWS OpsWorks o Stacks.

Esta seção descreve como usar o AWS OpsWorks Stacks para criar e gerenciar instâncias.

Tópicos

- Usando AWS OpsWorks instâncias Stacks
- Uso de recursos de cálculo criados fora do AWS OpsWorks Stacks
- Edição da configuração da instância
- Excluindo instâncias do AWS OpsWorks Stacks
- Uso do SSH para fazer login em uma instância Linux
- Uso do RDP para fazer login em uma instância Windows

Usando AWS OpsWorks instâncias Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode usar AWS OpsWorks pilhas para criar instâncias e adicioná-las à pilha.

Tópicos

- AWS OpsWorks Sistemas operacionais de pilha
- Adicionar uma instância a uma camada
- Usando o Custom AMIs
- Descreve como iniciar, parar e reiniciar instâncias 24/7
- Gerenciamento de carga com instâncias baseadas em tempo e carga

AWS OpsWorks Sistemas operacionais de pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks é compatível com as versões de 64 bits de vários sistemas operacionais integrados, incluindo distribuições Amazon e Ubuntu Linux e Microsoft Windows Server. Algumas observações gerais:

- As instâncias da pilha podem executar tanto Linux quanto Windows.

Uma pilha pode ter diferentes versões do Linux ou distribuições em diferentes instâncias, mas não é possível combinar as instâncias Linux e Windows.

- Você pode usar imagens personalizadas AMIs (Amazon Machine Images), mas elas devem ser baseadas em uma das AWS OpsWorks pilhas suportadas AMIs , descritas nos tópicos desta seção. Embora seja possível criar ou registrar instâncias com outros sistemas operacionais (como o CentOS 6). x) que foram criados de forma personalizada ou gerada pela comunidade AMIs, não são oficialmente suportados.
  - Sistemas operacionais Linux
  - Microsoft Windows Server
- É possível iniciar e parar instâncias manualmente ou deixar que o AWS OpsWorks Stacks dimensione automaticamente o número de instâncias.

  É possível usar esse tipo de escalabilidade automática com base em tempo com qualquer pilha. As pilhas do Linux também podem usar escalabilidade com base na carga.
- Além de usar o AWS OpsWorks Stacks para criar EC2 instâncias da Amazon, você também pode registrar instâncias com uma pilha Linux que foram criadas fora do AWS OpsWorks Stacks.

  Isso inclui EC2 instâncias da Amazon e instâncias executadas em seu próprio hardware. No entanto, elas devem ser executadas uma das distribuições do Linux compatíveis. Você não pode registrar a Amazon EC2 ou instâncias locais do Windows.

Você pode executar a `DescribeOperatingSystems`API AWS OpsWorks Stacks para retornar uma lista dos sistemas operacionais compatíveis e suas versões compatíveis do Chef. Veja a seguir um comando de exemplo que usa a AWS CLI.

```
aws opsworks describe-operating-systems
```

O seguinte é um exemplo de resposta.

```
{
    "OperatingSystems": [
        {
            "Name": "Amazon Linux",
            "Id": "Amazon Linux",
            "Type": "Linux",
            "ConfigurationManagers": [
                {
```

```
                "Name": "Chef",
                "Version": "11.10"
            },
            {
                "Name": "Chef",
                "Version": "11.4"
            },
            {
                "Name": "Chef",
                "Version": "0.9"
            }
        ],
        "ReportedName": "amazon",
        "ReportedVersion": "2014.03",
        "Supported": false
    },
    {
        "Name": "Amazon Linux 2",
        "Id": "Amazon Linux 2",
        "Type": "Linux",
        "ConfigurationManagers": [
            {
                "Name": "Chef",
                "Version": "12"
            }
        ],
        "ReportedName": "amazon",
        "ReportedVersion": "2"
    },
    {
        "Name": "Amazon Linux 2014.09",
        "Id": "Amazon Linux 2014.09",
        "Type": "Linux",
        "ConfigurationManagers": [
            {
                "Name": "Chef",
                "Version": "11.10"
            },
            {
                "Name": "Chef",
                "Version": "11.4"
            },
            {
                "Name": "Chef",
```

```
                "Version": "0.9"
            }
        ],
        "ReportedName": "amazon",
        "ReportedVersion": "2014.09",
        "Supported": false
    },
    {
        "Name": "Amazon Linux 2015.03",
        "Id": "Amazon Linux 2015.03",
        "Type": "Linux",
        "ConfigurationManagers": [
            {
                "Name": "Chef",
                "Version": "12"
            },
            {
                "Name": "Chef",
                "Version": "11.10"
            },
            {
                "Name": "Chef",
                "Version": "11.4"
            },
            {
                "Name": "Chef",
                "Version": "0.9"
            }
        ],
        "ReportedName": "amazon",
        "ReportedVersion": "2015.03",
        "Supported": false
    },
    {
        "Name": "Amazon Linux 2015.09",
        "Id": "Amazon Linux 2015.09",
        "Type": "Linux",
        "ConfigurationManagers": [
            {
                "Name": "Chef",
                "Version": "12"
            },
            {
                "Name": "Chef",
```

```
                "Version": "11.10"
            },
            {
                "Name": "Chef",
                "Version": "11.4"
            },
            {
                "Name": "Chef",
                "Version": "0.9"
            }
        ],
        "ReportedName": "amazon",
        "ReportedVersion": "2015.09",
        "Supported": false
    },
    {
        "Name": "Amazon Linux 2016.03",
        "Id": "Amazon Linux 2016.03",
        "Type": "Linux",
        "ConfigurationManagers": [
            {
                "Name": "Chef",
                "Version": "12"
            },
            {
                "Name": "Chef",
                "Version": "11.10"
            },
            {
                "Name": "Chef",
                "Version": "11.4"
            },
            {
                "Name": "Chef",
                "Version": "0.9"
            }
        ],
        "ReportedName": "amazon",
        "ReportedVersion": "2016.03"
    },
    {
        "Name": "Amazon Linux 2016.09",
        "Id": "Amazon Linux 2016.09",
        "Type": "Linux",
```

```
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                },
                {
                    "Name": "Chef",
                    "Version": "11.10"
                },
                {
                    "Name": "Chef",
                    "Version": "11.4"
                },
                {
                    "Name": "Chef",
                    "Version": "0.9"
                }
            ],
            "ReportedName": "amazon",
            "ReportedVersion": "2016.09"
        },
        {
            "Name": "Amazon Linux 2017.03",
            "Id": "Amazon Linux 2017.03",
            "Type": "Linux",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                },
                {
                    "Name": "Chef",
                    "Version": "11.10"
                },
                {
                    "Name": "Chef",
                    "Version": "11.4"
                },
                {
                    "Name": "Chef",
                    "Version": "0.9"
                }
            ],
            "ReportedName": "amazon",
```

```
            "ReportedVersion": "2017.03"
        },
        {
            "Name": "Amazon Linux 2017.09",
            "Id": "Amazon Linux 2017.09",
            "Type": "Linux",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                },
                {
                    "Name": "Chef",
                    "Version": "11.10"
                },
                {
                    "Name": "Chef",
                    "Version": "11.4"
                },
                {
                    "Name": "Chef",
                    "Version": "0.9"
                }
            ],
            "ReportedName": "amazon",
            "ReportedVersion": "2017.09"
        },
        {
            "Name": "Amazon Linux 2018.03",
            "Id": "Amazon Linux 2018.03",
            "Type": "Linux",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                },
                {
                    "Name": "Chef",
                    "Version": "11.10"
                }
            ],
            "ReportedName": "amazon",
            "ReportedVersion": "2018.03"
        },
```

```
        {
            "Name": "CentOS Linux 7",
            "Id": "CentOS Linux 7",
            "Type": "Linux",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                }
            ],
            "ReportedName": "CentOS Linux",
            "ReportedVersion": "7"
        },
        {
            "Name": "Microsoft Windows Server 2012 R2 Base",
            "Id": "Microsoft Windows Server 2012 R2 Base",
            "Type": "Windows",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12.2"
                }
            ],
            "ReportedName": "microsoft windows server",
            "ReportedVersion": "2012 r2 standard",
            "Supported": false
        },
        {
            "Name": "Microsoft Windows Server 2012 R2 with SQL Server Express",
            "Id": "Microsoft Windows Server 2012 R2 with SQL Server Express",
            "Type": "Windows",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12.2"
                }
            ],
            "ReportedName": "microsoft windows server",
            "ReportedVersion": "2012 r2 standard",
            "Supported": false
        },
        {
            "Name": "Microsoft Windows Server 2012 R2 with SQL Server Standard",
            "Id": "Microsoft Windows Server 2012 R2 with SQL Server Standard",
```

```
            "Type": "Windows",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12.2"
                }
            ],
            "ReportedName": "microsoft windows server",
            "ReportedVersion": "2012 r2 standard",
            "Supported": false
        },
        {
            "Name": "Microsoft Windows Server 2012 R2 with SQL Server Web",
            "Id": "Microsoft Windows Server 2012 R2 with SQL Server Web",
            "Type": "Windows",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12.2"
                }
            ],
            "ReportedName": "microsoft windows server",
            "ReportedVersion": "2012 r2 standard",
            "Supported": false
        },
        {
            "Name": "Microsoft Windows Server 2019 Base",
            "Id": "Microsoft Windows Server 2019 Base",
            "Type": "Windows",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12.2"
                }
            ],
            "ReportedName": "microsoft windows server",
            "ReportedVersion": "2019 datacenter"
        },
        {
            "Name": "Microsoft Windows Server 2019 with SQL Server Express",
            "Id": "Microsoft Windows Server 2019 with SQL Server Express",
            "Type": "Windows",
            "ConfigurationManagers": [
                {
```

```
                "Name": "Chef",
                "Version": "12.2"
            }
        ],
        "ReportedName": "microsoft windows server",
        "ReportedVersion": "2019 datacenter"
    },
    {
        "Name": "Microsoft Windows Server 2019 with SQL Server Standard",
        "Id": "Microsoft Windows Server 2019 with SQL Server Standard",
        "Type": "Windows",
        "ConfigurationManagers": [
            {
                "Name": "Chef",
                "Version": "12.2"
            }
        ],
        "ReportedName": "microsoft windows server",
        "ReportedVersion": "2019 datacenter"
    },
    {
        "Name": "Microsoft Windows Server 2019 with SQL Server Web",
        "Id": "Microsoft Windows Server 2019 with SQL Server Web",
        "Type": "Windows",
        "ConfigurationManagers": [
            {
                "Name": "Chef",
                "Version": "12.2"
            }
        ],
        "ReportedName": "microsoft windows server",
        "ReportedVersion": "2019 datacenter"
    },
    {
        "Name": "Microsoft Windows Server 2022 Base",
        "Id": "Microsoft Windows Server 2022 Base",
        "Type": "Windows",
        "ConfigurationManagers": [
            {
                "Name": "Chef",
                "Version": "12.2"
            }
        ],
        "ReportedName": "microsoft windows server",
```

```
            "ReportedVersion": "2022 datacenter"
        },
        {
            "Name": "Microsoft Windows Server 2022 with SQL Server Express",
            "Id": "Microsoft Windows Server 2022 with SQL Server Express",
            "Type": "Windows",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12.2"
                }
            ],
            "ReportedName": "microsoft windows server",
            "ReportedVersion": "2022 datacenter"
        },
        {
            "Name": "Microsoft Windows Server 2022 with SQL Server Standard",
            "Id": "Microsoft Windows Server 2022 with SQL Server Standard",
            "Type": "Windows",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12.2"
                }
            ],
            "ReportedName": "microsoft windows server",
            "ReportedVersion": "2022 datacenter"
        },
        {
            "Name": "Microsoft Windows Server 2022 with SQL Server Web",
            "Id": "Microsoft Windows Server 2022 with SQL Server Web",
            "Type": "Windows",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12.2"
                }
            ],
            "ReportedName": "microsoft windows server",
            "ReportedVersion": "2022 datacenter"
        },
        {
            "Name": "Red Hat Enterprise Linux 7",
            "Id": "Red Hat Enterprise Linux 7",
```

```
            "Type": "Linux",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                },
                {
                    "Name": "Chef",
                    "Version": "11.10"
                }
            ],
            "ReportedName": "Red Hat Enterprise Linux",
            "ReportedVersion": "7"
        },
        {
            "Name": "Ubuntu 12.04 LTS",
            "Id": "Ubuntu 12.04 LTS",
            "Type": "Linux",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                },
                {
                    "Name": "Chef",
                    "Version": "11.10"
                },
                {
                    "Name": "Chef",
                    "Version": "11.4"
                },
                {
                    "Name": "Chef",
                    "Version": "0.9"
                }
            ],
            "ReportedName": "ubuntu",
            "ReportedVersion": "12.04",
            "Supported": false
        },
        {
            "Name": "Ubuntu 14.04 LTS",
            "Id": "Ubuntu 14.04 LTS",
            "Type": "Linux",
```

```
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                },
                {
                    "Name": "Chef",
                    "Version": "11.10"
                }
            ],
            "ReportedName": "ubuntu",
            "ReportedVersion": "14.04"
        },
        {
            "Name": "Ubuntu 16.04 LTS",
            "Id": "Ubuntu 16.04 LTS",
            "Type": "Linux",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                }
            ],
            "ReportedName": "ubuntu",
            "ReportedVersion": "16.04"
        },
        {
            "Name": "Ubuntu 18.04 LTS",
            "Id": "Ubuntu 18.04 LTS",
            "Type": "Linux",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                }
            ],
            "ReportedName": "ubuntu",
            "ReportedVersion": "18.04"
        },
        {
            "Name": "Ubuntu 20.04 LTS",
            "Id": "Ubuntu 20.04 LTS",
            "Type": "Linux",
            "ConfigurationManagers": [
```

```
                {
                    "Name": "Chef",
                    "Version": "12"
                }
            ],
            "ReportedName": "ubuntu",
            "ReportedVersion": "20.04"
        },
        {
            "Name": "Custom",
            "Id": "Custom",
            "Type": "Linux",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12"
                },
                {
                    "Name": "Chef",
                    "Version": "11.10"
                },
                {
                    "Name": "Chef",
                    "Version": "11.4"
                },
                {
                    "Name": "Chef",
                    "Version": "0.9"
                }
            ]
        },
        {
            "Name": "CustomWindows",
            "Id": "CustomWindows",
            "Type": "Windows",
            "ConfigurationManagers": [
                {
                    "Name": "Chef",
                    "Version": "12.2"
                }
            ]
        }
    ]
```

```
}
```

Tópicos

- Sistemas operacionais Linux
- Microsoft Windows Server

Sistemas operacionais Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks é compatível com as versões de 64 bits dos seguintes sistemas operacionais Linux.

- Amazon Linux e Amazon Linux 2 (consulte o console do AWS OpsWorks Stacks para saber quais são as versões atualmente compatíveis)
- Ubuntu 20.04 LTS
- CentOS 7
- Red Hat Enterprise Linux 7

Você também pode usar o personalizado AMIs com base nesses sistemas operacionais.

Algumas notas gerais sobre as instâncias Linux:

Versões do pacote compatível

As versões compatíveis e os níveis de patch para pacotes, como Ruby, dependem do sistema operacional e da versão, conforme descrito nas seções a seguir.

Atualizações

Por padrão, o AWS OpsWorks Stacks garante que as instâncias Linux tenham os patches de segurança mais recentes chamando automaticamente `yum update` ou `apt-get update` após a inicialização de uma instância. Para desativar as atualizações automáticas, use as UpdateLayerações CreateInstanceUpdateInstance, CreateLayer, ou — ou os métodos equivalentes do AWS SDK ou comandos da AWS CLI — para definir o parâmetro como. `InstallUpdatesOnBoot` `false`

Para evitar interrupções no serviço, o AWS OpsWorks Stacks não instala automaticamente as atualizações depois que uma instância está on-line. É possível atualizar manualmente um sistema operacional de uma instância online a qualquer momento, executando o comando da pilha para fazer upgrade do sistema operacional. Para obter mais informações sobre como gerenciar as atualizações de segurança, consulte Gerenciamento de atualizações de segurança.

Para ter mais controle sobre como o AWS OpsWorks Stacks atualiza suas instâncias, crie uma AMI personalizada com base em um dos sistemas operacionais compatíveis. Por exemplo, com a personalização, AMIs você pode especificar quais versões do pacote estão instaladas em uma instância. Cada distribuição do Linux tem diferentes cronogramas de suporte e políticas de mesclagem de pacotes, por isso é preciso considerar qual é a abordagem que melhor atende às suas necessidades. Para obter mais informações, consulte Usando o Custom AMIs.

Arquivo Hosts

Cada instância on-line tem um `/etc/hosts` arquivo que mapeia endereços IP para nomes de host. AWS OpsWorks As pilhas incluem os endereços públicos e privados de todas as instâncias on-line da pilha no arquivo de `hosts` cada instância. Por exemplo, suponha que exista uma pilha com duas instâncias Node.js App Server, nodejs-app1 e nodejs-app2, e uma instância MySQL db-master1. O arquivo `hosts` da instância nodejs-app1 será semelhante ao exemplo a seguir, e o das outras instâncias terão arquivos `hosts` semelhantes.

```
...
# OpsWorks Layer State
192.0.2.0 nodejs-app1.localdomain nodejs-app1
10.145.160.232 db-master1
198.51.100.0 db-master1-ext
10.243.77.78 nodejs-app2
203.0.113.0 nodejs-app2-ext
10.84.66.6 nodejs-app1
```

```
192.0.2.0 nodejs-app1-ext
```

AWS OpsWorks Suporte ao Stacks Agent Proxy

O agente AWS OpsWorks Stacks para pilhas do Chef 11.10 e posteriores inclui suporte básico para servidores proxy, que normalmente são usados com sistemas isolados. VPCs Para ativar o suporte do servidor de proxy, uma instância deve ter um arquivo `/etc/environment` que fornece as configurações adequadas para tráfego HTTP e HTTPS. O arquivo deve ser semelhante ao seguinte, onde o texto destacado é substituído pelo URL do servidor de proxy e a porta:

```
http_proxy="http://myproxy.example.com:8080/"
https_proxy="http://myproxy.example.com:8080/"
no_proxy="169.254.169.254"
```

Para ativar o suporte de proxy, recomendamos a criação de uma AMI personalizada que inclua um arquivo `/etc/environment` adequado e usar essa AMI para criar as instâncias.

> **Note**
>
> Não recomendamos o uso de uma receita personalizada para criar um `/etc/environment` arquivo em suas instâncias. AWS OpsWorks O Stacks precisa dos dados do servidor proxy logo no início do processo de configuração, antes que qualquer receita personalizada seja executada.

Tópicos

- Amazon Linux
- Ubuntu LTS
- CentOS
- Red Hat Enterprise Linux

Amazon Linux

AWS OpsWorks O Stacks é compatível com as versões de 64 bits do Amazon Linux e do Amazon Linux 2. Além das atualizações e patches regulares, o Amazon Linux lança uma nova versão a cada

seis meses, aproximadamente, o que pode envolver alterações significativas. Ao criar uma pilha ou uma nova instância, é preciso especificar qual versão Amazon Linux deve ser usada. Quando a AWS libera uma nova versão, as instâncias continuam a executar a versão especificada até que ela seja alterada explicitamente. Depois que uma versão da Amazon Linux é lançada, há um período de migração de quatro semanas, em que a AWS continua a fornecer atualizações regulares para a versão antiga. Após o período de migração, suas instâncias podem continuar a executar a versão antiga, mas a AWS não fornecerá atualizações adicionais. Para obter mais informações, consulte AMI do Amazon Linux FAQs.

Quando uma nova versão do Amazon Linux for lançada, atualize para a nova versão dentro do período de migração para que as instâncias continuem a receber atualizações de segurança. Antes de atualizar suas instâncias da pilha de produção, inicie uma nova instância e verifique se o seu aplicativo é executado corretamente na nova versão. Em seguida, atualize as instâncias da pilha de produção.

> **Note**
>
> Por padrão, as versões personalizadas AMIs baseadas no Amazon Linux são automaticamente atualizadas para a nova versão quando ela é lançada. Recomenda-se bloquear a AMI personalizada em uma versão específica do Amazon Linux para que seja possível adiar a atualização até que a nova versão tenha sido testada. Para obter mais informações, consulte Como bloquear a AMI em uma versão específica?.
> Se você usar um AWS CloudFormation modelo para criar pilhas com instâncias executando o Amazon Linux, os modelos devem especificar explicitamente uma versão do Amazon Linux. Em particular, se o modelo especifica `Amazon Linux`, as instâncias continuarão executando a versão 2016.09. Para ter mais informações, consulte AWS::OpsWorks::Stack e AWS::OpsWorks::Instance.

Para atualizar a versão do Amazon Linux de uma instância, siga as seguintes instruções:

- Para instâncias online, execute o comando de pilha Upgrade Operating System.

  Quando uma nova versão do Amazon Linux estiver disponível, as páginas Instances e Stack exibirão um aviso prévio com um link de acesso para a página Run Command. Em seguida, você poderá executar Upgrade Operating System para atualizar sua instância.
- Para instâncias off-line com base no Amazon Elastic Block Store (com base no EBS), inicie as instâncias e execute Atualizar sistema operacional, como descrito na declaração anterior.

- Para instâncias off-line com base em armazenamento de instâncias, incluindo instâncias com base em tempo e carga, edite a configuração Operating system da instância para especificar a nova versão.

  AWS OpsWorks O Stacks atualiza automaticamente as instâncias para a nova versão quando elas são reiniciadas.

Amazon Linux: versões Node.js compatíveis

Versão Amazon Linux	Versões do Node.js
2	(Not applicable to operating systems that are available for Chef 12 and higher stacks only)
2018.03	0.12.18
2017.09	0.12.18
2017.03	0.12.18
2016.09	0.12.18 0.12.17 0.12.16 0.12.15
2016.03	0.12.18 0.12.17 0.12.16 0.12.15 0.12.14 0.12.13 0.12.12 0.12.10

Amazon Linux: versões do Chef compatíveis

Versão do Chef	Versões compatíveis do Amazon Linux
12	Amazon Linux 2 Amazon Linux 2018.03 Amazon Linux 2017.09 Amazon Linux 2017.03 Amazon Linux 2016.09 Amazon Linux 2016.03
11.10	Amazon Linux 2018.03 Amazon Linux 2017.09 Amazon Linux 2017.03 Amazon Linux 2016.09 Amazon Linux 2016.03
11.4 (deprecated)	Amazon Linux 2016.09 Amazon Linux 2016.03

Important

Antes de atualizar instâncias t1.micro, certifique-se de que elas possuem um arquivo de troca temporário, `/var/swapfile`. As instâncias t1.micro nas pilhas do Chef 0.9 não têm um arquivo de troca. Para as pilhas do Chef 11.4 e do Chef 11.10, as versões recentes do agente da instância criam automaticamente um arquivo de troca para instâncias t1.micro. No entanto, esta alteração foi apresentada ao longo de um período de várias semanas, portanto, verifique se o arquivo `/var/swapfile` está presente em instâncias criadas antes de 24 de março de 2014, aproximadamente.

Para instâncias t1.micro que não têm um arquivo de troca, siga as instruções a seguir para criar um:

- Para o Chef 11.10 e pilhas mais recentes, crie novas instâncias t1.micro, que automaticamente tem um arquivo de troca.
- Para pilhas Chef 0.9 pilhas , execute os seguintes comandos em cada instância como usuário raiz.

```
dd if=/dev/zero of=/var/swapfile bs=1M count=256
 mkswap /var/swapfile
 chown root:root /var/swapfile
 chmod 0600 /var/swapfile
 swapon /var/swapfile
```

Você também pode usar esses comandos no Chef 11.10 e pilhas posteriores, se você não quiser criar novas instâncias.

Ubuntu LTS

O Ubuntu lança uma nova versão do Ubuntu LTS a cada dois anos, aproximadamente, e é compatível com cada versão por aproximadamente cinco anos. Ubuntu fornece patches de segurança e atualizações durante o suporte do sistema operacional. Para obter mais informações, consulte LTS - Ubuntu Wiki.

- Você não pode atualizar uma instância do Ubuntu existente para uma versão mais recente do Ubuntu.

  É preciso criar uma nova instância do Ubuntu e excluir as instâncias antigas.
- O Ubuntu 20.04 LTS é compatível apenas com o Chef 12 e pilhas mais altas.

CentOS

AWS OpsWorks O Stacks é compatível com a versão de 64 bits do CentOS 7. A versão inicial compatível é CentOS 7, e o CentOS lança uma nova versão a cada dois anos, aproximadamente.

Quando você inicia uma nova instância em uma pilha do CentOS, o AWS OpsWorks Stacks instala automaticamente a versão mais atual do CentOS. Como o AWS OpsWorks Stacks não atualiza automaticamente o sistema operacional nas instâncias existentes quando uma nova versão secundária do CentOS é lançada, uma instância recém-criada pode receber uma versão mais recente do que as instâncias existentes da pilha. A fim de manter a consistência das versões na sua pilha, é possível atualizar as instâncias existentes para a versão atual do CentOS da seguinte maneira:

- Para instâncias online, execute o comando de pilha Upgrade Operating System, que executa `yum update` nas instâncias especificadas, para atualizá-las para a versão mais recente.

Quando uma versão secundária do CentOS 7 estiver disponível, as páginas Instances e Stack exibirão um aviso prévio com um link de acesso para a página Run Command. Em seguida, você poderá executar Upgrade Operating System para atualizar suas instâncias.

- Para instâncias off-line com base no Amazon EBS, inicie as instâncias e execute Atualizar sistema operacional, como descrito no item de lista anterior.
- Para instâncias offline com armazenamento de instâncias, o AWS OpsWorks Stacks instala automaticamente a nova versão quando as instâncias são reiniciadas.

CentOS: versões do Chef compatíveis

Versão do Chef	Versão do CentOS compatível
`12`	`CentOS 7`
`11.10`	`(None supported)`
`11.4 (deprecated)`	`(None supported)`

> **Note**
> AWS OpsWorks O Stacks oferece suporte ao Apache 2.4 para instâncias do CentOS.

Red Hat Enterprise Linux

AWS OpsWorks O Stacks suporta a versão de 64 bits do Red Hat Enterprise Linux 7 (RHEL 7). A versão inicial compatível é RHEL 7.1 e o Red Hat lança uma versão menor nova a cada 9 meses, aproximadamente. Versões secundárias são compatíveis com o RHEL 7.0. Para obter mais informações, consulte Ciclo de vida e políticas de atualização.

Quando você inicia uma nova instância, o AWS OpsWorks Stacks instala automaticamente a versão atual do RHEL 7. Como o AWS OpsWorks Stacks não atualiza automaticamente o sistema operacional nas instâncias existentes quando uma nova versão secundária do RHEL 7 é lançada, uma instância recém-criada pode receber uma versão mais recente do que as instâncias existentes

da pilha. A fim de manter a consistência das versões na sua pilha, é possível atualizar as instâncias existentes para a versão atual do RHEL 7 da seguinte maneira:

- Para instâncias online, execute o comando de pilha Upgrade Operating System, que executa `yum update` nas instâncias especificadas, para atualizá-las para a versão mais recente.

  Quando uma nova versão do RHEL 7 estiver disponível, as páginas Instances e Stack exibirão um aviso prévio com um link de acesso para a página Run Command. Em seguida, você poderá executar Upgrade Operating System para atualizar suas instâncias.
- Para instâncias off-line com base no Amazon EBS, inicie as instâncias e execute Atualizar sistema operacional, como descrito no item de lista anterior.
- Para instâncias offline com armazenamento de instâncias, o AWS OpsWorks Stacks instala automaticamente a nova versão quando as instâncias são reiniciadas.

Red Hat Enterprise Linux: versões Node.js compatíveis

Versão da RHEL	Versões do Node.js
7	(Node.js versions only apply to Chef 11.10 stacks) 0.8.19 0.8.26 0.10.11 0.10.21 0.10.24 0.10.25 0.10.27 0.10.29 0.10.40 0.12.10 0.12.12 0.12.13 0.12.15

Red Hat Enterprise Linux: versões Chef compatíveis

Versão do Chef	Versão RHEL compatível
`12`	`Red Hat Enterprise Linux 7`
`11.10`	`Red Hat Enterprise Linux 7`
`11.4 (deprecated)`	`(None supported)`

Todas as versões do Node.js anteriores à 0.10.40 estão obsoletas. 0.12.7 e 0.12.9 também estão obsoletas.

> **Note**
>
> AWS OpsWorks O Stacks oferece suporte ao Apache 2.4 para instâncias do RHEL 7.

Microsoft Windows Server

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As notas a seguir descrevem o suporte do AWS OpsWorks Stacks para instâncias do Windows. As instâncias do Windows estão disponíveis apenas para pilhas do Chef 12.2. A versão exata do Chef em uma pilha do Windows é 12.22.

Atualmente, o agente AWS OpsWorks Stacks não pode ser instalado e o AWS OpsWorks Stacks não pode gerenciar instâncias baseadas em Windows que usam um idioma de interface de usuário do sistema diferente de inglês - Estados Unidos da América (en-US).

Versões

AWS OpsWorks O Stacks é compatível com as seguintes versões do Windows de 64 bits:

- Microsoft Windows Server 2022 Base
- Microsoft Windows Server 2022 com SQL Server Express
- Microsoft Windows Server 2022 com SQL Server Standard
- Microsoft Windows Server 2022 com SQL Server Web
- Microsoft Windows Server 2019 Base
- Microsoft Windows Server 2019 com SQL Server Express
- Microsoft Windows Server 2019 com SQL Server Standard
- Microsoft Windows Server 2019 com SQL Server Web

Criação de instâncias

Você cria instâncias do Windows com o console, a API ou a CLI do AWS OpsWorks Stacks. As instâncias do Windows são baseadas em Amazon EBS, mas não é possível montar volumes Amazon EBS extra.

As pilhas do Windows podem usar instâncias 24/7, que você inicia e para manualmente. Elas também podem usar a escalabilidade automática baseada em tempo, que inicia e interrompe instâncias automaticamente com base em um cronograma especificado pelo usuário. As pilhas baseadas no Windows não podem usar a escalabilidade automática com base em carga.

Você não pode registrar instâncias do Windows que foram criadas fora do AWS OpsWorks Stacks com uma pilha.

Atualizações

A AWS atualiza AMIs o Windows para cada conjunto de patches, portanto, quando você cria uma instância, ela terá as atualizações mais recentes. No entanto, o AWS OpsWorks Stacks não fornece uma forma de aplicar atualizações às instâncias online do Windows. A maneira mais simples de garantir que o Windows esteja atualizado é substituir as instâncias regularmente, para que elas executem sempre a AMI mais recente.

Camadas

A fim de tratar tarefas como a instalação e configuração de softwares ou a implantação de aplicativos, será necessário implantar uma ou mais camadas personalizadas com receitas personalizadas.

Chef

As instâncias do Windows usam Chef 12.22 e executam o cliente do Chef no modo local que executa um servidor do Chef na memória local chamado chef-zero. A presença desse servidor permite que as receitas personalizadas utilizem a pesquisa e os recipientes de dados do Chef.

Login remoto

AWS OpsWorks O Stacks fornece aos usuários autorizados do IAM uma senha que eles podem usar para fazer login nas instâncias do Windows. Essa senha expira após um tempo específico. Administradores podem usar uma chave SSH para recuperar a senha de administrador de uma instância, o que fornece acesso ao RDP ilimitado. Para obter mais informações, consulte Login com RDP.

SDK da AWS

AWS OpsWorks O Stacks o instala automaticamente AWS SDK para .NETem cada instância. Esse pacote inclui as bibliotecas do AWS .NET e o AWS Tools for Windows, incluindo o AWS Tools for PowerShell. Para usar o SDK do Ruby, é possível solicitar que uma receita personalizada instale o gem adequado.

Monitoramento e métricas

As instâncias do Windows oferecem suporte às métricas padrão da Amazon CloudWatch (CloudWatch), que você pode visualizar no CloudWatch console.

Ruby

O cliente Chef 12.22 que o AWS OpsWorks Stacks instala nas instâncias do Windows vem com o Ruby 2.3.6. No entanto, o AWS OpsWorks Stacks não adiciona o diretório do executável à variável de ambiente PATH. Para que seus aplicativos usem essa versão do Ruby, é possível encontrá-la em `C:\opscode\chef\embedded\bin\`.

AWS OpsWorks CLI do Stacks Agent

O agente AWS OpsWorks Stacks em instâncias do Windows não expõe uma interface de linha de comando.

Suporte de proxy

Siga as seguintes instruções para configurar o suporte de proxy para instâncias do Windows:

1. Modifique `machine.config` para adicionar o seguinte, que adiciona suporte a proxy aos aplicativos Windows PowerShell (bootstrap inicial) e .NET (agente AWS OpsWorks Stacks):

```
<system.net>
  <defaultProxy>
    <proxy autoDetect="false" bypassonlocal="true"
 proxyaddress="http://10.100.1.91:3128"  usesystemdefault="false" />
    <bypasslist>
      <add address="localhost" />
      <add address="169.254.169.254" />
    </bypasslist>
  </defaultProxy>
</system.net>
```

2. Execute os comando a seguir para definir as variáveis de ambiente para uso posterior pelo Chef e Git:

```
setx /m no_proxy "localhost,169.254.169.254"
setx /m http_proxy "http://10.100.1.91:3128"
setx /m https_proxy "http://10.100.1.91:3128"
```

> **Note**
>
> Para ter mais controle sobre como o AWS OpsWorks Stacks atualiza suas instâncias, crie uma AMI personalizada com base no Microsoft Windows Server 2022 Base. Por exemplo, com a personalização, AMIs você pode especificar qual software está instalado em uma instância, como o Servidor Web (IIS). Para obter mais informações, consulte Usando o Custom AMIs.

Adicionar uma instância a uma camada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Depois de criar uma camada, normalmente você adiciona pelo menos uma instância. Você poderá adicionar mais instâncias posteriormente se o conjunto atual não suportar a carga. Também é possível usar instâncias baseadas em carga ou tempo para escalonar automaticamente o número de instâncias.

Você pode adicionar instâncias novas ou existentes a uma camada:

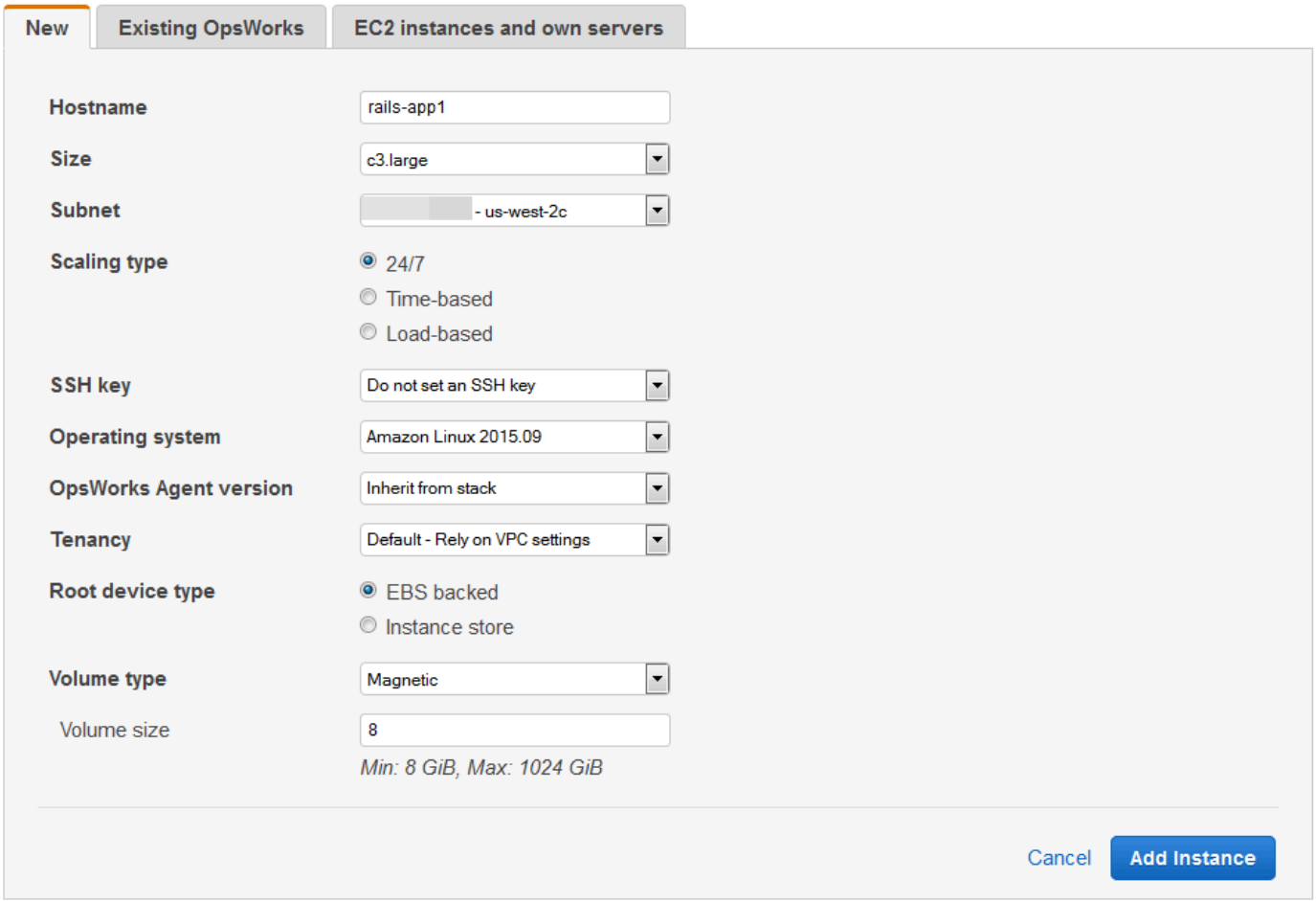
- Novo — OpsWorks cria uma nova instância, configurada de acordo com suas especificações, e a torna membro da camada.
- Existente: você pode adicionar uma instância existente em qualquer camada compatível, mas ela deve estar no estado offline (interrompido).

Se uma instância pertencer a várias camadas, o AWS OpsWorks Stacks executará as receitas referentes a cada camada da instância quando ocorrer um evento de ciclo de vida ou quando você executar um comando de pilha ou de implantação.

Para transformar uma instância em membro de várias camadas, edite a configuração correspondente. Para obter mais informações, consulte Edição da configuração da instância.

Para adicionar uma nova instância a uma camada

1. Na página Instances selecione +Instance para a camada apropriada e, se necessário, selecione a guia New. Se quiser configurar algo além de Host name, Size e Subnet/Availability Zone, escolha Advanced >> para ver outras opções. Veja a seguir o conjunto completo de opções:



2. Se desejar, substitua as configurações padrão, a maioria das quais você especificou quando criou a pilha. Para obter mais informações, consulte Criar uma nova pilha.

   Hostname

   Identifica a instância na rede. Por padrão, o AWS OpsWorks Stacks gera o nome de host de cada instância usando o tema Hostname que você especificou ao criar a pilha. Você pode substituir esse valor e especificar o nome do host de sua preferência.

   Tamanho

   Um tipo de EC2 instância da Amazon, que especifica os recursos da instância, como a quantidade de memória ou o número de núcleos virtuais. AWS OpsWorks As pilhas especificam um tamanho padrão para cada instância, que você pode substituir pelo tipo de instância de sua preferência.

   Os tipos de instância suportados pelas AWS OpsWorks pilhas dependem de a pilha estar ou não em uma VPC. Os tipos de instâncias também serão limitados se sua conta estiver

usando o nível gratuito da AWS. A lista suspensa Size (Tamanho) mostra os tipos de instância com suporte para a versão do Chef compatível com a pilha. Lembre-se de que as instâncias micro, como t1.micro, podem não ter recursos suficientes para oferecer suporte a algumas camadas. Para obter mais informações, consulte Tipos de instância do .

> **Note**
>
> Se você estiver usando instâncias de load balancer, observe que configurar eventos de ciclo de vida podem produzir um pico de carga de CPU significativo que pode durar um minuto ou mais. Esse pico de carga de instâncias menores pode ser o suficiente para acionar a ampliação, especialmente para grandes pilhas de balanceamento de carga com frequentes configurações de eventos. Veja a seguir algumas maneiras de reduzir a probabilidade de um evento ser configurado, causando uma ampliação desnecessária.
>
> - Use instâncias maiores, para que a carga adicional de um evento de configuração não seja suficiente para acionar a ampliação.
> - Não use tipos de instância, como o T2, que compartilham os recursos da CPU.
>
> Isso garante que quando uma configuração de evento ocorrer, todos os recursos de CPU da instância estejam imediatamente disponíveis.
> - Torne o `exceeded threshold` tempo significativamente mais longo que o tempo necessário para processar um evento de configuração, talvez 5 minutos.
>
> Para obter mais informações, consulte Uso da escalabilidade automática com base em carga.

Zona de disponibilidade/sub-rede

Se a pilha não estiver em uma VPC, essa configuração será identificada como Availability Zone e listará as zonas da região. Você pode usar essa configuração para substituir a zona de disponibilidade padrão que você especificou quando criou a pilha.

Se a pilha for executada em uma VPC, essa configuração será identificada como Subnet e listará as sub-redes da VPC. Você pode usar essa configuração para substituir a configuração padrão de sub-rede que você especificou quando criou a pilha.

> **Note**
>
> Por padrão, o AWS OpsWorks Stacks lista os intervalos CIDR da sub-rede. Para tornar a lista mais legível, use o console VPC ou a API para adicionar uma tag a cada sub-rede com a chave definida como e o valor definido como **`Name`** o nome da sub-rede. AWS OpsWorks Stacks acrescenta esse nome ao intervalo CIDR. No exemplo anterior, a sub-rede da tag Nome está definida como **`Private`**.

Tipo de Escala

Determina como a instância é iniciada e interrompida.

- O valor padrão é uma instância 24/7 que você inicia e interrompe manualmente.
- AWS OpsWorks O Stacks inicia e interrompe instâncias baseadas em tempo com base em um cronograma especificado.
- (Somente Linux) O AWS OpsWorks Stacks inicia e interrompe instâncias baseadas em carga com base em métricas de carga especificadas.

> **Note**
>
> Não se inicia ou se interrompe criando instâncias Linux no Amazon EC2, baseando-se em carga ou em tempo por conta própria. Em vez disso, você configura as instâncias, e o AWS OpsWorks Stacks as inicia e interrompe com base na configuração. Para obter mais informações, consulte Gerenciamento de carga com instâncias baseadas em tempo e carga.

Chave de ativação do SSH

Um par de EC2 chaves da Amazon. AWS OpsWorks O Stacks instala a chave pública na instância.

- Para instâncias do Linux, você pode usar a chave privada correspondente com um cliente da SSH para efetuar o login na instância.
- Para instâncias do Windows, você pode usar a chave privada correspondente para recuperar a senha do administrador da instância. Em seguida, você pode usar essa senha com o RDP para fazer login na instância como administrador.

Inicialmente essa configuração é o valor Default SSH key que você especificou quando criou a pilha.

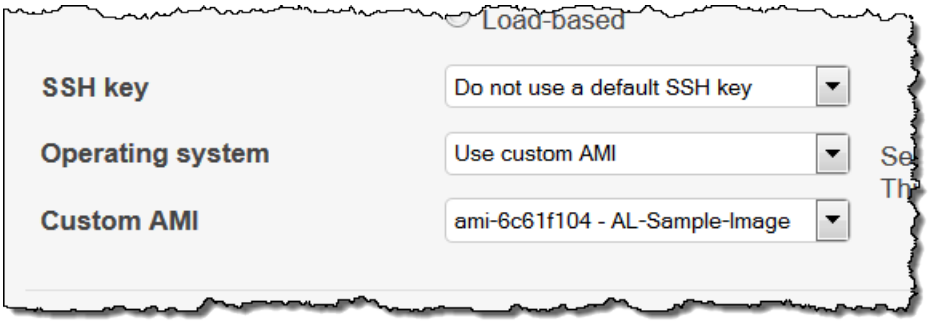
- Se o valor padrão estiver definido como Não use uma chave SSH padrão, você pode especificar uma das EC2 chaves da Amazon da sua conta.
- Se o valor padrão for definido como uma EC2 chave da Amazon, você poderá especificar uma chave diferente ou nenhuma chave.

Sistema operacional

O sistema operacional especifica qual sistema operacional a instância está executando. AWS OpsWorks O Stacks suporta somente sistemas operacionais de 64 bits.

Inicialmente essa configuração é o valor Default operating system que você especificou quando criou a pilha. Você pode substituir o valor padrão para especificar um sistema operacional diferente da Linux ou de uma Imagem de máquina da Amazon (AMI) personalizada. No entanto, você não pode mudar do Linux para o Windows ou do Windows para o Linux.

Se você selecionar Usar AMI personalizada, a página exibirá uma lista de tipos de dispositivo raiz e de arquitetura personalizados AMIs em vez de arquitetura.



Para obter mais informações, consulte Usando o Custom AMIs.

OpsWorks Versão do agente

OpsWorks A versão do agente especifica a versão do agente AWS OpsWorks Stacks que você deseja executar na instância. Se você desejar que o AWS OpsWorks Stacks atualize o agente automaticamente, escolha Inherit from stack (Herdar da pilha). Para instalar uma versão específica do agente e atualizar manualmente o agente na instância, escolha uma versão da lista suspensa.

> **Note**
>
> Nem todos as versões do agente funcionam com todas as versões de sistemas operacionais. Se sua instância estiver executando um agente, ou se você instalar um agente em uma instância, que não é totalmente compatível com o sistema operacional da instância, o console do AWS OpsWorks Stacks exibirá mensagens de erro que instruem você a instalar um agente compatível.

Locação

Escolha a opção de locação para sua instância. Você pode optar por executar suas instâncias em servidores físicos totalmente dedicados ao seu uso.

- Default - Rely on VPC settings. Sem configurações de locação ou de locação de heranças na sua VPC.
- Dedicated - Run a dedicated instance. Pague por hora para instâncias que são executadas no hardware de um único locatário. Para obter mais informações, consulte Instâncias dedicadas no Guia do usuário da Amazon VPC e Instâncias EC2 dedicadas da Amazon.
- Dedicated host - Run this instance on a dedicated host. pague por um host físico que seja totalmente dedicado à execução de suas instâncias e traga suas licenças de software existentes por soquete, por núcleo ou por VM para reduzir custos. Para obter mais informações, consulte Visão geral de hosts dedicados na EC2 documentação da Amazon e Hosts EC2 dedicados da Amazon.

Tipo de dispositivo raiz

Especifica o dispositivo raiz da instância de armazenamento.

- As instâncias do Linux podem obter suporte do Amazon EBS ou do armazenamento de instâncias.
- As instâncias do Windows obtêm suporte do Amazon EBS.

Para obter mais informações, consulte Armazenamento.

> **Note**
>
> Após a inicialização inicial, as instâncias apoiadas pelo Amazon EBS inicializam mais rápido do que as instâncias apoiadas pelo armazenamento de instâncias, porque o

AWS OpsWorks Stacks não precisa reinstalar o software da instância do zero. Para obter mais informações, consulte Armazenamento do dispositivo raiz.

Tipo de volume

Especifica o tipo de volume do dispositivo raiz: Magnetic, Provisioned IOPS (SSD) ou General Purpose (SSD). Para obter mais informações, consulte Tipos de volumes do Amazon EBS.

Tamanho do volume

Especifica o tamanho do volume raiz para o tipo de volume.especificado. Para obter mais informações, consulte Tipos de volumes do Amazon EBS.

- Finalidade geral (SSD). O tamanho mínimo permitido é: 8 GiB; o tamanho máximo é 16384 GiB.
- IOPS provisionado (SSD). O tamanho mínimo permitido é: 8 GiB; o tamanho máximo é 16384 GiB. Você pode definir um mínimo de 100 operações de entrada/saída por segundo (IOPS) e um máximo de 240 IOPS.
- Magnético. O tamanho mínimo permitido é 8 GiB; o tamanho máximo é 1024 GiB.

3. Selecione Add Instance para criar a instância nova.

> **Note**
>
> Você não pode substituir a configuração da versão do agente padrão da pilha quando cria uma instância. Para especificar uma configuração de versão do agente personalizado, você deve criar a instância e, em seguida, editar sua configuração.

Para adicionar uma instância existente a uma camada

1. Na página Instances selecione +Instance da camada apropriada e, se necessário, selecione a guia Existing.

> **Note**
>
> Caso mude de ideia sobre o uso de uma instância existente, selecione New para criar uma nova instância, conforme descrito no procedimento anterior.

2. Na guia Existing, selecione uma instância na lista.
3. Selecione Add Instance para criar a instância nova.

Uma instância representa uma EC2 instância da Amazon, mas é basicamente apenas uma estrutura de dados do AWS OpsWorks Stacks. Uma instância deve ser iniciada para criar uma EC2 instância da Amazon em execução, conforme descrito nas seções a seguir.

> **Important**
>
> Se você ativar instâncias em uma VPC padrão, você deve ter cuidado ao modificar a configuração da VPC. As instâncias sempre devem ser capazes de se comunicar com o serviço AWS OpsWorks Stacks, o Amazon S3 e com os repositórios de pacotes. Se, por exemplo, você remover um gateway padrão, as instâncias perderão a conexão com o serviço AWS OpsWorks Stacks, que então tratará as instâncias como falhadas e as recuperará automaticamente. No entanto, o AWS OpsWorks Stacks não conseguirá instalar o agente da instância nas instâncias corrigidas. Sem um agente, as instâncias não podem se comunicar com o serviço, e o processo de startup não avançará além do estado `booting` Para obter mais informações sobre a VPC padrão, consulte Plataformas suportadas.

Você também pode incorporar recursos de computação Linux em uma pilha que foi criada fora das AWS OpsWorks pilhas:

- EC2 Instâncias da Amazon que você criou diretamente usando o EC2 console, a CLI ou a API da Amazon.
- Instâncias on-premises executadas em seu próprio hardware, inclusive instâncias executadas em máquinas virtuais.

Para obter mais informações, consulte Uso de recursos de cálculo criados fora do AWS OpsWorks Stacks.

Usando o Custom AMIs

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks oferece suporte a duas formas de personalizar instâncias: Amazon Machine Images (AMIs) personalizadas e receitas do Chef. Ambas as abordagens oferecem a você o controle sobre quais pacotes e versões são instalados, como eles são configurados e assim por diante. No entanto, cada abordagem tem vantagens diferentes, de modo que o melhor depende de seus requisitos.

A seguir, os principais motivos a serem considerados usando uma AMI personalizada:

- Você deseja criar pacotes específicos com antecedência em vez de instalá-los depois que a instância é inicializada.
- Você deseja controlar o momento de empacotar as atualizações para fornecer uma imagem base consistente para a camada.
- Você quer instâncias — instâncias baseadas em carga em particular — para inicializar o mais rápido possível.

A seguir, os principais motivos para considerar o uso de receitas do Chef:

- Eles são mais flexíveis do que personalizados AMIs.
- Elas são mais fáceis de atualizar.
- Elas podem executar atualizações em instâncias em execução.

Na prática, a solução ideal pode ser uma combinação de ambas as abordagens. Para obter mais informações sobre receitas, consulte Livros de receitas e receitas.

Tópicos

- Como o Custom AMIs funciona com AWS OpsWorks Stacks

- Criação de uma AMI personalizada para AWS OpsWorks pilhas

Como o Custom AMIs funciona com AWS OpsWorks Stacks

Para especificar uma AMI personalizada para suas instâncias, selecione Usar AMI personalizada como sistema operacional da instância ao criar uma nova instância. AWS OpsWorks Em seguida, Stacks exibe uma lista dos personalizados AMIs na região da pilha e você seleciona o apropriado na lista. Para obter mais informações, consulte Adicionar uma instância a uma camada.

> **Note**
>
> Você não pode especificar uma AMI personalizada em particular como um sistema operacional padrão de uma pilha. Você pode definir `Use custom AMI` como o sistema operacional padrão da stack, mas você só poderá especificar uma AMI em particular quando adicionar novas instâncias a camadas. Para ter mais informações, consulte Adicionar uma instância a uma camada e Criar uma nova pilha. Embora seja possível criar instâncias com outros sistemas operacionais (como o CentOS 6). x) que foram criados de forma personalizada ou gerada pela comunidade AMIs, não são oficialmente suportados.

Este tópico discute alguns problemas gerais que você deve considerar antes de criar ou usar uma AMI personalizada.

Tópicos

- Comportamento de startup
- Escolha de uma camada
- Tratamento de aplicativos

Comportamento de startup

Quando você inicia a instância, o AWS OpsWorks Stacks usa a AMI personalizada especificada para iniciar uma nova EC2 instância da Amazon. AWS OpsWorks Em seguida, o Stacks usa cloud-init para instalar o agente AWS OpsWorks Stacks na instância e o agente executa as receitas de configuração da instância seguidas pelas receitas de implantação. Depois que a instância estiver online, o agente executará Configurar receitas para todas as instâncias da pilha, incluindo a instância recém-adicionada.

Escolha de uma camada

O agente AWS OpsWorks Stacks geralmente não entra em conflito com os pacotes instalados. No entanto, a instância deve ser membro de pelo menos uma camada. AWS OpsWorks O Stacks sempre executa as receitas dessa camada, o que pode causar problemas. Você deve compreender exatamente o que as receitas de uma camada fazem a uma instância antes de adicionar uma instância com uma AMI personalizada a essa camada.

Para ver quais receitas um determinado tipo de camada executa em sua instância, abra uma pilha que inclua essa camada. Em seguida, clique em Layers no painel de navegação e clique em Recipes para a camada de interesse. Para ver o código real, clique no nome da receita.

> **Note**
>
> Para Linux AMIs, uma forma de reduzir a possibilidade de conflitos é usar AWS OpsWorks pilhas para provisionar e configurar a instância que é a base da sua AMI personalizada. Para obter mais informações, consulte Crie uma AMI Linux personalizada a partir de uma instância do AWS OpsWorks Stacks.

Tratamento de aplicativos

Além de pacotes, você também pode incluir um aplicativo na AMI. Se você tiver um aplicativo grande e complexo, incluí-lo na AMI poderá reduzir o tempo de startup da instância. Você pode incluir aplicativos pequenos em sua AMI, mas geralmente há pouca ou nenhuma vantagem de tempo em relação à implantação do aplicativo pelo AWS OpsWorks Stacks.

Uma opção é incluir o aplicativo na sua AMI e também criar um aplicativo que implante o aplicativo nas instâncias desde um repositório. Essa abordagem reduz o tempo de inicialização, mas também oferece uma forma conveniente para atualizar o aplicativo depois que a instância estiver em execução. Observe que as receitas do Chef são imutáveis, de modo que a implantação de receitas não modificará o aplicativo, desde que a versão no repositório seja igual à da instância.

Criação de uma AMI personalizada para AWS OpsWorks pilhas

Para usar uma AMI personalizada com AWS OpsWorks pilhas, primeiro você deve criar uma AMI a partir de uma instância personalizada. Você pode escolher entre duas opções:

- Use o EC2 console ou a API da Amazon para criar e personalizar uma instância, com base em uma versão de 64 bits de uma das suportadas pelo AWS OpsWorks Stacks AMIs.

- Para Linux AMIs, use OpsWorks para criar uma EC2 instância da Amazon, com base na configuração de suas camadas associadas.

Antes de criar uma AMI Linux personalizada, desative-a `noexec` na `/tmp` partição para permitir que o AWS OpsWorks Stacks instale seu agente em instâncias Linux personalizadas.

> **Note**
>
> Lembre-se de que uma AMI pode não trabalhar com todos os tipos de instância; portanto, certifique-se de que sua AMI inicial é compatível com os tipos de instância que você planeja usar. Especificamente, os tipos de instância R3 exigem uma AMI com virtualização assistida por hardware (HVM).

Em seguida, você usa o EC2 console ou a API da Amazon para criar uma AMI personalizada a partir da instância personalizada. Você pode usar sua personalização AMIs em qualquer pilha que esteja na mesma região adicionando uma instância a uma camada e especificando sua AMI personalizada. Para obter mais informações sobre como criar uma instância que usa uma AMI personalizada, consulte Adicionar uma instância a uma camada.

> **Note**
>
> Por padrão, o AWS OpsWorks Stacks instala todas as atualizações do Amazon Linux na inicialização, o que fornece a versão mais recente. Além disso, o Amazon Linux libera uma nova versão aproximadamente a cada seis meses, o que pode envolver alterações significativas. Por padrão, as versões personalizadas AMIs baseadas no Amazon Linux são automaticamente atualizadas para a nova versão quando ela é lançada. Recomenda-se bloquear a AMI personalizada em uma versão específica do Amazon Linux, o que permite a você adiar a atualização até que a nova versão tenha sido testada. Para obter mais informações, consulte Como bloquear a AMI em uma versão específica?.

Tópicos

- Crie uma AMI personalizada usando a Amazon EC2
- Crie uma AMI Linux personalizada a partir de uma instância do AWS OpsWorks Stacks
- Criar uma AMI do Windows personalizada

Crie uma AMI personalizada usando a Amazon EC2

A maneira mais simples de criar uma AMI personalizada — e a única opção para Windows AMIs — é realizar toda a tarefa usando o EC2 console ou a API da Amazon. Para obter mais detalhes sobre as etapas a seguir, consulte Criando suas próprias AMIs.

Para criar uma AMI personalizada usando o EC2 console ou a API da Amazon

1. Crie uma instância usando uma versão de 64 bits de uma das suportadas pelo AWS OpsWorks Stacks AMIs.
2. Personalize a instância na Etapa 1, configurando-a, instalando pacotes e assim por diante. Lembre-se de que tudo o que você instalar será reproduzido em todas as instâncias baseadas com base na AMI, portanto não inclua itens específicos de uma determinada instância.
3. Pare a instância e crie uma AMI personalizada.

Crie uma AMI Linux personalizada a partir de uma instância do AWS OpsWorks Stacks

Para usar uma instância personalizada do AWS OpsWorks Stacks Linux para criar uma AMI, esteja ciente de que cada EC2 instância da Amazon criada pela OpsWorks inclui uma identidade exclusiva. Se você criar uma AMI personalizada de tal instância, ela inclui a identidade, e todas as instâncias com base na AMI tem a mesma identidade. Para garantir que as instâncias com base em sua AMI personalizada tenham uma identidade exclusiva, remova a identidade da instância personalizada antes de criar a AMI.

Para criar uma AMI personalizada a partir de uma instância do AWS OpsWorks Stacks

1. Crie uma pilha do Linux e adicione uma ou mais camadas para definir a configuração da instância personalizada. Você pode usar layers integrados, personalizados como apropriado, bem como layers totalmente personalizadas. Para obter mais informações, consulte Personalizando pilhas AWS OpsWorks.
2. Edite as camadas e desative AutoHealing.
3. Adicione uma instância com a sua distribuição preferida do Linux à camada ou camadas e a inicie. Recomendamos o uso de uma instância baseada no Amazon EBS. Abra a página de detalhes da instância e registre sua Amazon EC2 ID para mais tarde.
4. Quando a instância estiver online, faça login com o SSH e execute um dos quatro comandos a seguir, dependendo do sistema operacional da instância.

5. Para uma instância do Amazon Linux em uma pilha do Chef 11 ou do Chef 12 ou uma instância do Red Hat Enterprise Linux 7 em uma pilha do Chef 11, faça o seguinte:

   a. `sudo /etc/init.d/monit stop`

   b. `sudo /etc/init.d/opsworks-agent stop`

   c. `sudo rm -rf /etc/aws/opsworks/ /opt/aws/opsworks/ /var/log/aws/opsworks/ /var/lib/aws/opsworks/ /etc/monit.d/opsworks-agent.monitrc /etc/monit/conf.d/opsworks-agent.monitrc /var/lib/cloud/ /etc/chef`

   > **Note**
   >
   > Para instâncias em uma pilha do Chef 12, adicione as duas pastas a seguir a este comando:
   >
   > - `/var/chef`
   > - `/opt/chef`

   d. `sudo rpm -e opsworks-agent-ruby`

   e. `sudo rpm -e chef`

6. Para uma instância do Ubuntu 16.04 LTS ou 18.04 LTS em uma pilha do Chef 12, faça o seguinte:

   a. `sudo systemctl stop opsworks-agent`

   b. `sudo rm -rf /etc/aws/opsworks/ /opt/aws/opsworks/ /var/log/aws/opsworks/ /var/lib/aws/opsworks/ /etc/monit.d/opsworks-agent.monitrc /etc/monit/conf.d/opsworks-agent.monitrc /var/lib/cloud/ /var/chef /opt/chef /etc/chef`

   c. `sudo apt-get -y remove chef`

   d. `sudo dpkg -r opsworks-agent-ruby`

   e. `systemctl stop apt-daily.timer`

   f. `systemctl stop apt-daily-upgrade.timer`

   g. `rm /var/lib/systemd/timers/stamp-apt-daily.timer`

   h. `rm /var/lib/systemd/timers/stamp-apt-daily-upgrade.timer`

7. Para outras versões do Ubuntu compatíveis em uma pilha do Chef 12, faça o seguinte:

a. `sudo /etc/init.d/monit stop`

b. `sudo /etc/init.d/opsworks-agent stop`

c. `sudo rm -rf /etc/aws/opsworks/ /opt/aws/opsworks/ /var/log/aws/opsworks/ /var/lib/aws/opsworks/ /etc/monit.d/opsworks-agent.monitrc /etc/monit/conf.d/opsworks-agent.monitrc /var/lib/cloud/ /var/chef /opt/chef /etc/chef`

d. `sudo apt-get -y remove chef`

e. `sudo dpkg -r opsworks-agent-ruby`

8. Para uma instância do Red Hat Enterprise Linux 7 em uma pilha do Chef 12, faça o seguinte:

   a. `sudo systemctl stop opsworks-agent`

   b. `sudo rm -rf /etc/aws/opsworks/ /opt/aws/opsworks/ /var/log/aws/opsworks/ /var/lib/aws/opsworks/ /etc/monit.d/opsworks-agent.monitrc /etc/monit/conf.d/opsworks-agent.monitrc /var/lib/cloud/ /etc/chef /var/chef`

   c. `sudo rpm -e opsworks-agent-ruby`

   d. `sudo rpm -e chef`

9. Esta etapa depende do tipo de instância:

   - Para uma instância baseada no Amazon EBS, use o console AWS OpsWorks Stacks para interromper a instância e criar a AMI conforme descrito em Criação de uma AMI Linux baseada no Amazon EBS.
   - Para uma instância baseada em armazenamento de instâncias, crie a AMI conforme descrito em Criação de uma AMI Linux com armazenamento de instâncias e, em seguida, use AWS OpsWorks o console Stacks para interromper a instância.

     Quando criar a AMI, inclua os arquivos de certificado. Por exemplo, você pode chamar o comando `ec2-bundle-vol` com o argumento `-i` definido como `-i $(find /etc /usr /opt -name '*.pem' -o -name '*.crt' -o -name '*.gpg' | tr '\n' ',')`. Não remova as chaves públicas apt durante o empacotamento. O comando `ec2-bundle-vol` padrão lida com esta tarefa.

10. Limpe sua pilha retornando ao console do AWS OpsWorks Stacks e excluindo a instância da pilha.

Criar uma AMI do Windows personalizada

Os procedimentos a seguir são personalizados AMIs para o Windows Server 2022 Base. Você pode escolher outros sistemas operacionais Windows Server no console EC2 de gerenciamento da Amazon.

> **Important**
>
> Atualmente, o agente AWS OpsWorks Stacks não pode ser instalado e o AWS OpsWorks Stacks não pode gerenciar instâncias baseadas em Windows que usam um idioma de interface de usuário do sistema diferente de inglês - Estados Unidos da América (en-US).

Tópicos

- Criação de uma AMI personalizada do Windows com o Sysprep
- Criação de uma AMI personalizada do Windows sem o Sysprep
- Adição de uma nova instância usando uma AMI personalizada do Windows

Criação de uma AMI personalizada do Windows com o **`Sysprep`**

Criar um Windows personalizado AMIs usando o Sysprep normalmente resulta em uma inicialização mais lenta da instância, mas em um processo mais limpo. A primeira inicialização de uma instância criada a partir de uma imagem criada com `Sysprep` leva mais tempo devido `Sysprep` às atividades, reinicializações, provisionamento de AWS OpsWorks pilhas e primeira execução de AWS OpsWorks pilhas, incluindo instalação e configuração. Conclua as etapas para criar uma AMI personalizada do Windows no EC2 console da Amazon.

Para criar uma AMI personalizada do Windows com o Sysprep

1. No EC2 console da Amazon, escolha Launch Instance.
2. Localize Microsoft Windows Server 2022 Base e escolha Selecionar.
3. Selecione o tipo de instância que desejar e escolha Configure Instance Details. Faça as alterações de configuração na AMI, incluindo as configurações de nome da máquina, de armazenamento e de grupo de segurança. Escolha Executar.
4. Após o término do processo de inicialização da instância, obtenha sua senha e então conecte-se à instância em uma janela Conexão de Área de Trabalho Remota do Windows.

5. Na tela inicial do Windows, escolha Iniciar e comece a digitar **`ec2configservice`** até que os resultados mostrem o EC2ConfigServiceSettingsconsole. Abra o console de .
6. Na guia Geral, verifique se a caixa de seleção Habilitar UserData execução está preenchida (embora essa opção não seja necessária`Sysprep`, ela é necessária para que o AWS OpsWorks Stacks instale seu agente). Desmarque a caixa de seleção da opção Set the computer name of the instance... (Definir o nome do computador da instância ...), pois essa opção pode causar um loop de reinicialização com o AWS OpsWorks Stacks.
7. Na guia Imagem, defina a Senha do Administrador como Aleatória para permitir que EC2 a Amazon gere automaticamente uma senha que você possa recuperar com uma chave SSH ou Especifique para especificar sua própria senha. `Sysprep`salva essa configuração. Se você especificar sua própria senha, armazene a senha em um local conveniente. Recomendamos que você não escolha Keep Existing.
8. Escolha Apply e, em seguida, Shutdown with Sysprep. Quando for solicitada sua confirmação, escolha Yes.
9. Depois que a instância for interrompida, no EC2 console da Amazon, clique com o botão direito do mouse na instância na lista de Instâncias, escolha Imagem e, em seguida, escolha Criar imagem.
10. Na página Create Image, forneça um nome e uma descrição para a imagem e especifique a configuração do volume. Quando terminar, selecione Create Image.
11. Abra a página Images e espere que sua imagem mude do estágio pending para available. Sua nova AMI está pronta para uso.

Criação de uma AMI personalizada do Windows sem o **`Sysprep`**

Conclua as etapas para criar uma AMI personalizada do Windows no EC2 console da Amazon.

Para criar uma AMI personalizada do Windows sem o Sysprep

1. No EC2 console da Amazon, escolha Launch Instance.
2. Localize Microsoft Windows Server 2022 Base e escolha Selecionar.
3. Selecione o tipo de instância que desejar e escolha Configure Instance Details. Faça as alterações de configuração na AMI, incluindo as configurações de nome da máquina, de armazenamento e de grupo de segurança. Escolha Executar.
4. Após o término do processo de inicialização da instância, obtenha sua senha e então conecte-se à instância em uma janela Conexão de Área de Trabalho Remota do Windows.

5. Na instância, abra `C:\Program Files\Amazon\Ec2ConfigService\Settings\config.xml`, altere as duas configurações a seguir e salve e feche o arquivo:
   - `Ec2SetPassword` para `Enabled`
   - `Ec2HandleUserData` para `Enabled`
6. Desconecte-se da sessão do Remote Desktop e retorne ao EC2 console da Amazon.
7. Na lista Instances, interrompa a instância.
8. Depois que a instância for interrompida, no EC2 console da Amazon, clique com o botão direito do mouse na instância na lista de Instâncias, escolha Imagem e, em seguida, escolha Criar imagem.
9. Na página Create Image, forneça um nome e uma descrição para a imagem e especifique a configuração do volume. Quando terminar, selecione Create Image.
10. Abra a página Images e espere que sua imagem mude do estágio pending para available. Sua nova AMI está pronta para uso.

Adição de uma nova instância usando uma AMI personalizada do Windows

Depois que sua imagem mudar para o estado available, você poderá criar novas instâncias que estejam de acordo com a AMI personalizada do Windows. Quando você escolhe Usar AMI personalizada do Windows na lista de sistemas operacionais, o AWS OpsWorks Stacks exibe uma lista de itens personalizados AMIs.

Para adicionar uma nova instância com base em uma AMI personalizada do Windows

1. Quando sua nova AMI estiver disponível, acesse o console AWS OpsWorks Stacks, abra a página Instâncias para uma pilha do Windows e escolha + Instance na parte inferior da página para adicionar uma nova instância.
2. Na guia New, selecione Advanced.
3. Na lista suspensa Operating system, escolha Use custom Windows AMI.
4. Na lista suspensa Custom AMI, escolha a AMI criada e então selecione Add Instance.

Agora você pode iniciar e executar a instância.

Descreve como iniciar, parar e reiniciar instâncias 24/7

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Você pode usar instâncias 24/7, com pilhas do Linux e Windows.

Depois de adicionar uma instância 24 horas por dia, 7 dias por semana a uma camada, você deve iniciar manualmente a instância para iniciar a instância correspondente do Amazon Elastic Compute Cloud EC2 (Amazon) e interrompê-la manualmente para encerrar a instância da Amazon. EC2 Você também pode reinicializar manualmente as instâncias que não estão funcionando corretamente. AWS OpsWorks O Stacks inicia e interrompe automaticamente as instâncias baseadas em tempo e carga. Para obter mais informações, consulte Gerenciamento de carga com instâncias baseadas em tempo e carga.

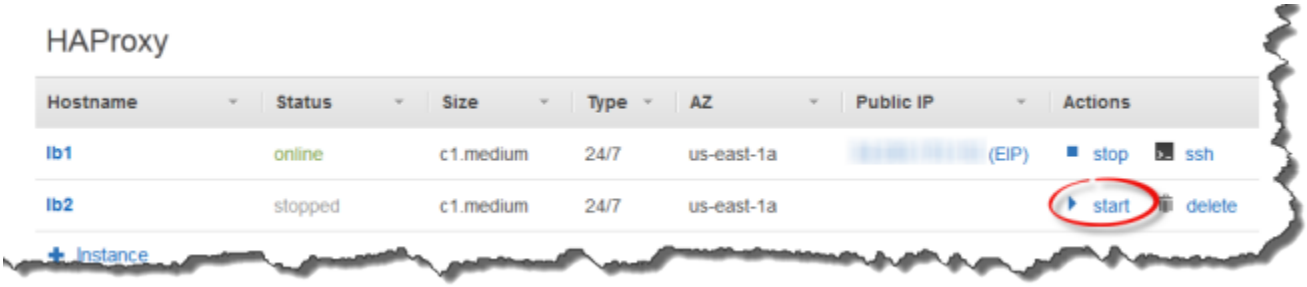
> **Important**
>
> AWS OpsWorks As instâncias do Stacks devem ser iniciadas, interrompidas e reiniciadas somente no AWS OpsWorks console. AWS OpsWorks não reconhece operações de início, parada ou reinicialização realizadas no EC2 console da Amazon.

Tópicos

- Iniciando ou reiniciando uma instância
- Interromper uma instância
- Como reinicializar uma instância

Iniciando ou reiniciando uma instância

Para iniciar uma nova instância, na página Instances, clique em start na coluna Actions da instância.



Você também pode criar várias instâncias e então inicia-las todas ao mesmo tempo clicando em Start all Instances.

Depois de iniciar a instância, o AWS OpsWorks Stacks inicia uma EC2 instância da Amazon e inicializa o sistema operacional. O processo de startup geralmente leva alguns minutos e é, de certa forma, geralmente mais lento para instâncias do Windows que para instâncias do Linux. Conforme o startup progride, o campo Status da instância exibe as séries dos valores a seguir:

1. solicitado - AWS OpsWorks Stacks chamou o EC2 serviço da Amazon para criar a EC2 instância da Amazon.
2. pendente - O AWS OpsWorks Stacks está aguardando o início da EC2 instância da Amazon.
3. inicialização - A EC2 instância da Amazon está inicializando.
4. running_setup - AWS OpsWorks Stacks acionou o evento Setup e está executando o Setup da camada, seguido por suas Deploy personalizada. Para obter mais informações, consulte Execução de receitas. Se você adicionou livros de receitas personalizados à pilha, o AWS OpsWorks Stacks instala a versão atual do seu repositório antes de executar o Setup and Deploy personalizada.
5. online - A instância está pronta para uso.

Quando o Status mudar para online, a instância ficará completamente operacional.

- Se a camada tiver um balanceador de carga conectado, o AWS OpsWorks Stacks adicionará a instância a ela.
- AWS OpsWorks As pilhas acionam um Configure evento, que executa as receitas Configure personalizada.

  Conforme necessário, essas receitas atualizam a instância para acomodar a nova instância.

- AWS OpsWorks As pilhas substituem a ação inicial da instância por parar, que você pode usar para interromper a instância.

Se a instância não iniciar com sucesso ou o Configurar receitas falhar, o status será definido como start_failed ou setup_failed, respectivamente. Você pode examinar os logs para determinar a causa. Para obter mais informações, consulte Guia de depuração e solução de problemas.

Uma instância interrompida permanece parte da pilha e retém todos recursos. Por exemplo, volumes do Amazon EBS e endereços IP elásticos ainda estão associados com uma instância interrompida. Você pode reiniciar uma instância interrompida escolhendo start na coluna Actions da instância. Reiniciar uma instância interrompida aciona o seguinte:

- Instâncias baseadas em armazenamento de instâncias — o AWS OpsWorks Stacks lança uma nova EC2 instância da Amazon com a mesma configuração.
- Instâncias suportadas pelo Amazon EBS — o AWS OpsWorks Stacks reinicia a instância da EC2 Amazon, que reconecta o volume raiz.

Depois que a instância termina de inicializar, o AWS OpsWorks Stacks instala as atualizações do sistema operacional e executa o Setup and Deploy receitas, assim como no início inicial. AWS OpsWorks O Stacks também faz o seguinte para instâncias reiniciadas, conforme apropriado.

- Associa novamente endereços IP elásticos.
- Reconecta volumes do Amazon Elastic Block Store (Amazon EBS).
- Para instâncias respaldadas pelo armazenamento de instâncias, instala as versões de livro de receitas mais recentes.

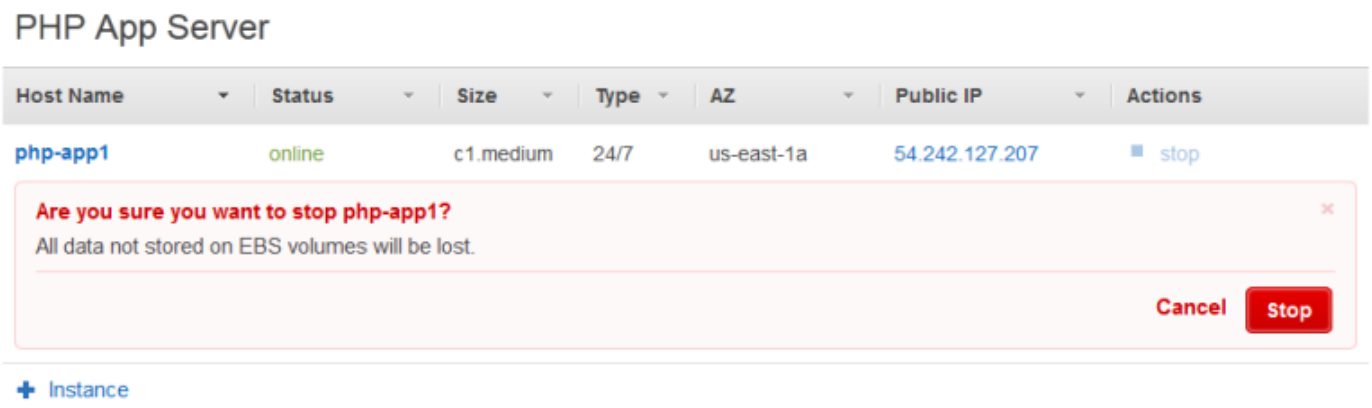
  Instâncias respaldadas pelo Amazon EBS continuam a usar livros de receita personalizados que foram armazenados no volume raiz. Se seus livros de receita personalizados mudaram desde que você parou a instância, você deve atualiza-las manualmente após a instância ficar online. Para obter mais informações, consulte Atualizar livros de receitas personalizadas.

> **Note**
>
> Pode levar vários minutos para um endereço IP elástico ser reassociado com uma instância reiniciada. Esteja ciente de que a configuração do Elastic IP da instância representa metadados, e simplesmente indica que o endereço deve ser associado com a instância.

A configuração de Public IP reflete o estado da instância, e pode estar vazio, inicialmente. Quando o endereço Elastic IP está associado com a instância, o endereço é atribuído à configuração do Public IP, seguido por (EIP).

Interromper uma instância

Na página Instâncias, clique em parar na coluna Ações da instância, que notifica AWS OpsWorks Stacks para executar as receitas de desligamento e encerrar a instância. EC2



Você também pode interromper todas instâncias na pilha clicando em Stop All Instances.

Depois de interromper a instância, o AWS OpsWorks Stacks executa várias tarefas:

1. Se a camada da instância tiver um balanceador de carga do Elastic Load Balancing anexado, o AWS OpsWorks Stacks cancelará o registro da instância.

   Se a camada é compatível com o recurso de drenagem de conexão do load balancer, o AWS OpsWorks Stacks atrasa o acionamento do evento Shutdown evento até que a drenagem da conexão seja concluída. Para obter mais informações, consulte Camada Elastic Load Balancing.
2. AWS OpsWorks As pilhas acionam um Shutdown , que executa as receitas Shutdown personalizada.
3. Após acionar o evento Shutdown evento, AWS OpsWorks Stacks espera por um tempo especificado para permitir o Shutdown termine e, em seguida, faz o seguinte:
   - Finalize instâncias respaldadas pelo armazenamento de instâncias, que exclui todos os dados.
   - Interrompa instâncias respaldadas em Amazon EBS, que preserva os dados no volume raiz.

   Para obter mais informações sobre o armazenamento de instâncias, consulte Armazenamento.

> **Note**
>
> A definição de tempo limite padrão de shutdown é 120 segundos. Se suas receitas Shutdown precisarem de mais tempo, você poderá editar a configuração da camada para alterar a configuração.

Você pode monitorar o processo de desligamento observando a coluna Status da instância. Conforme o desligamento progride, ele exibe as séries de valores a seguir:

1. encerramento - o AWS OpsWorks Stacks está encerrando a instância da Amazon. EC2
2. shutting_down - As AWS OpsWorks pilhas estão executando a camada Shutdown personalizada.
3. encerrada - A EC2 instância da Amazon é encerrada.
4. stopped - A instância foi parada.

Como reinicializar uma instância

Na página Instances, clique no nome da instância que não está funcionando para abrir a página de detalhes e clique em Reboot.

Esse comando executa uma reinicialização suave da EC2 instância Amazon associada. Ele não exclui os dados da instância, até mesmo de instâncias respaldadas pelo armazenamento de instâncias, e não aciona quaisquer eventos de ciclo de vida.

> **Note**
>
> Para que AWS OpsWorks as pilhas substituam automaticamente as instâncias com falha, ative a recuperação automática. Para obter mais informações, consulte Como usar a correção automática.

Gerenciamento de carga com instâncias baseadas em tempo e carga

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

À medida que o tráfego de entrada varia, a pilha pode ter poucas instâncias para processar confortavelmente a carga ou mais instâncias do que o necessário. Você pode economizar tempo e dinheiro usando instâncias com base em tempo ou carga para aumentar ou diminuir automaticamente as instâncias de uma camada, de forma que você sempre tenha instâncias o suficiente para processar o tráfego de entrada sem pagar por capacidade desnecessária. Não é necessário monitorar as cargas do servidor, ou iniciar ou interromper manualmente as instâncias. Além disso, as instâncias com base em tempo e carga distribuem, escalam e equilibram automaticamente aplicativos em diversas Zonas de disponibilidade em uma região, proporcionando redundância geográfica e escalabilidade.

A escalabilidade automática é baseada em dois tipos de instância, que se ajustam às instâncias online da camada com base em critérios diferentes:

- Instâncias Time-based

  Elas permitem que uma pilha processe cargas que seguem um padrão previsível ao incluir instâncias que são executadas apenas em momentos ou dias específicos. Por exemplo, você pode iniciar algumas instâncias após as 18:00 para executar tarefas de backup noturnas ou interromper algumas instâncias nos finais de semana quando o tráfego é menor.
- Instâncias Load-based

  Elas permitem que uma pilha processe cargas de variáveis ao iniciar instâncias adicionais quando o tráfego é alto e parar instâncias quando o tráfego é baixo, com base em qualquer uma das diversas métricas de carga. Por exemplo, você pode fazer com que o AWS OpsWorks Stacks inicie instâncias quando a utilização média da CPU exceda 80% e interrompa as instâncias quando a carga média da CPU cair abaixo de 60%.

As instâncias baseadas em tempo e em carga são compatíveis com pilhas do Linux, enquanto somente as instâncias baseadas em tempo são compatíveis com pilhas do Windows.

Diferente das instâncias 24 horas por dia, 7 dias por semana que devem ser iniciadas e paradas manualmente, as instâncias com base em tempo ou carga não podem ser paradas ou iniciadas por você. Em vez disso, você configura as instâncias e o AWS OpsWorks Stacks as inicia ou interrompe com base na configuração. Por exemplo, você configura instâncias baseadas em tempo para iniciar e parar em um cronograma especificado. AWS OpsWorks Em seguida, o Stacks inicia e interrompe as instâncias de acordo com essa configuração.

Uma prática comum é usar todos os três tipos de instância da seguinte forma.

- Um conjunto de instâncias 24 horas por dia, 7 dias por semana para processar a carga base. Em geral, você apenas inicia essas instâncias e permite que sejam executadas continuamente.
- Um conjunto de instâncias baseadas em tempo, que o AWS OpsWorks Stacks inicia e interrompe para lidar com variações previsíveis de tráfego. Por exemplo, se o tráfego for maior durante o horário de trabalho, você poderia configurar as instâncias com base em tempo para iniciar durante a manhã e desligar à noite.
- Um conjunto de instâncias baseadas em carga, que o AWS OpsWorks Stacks inicia e interrompe para lidar com variações de tráfego imprevisíveis. AWS OpsWorks As pilhas as iniciam quando a carga se aproxima da capacidade das instâncias baseadas em horário e 24 horas por dia, 7 dias por semana, e as interrompem quando o tráfego volta ao normal.

Para obter mais informações sobre como usar esses períodos de escalabilidade, consulte Otimização do número de servidores.

> **Note**
>
> Se você criou aplicativos para a camada de instâncias ou criou livros de receitas personalizados, o AWS OpsWorks Stacks implanta automaticamente a versão mais recente em instâncias baseadas em tempo e carga quando elas são iniciadas pela primeira vez. No entanto, o AWS OpsWorks Stacks não necessariamente implanta os livros de receitas mais recentes para reiniciar instâncias off-line. Para ter mais informações, consulte Editar aplicativos e Atualizar livros de receitas personalizadas.

Tópicos

- Uso da escalabilidade automática com base em tempo
- Uso da escalabilidade automática com base em carga
- Como a escalabilidade com base em carga difere da autorrecuperação

Uso da escalabilidade automática com base em tempo

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O escalonamento baseado em tempo permite controlar quantas instâncias uma camada deve ter on-line em determinados horários do dia ou dias da semana, iniciando ou interrompendo instâncias em uma programação específica. AWS OpsWorks O Stacks verifica a cada dois minutos e inicia ou interrompe as instâncias conforme necessário. Você especifica a programação separadamente para cada instância da seguinte forma:

- Horário do dia. Você pode ter mais instâncias em execução durante o dia do que à noite, por exemplo.
- Dia da semana. Você pode ter mais instâncias em execução em dias da semana do que nos fins de semana, por exemplo.

> **Note**
>
> Você não pode especificar datas específicas.

Tópicos

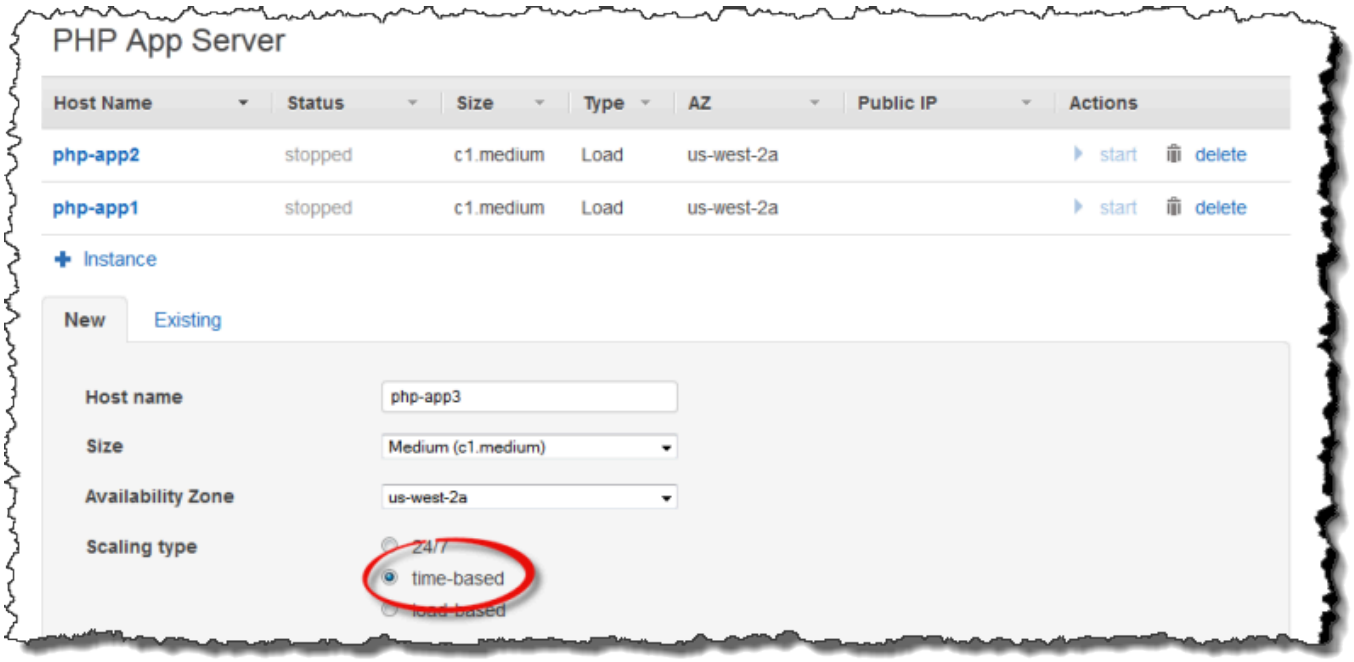
- Adicionar uma instância com base em tempo a uma camada
- Configurar uma instância com base em tempo

Adicionar uma instância com base em tempo a uma camada

Você pode adicionar uma nova instância com base em tempo a uma camada ou usar uma instância existente.

Para adicionar uma nova instância com base em tempo

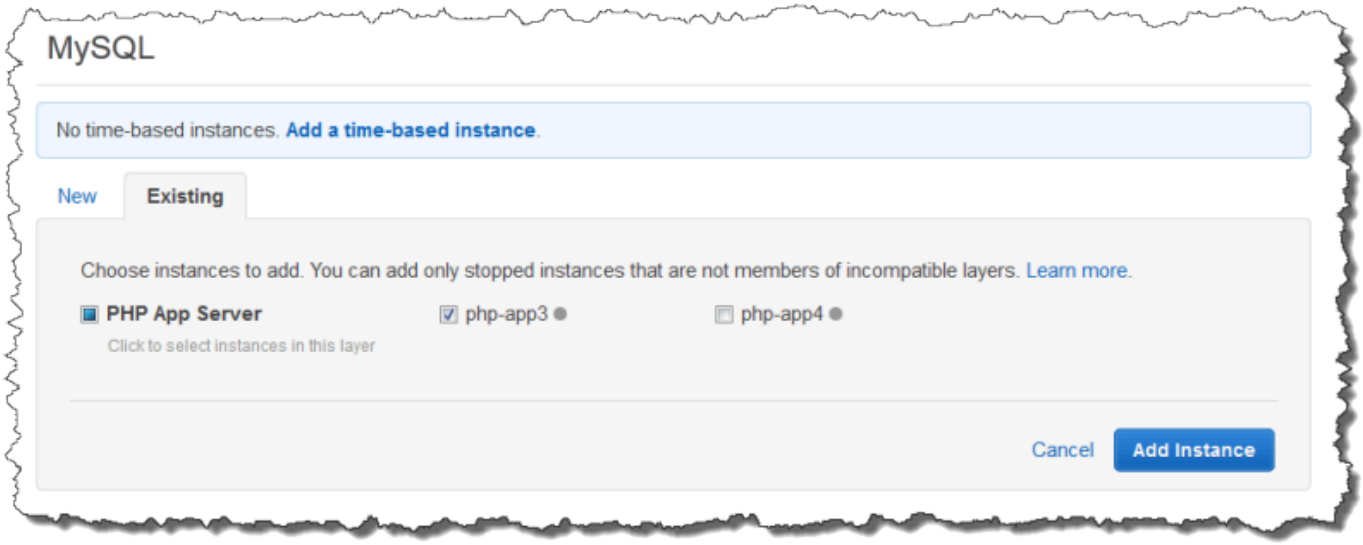
1. Na página Instâncias, clique em + Instância para adicionar uma instância. Na guia Novo, escolha Avançado e, em seguida, escolha Com base no tempo.



2. Configure a instância. Em seguida, escolha Adicionar instância para adicionar a instância à camada.

Para adicionar uma instância com base em tempo existentes a uma camada

1. Na página Instâncias com base no tempo, escolha + Instância se uma camada já tiver uma instância com base em tempo. Caso contrário, escolha Adicionar instância com base no tempo. Em seguida, escolha a guia Existente.



2. Na guia Existente, escolha uma instância na lista. A lista mostra apenas as instâncias com base em tempo.

> **Note**
>
> Caso mude de ideia sobre o uso de uma instância atual, na guia Novo crie uma nova instância, conforme descrito no procedimento anterior.

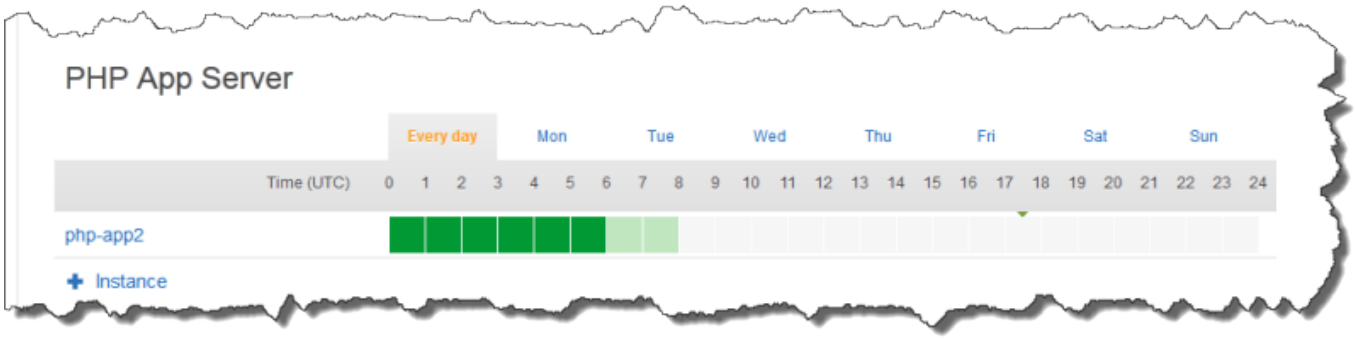
3. Clique em Adicionar instância para adicionar a instância à camada.

Configurar uma instância com base em tempo

Depois de adicionar uma instância com base em tempo a uma layer, você configura sua programação da seguinte forma.

Para configurar uma instância com base em tempo

1. No painel de navegação, em Instâncias, escolha Com base no tempo.
2. Especifique os períodos on-line de cada instância com base em tempo ao preencher as caixas adequadas abaixo da hora desejada.
   - Para usar a mesma programação todos os dias, escolha a guia Todos os dias e especifique os períodos de tempo on-line.
   - Para usar programações diferentes em dias diferentes, escolha cada dia e depois escolha os períodos adequados.



> **Note**
>
> Considere o tempo necessário para iniciar uma instância e que o AWS OpsWorks Stacks verifique somente a cada poucos minutos se as instâncias devem ser iniciadas ou interrompidas. Por exemplo, se uma instância deve estar em execução à 1:00 UTC, inicie-a à 0:00 UTC. Caso contrário, o AWS OpsWorks Stacks pode não iniciar a instância até alguns minutos depois da 1h UTC, e a instância demora mais alguns minutos para ficar on-line.

Você pode alterar os períodos on-line de uma instância a qualquer momento executando as etapas anteriores. Na próxima vez que o AWS OpsWorks Stacks fizer uma verificação, ele usará a nova programação para determinar se as instâncias devem ser iniciadas ou interrompidas.

> **Note**
>
> É possível adicionar uma nova instância com base em tempo a uma camada abrindo a página Com base no tempo e ao escolhendo Adicionar instância com base no tempo (se você ainda não adicionou uma instância com base no tempo à camada) ou + Instância (se a camada já tem uma ou mais instâncias com base no tempo). Em seguida, configure a instância conforme descrito em procedimentos anteriores.

Uso da escalabilidade automática com base em carga

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As instâncias baseadas em carga permitem que você inicie ou interrompa instâncias rapidamente em resposta às mudanças no tráfego de entrada. AWS OpsWorks O Stacks usa CloudWatch dados da Amazon para calcular as seguintes métricas para cada camada, que representam valores médios em todas as instâncias da camada:

- CPU: o consumo médio de CPU, como 80%
- Memória: o consumo média de memória, como 60%
- Carga: a média de trabalho de cálculo que um sistema executa em um minuto.

Você define os limites de ampliação e redução de qualquer uma dessas métricas. Você também pode usar CloudWatch alarmes personalizados como limites.

Um evento de escalabilidade é acionado ao ultrapassar um limite. Você determina como o AWS OpsWorks Stacks responde a eventos de escalabilidade ao especificar o seguinte:

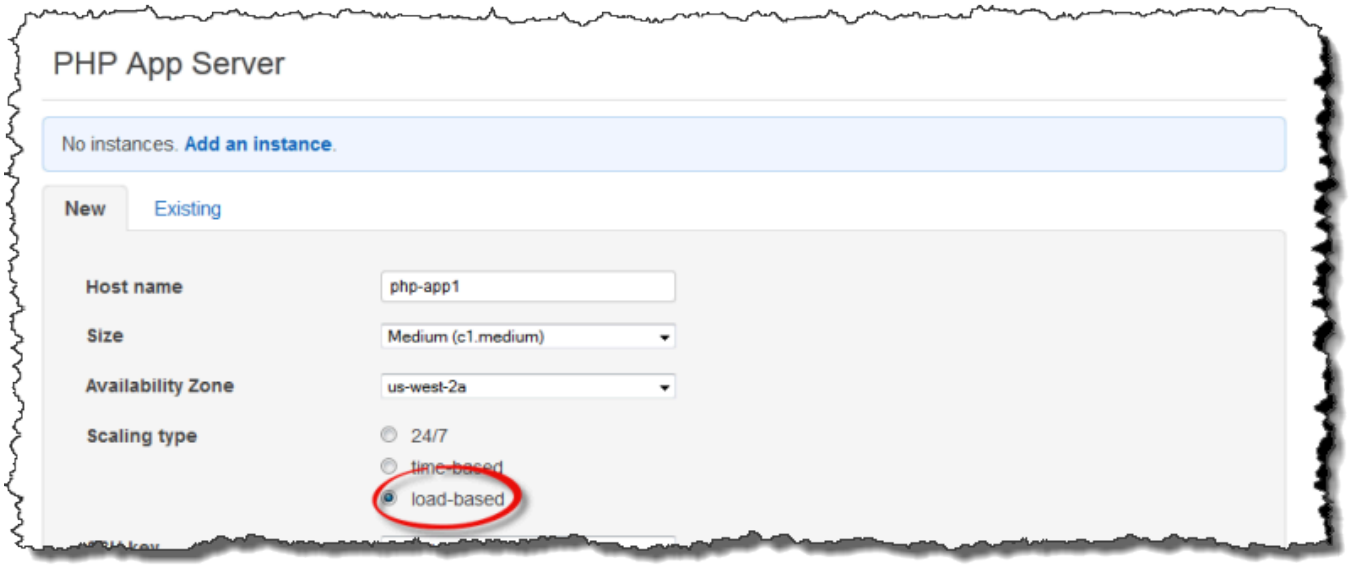
- Quantas instâncias para iniciar ou parar.
- Quanto tempo AWS OpsWorks as pilhas devem esperar depois de ultrapassar um limite antes de iniciar ou excluir instâncias. Por exemplo, o uso da CPU deve exceder o limite para pelo menos 15 minutos. Esse valor permite que você ignore breves flutuações de tráfego.
- Quanto tempo AWS OpsWorks as pilhas devem esperar após iniciar ou interromper as instâncias antes de monitorar as métricas novamente. Em geral, o ideal é permitir tempo suficiente para que instâncias iniciadas fiquem online ou instâncias interrompidas sejam antes de avaliar se a camada ainda excede o limite.

Quando ocorre um evento de escalabilidade, o AWS OpsWorks Stacks inicia ou interrompe somente instâncias baseadas em carga. Ele não iniciar ou interrompe instâncias 24 horas por dia, 7 dias por semana ou com base em tempo.

> **Note**
>
> A escalabilidade automática com base em carga não cria novas instâncias; ela inicia e interrompe somente as instâncias criadas. Portanto, você deve provisionar instâncias com base em carga suficientes com antecedência para processar a carga máxima antecipada.

Para criar uma instância com base em carga

1. Na página Instâncias, escolha + Instância para adicionar uma instância. Escolha Avançado e, em seguida, escolha Com base em carga.



2. Configure a instância, depois escolha Adicionar instância para adicionar outra instância à camada e a inicie.

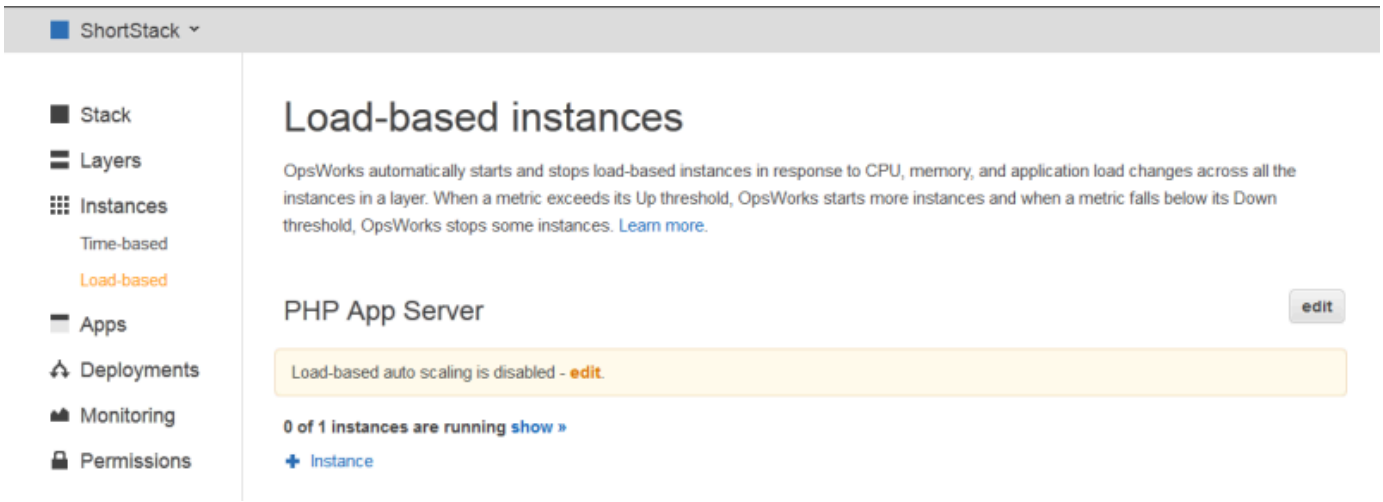
Repita esse procedimento até criar uma quantidade suficiente de instâncias. Você pode adicionar ou remover instâncias posteriormente, de acordo com o necessário.

Depois de adicionar instâncias com base em carga para a uma camada, você deve permitir a escalabilidade com base em carga e especificar a configuração. A configuração de escalabilidade com base em carga é uma propriedade de camada, não uma propriedade de instância que

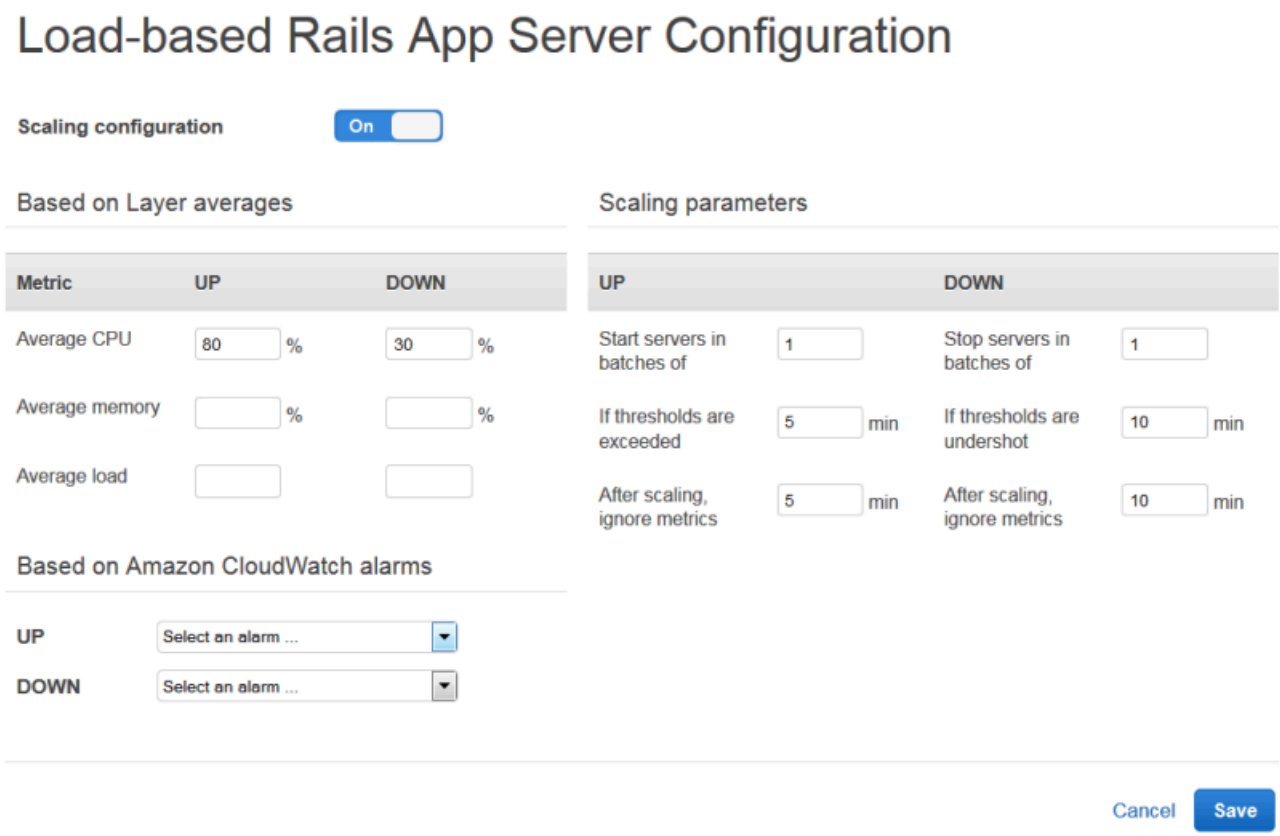
especifica quando uma camada deve iniciar ou interromper instâncias com base em carga. Ele deve ser especificada separadamente para cada camada que usa instâncias com base em carga.

Para habilitar e configurar a escalabilidade com base em carga

1. No painel de navegação, em Instâncias, escolha Com base em carga e, em seguida, escolha Editar na camada apropriada.



2. Defina Ajuste de escala automático com base em carga habilitado como Ativado. Em seguida, defina os parâmetros de limite e escalabilidade a fim de definir como e quando adicionar ou excluir instâncias.



Limites médios de camada

Você pode definir os limites de escalabilidade com base nos seguintes valores, com média calculada para todas as instâncias de camada.

- CPU média: o uso médio da CPU da camada, como porcentagem do total.
- Memória média: o uso médio da memória da camada, como porcentagem do total.
- Carga média: a carga média da camada.

  Para obter mais informações sobre como a carga é calculada, consulte Carga (computação) na Wikipedia.

Ultrapassar um limite causa um evento de escalabilidade, aumento de escala se forem necessárias mais instâncias e redução de escala se forem necessárias menos instâncias. AWS OpsWorks Em seguida, as pilhas adicionam ou excluem instâncias com base nos parâmetros de escalabilidade.

CloudWatch Alarmes personalizados

Você pode usar até cinco CloudWatch alarmes personalizados como limites de aumento ou redução de escala. Eles devem estar na mesma região da pilha. Para obter mais

informações sobre como criar alarmes personalizados, consulte Criação de alarmes na Amazon CloudWatch .

> **Note**
>
> Para usar alarmes personalizados, é necessário atualizar a função do serviço para permitir `cloudwatch:DescribeAlarms`. Você pode fazer com que o AWS OpsWorks Stacks atualize a função para você na primeira vez que usar esse recurso ou pode editar a função manualmente. Para obter mais informações, consulte Permitindo que AWS OpsWorks as pilhas ajam em seu nome.
> Quando há vários alarmes configurados para configuração baseada em carga, se um alarme estiver no estado de alarme `INSUFFICIENT_DATA` métrico, o escalonamento da instância com base na carga não poderá ocorrer mesmo se outro alarme estiver no estado `ALARM`. O ajuste de escala automático só pode prosseguir se todos os alarmes estiverem nos estados `OK` ou `ALARM`. Para obter mais informações sobre o uso de CloudWatch alarmes da Amazon, consulte Como usar CloudWatch alarmes da Amazon no Guia CloudWatch do usuário da Amazon.

Parâmetros de escalabilidade

Os parâmetros a seguir controlam como o AWS OpsWorks Stacks gerencia os eventos de escalabilidade.

- Iniciar servidores em lotes de: a quantidade de instâncias para adicionar ou remover quando o evento de escalabilidade ocorre.
- Se os limites forem excedidos — a quantidade de tempo (em minutos) em que a carga deve permanecer acima de um limite de aumento ou abaixo de um limite de redução de escala antes que as pilhas acionem um evento de escalabilidade. AWS OpsWorks
- Após a escalabilidade, ignore as métricas — A quantidade de tempo (em minutos) após a ocorrência de um evento de escalabilidade em que AWS OpsWorks as pilhas devem ignorar as métricas e suprimir eventos de escalabilidade adicionais.

  Por exemplo, o AWS OpsWorks Stacks adiciona novas instâncias após um evento de upscaling, mas as instâncias não começarão a reduzir a carga até serem inicializadas e configuradas. Não é necessário criar mais eventos de escalabilidade até as novas instâncias estarem online e processando solicitações, o que demora em geral alguns minutos. Essa configuração permite que você direcione o AWS OpsWorks Stacks para

suprimir eventos de escalabilidade por um período suficiente para que as novas instâncias fiquem online.

É possível aumentar essa configuração para impedir oscilações súbitas de escalabilidade quando as médias da camada, como CPU média, Memória média ou Carga média, estiverem em discordância temporária.

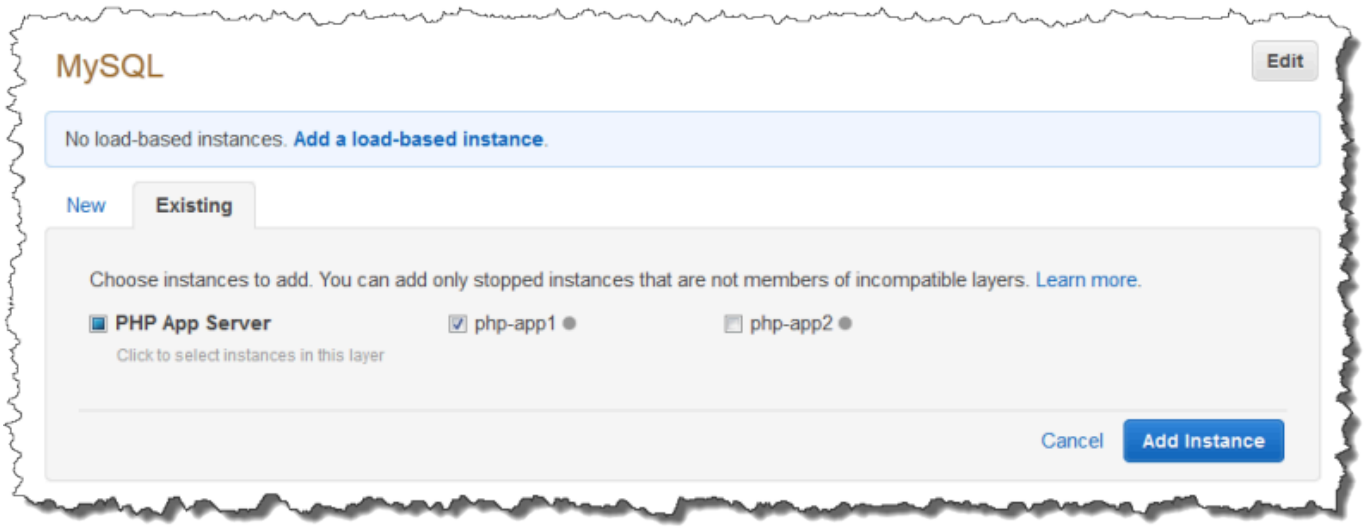
Por exemplo, se o uso da CPU estiver acima do limite e uso de memória estiver sendo reduzido, um evento de ampliação de instância pode ser seguido imediatamente por um evento de redução de memória. Para evitar isso, você pode aumentar a quantidade de minutos na configuração Após escalabilidade, ignorar as métricas. Neste exemplo, a escalabilidade da CPU ocorreria, mas o evento de redução de memória não.

3. Para adicionar mais instâncias com base em carga, escolha + Instância, defina as configurações e depois escolha Adicionar Instância. Repita até você ter instâncias com base em carga suficientes para processar a carga máxima antecipada. Em seguida, escolha Salvar.

> **Note**
>
> Também é possível adicionar uma nova instância com base em carga a uma camada abrindo a página Com base em carga e ao escolher Adicionar instância com base em carga (se você ainda não adicionou uma instância com base em carga à camada) ou + Instância (se a camada já tem uma ou mais instâncias com base em carga). Em seguida, configure a instância conforme descrito anteriormente nesta sessão.

Para adicionar uma instância com base em carga existente a uma camada

1. No painel de navegação, em Instâncias, escolha Com base em carga.
2. Se você já tiver habilitado a escalabilidade automática com base em carga para uma camada, escolha + Instância. Caso contrário, escolha Adicionar instância com base em carga. Escolha a guia Existente.



3. Na guia Existente, escolha uma instância. A lista mostra apenas as instâncias com base em carga.

> **Note**
>
> Caso mude de ideia sobre o uso de uma instância atual, na guia Novo para criar uma nova instância, conforme descrito no procedimento anterior.

4. Clique em Adicionar instância para adicionar a instância à camada.

Você pode modificar a configuração ou desativar a escalabilidade automática com base em carga a qualquer momento.

Para desativar a escalabilidade automática com base em carga

1. No painel de navegação, em Instâncias, escolha Com base em carga e, em seguida, escolha Editar na camada apropriada.
2. Mude Ajuste de escala automático com base em carga habilitado para Não.

Como a escalabilidade com base em carga difere da autorrecuperação

A escalabilidade automática com base em carga usa as métricas de carga calculadas em todas as instâncias em execução. Se as métricas permanecerem entre os limites especificados, o AWS OpsWorks Stacks não iniciará nem interromperá nenhuma instância. Com a recuperação automática, por outro lado, o AWS OpsWorks Stacks inicia automaticamente uma nova instância com a mesma

configuração quando uma instância para de responder. A instância pode não responder devido a um problema de rede ou algum problema com a instância.

Por exemplo, considere que o limite de ampliação de sua CPU seja de 80% e uma instância para de responder.

- Se a recuperação automática estiver desativada e as instâncias restantes em execução conseguirem manter a utilização média da CPU abaixo de 80%, o AWS OpsWorks Stacks não iniciará uma nova instância. Em vez disso, inicia uma instância de substituição somente se o uso médio da CPU nas instâncias restantes exceder 80%.
- Se a recuperação automática estiver ativada, o AWS OpsWorks Stacks iniciará uma instância de substituição, independentemente dos limites de carga.

Uso de recursos de cálculo criados fora do AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

Instânciasdescreve como usar AWS OpsWorks Stacks para criar e gerenciar grupos de instâncias do Amazon Elastic Compute Cloud EC2 (Amazon). Você também pode incorporar recursos de computação Linux em uma pilha criada fora das AWS OpsWorks pilhas:

- EC2 Instâncias da Amazon que você criou diretamente usando o EC2 console, a CLI ou a API da Amazon.
- Instâncias on-premises executadas em seu próprio hardware, inclusive instâncias executadas em máquinas virtuais.

Esses recursos de computação se tornam instâncias AWS OpsWorks gerenciadas pelo Stacks, e você pode gerenciá-los da mesma forma que as instâncias normais AWS OpsWorks do Stacks:

- Gerenciar permissões dos usuários: você pode usar o gerenciamento de usuário do AWS OpsWorks Stacks para especificar quais usuários podem acessar suas pilhas, quais ações podem executar nas instâncias da pilha e se têm acesso SSH e privilégios sudo.
- Automatize tarefas — Você pode fazer com que o AWS OpsWorks Stacks execute receitas personalizadas do Chef para realizar tarefas como executar scripts em qualquer uma ou em todas as instâncias de uma pilha com um único comando.

  Se você atribuir a instância a uma camada, o AWS OpsWorks Stacks executará automaticamente um conjunto específico de receitas do Chef na instância em pontos-chave de seu ciclo de vida, incluindo suas receitas personalizadas. Observe que você pode atribuir EC2 instâncias registradas da Amazon somente a camadas personalizadas.
- Gerenciar recursos — Uma pilha permite agrupar e gerenciar recursos em uma Região da AWS, e o OpsWorks painel mostra o status de suas pilhas em todas as regiões.
- Instalar pacotes: você pode usar as receitas do Chef para instalar pacotes em qualquer instância em uma pilha.
- Atualize o sistema operacional — o AWS OpsWorks Stacks fornece uma maneira simples de instalar patches e atualizações de segurança do sistema operacional nas instâncias de uma pilha.
- Implantar aplicativos — O AWS OpsWorks Stacks implanta aplicativos de forma consistente em todas as instâncias do servidor de aplicativos da pilha.
- Monitoramento — O AWS OpsWorks Stacks cria CloudWatchmétricas personalizadas para monitorar todas as instâncias da sua pilha.

Para obter informações sobre preços, consulte os OpsWorks preços da AWS.

Veja a seguir o procedimento básico para trabalhar com uma instância registrada.

1. Registre a instância com uma pilha.

   A instância agora faz parte da pilha e é gerenciada pelo AWS OpsWorks Stacks.
2. Como opção, atribua a instância a uma camada.

   Essa etapa permite que você aproveite ao máximo a funcionalidade de gerenciamento de AWS OpsWorks pilhas. Você pode atribuir instâncias locais registradas a qualquer camada; EC2 instâncias registradas da Amazon podem ser atribuídas somente a camadas personalizadas.

3. Use AWS OpsWorks Stacks para gerenciar a instância.
4. Quando você não precisar mais da instância na pilha, cancele o registro, o que remove a instância das pilhas. AWS OpsWorks

As seções a seguir descrevem este processo em detalhes.

Tópicos

- Registrando uma instância com um AWS OpsWorks Stacks Stack
- Gerenciar instâncias registradas
- Atribuir uma instância registrada a uma camada
- Cancelar a atribuição de uma instância registrada
- Cancelar o registro de uma instância registrada
- Ciclo de vida de instância registrada

Registrando uma instância com um AWS OpsWorks Stacks Stack

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

Para registrar uma instância que está fora do AWS OpsWorks Stacks, você executa o AWS CLI aws opsworks register comando. Você pode executar esse comando a partir da instância que deseja registrar ou de outro computador. Você aplica as `AWSOpsWorksRegisterCLI_OnPremises` políticas `AWSOpsWorksRegisterCLI_EC2` ou a um usuário ou grupo para conceder as permissões

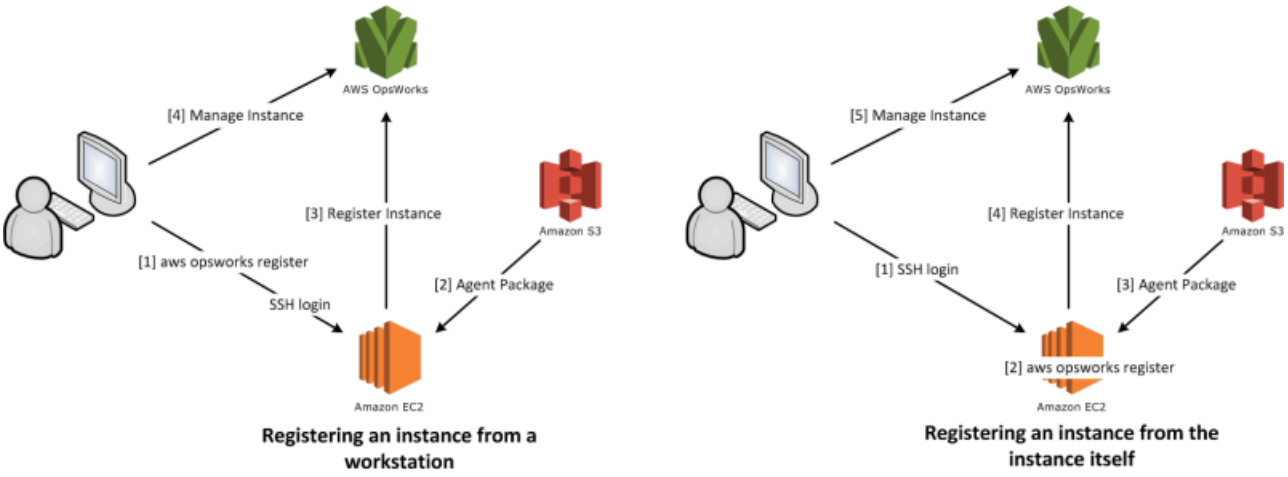
necessárias AWS CLI para registrar EC2 ou instâncias locais, respectivamente. Essas políticas exigem a versão 1.16.180 AWS CLI ou mais recente.

> **Note**
>
> Para impedir que usuários ou funções registrem instâncias, atualize o perfil de instância para negar acesso ao comando register.

O processo de registro instala um agente em uma instância que você deseja gerenciar usando AWS OpsWorks pilhas e registra a instância com uma AWS OpsWorks pilha especificada por você. Depois de registrar uma instância, a instância será parte da pilha e será gerenciada pelo AWS OpsWorks Stacks. Para obter mais informações, consulte Gerenciar instâncias registradas.

> **Note**
>
> Embora o AWS Tools for PowerShell inclua o `Register-OpsInstance`cmdlet, que chama a ação da `register` API, recomendamos que você use o AWS CLI para executar o `register` comando em vez disso.

O diagrama a seguir mostra as duas abordagens para registrar uma EC2 instância da Amazon. Você pode usar as mesmas abordagens para registrar uma instância on-premises.



> **Note**
>
> Você pode usar o console do AWS OpsWorks Stacks para gerenciar uma instância registrada, mas é necessário usar um comando `register` da CLI da AWS para registrar a

instância. O motivo é que o processo de registro deve ser executado na instância, o que não pode ser feito pelo console.

As seções a seguir descrevem o procedimento em detalhes.

Tópicos

- Passo a passo: registrar uma instância da estação de trabalho
- Registrando Amazon EC2 e instâncias locais

Passo a passo: registrar uma instância da estação de trabalho

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

O processo de registro oferece suporte a vários cenários. Esta seção mostra um end-to-end exemplo de um cenário: como usar sua estação de trabalho para registrar uma EC2 instância da Amazon. Os outros cenários de registro usam um procedimento semelhante. Para obter mais informações, consulte Registrando Amazon EC2 e instâncias locais.

> **Note**
>
> Normalmente, você deseja registrar uma EC2 instância existente da Amazon. Entretanto, é possível criar apenas uma nova instância e uma nova pilha para o passo a passo e excluí-los ao concluir.

Tópicos

- Etapa 1: criar uma pilha e uma instância
- Etapa 2: instalar e configurar a AWS CLI
- Etapa 3: registrar a instância com a pilha de EC2 registros

Etapa 1: criar uma pilha e uma instância

Para começar, você precisa de uma pilha e de uma EC2 instância da Amazon para ser registrada nessa pilha.

Para criar a pilha e a instância

1. Use o console do AWS OpsWorks Stacks para criar uma nova pilha chamada **`EC2Register`**. Você pode aceitar os valores padrão para as outras configurações da pilha.
2. Inicie uma nova instância no EC2 console da Amazon. Observe o seguinte.

   - A instância deve estar da mesma região e VPC da pilha.

     Se você estiver usando um VPC, selecione uma sub-rede pública para o passo a passo.
   - Se for necessário criar uma chave SSH, salve o arquivo da chave privada na estação de trabalho e registre o nome e o local do arquivo.

     Se você usar uma chave existente, registre o nome e a localização do arquivo da chave privada. Você precisará desses valores mais adiante.
   - A instância deve ter por base um dos sistemas operacionais Linux suportados. Por exemplo, se a pilha estiver no Oeste dos EUA (Oregon), você pode usar `ami-35501205` para iniciar uma instância 14.04 LTS do Ubuntu nessa região.

   Caso contrário, aceite os valores padrão.

Enquanto a instância está inicializando, você pode prosseguir para a próxima seção.

Etapa 2: instalar e configurar a AWS CLI

O registro é realizado usando o AWS CLI aws opsworks register comando. Antes de registrar sua primeira instância, você deve estar executando a versão 1.16.180 AWS CLI ou mais recente. Os detalhes da instalação dependem do sistema operacional da sua estação de trabalho. Para obter

mais informações sobre a instalação do AWS CLI, consulte Instalando a interface de linha de comando da AWS. Para verificar a versão da AWS CLI que você está executando, insira `aws --version` em uma sessão de shell.

> **Note**
>
> Para impedir que usuários ou funções registrem instâncias, atualize o perfil de instância para negar acesso ao comando register.

É altamente recomendável que você não pule essa etapa, mesmo que já esteja executando o AWS CLI em sua estação de trabalho. Usar a versão mais atual da AWS CLI é uma prática recomendada de segurança.

Você deve fornecer ao `register` um conjunto de credenciais da AWS com as permissões adequadas. A maneira recomendada de fazer isso, para evitar a instalação de credenciais diretamente em uma instância, é registrar as instâncias iniciadas em um perfil de instância e depois adicionar a opção `--use-instance-profile` ao comando `register`. Se estiver obtendo credenciais de um perfil de instância, pule para Etapa 3: registrar a instância com a pilha de EC2 registros neste tópico. No entanto, se a sua instância não tiver sido iniciada com um perfil de instância, você poderá criar um usuário do IAM. O procedimento a seguir cria um novo usuário com as permissões apropriadas, instalando as credenciais desse usuário na estação de trabalho e depois transmitindo essas credenciais para `register`.

> **Warning**
>
> Os usuários do IAM têm credenciais de longo prazo, o que representa um risco de segurança. Para ajudar a reduzir esse risco, recomendamos que você forneça a esses usuários somente as permissões necessárias para realizar a tarefa e que você os remova quando não forem mais necessários.

Para criar o usuário

1. No console do IAM, selecione Users (Usuários) no painel de navegação e selecione Add user (Adicionar usuário).
2. Adicione um usuário chamado **`EC2Register`**.
3. Escolha Próximo.

4. Na página Definir permissões, selecione Anexar políticas diretamente.
5. Entre **`OpsWorks`** na caixa Filtro da política de permissões para exibir as AWS OpsWorks políticas, selecione uma das políticas a seguir e escolha Avançar: revisão. Essa política confere aos usuários as permissões necessárias para executar o `register`.
   - Escolha `AWSOpsWorksRegisterCLI_EC2` permitir que o usuário registre EC2 instâncias que usam perfis de instância.
   - Escolha `AWSOpsWorksRegisterCLI_OnPremises` para permitir que as permissões do usuário registrem instâncias on-premises.
6. Escolha Próximo.
7. Na página Review (Revisar), selecione Create user (Criar usuário).
8. Agora crie chaves de acesso para seu usuário. Do painel de navegação, escolha Users e depois escolha o usuário para o qual você deseja criar a chave.
9. Escolha a guia Credenciais de segurança, selecione Criar chave de acesso.
10. Escolha as melhores práticas e alternativas da chave de acesso que melhor correspondam à sua tarefa.
11. Escolha Próximo.
12. (Opcionalmente) Insira uma tag para identificar as chaves de acesso.
13. Escolha Próximo.
14. Escolha Fazer download do arquivo .csv, salve o arquivo de credenciais em um local conveniente no seu sistema e escolha Fechar.

Você precisa fornecer as credenciais de usuário do IAM para `register`. Este passo a passo trata da tarefa instalando as credenciais do EC2 Registro no arquivo da sua estação de trabalho. `credentials` Para obter informações sobre outras formas de gerenciar credenciais para o AWS CLI, consulte Arquivos de configuração e credenciais.

Para instalar as credenciais do usuário

1. Crie ou abra o arquivo `credentials` da estação de trabalho. O arquivo está localizado em `~/.aws/credentials` (Linux, Unix e OS X) ou `C:\Users\`*`User_Name`*`\.aws\credentials` (sistema Windows).
2. Adicione um perfil do usuário do EC2 Registro ao `credentials` arquivo, usando o formato a seguir.

```
[ec2register]
aws_access_key_id = access_key_id
aws_secret_access_key = secret_access_key
```

Substitua *access_key_id* e *secret_access_key* pelas chaves de EC2 registro que você baixou anteriormente.

Etapa 3: registrar a instância com a pilha de EC2 registros

Agora, você está pronto para registrar a instância.

Para registrar a instância

1. Em AWS OpsWorks Pilhas, retorne à pilha de EC2 registros, escolha Instâncias no painel de navegação e, em seguida, escolha Registrar uma instância.
2. Selecione EC2 Instâncias, escolha Avançar: Selecionar instâncias e selecione sua instância na lista.
3. Escolha Avançar: Instalar o AWS CLI e Avançar: Registrar instâncias. AWS OpsWorks O Stacks usa automaticamente as informações disponíveis, como o ID da pilha e o ID da instância, para criar um modelo de `register` comando, que é exibido na página Registrar instâncias. Neste exemplo, você usa o `register` para entrar na instância com uma chave SSH e especifica explicitamente o arquivo da chave. Portanto, defina I use SSH keys to connect to my instances (Eu uso chaves SSH para conectar minhas instâncias) como Yes (Sim). O modelo de comando é semelhante ao seguinte.

```
aws opsworks register --infrastructure-class ec2 --region region endpoint ID
  --stack-id 247be7ea-3551-4177-9524-1ff804f453e3 --ssh-username [username]
  --ssh-private-key [key-file] i-f1245d10
```

> **Note**
>
> Você deve definir a região como a região do endpoint do serviço AWS OpsWorks Stacks, não a região da pilha, se a pilha estiver dentro de uma região clássica associada ao endpoint regional. `us-east-1` AWS OpsWorks As pilhas determinam a região da pilha a partir do ID da pilha.

4. O modelo de comando contém diversos valores de argumento específicos do usuário, que são indicados por parênteses e devem ser substituídos pelos valores adequados. Copie o modelo do comando para um editor de texto e edite-o da seguinte forma.

> **Important**
>
> O usuário do IAM; criado durante o processo de registro é necessário durante a vida de uma instância registrada. A exclusão do usuário faz com que o agente do AWS OpsWorks Stacks não consiga se comunicar com o serviço. Para ajudar a evitar problemas de gerenciamento de instâncias registrada caso o usuário do seja excluído, adicione o parâmetro `--use-instance-profile` ao comando `register` para usar o perfil de instância interno da instância. A adição do `--use-instance-profile` parâmetro também evita que ocorram erros ao alternar as chaves de acesso da AWS conta a cada 90 dias (uma prática recomendada), pois evita incompatibilidades entre as chaves de acesso disponíveis para o AWS OpsWorks agente e o usuário do IAM necessário.

- *`key file`*Substitua pelo caminho totalmente qualificado do arquivo de chave privada do par de EC2 chaves da Amazon que você salvou ao criar a instância.

  Você pode usar um caminho relativo, se preferir.
- *`username`*Substitua pelo nome de usuário da instância.

  Para este exemplo, o nome de usuário é `ubuntu` para uma instância do Ubuntu ou `ec2-user` para uma instância Red Hat Enterprise Linux (RHEL) ou Amazon Linux.
- Adicione `--use-instance-profile`, que executa `register` com o perfil de instância para evitar erros durante a rotação de chaves ou se o usuário do IAM principal for acidentalmente excluído.

Seu comando deve se parecer com o seguinte.

```
aws opsworks register --use-instance-profile --infrastructure-class ec2 \
  --region us-west-2  --stack-id 247be7ea-3551-4177-9524-1ff804f453e3 --ssh-
username ubuntu \
  --ssh-private-key "./keys/mykeys.pem" i-f1245d10
```

5. Abra uma janela de terminal em sua estação de trabalho, cole o comando `register` do seu editor e execute o comando.

   Em geral, o registro demora aproximadamente cinco minutos. Quando estiver concluído, retorne ao console do AWS OpsWorks Stacks e escolha Concluído. Em seguida, escolha Instances (Instâncias) no painel de navegação. A instância deve estar listada em Unassigned Instances. Em seguida, você pode atribuir a instância à camada ou deixá-la onde está, dependendo de como você pretende gerenciar a instância.
6. Ao terminar, interrompa a instância e exclua-a usando o console ou os comandos do AWS OpsWorks Stacks. Isso encerra a EC2 instância da Amazon, para que você não incorra em nenhuma cobrança adicional.

Registrando Amazon EC2 e instâncias locais

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

Esta seção descreve como registrar uma instância local EC2 ou da Amazon com uma AWS OpsWorks pilha Stacks.

Tópicos

- Preparar a instância
- Instalar e configurar a AWS CLI
- Registrar a instância
- Usar o comando register

- Exemplos de comandos de registro
- Políticas de registro de instância

Preparar a instância

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

Antes de registrar uma instância, você deve garantir que ela seja compatível com o AWS OpsWorks Stacks. Os detalhes dependem se você está registrando uma instância local ou da Amazon EC2 .

Instâncias on-premises

Uma instância on-premises deve atender aos seguintes critérios:

- A instância deve executar um dos sistemas operacionais Linux suportados. Embora seja possível criar ou registrar instâncias com outros sistemas operacionais (como o CentOS 6). x) que foram criados de forma personalizada ou gerada pela comunidade AMIs, não são oficialmente suportados.

  Você deve instalar o pacote `libyaml` na instância. Para instâncias do Ubuntu, o pacote é chamado de `libyaml-0-2`. Para instâncias CentOS e Red Hat Enterprise Linux, o pacote é chamado de `libyaml`.
- A instância deve ter um tipo de instância compatível (às vezes chamada do tamanho da instância). Tipos de instâncias suportados podem variar de acordo com o sistema operacional e dependem se a pilha está em uma VPC. Para ver uma lista dos tipos de instância compatíveis, veja os valores da lista suspensa Tamanho que são mostrados no console AWS OpsWorks Stacks quando você

tenta criar uma nova instância na pilha de destino. Se um tipo de instância está esmaecido e não pode ser criado na pilha de destino, você não pode registrar uma instância desse tipo.

- A instância deve ter acesso à Internet que permita a comunicação com o endpoint do serviço AWS OpsWorks Stacks,. `opsworks.us-east-1.amazonaws.com (HTTPS)` A instância também deve ser compatível com conexões de saída com recursos da AWS, como o Amazon S3.
- Se você pretende registrar a instância a partir de uma estação separada, a instância registrada deve ser compatível com o login de SSH da estação de trabalho.

  O login de SSH não é necessário se você executar o comando de registro a partir da instância.
- A chave de AWS acesso é usada para autenticação do AWS OpsWorks agente no serviço AWS OpsWorks Stacks. Se você alternar as chaves de acesso conforme recomendado a cada 90 dias, atualize o AWS OpsWorks agente manualmente para usar a nova chave. Em um computador ou instância on-premises, edite o arquivo `/etc/aws/opsworks/instance-agent.yml` com a nova chave de acesso e a chave secreta. O comando a seguir mostra a chave de acesso e a chave secreta neste arquivo. Um agente que está usando chaves antigas pode causar erros.

```
cat /etc/aws/opsworks/instance-agent.yml | egrep "access_key|secret_key"
:access_key_id: AKIAIOSFODNN7EXAMPLE
:secret_access_key: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
```

EC2Instâncias da Amazon

Uma EC2 instância da Amazon deve atender aos seguintes critérios:

- A AMI deve ter por base um dos sistemas operacionais Linux suportados. Para ver uma lista atual, consulte AWS OpsWorks Sistemas operacionais de pilha.

  Para obter mais informações, consulte Usando o Custom AMIs.

  Se a instância é baseada em uma AMI personalizada que deriva a partir de uma AMI padrão suportada, ou se a instância contém uma configuração mínima, você deve instalar o pacote `libyaml` na instância. Para instâncias do Ubuntu, o pacote é chamado de `libyaml-0-2`. Para instâncias Amazon Linux e Red Hat Enterprise Linux, o pacote é chamado de `libyaml`.
- A instância deve ter um tipo de instância compatível (às vezes chamada do tamanho da instância). Tipos de instâncias suportados podem variar de acordo com o sistema operacional e dependem se a pilha está em uma VPC. Para ver uma lista dos tipos de instância compatíveis, veja os valores da lista suspensa Tamanho que são mostrados no console AWS OpsWorks Stacks quando você

tenta criar uma nova instância na pilha de destino. Se um tipo de instância está esmaecido e não pode ser criado na pilha de destino, você não pode registrar uma instância desse tipo.

- A instância deve estar no estado `running`.
- A instância não deve fazer parte de um Grupo do Auto Scaling.

  Para obter mais informações, consulte Separar EC2 instâncias do seu grupo de Auto Scaling.
- A instância pode fazer parte de uma VPC, mas precisa estar na mesma VPC da pilha e a VPC deve estar configurada para funcionar corretamente com as pilhas. AWS OpsWorks
- As instâncias spot não são compatíveis, pois não funcionam com a autorrecuperação.

Quando você registra uma EC2 instância da Amazon, o AWS OpsWorks Stacks não modifica os grupos de segurança ou as regras da instância. Certifique-se de que as regras do grupo de segurança da instância atendam aos seguintes requisitos do AWS OpsWorks Stacks.

Ingress Rules (Regras de entrada)

As regras de entrada devem permitir o seguinte.

- Login de SSH.
- O tráfego das camadas adequadas.

  Por exemplo, um servidor de banco de dados normalmente permite o tráfego de entrada das camadas do servidor de aplicativos da pilha.
- O tráfego para as portas apropriadas.

  Por exemplo, as instâncias do servidor de aplicativos normalmente permitem todo o tráfego de entrada para as portas 80 (HTTP) e 443 (HTTPS).

Egress Rules (Regras de saída)

As regras de saída devem permitir o seguinte.

- Tráfego para o serviço AWS OpsWorks Stacks a partir de aplicativos em execução na instância.
- O tráfego para acessar os recursos da AWS, como o Amazon S3 de aplicativos usando a API da AWS.

Uma abordagem comum é não especificar nenhuma regra de saída, o que impõe restrições sobre o tráfego de saída.

Instalar e configurar a AWS CLI

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Antes de registrar sua primeira instância, você deve estar executando a versão 1.16.180 AWS CLI ou mais recente no computador a partir do qual você executa. `register` Os detalhes da instalação dependem do sistema operacional da sua estação de trabalho. Para obter mais informações sobre a instalação do AWS CLI, consulte Instalação da interface da linha de comando da AWS e configuração da interface da linha de comando da AWS. Para verificar a versão da AWS CLI que você está executando, insira `aws --version` em uma sessão de shell.

> **Note**
>
> Embora o AWS Tools for PowerShell inclua o `Register-OpsInstance`cmdlet, que chama a ação da `register` API, recomendamos que você use o AWS CLI para executar o `register` comando em vez disso.

É necessário executar o `register` com as permissões adequadas. Você pode obter as permissões usando um perfil do IAM ou, de forma menos ideal, instalando as credenciais do usuário com as permissões adequadas na estação de trabalho ou instância que será registrada. Você poderá executar o `register` com essas credenciais conforme descrito posteriormente. Especifique as permissões anexando uma política do IAM ao usuário ou perfil. Pois`register`, você usa `AWSOpsWorksRegisterCLI_OnPremises` as políticas `AWSOpsWorksRegisterCLI_EC2` ou, que concedem permissões para registrar instâncias locais EC2 ou da Amazon, respectivamente.

> **Note**
>
> Se você executa `register` em uma EC2 instância da Amazon, o ideal é usar uma função do IAM para fornecer credenciais. Para obter mais informações sobre como anexar uma

função do IAM a uma instância existente, consulte Anexar uma função do IAM a uma instância ou Substituir uma função do IAM no Guia EC2 do usuário da Amazon.

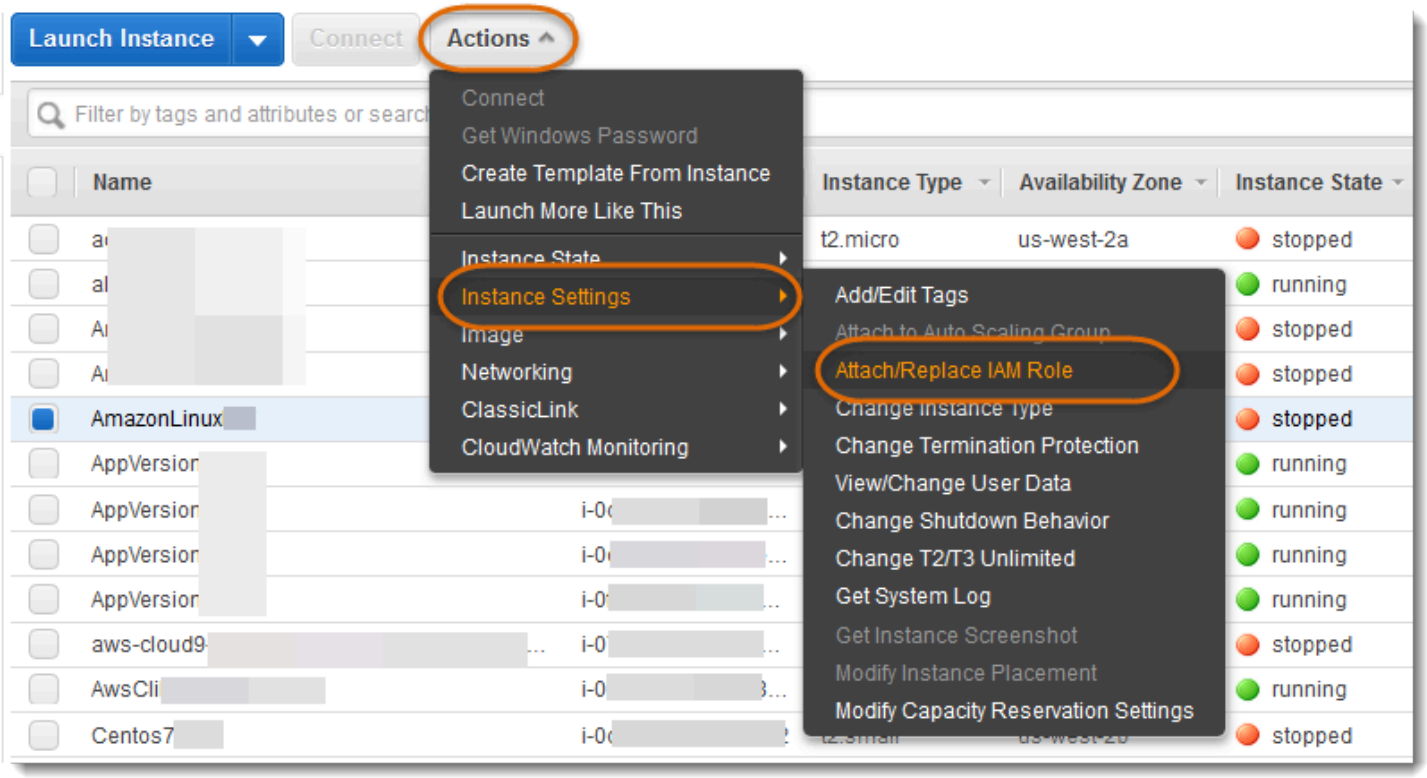
Para trechos de exemplo das políticas `AWSOpsWorksRegisterCLI_EC2` e `AWSOpsWorksRegisterCLI_OnPremises`, consulte Políticas de registro de instância. Para obter mais informações sobre como criar e gerenciar credenciais da AWS, consulte Credenciais de segurança da AWS.

Tópicos

- Usar uma função de IAM
- Usar credenciais instaladas

Usar uma função de IAM

Se você estiver executando o comando a partir da EC2 instância da Amazon que pretende registrar, a estratégia preferida para fornecer credenciais `register` é usar uma função do IAM que tenha a `AWSOpsWorksRegisterCLI_EC2` política ou permissões equivalentes anexadas. Essa abordagem permite que você evite instalar suas credenciais na instância. Uma forma de fazer isso é usar o comando Attach/Replace IAM Role no EC2 console, conforme mostrado na imagem a seguir.



Para obter mais informações sobre como anexar uma função do IAM a uma instância existente, consulte Anexar uma função do IAM a uma instância ou Substituir uma função do IAM no Guia EC2 do usuário da Amazon. Para instâncias que foram executadas com um perfil de instância (recomendado), adicione a opção `--use-instance-profile` ao comando `register` para fornecer credenciais. Não use o parâmetro `--profile`.

Se a instância estiver em execução e tiver uma função, você poderá conceder as permissões necessárias anexando a política `AWSOpsWorksRegisterCLI_EC2` à função. A função fornece um conjunto de credenciais padrão para a instância. Contanto que você não tiver instalado qualquer credencial na instância, o `register` considera automaticamente a função e é executado com suas permissões.

> **Important**
>
> Recomendamos que você não instale credenciais na instância. Além de criar um risco de segurança, a função da instância está no final da cadeia de provedores padrão que ela AWS CLI usa para localizar as credenciais padrão. As credenciais instaladas podem ter precedência sobre a função e, portanto, `register` pode não ter as permissões necessárias. Para obter mais informações, consulte Conceitos básicos do AWS CLI.

Se uma instância em execução não tem uma função, é necessário instalar credenciais com as permissões necessárias na instância, conforme descrito em Usar credenciais instaladas. Um procedimento recomendável, mais fácil e menos propenso a erros é usar instâncias que são iniciadas com um perfil de instância.

Usar credenciais instaladas

Há várias maneiras de instalar as credenciais do usuário em um sistema e fornecê-las a um AWS CLI comando. A seguir, descrevemos uma abordagem que não é mais recomendada, mas pode ser usada se você estiver registrando EC2 instâncias que foram executadas sem um perfil de instância. Também é possível usar as credenciais existentes do usuário do , contanto que as políticas anexadas concedam as permissões necessárias. Para obter mais informações, incluindo uma descrição de outras formas de instalar credenciais, consulte Arquivos de configuração e credenciais.

Para usar as credenciais instaladas

1. Crie um usuário do IAM e salve o ID da chave de acesso e a chave de acesso secreta em um local seguro.

   > **Warning**
   >
   > Os usuários do IAM têm credenciais de longo prazo, o que representa um risco de segurança. Para ajudar a reduzir esse risco, recomendamos que você forneça a esses usuários somente as permissões necessárias para realizar a tarefa e que você os remova quando não forem mais necessários.

2. Anexe a AWSOpsWorksRegisterCLI_OnPremises política ao usuário. Se preferir, você poderá anexar uma política que conceda permissões mais amplas, contanto que ela inclua as permissões `AWSOpsWorksRegisterCLI_OnPremises`.
3. Crie um perfil para o usuário no arquivo `credentials` do sistema. O arquivo está localizado em `~/.aws/credentials` (Linux, Unix e OS X) ou `C:\Users\`*`User_Name`*`\.aws\credentials` (sistema Windows). O arquivo contém um ou mais perfis no seguinte formato, cada um com uma ID de chave de acesso e uma chave de acesso secreta.

   ```
   [profile_name]
   aws_access_key_id = access_key_id
   aws_secret_access_key = secret_access_key
   ```

   Substitua as credenciais do IAM que você salvou anteriormente pelos *`secret_access_key`* valores *`access_key_id`* e. Você pode especificar qualquer nome de sua preferência para um nome de perfil, com duas limitações: o nome deve ser exclusivo e o perfil padrão deve se chamar `default`. Também é possível usar um perfil existente, contanto que ele tenha as permissões necessárias.
4. Use o parâmetro `--profile` do comando `register` para especificar o nome do perfil. O comando `register` é executado com as permissões concedidas às credenciais associadas.

   Também é possível omitir `--profile`. Nesse caso, `register` é executado com as credenciais padrão. Lembre-se de que não são necessariamente as credenciais do perfil padrão, portanto, verifique se as credenciais padrão têm as permissões necessárias. Para obter mais informações sobre como AWS CLI determina as credenciais padrão, consulte Configurando a interface de linha de comando da AWS.

Registrar a instância

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

Você registra uma instância ao executar o comando AWS CLI `register` a partir da estação de trabalho ou da instância. A maneira mais simples de processar a operação é usar o assistente de registro do console do AWS OpsWorks Stacks, que simplifica o processo de construção da string de comando. Depois que estiver familiarizado com o procedimento de registro, você poderá ignorar o assistente se preferir e executar o comando `register`.

Veja a seguir como usar o assistente de registro para registrar uma instância com uma pilha existente.

> **Note**
>
> Para registrar uma instância com uma nova pilha, você pode fazer isso escolhendo Registrar instâncias no painel de AWS OpsWorks pilhas. Isso inicia um assistente idêntico a das pilhas existentes, exceto pela página adicional que configura a nova pilha.

Para usar o assistente de registro e registrar uma instância

1. No console do AWS OpsWorks Stacks, crie uma pilha ou abra uma existente.
2. Escolha Instances no painel de navegação e, em seguida, register an instance.
3. Na página Escolha um tipo de instância, especifique se você deseja registrar uma Amazon EC2 ou uma instância local:

- Se você estiver registrando uma EC2 instância da Amazon, escolha Avançar: Selecionar instâncias.
- Se você estiver registrando uma instância on-premises, selecione Próximo: instalar CLI da AWS e vá para a Etapa 5.

4. Se você estiver registrando uma EC2 instância da Amazon, abra a página Selecionar instâncias para selecionar a instância a ser registrada. AWS OpsWorks O Stacks coleta as informações necessárias para criar o comando. Quando terminar, escolha Next:Install AWS CLI (Próximo: Instalar a CLI da AWS).
5. A instância na qual você planeja executar `register` deve estar executando a versão 1.16.180 AWS CLI ou mais recente. Para instalar ou atualizar a AWS CLI, a página do assistente de instalação fornece links para instruções de instalação e configuração. Depois de verificar a instalação da AWS CLI , especifique se você está executando o comando na instância a ser registrada ou em uma estação de trabalho separada e depois escolha Next: Register Instances (Próximo: Registrar instâncias).
6. A página Register Instances (Registrar instâncias) exibe um modelo para uma string de comando `register` que incorpora as opções selecionadas. Por exemplo, se você estiver registrando uma EC2 instância da Amazon em uma estação de trabalho separada, o modelo padrão se assemelha ao seguinte.

```
aws opsworks register --infrastructure-class ec2 --region us-west-2
  --stack-id 247be7ea-3551-4177-9524-1ff804f453e3 --ssh-username [username] i-
f1245d10
```

> **Important**
>
> O usuário do IAM; criado durante o processo de registro é necessário durante a vida de uma instância registrada. A exclusão do usuário faz com que o agente do AWS OpsWorks Stacks não consiga se comunicar com o serviço. Para ajudar a evitar problemas de gerenciamento de instâncias registrada caso o usuário do seja excluído, adicione o parâmetro `--use-instance-profile` ao comando `register` para usar o perfil de instância interno da instância. A adição do `--use-instance-profile` parâmetro também evita que ocorram erros ao alternar as chaves de acesso da AWS conta a cada 90 dias (uma prática recomendada), pois evita incompatibilidades entre as chaves de acesso disponíveis para o AWS OpsWorks agente e o usuário do IAM necessário.

Se você definir Eu uso chaves SSH como Sim, o AWS OpsWorks Stacks adiciona o `--ssh-private-key` argumento à string, que você pode usar para especificar um arquivo de chave SSH privado.

> **Note**
>
> Se você quiser `register` fazer login com uma senha, defina Eu uso chaves SSH como Não. Ao executar `register`, a senha será solicitada.

Copie essa string em um editor de texto e edite-o conforme necessário. Observe o seguinte.

- O texto entre parênteses representa informações que você deve fornecer, como a localização do arquivo chave de SSH.
- O modelo presume que você esteja executando o `register` com as credenciais padrão da AWS. Caso contrário, adicione um argumento `--profile` à string de comando e especifique o nome do perfil da credencial que você deseja usar.

Para outros cenários, pode ser necessário alterar o comando ainda mais. Para obter uma explicação sobre os argumentos `register` disponíveis e formas alternativas de construir a string de comando, consulte Usar o comando `register`. Você também pode exibir a documentação do comando ao executar `aws opsworks help register` na linha de comando. Para obter alguns exemplos de strings de comando, consulte Exemplos de comandos de registro.

7. Depois de terminar de editar a string de comando, abra uma janela no terminal na estação de trabalho ou use o SSH para fazer login na instância e executar o comando. A operação inteira demora em geral cinco minutos, durante os quais a instância fica no estado Registering.
8. Quando a operação é concluída, selecione Done. Agora, a instância está com o estado Registered e aparece listada como uma instância não atribuída na página Instances da pilha.

O comando `register` faz o seguinte.

1. Se `register` estiver funcionando em uma estação de trabalho, o comando primeiro usa SSH para fazer login na instância que será registrada.

O restante do processo ocorre na instância e é o mesmo independentemente de onde o comando é executado.

2. Faz o download do pacote do agente AWS OpsWorks Stacks do Amazon S3.
3. Desempacota e instala o agente e suas dependências, como o AWS SDK para Ruby.
4. Cria o seguinte:
   - Um usuário do IAM que inicializa o agente com o serviço AWS OpsWorks Stacks para fornecer comunicação segura.

     As permissões do usuário permitem somente a ação `opsworks:RegisterInstance` e expiram após 15 minutos.
   - Um grupo do IAM da pilha, que contém os usuários das instâncias registradas.
5. Cria um par de chaves RSA e envia a chave pública para o AWS OpsWorks Stacks.

   Este par de chaves é usado para criptografar as comunicações entre o agente e o AWS OpsWorks Stacks.
6. Registra a instância com AWS OpsWorks Stacks. Em seguida, a pilha executa um conjunto de receitas de instalação inicial para configurar a instância, que inclui o seguinte.
   - Substituir o arquivo de hosts da instância.

     Ao registrar a instância, você transferiu o gerenciamento de usuários para o AWS OpsWorks Stacks, que deve ter seu próprio arquivo de hosts para controlar as permissões de login SSH.
   - Para EC2 instâncias da Amazon, a configuração inicial também inclui o registro de todos os volumes anexados do Amazon EBS ou endereços IP elásticos com a pilha.

     Você deve garantir que os volumes do Amazon EBS não são montados em pontos de montagem reservados, incluindo `/var/www` e qualquer ponto de montagem reservado pelas camadas da instância. Para obter mais informações sobre o gerenciamento de recursos da pilha, consulte Gerenciamento de recursos. Para obter mais informações sobre pontos de montagem de camada, consulte AWS OpsWorks Referência de camadas de pilhas.

   Para obter uma descrição completa das alterações da configuração inicial, consulte Alterações de configuração inicial.

> **Note**
>
> A configuração inicial não atualiza o sistema operacional de uma instância registrada; você deve processar essa tarefa por conta própria. Para obter mais informações, consulte Gerenciamento de atualizações de segurança.

Usar o comando **`register`**

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

Para registrar uma instância, verifique se você está executando pelo menos a versão 1.16.180 da AWS CLI. Veja a seguir a sintaxe geral do comando `register`.

```
aws opsworks register \
  [--profile profile_name] \
  [--region region_name] \
  --infrastructure-class instance_type \
  --stack-id stack ID \
  [--local] | [--ssh-private-key key_file --ssh-username username] | [--override-
ssh command_string] \
  [--override-hostname hostname] \
  [--debug] \
  [--override-public-ip public IP] \
  [--override-private-ip private IP] \
..[--use-instance-profile] \
```

```
[ [IP address] | [hostname] | [instance ID]
```

Os argumentos a seguir são comuns a todos os AWS CLI comandos.

`--profile`

(Opcional) O nome do perfil de credencial. Se você omitir esse argumento, o comando será executado com as credenciais padrão. Para obter mais informações sobre como AWS CLI determina as credenciais padrão, consulte Configurando a interface de linha de comando da AWS.

`--region`

(Opcional) A região do endpoint do serviço AWS OpsWorks Stacks. Não `--region` defina para a região da pilha. AWS OpsWorks As pilhas determinam automaticamente a região da pilha a partir do ID da pilha.

> **Note**
>
> Se a região padrão já estiver definida, é possível omitir esse argumento. Para obter mais informações sobre como especificar uma região padrão, consulte Configurar a AWS Command Line Interface.

Use os argumentos a seguir para instâncias da Amazon EC2 e locais.

`--infrastructure-class`

(Obrigatório) Esse parâmetro deve ser definido como `ec2` ou`on-premises`, para indicar se você está registrando uma instância Amazon EC2 ou local, respectivamente.

`--stack-id`

(Obrigatório) O ID da pilha com a qual a instância será registrada.

> **Note**
>
> Para encontrar o ID de uma pilha, na página Stack, escolha Settings (Configurações). O ID da pilha é rotulado como OpsWorks ID e é um GUID parecido com. `ad21bce6-7623-47f1-bf9d-af2affad8907`

Argumentos de login de SSH

Use os seguintes argumentos para especificar como `register` deve efetuar o login na instância.

`--local`

(Opcional) Use este argumento para registrar a instância na qual você executa o comando.

Neste caso, `register` não precisa fazer login na instância.

`--ssh-private-key` e **`--ssh-username`**

(Opcional) Use esses argumentos se você estiver registrando a instância de uma estação de trabalho separada e deseja especificar explicitamente o nome de usuário ou o arquivo de chave privada.

- `--ssh-username`: use esse argumento para especificar um nome de usuário de SSH.

  Se você omitir `--ssh-username`, `ssh` usa o nome do usuário padrão.

- `--ssh-private-key`: use esse argumento para especificar explicitamente um arquivo de chave privada.

  Se você omitir `--ssh-private-key`, `ssh` tentará fazer login utilizando técnicas de autenticação que não exigem uma senha, incluindo o uso da chave privada. Se não houver suporte para nenhuma dessas técnicas, o `ssh` faz a consulta pela senha. Para obter mais informações sobre como o `ssh` processa a autenticação, consulte O protocolo de autenticação de Secure Shell (SSH).

`--override-ssh`

(Opcional) Use esse argumento se você estiver registrando a instância de uma estação de trabalho separada e deseja especificar uma string de comando `ssh` personalizada. O comando `register` usa essa string de comando para fazer login na instância registrada.

Para obter mais informações sobre `ssh`, consulte SSH.

`--override-hostname`

(Opcional) Especifica um nome de host para a instância, que é usado somente pelo AWS OpsWorks Stacks. O valor padrão é o nome do host da instância.

`--debug`

(Opcional) Fornece informações de depuração se houver falha no processo de registro. Para obter informações sobre a solução de problemas, consulte Solução de problemas do registro da instância.

`--use-instance-profile`

(Opcional, mas altamente recomendado para EC2 instâncias da Amazon) Permite que o `register` comando use um perfil de instância anexado, em vez de criar um usuário do IAM. Adicionar este parâmetro pode ajudar a evitar erros que ocorrem se você tentar gerenciar uma instância registrada quando o usuário do IAM é excluído acidentalmente.

> **Important**
>
> O usuário do IAM; criado durante o processo de registro é necessário durante a vida de uma instância registrada. A exclusão do usuário faz com que o agente do AWS OpsWorks Stacks não consiga se comunicar com o serviço. Para ajudar a evitar problemas de gerenciamento de instâncias registrada caso o usuário do seja excluído, adicione o parâmetro `--use-instance-profile` ao comando `register` para usar o perfil de instância interno da instância. A adição do `--use-instance-profile` parâmetro também evita que ocorram erros ao alternar as chaves de acesso da AWS conta a cada 90 dias (uma prática recomendada), pois evita incompatibilidades entre as chaves de acesso disponíveis para o AWS OpsWorks agente e o usuário necessário.

Alvo

(Condicional) Se você executar esse comando a partir de uma estação de trabalho, o valor final na string de comando especifica o destino de registro em uma das seguintes formas.

- O endereço IP público da instância.
- O nome de host da instância.
- Para EC2 instâncias da Amazon, o ID da instância.

  AWS OpsWorks O Stacks usa o ID da instância para obter a configuração da instância, incluindo o endereço IP público da instância. Por padrão, o AWS OpsWorks Stacks usa esse endereço para criar a cadeia de `ssh` comando usada para fazer login na instância. Se for necessário conectar a um endereço IP privado, é necessário usar `--override-ssh` para

fornecer uma string de comando personalizada. Para obter um exemplo, consulte Registrar uma instância on-premises a partir de uma estação de trabalho.

> **Note**
>
> Se você especificar um nome de host, `ssh` depende do servidor DNS para resolver o nome para uma instância específica. Caso não tenha certeza se o nome do host é exclusivo, use `ssh` para verificar se o nome do host é resolvido de acordo com a instância correta.

Se você executar esse comando a partir da instância que será registrada, omita o identificador da instância e use o argumento `--local`.

Os seguintes argumentos são apenas para instâncias on-premises.

`--override-public-ip`

(Opcional) O AWS OpsWorks Stacks exibe o endereço especificado como o endereço IP público da instância. Ele não altera o endereço IP público da instância. No entanto, se um usuário usa o console para se conectar à instância, por exemplo, escolhendo o endereço na página Instâncias, o AWS OpsWorks Stacks usa o endereço especificado. AWS OpsWorks As pilhas determinam automaticamente o valor padrão do argumento.

`--override-private-ip`

(Opcional) O AWS OpsWorks Stacks exibe o endereço especificado como o endereço IP privado da instância. Isso não altera o endereço IP privado da instância. AWS OpsWorks As pilhas determinam automaticamente o valor padrão do argumento.

Exemplos de comandos de registro

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

Esta seção contém alguns exemplos de strings de comando de `register`.

Registre uma EC2 instância da Amazon a partir de uma estação de trabalho

O exemplo a seguir registra uma EC2 instância da Amazon em uma estação de trabalho. A string de comando usa credenciais padrão e identifica a instância por seu ID de instância da Amazon EC2 . Você pode usar o exemplo para instância on-premises ao alterar `ec2` para `on-premises`.

```
aws opsworks register \
  --region us-west-2 \
  --use-instance-profile \
  --infrastructure-class ec2 \
  --stack-id ad21bce6-7623-47f1-bf9d-af2affad8907 \
  --ssh-user-name my-sshusername \
  --ssh-private-key "./keys/mykeys.pem" \
  i-2422b9c5
```

Registrar uma instância on-premises a partir de uma estação de trabalho

O exemplo a seguir registra uma instância on-premises a partir de uma estação de trabalho separada. A string de comando usa as credenciais padrão e faz login na instância com a string de comando `ssh` especificada. Se a instância exigir uma senha, o `register` informa você. Você pode usar o exemplo para EC2 instâncias da Amazon alterando `on-premises` para`ec2`.

```
aws opsworks register \
  --region us-west-2 \
  --infrastructure-class on-premises \
  --stack-id ad21bce6-7623-47f1-bf9d-af2affad8907 \
  --override-ssh "ssh your-user@192.0.2.0"
```

> **Note**
>
> Você pode usar `--override-ssh` para especificar qualquer string de comando SSH personalizada. AWS OpsWorks Em seguida, o Stacks usa a string especificada para fazer login na instância em vez de criar uma string de comando. Para obter outro exemplo, consulte Registrar uma instância usando uma string de comando SSH personalizada.

Registrar uma instância usando uma string de comando SSH personalizada

O exemplo a seguir registra uma instância on-premises a partir de uma estação de trabalho e usa o argumento `--override-ssh` para especificar um comando SSH personalizado que `register` usa para fazer login na instância. Este exemplo usa `sshpass` para fazer login com um nome de usuário e senha, mas é possível especificar qualquer string de comando `ssh` válida.

```
aws opsworks register \
  --region us-west-2 \
  --infrastructure-class on-premises \
  --stack-id 2f92ff9d-04f2-4728-879b-f4283b40783c \
  --override-ssh "sshpass -p 'mypassword' ssh your-user@192.0.2.0"
```

Registrar uma instância ao executar **`register`** da instância

O exemplo a seguir mostra como registrar uma EC2 instância da Amazon executando a `register` partir da própria instância. A string de comando depende das credenciais padrão para suas permissões. Para usar o exemplo de uma instância on-premises, altere `--infrastructure-class` para `on-premises`.

```
aws opsworks register \
  --region us-west-2 \
  --infrastructure-class ec2 \
  --stack-id ad21bce6-7623-47f1-bf9d-af2affad8907 \
  --local
```

Registrar uma instância com um endereço IP privado

Por padrão, o `register` usa o endereço IP público da instância para fazer login na instância. Para registrar uma instância com um endereço IP privado, como uma instância em uma sub-rede privada da VPC, é necessário usar o `--override-ssh` para especificar uma string de comando `ssh` personalizada.

```
aws opsworks register \
  --region us-west-2 \
  --infrastructure-class ec2 \
  --stack-id 2f92ff9d-04f2-4728-879b-f4283b40783c \
  --override-ssh "ssh -i mykey.pem ec2-user@10.183.201.93" \
  i-2422b9c5
```

Políticas de registro de instância

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As `AWSOpsWorksRegisterCLI_OnPremises` políticas `AWSOpsWorksRegisterCLI_EC2` e fornecem as permissões corretas para registro EC2 e instâncias locais, respectivamente. Você adiciona `AWSOpsWorksRegisterCLI_EC2` ao seu usuário do IAM para registrar EC2 instâncias, mas adiciona `AWSOpsWorksRegisterCLI_OnPremises` ao seu usuário para registrar instâncias locais. Para usar essas políticas, você deve estar executando pelo menos a versão 1.16.180 do AWS CLI ou mais recente.

A política **`AWSOpsWorksRegisterCLI_EC2`**

Adicione `AWSOpsWorksRegisterCLI_EC2` ao seu usuário para registrar EC2 instâncias. Você deve usar esse perfil se planeja registrar somente EC2 instâncias. Quando você usa essa política, as permissões são fornecidas pelo perfil da EC2 instância.

```
{
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": [
            "opsworks:AssignInstance",
            "opsworks:CreateLayer",
```

```
            "opsworks:DeregisterInstance",
            "opsworks:DescribeInstances",
            "opsworks:DescribeStackProvisioningParameters",
            "opsworks:DescribeStacks",
            "opsworks:UnassignInstance"
          ],
          "Resource": [
            "*"
          ]
        },
        {
          "Effect": "Allow",
          "Action": [
            "ec2:DescribeInstances"
          ],
          "Resource": [
            "*"
          ]
        }
      ]
    }
```

A política **`AWSOpsWorksRegisterCLI_OnPremises`**

Adicione `AWSOpsWorksRegisterCLI_OnPremises` ao seu usuário para registrar instâncias on-premises. Essa política inclui permissões do IAM, como `AttachUserPolicy`, mas os recursos nos quais as permissões funcionam são restritos.

```
    {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": [
            "opsworks:AssignInstance",
            "opsworks:CreateLayer",
            "opsworks:DeregisterInstance",
            "opsworks:DescribeInstances",
            "opsworks:DescribeStackProvisioningParameters",
            "opsworks:DescribeStacks",
            "opsworks:UnassignInstance"
          ],
          "Resource": [
```

```
            "*"
          ]
        },
        {
          "Effect": "Allow",
          "Action": [
            "ec2:DescribeInstances"
          ],
          "Resource": [
            "*"
          ]
        },
        {
          "Effect": "Allow",
          "Action": [
            "iam:CreateGroup",
            "iam:AddUserToGroup"
          ],
          "Resource": [
            "arn:aws:iam::*:group/AWS/OpsWorks/OpsWorks-*"
          ]
        },
        {
          "Effect": "Allow",
          "Action": [
            "iam:CreateUser",
            "iam:CreateAccessKey"
          ],
          "Resource": [
            "arn:aws:iam::*:user/AWS/OpsWorks/OpsWorks-*"
          ]
        },
        {
          "Effect": "Allow",
          "Action": [
            "iam:AttachUserPolicy"
          ],
          "Resource": [
            "arn:aws:iam::*:user/AWS/OpsWorks/OpsWorks-*"
          ],
          "Condition": {
            "ArnEquals":
              {
```

```
                "iam:PolicyARN": "arn:aws:iam::aws:policy/
AWSOpsWorksInstanceRegistration"
              }
            }
        }
      ]
    }
```

(Obsoleta) A política **`AWSOpsWorksRegisterCLI`**

> **Important**
>
> A política `AWSOpsWorksRegisterCLI` foi descontinuada e não pode ser usada para registrar novas instâncias. Ela está disponível somente para compatibilidade com versões anteriores em instâncias que já foram registradas. A política `AWSOpsWorksRegisterCLI` inclui várias permissões do IAM, incluindo `CreateUser`, `PutUserPolicy` e `AddUserToGroup`. Como essas são permissões no nível do administrador, você só deve atribuir a política `AWSOpsWorksRegisterCLI` aos usuários administrativos confiáveis.

Gerenciar instâncias registradas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

Quando você registra uma instância, ela se torna uma instância do AWS OpsWorks Stacks e você pode gerenciá-la da mesma forma que as instâncias criadas com o AWS OpsWorks Stacks. Existem duas diferenças principais:

- As instâncias registradas não precisam ser atribuídas a uma camada.
- É possível cancelar o registro de uma instância registrada e retorná-la ao seu controle direto.

Depois de registrar uma instância, ela fica no estado Registrado. AWS OpsWorks O Stacks fornece a seguinte funcionalidade de gerenciamento para todas as instâncias registradas:

- Health Checks — o AWS OpsWorks Stacks monitora o agente para avaliar se a instância continua funcionando.

  Se uma instância falhar em uma verificação de saúde, o AWS OpsWorks Stacks recupera automaticamente as EC2 instâncias registradas da Amazon e altera o status das instâncias locais registradas para. `connection lost`
- CloudWatch monitoramento — o CloudWatch monitoramento está habilitado para a instância registrada.

  É possível monitorar métricas como o uso da CPU e a memória disponível, além de receber como opção uma notificação se uma métrica ultrapassar um limite especificado.
- Gerenciamento de usuários — O AWS OpsWorks Stacks fornece uma maneira simples de especificar quais usuários podem acessar a instância e quais operações eles podem realizar. Para obter mais informações, consulte Gerenciamento de permissões de usuário.
- Execução de receitas: você pode usar o comando de pilha Executar receitas para executar receitas do Chef na instância.
- Atualizações do sistema operacional: você pode usar o comando de pilha Atualizar dependências para atualizar o sistema operacional da instância.

Para aproveitar ao máximo a funcionalidade de gerenciamento de AWS OpsWorks pilhas, você pode atribuir a instância a uma camada. Para obter mais informações, consulte Atribuir uma instância registrada a uma camada.

Há diferenças entre a forma como o AWS OpsWorks Stacks gerencia a Amazon EC2 e as instâncias locais.

EC2 Instâncias da Amazon

- Se você interromper uma EC2 instância registrada da Amazon, o AWS OpsWorks Stacks encerra as instâncias apoiadas pelo armazenamento de instâncias e interrompe as instâncias baseadas no Amazon EBS.

A instância ainda está registrada com a pilha e é atribuída às camadas para que você possa reiniciá-la, se necessário. Você deve cancelar o registro de uma instância registrada para removê-la de uma pilha, explicitamente ou ao excluir a instância, o que automaticamente cancela o registro.

- Se você reiniciar uma EC2 instância registrada da Amazon ou se a instância falhar e for recuperada automaticamente, o resultado será o mesmo que parar e reiniciar a instância usando a Amazon. EC2 Observe estas diferenças:
  - Instâncias baseadas em armazenamento de instâncias — o AWS OpsWorks Stacks inicia uma nova instância com a mesma AMI.

    Observe que o AWS OpsWorks Stacks não tem conhecimento de nenhuma operação que você executou na instância antes de ela ser registrada, como instalar pacotes de software. Se você quiser que o AWS OpsWorks Stacks instale pacotes ou execute outras tarefas de configuração na inicialização, você deve fornecer receitas personalizadas do Chef que executem as tarefas necessárias e as atribuam aos eventos de configuração das camadas apropriadas.
  - Instâncias suportadas pelo Amazon EBS — o AWS OpsWorks Stacks inicia uma nova instância com a mesma AMI e reconecta o volume raiz, o que restaura a instância à configuração anterior.
- Se você cancelar o registro de uma EC2 instância da Amazon registrada, ela voltará a ser uma instância normal da Amazon EC2 .

Instâncias on-premises

- AWS OpsWorks As pilhas não podem parar ou iniciar uma instância local registrada.

  Um evento Encerrar é acionado ao cancelar a atribuição de uma instância on-premises registrada. Entretanto, esse evento apenas executa as receitas de Encerrar das camadas atribuídas. Elas executam tarefas como encerrar serviços, mas não interrompem a instância.
- AWS OpsWorks As pilhas não podem reparar automaticamente uma instância local registrada se ela falhar, mas a instância é marcada como conexão perdida.
- As instâncias on-premises não podem usar o balanceador de carga elástica, o Amazon EBS ou os serviços de endereço IP elástico.

Atribuir uma instância registrada a uma camada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

Depois de registrar uma instância, você poderá atribuí-la a uma ou mais camadas. A vantagem de atribuir uma instância a uma camada em vez de deixá-la sem atribuição é que você pode atribuir receitas personalizadas aos eventos do ciclo de vida da camada. AWS OpsWorks Em seguida, as pilhas as executam automaticamente no momento apropriado, após as receitas da camada para esse evento.

- Você pode atribuir qualquer instância registrada a uma camada personalizada. Uma camada personalizada tem um conjunto mínimo de receitas que não instalam quaisquer pacotes, portanto, não devem criar conflitos com a configuração atual da instância.
- Você pode atribuir instâncias locais às camadas integradas do AWS OpsWorks Stacks.

  Cada camada integrada inclui receitas que instalam automaticamente um ou mais pacotes. Por exemplo, as receitas de Configuração Java App Server instalam Apache e Tomcat. As receitas da camada também pode executar outras operações como reiniciar serviços e implantar aplicativos. Antes de atribuir uma instância on-premises a uma camada interna, é necessário garantir que as receitas da camada não criem conflitos, como tentar instalar uma versão do servidor de aplicativo diferente da que está atualmente na instância. Para ter mais informações, consulte Camadas e AWS OpsWorks Referência de camadas de pilhas.

Como atribuir uma instância registrada a uma camada

1. Adicione as camadas que você deseja usar à pilha, se você ainda não tiver feito isso.

2. Escolha Instances (Instâncias) no painel de navegação e, em seguida, assign na coluna Actions (Ações) da instância.
3. Selecione as camadas apropriadas e escolha Save (Salvar).

Quando você atribui uma instância a uma camada, o AWS OpsWorks Stacks faz o seguinte.

- Executa Configurar receitas da camada.
- Adicionar qualquer endereço IP elástico ou volumes de Amazon EBS; aos recursos da pilha.

  Em seguida, você pode usar o AWS OpsWorks Stacks para gerenciar esses recursos. Para obter mais informações, consulte Gerenciamento de recursos.

Depois de concluídas, a instância fica no status on-line e é totalmente incorporada à pilha. AWS OpsWorks Em seguida, as pilhas executam as receitas atribuídas à camada sempre que ocorre um evento de ciclo de vida.

Cancelar a atribuição de uma instância registrada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este recurso é suportado somente para pilhas do Linux.

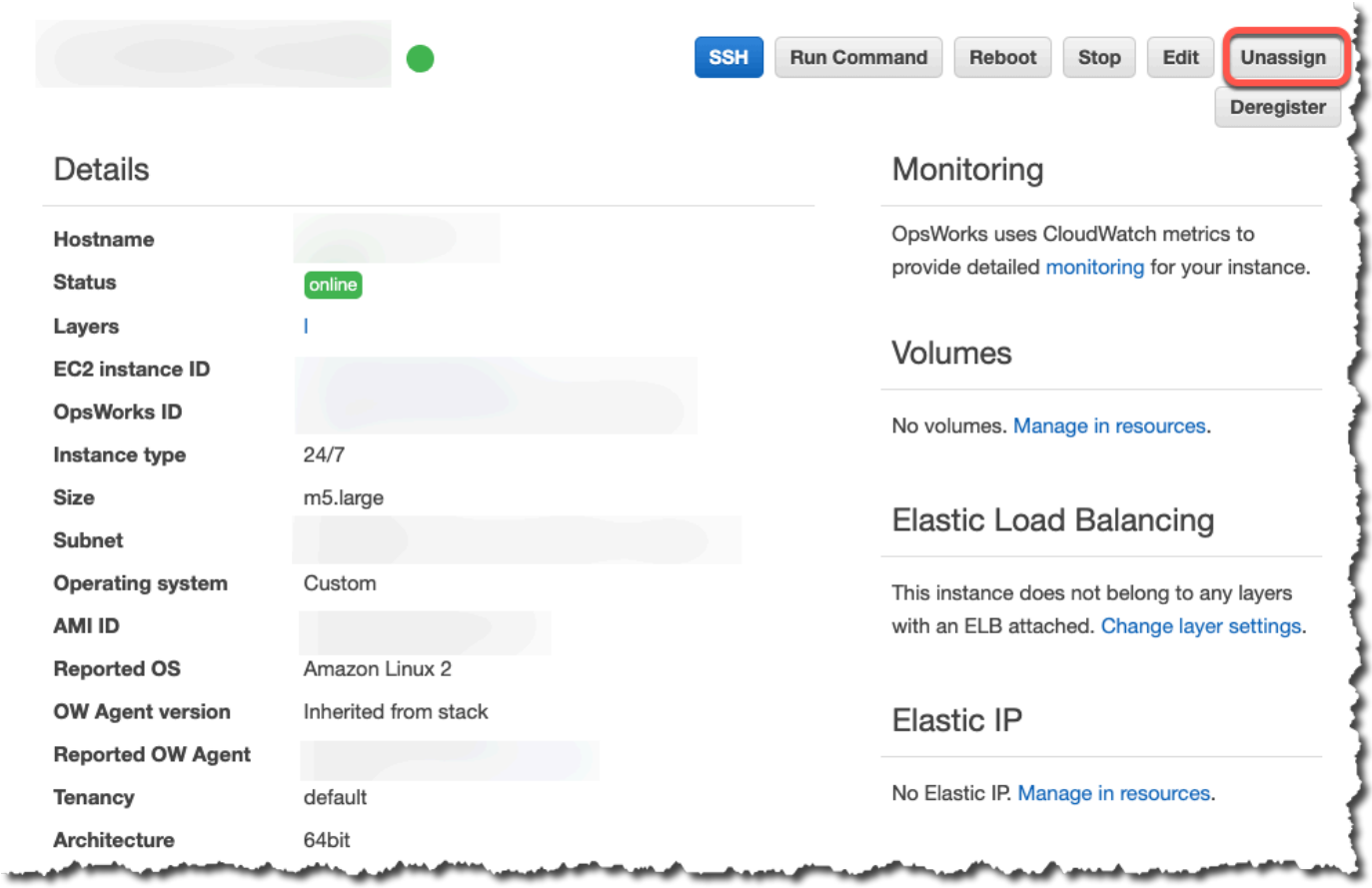
Você pode cancelar a atribuição de uma instância registrada de suas camadas usando o AWS OpsWorks console ou a AWS CLI operação do SDK.

Quando você cancela a atribuição de uma instância, o AWS OpsWorks Stacks executa as receitas de desligamento da camada na instância. Essas receitas executam tarefas como o encerramento

de serviços, mas não interrompem a instância. Se a instância for atribuída a diversas camadas, o cancelamento de atribuição se aplica a cada camada; não é possível cancelar a atribuição de uma instância a partir de um subconjunto de suas camadas. No entanto, a instância ainda está registrada com a pilha e você pode atribuí-la a outra camada, se desejar.

Para cancelar a atribuição de uma instância registrada usando o console

1. No painel de navegação, escolha Instances (Instâncias).
2. Escolha a instância que você deseja cancelar a atribuição.
3. Na página Detalhes da instância, escolha Cancelar atribuição.



Para cancelar a atribuição de uma instância registrada usando o AWS CLI

Execute o comando aws opsworks unassign-instance para cancelar a atribuição de uma instância registrada de todas as camadas que estão usando a instância.

```
aws opsworks unassign-instance --region region --instance-id instance-id
```

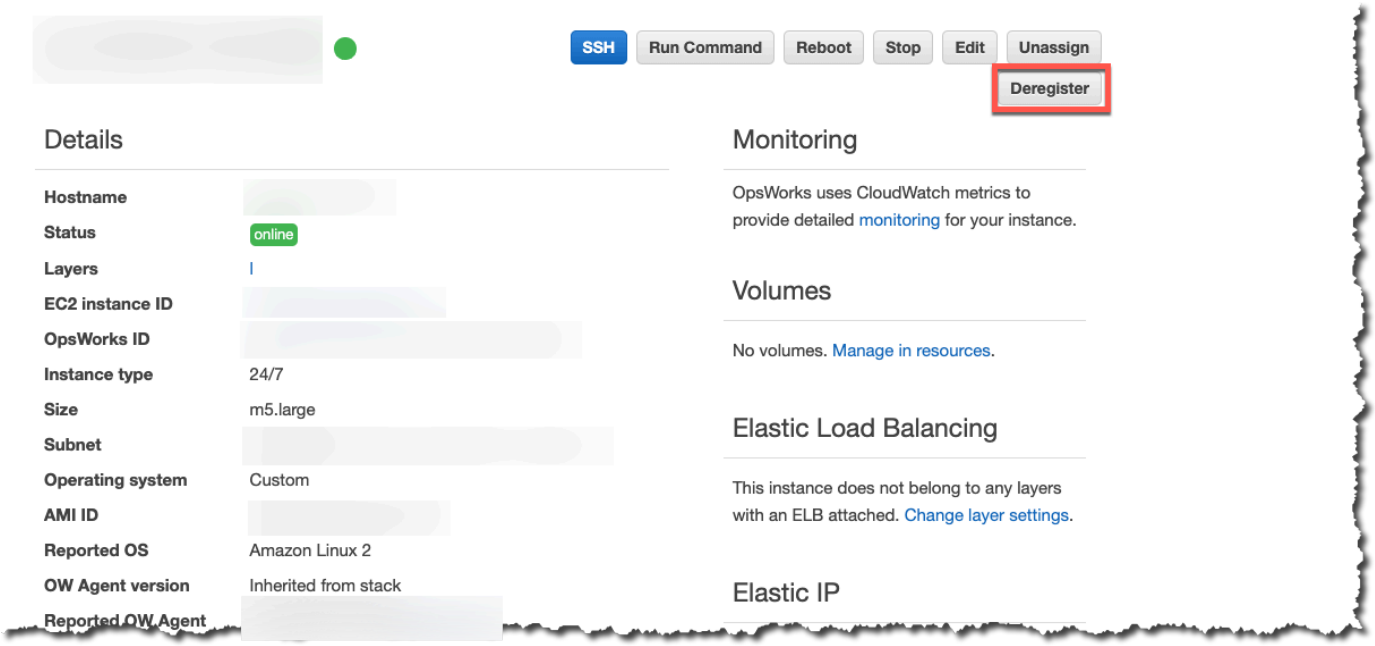
Cancelar o registro de uma instância registrada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode cancelar o registro de uma instância usando o AWS OpsWorks console ou a operação AWS CLI do SDK.

Para cancelar o registro de uma instância usando o console

1. No painel de navegação, escolha Instances (Instâncias).
2. Escolha a instância que você deseja cancelar o registro.
3. Na página Detalhes da instância, escolha Cancelar registro.



Para cancelar o registro de uma instância usando o AWS CLI

Execute o comando aws opsworks deregister-instance para cancelar o registro de uma instância da pilha.

```
aws opsworks deregister-instance --region region --instance-id instance-id
```

Quando você cancela o registro de uma instância, o AWS OpsWorks Stacks faz o seguinte:

- Remove a instância da pilha.
- Cancela a atribuição da instância de qualquer camada atribuída.
- Encerra e desinstala o agente.
- Cancela o registro de qualquer recurso vinculado (endereços IP elástico e volumes de Amazon EBS).

  Esse procedimento inclui recursos que foram anexados à instância antes do registro e recursos que você usou AWS OpsWorks pilhas para anexar à instância enquanto ela fazia parte da pilha. Após o cancelamento do registro, os recursos não fazem mais parte dos recursos da pilha, mas permanecem vinculados à instância.
- Para instâncias on-premises, interrompe as cobranças.
- Remove todas as tags OpsWorks adicionadas à instância.

A instância permanece em execução, mas está sob seu controle direto e não é mais gerenciada pelo AWS OpsWorks Stacks.

> **Note**
>
> Tanto o registro quanto o cancelamento de registro de computadores ou instâncias só podem ser realizados totalmente nas pilhas do Linux. Para pilhas do Windows, o cancelamento do registro de instâncias é permitido, mas isso não desinstala o OpsWorks agente da instância. O cancelamento do registro não remota todos os arquivos alterados e não reverte completamente para cópias em backup de alguns arquivos. A lista se aplica às pilhas do Chef 11.10 e Chef 12; as diferenças entre as duas versões são anotadas aqui.
>
> - O backup de `/etc/hosts` é salvo em `/var/lib/aws/opsworks/local-mode-cache/backup/etc/`, mas não é restaurado.
> - As entradas permanecem para `aws` e `opsworks` em senha, grupo e arquivos sombra etc.
> - `/etc/sudoers`contém uma referência a um diretório AWS OpsWorks Stacks.

- Os arquivos a seguir podem ser deixados; a longo prazo, considere excluir `/var/lib/aws/opsworks`.
  - `/var/log/aws/opsworks` permanece em instância nas pilhas do Chef 11.10.
  - `/var/lib/aws/opsworks` permanece nas pilhas do Chef 11.10 e Chef 12.
  - `/var/chef` permanece em instância nas pilhas do Chef 12.
- Outros arquivos abandonados:
  - `/etc/logrotate.d/opsworks-agent`
  - `/etc/cron.d/opsworks-agent-updater`
  - `/etc/ld.so.conf.d/opsworks-user-space.conf`
  - `/etc/motd.opsworks-static`
  - `/etc/aws/opsworks`
  - `/etc/sudoers.d/opsworks`
  - `/etc/sudoers.d/opsworks-agent`

Ciclo de vida de instância registrada

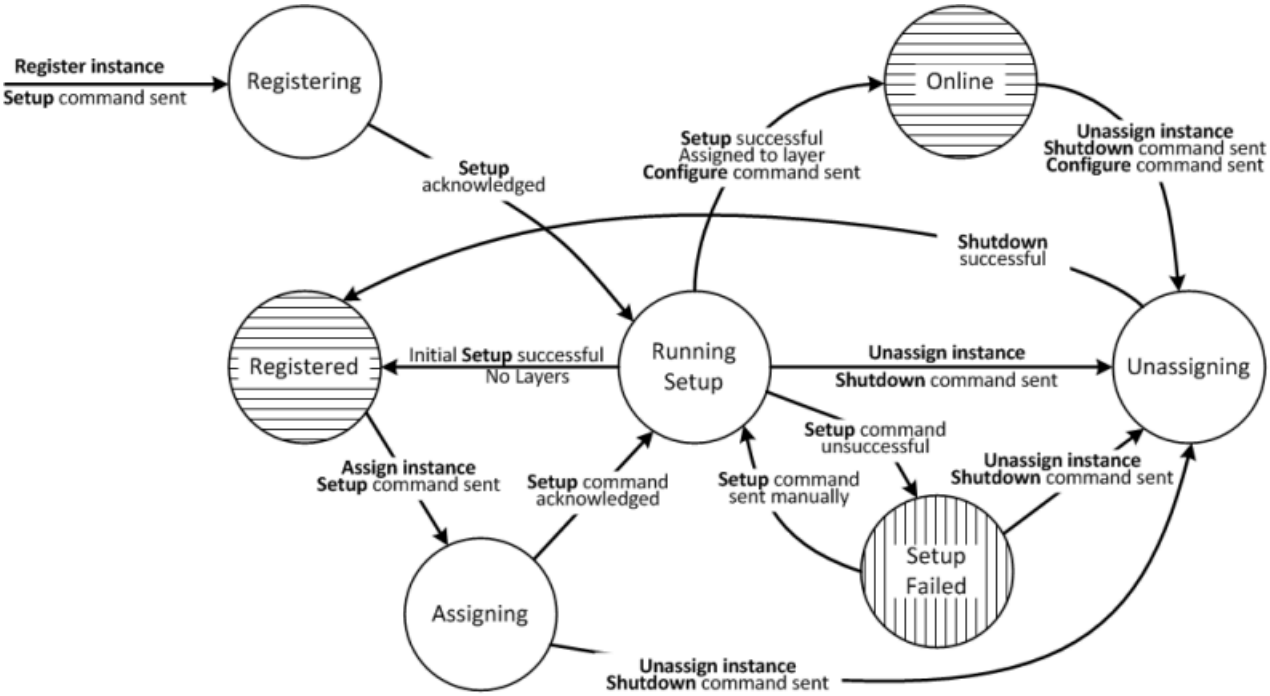
Important

O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Note

Este recurso é suportado somente para pilhas do Linux.

O ciclo de vida da instância registrada inicia depois que o agente é instalado e está em execução. Nesse ponto, ele direciona AWS OpsWorks Stacks a registrar a instância com a pilha. O diagrama de estado a seguir resume os principais elementos de ciclo de vida.



Cada estado corresponde a um status de instância. As bordas representam um dos seguintes comandos do AWS OpsWorks Stacks. Os detalhes são discutidos nas seguintes seções.

- Instalação: este comando corresponde ao evento de ciclo de vida de instalação e executa as Receitas de instalação da instância.
- Configurar: esse comando corresponde ao evento de ciclo de vida Configurar.

  AWS OpsWorks As pilhas acionam esse evento em todas as instâncias da pilha quando uma instância entra ou sai do estado on-line. As instâncias executam Configurar receitas, que faz alterações necessárias para acomodar a nova instância.
- Encerrar: este comando corresponde ao evento de ciclo de vida Encerrar, que executa Encerrar receitas da instância.

  Essas receitas executam tarefas como encerrar serviços, mas não interrompem a instância.
- Cancelar registro : esse comando cancela o registro de uma instância e não corresponde a um evento de ciclo de vida.

> **Note**
>
> Para simplificar, o diagrama não exibe os estados Cancelar registro e Excluído. Você pode cancelar o registro de uma instância a partir de qualquer um dos estados no diagrama, que

envia um comando Cancelar o registro para a instância e o move para o estado Cancelar o registro.

- Se você cancelar o registro de uma instância on-line, o AWS OpsWorks Stacks enviará um comando Configure para as instâncias restantes na pilha para notificá-las de que a instância está ficando off-line.
- Depois de confirmar o comando Cancelar registro, a instância ainda está funcionando, mas está no estado Excluído e não faz mais parte da pilha. Se você deseja incorporar a instância à pilha novamente, é necessário registrá-la novamente.

Tópicos

- Registrar
- Execução de configuração
- Registrado
- Atribuição
- Online
- Falha na configuração
- Cancelar atribuição
- Alterações de configuração inicial

Registrar

Depois que o agente envia uma solicitação de registro, o AWS OpsWorks Stacks inicia o ciclo de vida da instância enviando um comando de configuração para a instância, colocando-a no estado Registro. Depois que a instância reconhecer o comando Configuração, ela entra no estado Execução de configuração.

Execução de configuração

O estado Execução de configuração executa Configurar receitas da instância. A configuração funcionará dependendo do estado anterior.

> **Note**
>
> Se você cancelar a atribuição da instância enquanto ela estiver no estado Running Setup, o AWS OpsWorks Stacks enviará um comando Shutdown, que executa as receitas de

desligamento da instância, mas não interrompe a instância. O instância entrará no estado Cancelar atribuição.

Tópicos

- Registrar
- Atribuição
- Falha na configuração

Registrar

Durante o processo de registro, a configuração cria uma instância do AWS OpsWorks Stacks para representar a instância registrada na pilha e executa um conjunto de receitas principais de configuração na instância.

Uma alteração importante executada pela configuração inicial é substituir o arquivo de hosts da instância. Ao registrar a instância, você passa o gerenciamento do usuário para o AWS OpsWorks Stacks, que deve ter seu próprio arquivo de hosts para controlar as permissões de login SSH. A configuração inicial também cria ou modifica diversos arquivos e, em sistema Ubuntu, modifica as origens do pacote e instala um conjunto de pacotes. Para obter detalhes, consulte Alterações de configuração inicial.

Durante o registro, o processo chama a `AttachUserPolicy` do IAM, que faz parte das permissões anexadas ao usuário do IAM que você cria como um pré-requisito. Se `AttachUserPolicy` não existir (provavelmente porque você está executando uma versão mais antiga da CLI da AWS), o processo passará a chamar `PutUserPolicy`.

> **Note**
>
> Para manter a consistência, o AWS OpsWorks Stacks executa todas as receitas básicas de configuração. Entretanto, alguns deles executam algumas ou todas as suas tarefas somente se uma instância foi atribuída a pelo menos uma camada para não afetarem necessariamente a configuração inicial.

- Se a configuração for bem-sucedida, a instância muda para o estado Registrado.
- Se a configuração não for bem-sucedida, a instância muda para o estado Falha na configuração.

Atribuição

A instância tem pelo menos uma camada atribuída. AWS OpsWorks O Stacks executa as receitas de configuração de cada camada, incluindo todas as receitas personalizadas que você atribuiu ao evento de configuração das camadas.

- Se a configuração for bem-sucedida, a instância entra no estado Online e o AWS OpsWorks Stacks aciona um evento de ciclo de vida Configurar em cada instância na pilha para notificá-los sobre a nova instância.
- Se a configuração não for bem-sucedida, a instância muda para o estado Falha na configuração.

> **Note**
>
> Esse processo de configuração executa as principais receitas uma segunda vez. Entretanto, as receitas do Chef são idempotentes, portanto, elas não repetem tarefas já executadas.

Falha na configuração

Se ocorrer uma falha no processo de configuração de uma instância no estado Atribuição, é possível tentar novamente ao usar o comando de pilha Configuração para executar manualmente Configurar receitas da instância.

- Se a configuração for bem-sucedida, a instância atribuída entra no estado Online e o AWS OpsWorks Stacks aciona um evento de ciclo de vida Configure em cada instância na pilha para notificá-los sobre a nova instância.
- Se a tentativa de configuração não for bem-sucedida, a instância muda para o estado de Falha na configuração.

Registrado

As instâncias no estado Registrado fazem parte da pilha e são gerenciadas pelas AWS OpsWorks Pilhas, mas não são atribuídas a uma camada. Elas podem permanecer nesse estado indefinidamente.

Se você atribuir a instância a uma ou mais camadas, o AWS OpsWorks Stacks enviará um comando de configuração para a instância e ela será movida para o Atribuição estado.

Atribuição

Depois que a instância reconhecer o comando Configuração, ela entra no estado Execução de configuração.

Se você cancelar a atribuição da instância enquanto ela estiver no estado Atribuição, o AWS OpsWorks Stacks encerrará o processo de configuração e enviará um comando Shutdown. O instância entrará no estado Cancelar atribuição.

Online

A instância agora é um membro de pelo menos uma camada e é tratada como uma instância regular do AWS OpsWorks Stacks. Ela pode permanecer nesse estado indefinidamente.

Se você cancelar a atribuição da instância enquanto ela estiver no estado Online, o AWS OpsWorks Stacks enviará um comando Shutdown para a instância e um comando Configure para o resto das instâncias da pilha. O instância entrará no estado Cancelar atribuição.

Falha na configuração

Ocorreu um erro no comando Configuração.

- Você pode tentar novamente ao executar o comando de pilha Configuração.

  A instância retorna para o estado Execução de configuração.
- Se você cancelar a atribuição da instância, o AWS OpsWorks Stacks enviará um comando Shutdown para a instância.

  O instância entrará no estado Cancelar atribuição.

Cancelar atribuição

Após a conclusão do comando Encerramento, a instância não está mais atribuída a quaisquer layers e retorna para o estado Registrado.

> **Note**
>
> Se uma instância for atribuída a várias camadas, o cancelamento da atribuição se aplica a todas as camadas; não é possível cancelar a atribuição de um subconjunto de camadas

atribuídas. Se você deseja um conjunto diferente de camadas atribuídas, cancele a atribuição da instância e reatribua as camadas desejadas.

Alterações de configuração inicial

A configuração inicial cria ou modifica os seguintes arquivos e diretórios registrados em todas as instâncias.

Arquivos criados

```
/etc/apt/apt.conf.d/99-no-pipelining
/etc/aws/
/etc/init.d/opsworks-agent
/etc/motd
/etc/motd.opsworks-static
/etc/sudoers.d/opsworks
/etc/sudoers.d/opsworks-agent
/etc/sysctl.d/70-opsworks-defaults.conf
/opt/aws/opsworks/
/usr/sbin/opsworks-agent-cli
/var/lib/aws/
/var/log/aws/
/vol/
```

Arquivos modificados

```
/etc/apt/apt.conf.d/99-no-pipelining
/etc/crontab
/etc/default/monit
/etc/group
/etc/gshadow
/etc/monit/monitrc
/etc/passwd
/etc/security/limits.conf (removing limits only for EC2 micro instances)
/etc/shadow
/etc/sudoers
```

A configuração inicial também cria um arquivo de troca nas EC2 microinstâncias da Amazon.

A configuração inicial faz as seguintes alterações em sistemas Ubuntu.

Origens de pacote

A configuração inicial altera as fontes de pacote para o seguinte.

- `deb http://archive.ubuntu.com/ubuntu/ ${code_name} main universe`

  Para: `deb-src http://archive.ubuntu.com/ubuntu/ ${code_name} main universe`
- `deb http://archive.ubuntu.com/ubuntu/ ${code_name}-updates main universe`

  Para: `deb-src http://archive.ubuntu.com/ubuntu/ ${code_name}-updates main universe`
- `deb http://archive.ubuntu.com/ubuntu ${code_name}-security main universe`

  Para: `deb-src http://archive.ubuntu.com/ubuntu ${code_name}-security main universe`
- `deb http://archive.ubuntu.com/ubuntu/ ${code_name}-updates multiverse`

  Para: `deb-src http://archive.ubuntu.com/ubuntu/ ${code_name}-updates multiverse`
- `deb http://archive.ubuntu.com/ubuntu ${code_name}-security multiverse`

  Para: `deb-src http://archive.ubuntu.com/ubuntu ${code_name}-security multiverse`
- `deb http://archive.ubuntu.com/ubuntu/ ${code_name} multiverse`

  Para: `deb-src http://archive.ubuntu.com/ubuntu/ ${code_name} multiverse`
- `deb http://security.ubuntu.com/ubuntu ${code_name}-security multiverse`

  Para: `deb-src http://security.ubuntu.com/ubuntu ${code_name}-security multiverse`

Pacotes

A configuração inicial desinstala `landscape` e instalar os seguintes pacotes.

`autofs`	`libicu-dev`	`libopenssl-ruby`

libssl-dev	libxml2-dev	libxslt-dev
libyaml-dev	monit	ntpd
procps	ruby	ruby-dev
rubygems	screen	sqlite
vim	xfs	

Edição da configuração da instância

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode editar configurações de instância, incluindo instâncias registradas do Amazon Elastic Compute Cloud EC2 (Amazon), com as seguintes limitações:

- A instância deve estar no estado interrompido.

  Embora você não possa modificar as propriedades de uma instância online, pode alterar alguns aspectos da configuração editando os camadas da instância. Para obter mais informações, consulte Como editar a configuração de uma OpsWorks camada.
- Algumas configurações, como Availability Zone e Scaling Type, são determinadas quando você cria a instância e não podem ser modificadas posteriormente.
- Algumas configurações podem ser modificadas somente para as instâncias baseadas no armazenamento de instância, não para as instâncias baseadas no Amazon Elastic Block Store.

  Por exemplo, você pode alterar o sistema operacional de uma instância baseada no armazenamento de instância. As instâncias baseadas no Amazon EBS devem usar o sistema

operacional que você especificou quando criou a instância. Para obter mais informações sobre o armazenamento de instâncias, consulte Armazenamento.

- Por padrão, as instâncias herdam a configuração da versão do agente da pilha.

  Você pode usar a versão do OpsWorks agente para substituir a configuração da versão do agente da pilha e especificar uma versão específica do agente para uma instância. Se você especificar a versão do agente de uma instância, o AWS OpsWorks Stacks não atualizará automaticamente o agente quando uma nova versão estiver disponível, mesmo que a configuração da versão do agente da pilha seja Atualização automática. Você deve atualizar a versão do agente da instância manualmente editando a configuração da instância. AWS OpsWorks Em seguida, o Stacks instala a versão especificada do agente na instância.

> **Note**
>
> Você não pode editar a configuração de instâncias no local registradas.

Para editar a configuração de uma instância

1. Interrompa a instância, se ela ainda não tiver parado.
2. Na página Instances, clique no nome do load balancer para abrir sua página Details.
3. Clique em Edit para exibir a página de edição.
4. Edite a configuração da instância, conforme apropriado.

Para uma descrição das configurações Host name, Size, SSH key e Operating system, consulte Adicionar uma instância a uma camada. A configuração Layers permite que você adicione ou remova camadas. A camada atual da instância aparece seguindo a lista de camadas.

- Para adicionar outra camada, selecione-a na lista.
- Para remover a instância de uma das suas camadas, clique no x perto da camada apropriada.

  Uma instância deve ser membro de pelo menos uma camada, por isso não é possível remover a última camada.

Quando você reinicia a instância, o AWS OpsWorks Stacks inicia uma nova EC2 instância da Amazon com a configuração atualizada.

Excluindo instâncias do AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode usar o AWS OpsWorks Stacks para interromper uma instância, incluindo EC2 instâncias registradas da Amazon. Isso interrompe a EC2 instância, mas ela permanece na pilha. Você pode reiniciá-la clicando em start (iniciar) na coluna Actions (Ações) da instância. Se você não precisar mais de uma instância e quiser removê-la da pilha, você pode excluí-la, o que remove a instância da pilha e encerra a instância associada da Amazon. EC2 A exclusão de uma instância também exclui os logs ou os dados associados e todos os volumes do Amazon Elastic Block Store (EBS) na instância.

> **Important**
>
> Este tópico se aplica somente às EC2 instâncias da Amazon que são gerenciadas pelo AWS OpsWorks Stacks. Para obter mais informações sobre como excluir instâncias gerenciadas pelo EC2 console ou pela API da Amazon, consulte Encerrar sua instância.

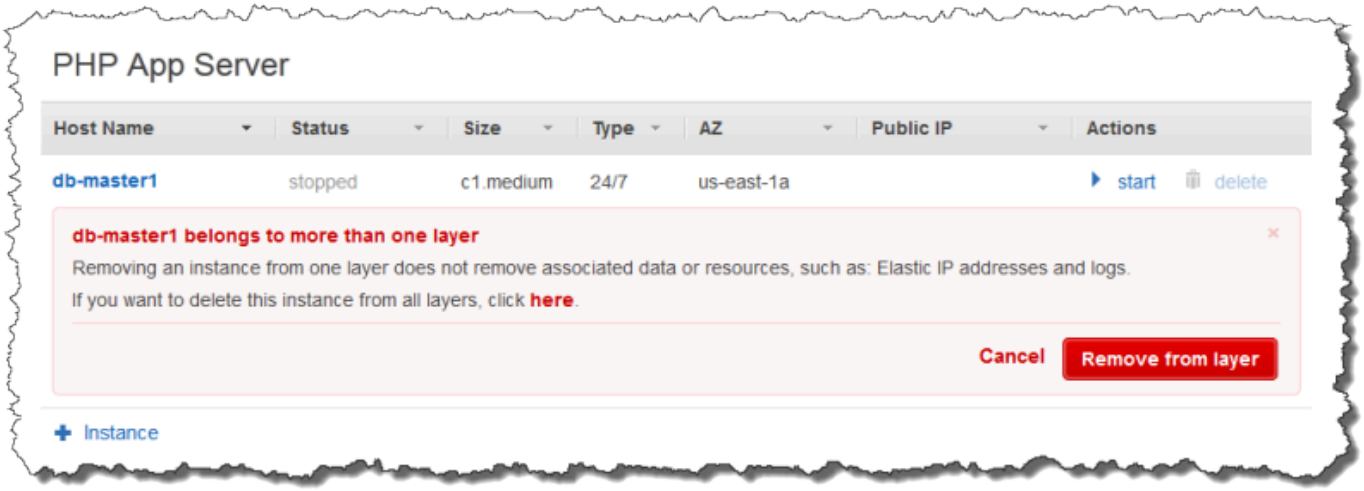
> **Note**
>
> Você não pode usar o AWS OpsWorks Stacks para excluir uma instância local registrada.

Caso uma instância pertença a várias layers, você pode excluí-la da pilha ou apenas remover camadas determinada layer. Você também pode remover camadas de instâncias editando a configuração da instância, conforme descrito em Edição da configuração da instância.

> **Important**
>
> Você deve excluir instâncias do AWS OpsWorks Stacks somente usando o console ou a AWS OpsWorks API do Stacks. Em particular, você não deve excluir instâncias do AWS OpsWorks Stacks usando o EC2 console ou a API da Amazon porque EC2 as ações da Amazon não são sincronizadas automaticamente com AWS OpsWorks o Stacks. Por exemplo, se a recuperação automática estiver ativada e você encerrar uma instância usando o EC2 console da Amazon, o AWS OpsWorks Stacks tratará a instância encerrada como uma instância com falha e iniciará outra instância da Amazon para EC2 substituí-la. Para obter mais informações, consulte Como usar a correção automática.

Para excluir uma instância

1. Na página Instances, localize a instância na camada apropriada. Caso a instância esteja em execução, clique em stop na coluna Actions.
2. Depois que o jogo for concluído, o status mudará para stopped, clique em delete. Se a instância for membro de mais de uma camada, Layer AWS OpsWorks Stacks exibirá a seção a seguir.



   - Para remover a instância somente de uma camada selecionada, clique em Remove from layer.

     A instância continua sendo um membro das outras camadas e pode ser reiniciada.
   - Para excluir a instância de todas as camadas, o que a remove da pilha, clique here.
3. Se você optar por remover completamente uma instância da pilha ou se a instância for membro de apenas uma camada, o AWS OpsWorks Stacks solicitará que você confirme a exclusão.

Escolha Delete para confirmar. Além de excluir a instância da pilha, essa ação exclui todos os logs ou dados associados, além de volumes raiz anexados à instância. Para remover todos os volumes da instância, escolha Excluir volumes do EBS da instância (os snapshots não serão excluídos) antes de escolher Excluir.

Uso do SSH para fazer login em uma instância Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode fazer login em suas instâncias Linux on-line com SSH usando o MindTerm cliente integrado ou um cliente de terceiros, como o PuTTY. Normalmente, o SSH depende de um par de chaves RSA para autenticação. Você instala a chave pública na instância e fornece a chave privada correspondente ao cliente SSH. AWS OpsWorks O Stacks gerencia a instalação de chaves públicas nas instâncias da sua pilha para você, da seguinte maneira.

- Par de chaves Amazon Elastic Compute Cloud (Amazon EC2) — Se a região da pilha tiver um ou mais pares de EC2 chaves da Amazon, você poderá especificar um par de chaves SSH padrão para a pilha.

  Como opção, você pode substituir o par de chaves padrão e especificar um par diferente ao criar uma instância. Em ambos os casos, o AWS OpsWorks Stacks instala a chave pública do par de chaves especificado na instância. Para obter mais informações sobre como criar pares de EC2 chaves da Amazon, consulte Amazon EC2 Key Pairs.
- Par de chaves pessoais: cada usuário pode registrar um par de chaves pessoais no AWS OpsWorks Stacks.

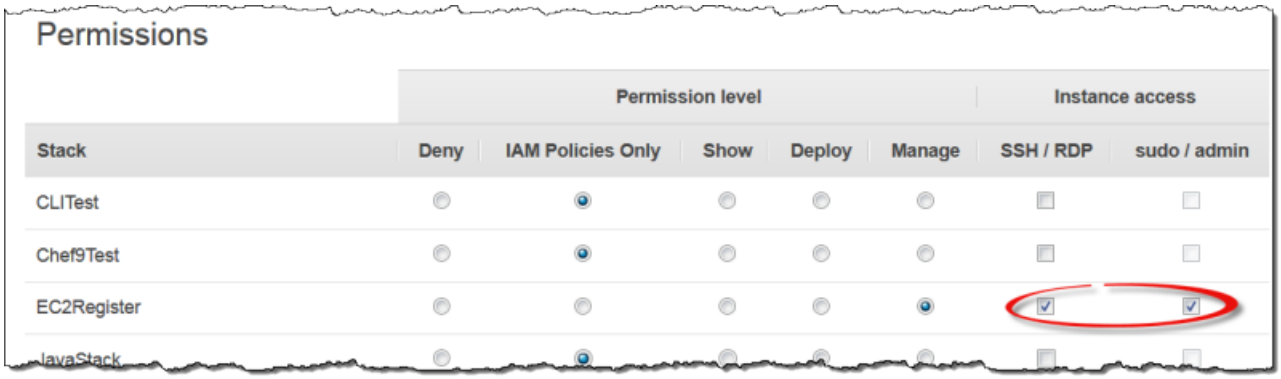
  O usuário ou administrador registra a chave pública no AWS OpsWorks Stacks, e o usuário armazena a chave privada localmente. Ao definir permissões para uma pilha, o administrador especifica que os usuários devem ter acesso SSH às instâncias da pilha. AWS OpsWorks O

Stacks cria automaticamente um usuário do sistema nas instâncias da pilha para cada usuário autorizado e instala a chave pública pessoal dos usuários.

Um usuário deve ter autorização SSH para usar o cliente MindTerm SSH ou usar seu par de chaves pessoais para fazer login nas instâncias de uma pilha.

Para autorizar o SSH para um usuário

1. No painel de navegação AWS OpsWorks Stacks, clique em Permissões.
2. Selecione SSH/RDP para o usuário do IAM desejado a fim de conceder as permissões necessárias. Se você deseja permitir que o usuário utilize o `sudo` para elevar privilégios, por exemplo, para executar comandos de CLI do agente, selecione sudo/admin também.



Para obter mais informações sobre como usar o AWS OpsWorks Stacks para gerenciar o acesso SSH, consulte. Gerenciamento do acesso por SSH

Tópicos

- Usando o cliente MindTerm SSH integrado
- Uso de um cliente SSH de terceiros

Usando o cliente MindTerm SSH integrado

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

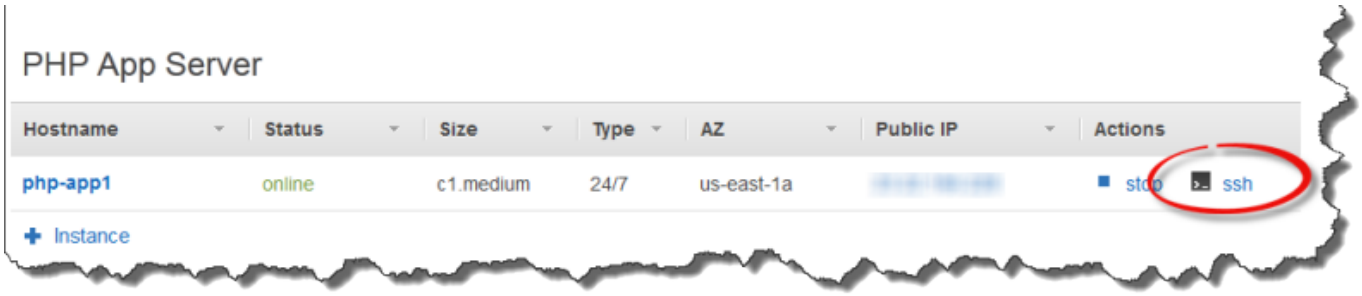
dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A maneira mais simples de fazer login em uma instância Linux é usar o cliente MindTerm SSH integrado. Cada instância on-line inclui uma ação ssh que você pode usar para iniciar o MindTerm cliente.

> **Note**
>
> Você deve ter o Java habilitado em seu navegador para usar o MindTerm cliente.

Para fazer login com o MindTerm cliente

1. Se você ainda não tiver feito isso, autorize o acesso SSH para o usuário do IAM que estabelecerá conexão com a instância, conforme descrito na seção anterior.
2. Faça login como o usuário.
3. Na página Instances, escolha ssh na coluna Actions da instância apropriada.



4. Em Chave privada, forneça um caminho para a chave privada pessoal do usuário ou uma chave EC2 privada da Amazon, dependendo das chaves públicas que você instalou na instância.
5. Escolha Launch Mindterm e use a janela do terminal para executar comandos na instância.

Uso de um cliente SSH de terceiros

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você também pode usar um cliente SSH de terceiros, como o PuTTY, para estabelecer conexão com instâncias Linux.

Para usar um cliente SSH de terceiros

1. Certifique-se de que o AWS OpsWorks Stacks tenha instalado uma chave EC2 pública da Amazon ou uma chave pública pessoal do usuário do IAM na instância, conforme discutido anteriormente.
2. Obtenha o nome DNS público da instância ou o endereço IP público da página de detalhes dessa instância.
3. Forneça ao cliente o nome de host da instância, que depende do sistema operacional, da seguinte forma:

   - Amazon Linux e Red Hat Enterprise Linux (RHEL): `ec2-user@`*`DNSName/Address`*.
   - Ubuntu: `ubuntu@`*`DNSName/Address`*.

   *`DNSName/Address`*Substitua pelo nome DNS público ou endereço IP da etapa anterior.
4. Forneça ao cliente uma chave privada que corresponda a uma chave pública instalada. Você pode usar uma chave EC2 privada da Amazon ou a chave privada pessoal de um usuário do IAM, dependendo de quais chaves públicas foram instaladas na instância.

Uso do RDP para fazer login em uma instância Windows

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode usar o protocolo de área de trabalho remota (RDP) do Windows para fazer login em uma instância Windows online, da seguinte forma:

- A instância deve ter um grupo de segurança com uma regra de entrada que permita o acesso por RDP.

  Para obter mais informações sobre como trabalhar com grupos de segurança, consulte Usar grupos de segurança.
- Usuários comuns — O AWS OpsWorks Stacks fornece aos usuários comuns autorizados uma senha RDP válida por um período limitado, que pode variar de 30 minutos a 12 horas.

  Além de serem autorizados, os usuários devem ter pelo menos um nível de permissão Mostrar ou suas políticas anexadas AWS Identity and Access Management (IAM) devem permitir a `opsworks:GrantAccess` ação.
- Administradores: você pode usar a senha de administrador para fazer login por quanto tempo quiser.

  Conforme descrito posteriormente, se você especificou um par de chaves do Amazon Elastic Compute Cloud (Amazon EC2) para a instância, você pode usá-lo para recuperar a senha do administrador.

> **Note**
>
> Este tópico descreve como usar o cliente de Conexão da área de trabalho remota do Windows para fazer login em uma estação de trabalho Windows. Você também pode usar um dos clientes RDP disponíveis para Linux ou OS X, mas o procedimento pode ser um pouco diferente. Para obter mais informações sobre clientes RDP compatíveis com Microsoft Windows Server 2012 R2, consulte Clientes da Área de trabalho remota da Microsoft.

Tópicos

- Fornecimento de um grupo de segurança que permite o acesso RDP
- Login como um usuário comum
- Login como administrador

Fornecimento de um grupo de segurança que permite o acesso RDP

Antes de usar o RDP para se conectar a uma instância Windows, as regras de entrada do grupo de segurança da instância devem permitir conexões RDP. Ao criar a primeira pilha em uma região, o AWS OpsWorks Stacks cria um conjunto de grupos de segurança. Eles incluem um chamado algo como`AWS-OpsWorks-RDP-Server`, que o AWS OpsWorks Stacks anexa a todas as instâncias do Windows para permitir acesso ao RDP. No entanto, por padrão, esse grupo de segurança não tem regras. Assim, você deve adicionar uma regra de entrada para permitir o acesso RDP para suas instâncias.

Para permitir o acesso RDP

1. Abra o EC2console da Amazon, configure-o para a região da pilha e escolha Security Groups no painel de navegação.
2. Selecione AWS- OpsWorks -RDP-Server, escolha a guia Entrada e escolha Editar.
3. Selecione Add Rule e especifique as seguintes configurações:
   - Tipo: RDP
   - Origem: os endereços IP de origem permitidos.

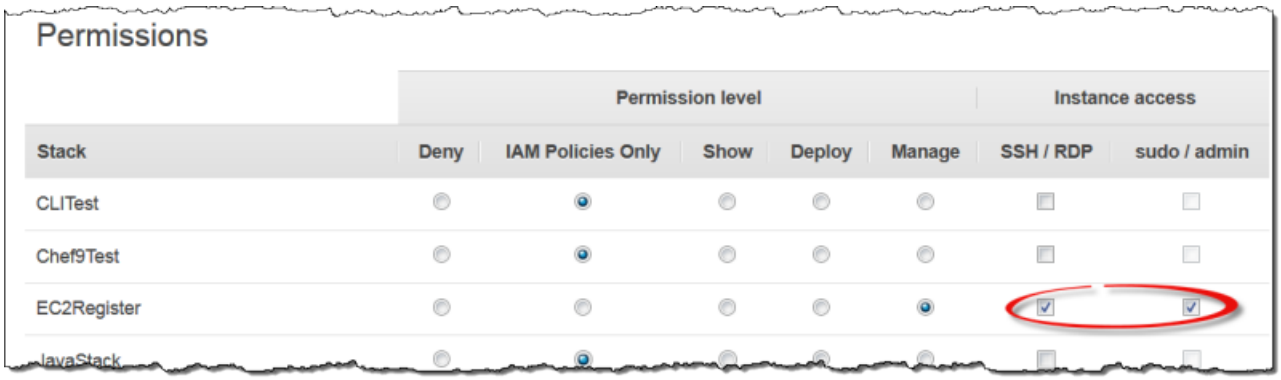
     Normalmente, você permite solicitações RDP de entrada de seu endereço IP ou de uma faixa especificada de endereços IP (normalmente a faixa de endereços IP de sua empresa).

Login como um usuário comum

Um usuário autorizado pode fazer login em instâncias usando uma senha temporária, fornecida pelo AWS OpsWorks Stacks.

Para autorizar o RDP para um usuário;

1. No painel de navegação AWS OpsWorks Stacks, clique em Permissões.
2. Marque a caixa de seleção SSH/RDP do usuário do desejado para conceder as permissões necessárias. Se você deseja que o usuário tenha permissões de administrador, também é necessário selecionar sudo/admin.



Os usuários autorizados podem fazer login em qualquer uma das instâncias online da pilha, do modo a seguir.

Para fazer login como um usuário do IAM comum

1. Faça login como um usuário do IAM.
2. Na página Instances, escolha rdp na coluna Actions da instância apropriada.
3. Especifique a duração da sessão, que pode variar de 30 minutos a 12 horas, e escolha Generate Password. A senha será válida apenas pela duração especificada para a sessão.
4. Registre os valores public DNS name, username e password e, em seguida, selecione Acknowledge and close.
5. Abra o cliente de Conexão da área de trabalho remota do Windows, escolha Show Options e forneça o seguinte a partir das informações que você anotou na Etapa 4:

   - Computador: o nome DNS público da instância.

     Você também pode usar o endereço IP público, se preferir. Escolha Instances e copie o endereço da coluna Public IP da instância.
   - Nome de usuário: o nome de usuário.
6. Quando o cliente solicitar suas credenciais, digite a senha que você salvou na Etapa 4.

> **Note**
>
> AWS OpsWorks O Stacks gera uma senha de usuário somente para instâncias on-line. Se você iniciar uma instância e, por exemplo, uma de suas receitas de configuração personalizada falhar, a instância ficará no estado `setup_failed`. Mesmo que a instância não esteja on-line no que diz respeito ao AWS OpsWorks EC2 Stacks, ela está em execução

e geralmente é útil fazer login para solucionar o problema. AWS OpsWorks Nesse caso, as pilhas não gerarão uma senha para você, mas se você tiver atribuído um key pair SSH à instância, poderá usar o EC2 console ou a CLI para recuperar a senha de administrador da instância e fazer login como administrador. Para obter mais informações, consulte a seção a seguir.

Login como administrador

Você pode fazer login em uma instância como administrador usando a senha apropriada. Se você atribuiu um EC2 key pair a uma instância, a Amazon o EC2 usa para criar e criptografar automaticamente uma senha de administrador quando a instância é iniciada. Em seguida, você pode usar a chave privada do par de chaves com o EC2 console, a API ou a CLI para recuperar e descriptografar a senha.

> **Note**
>
> Você não pode usar um par de chaves SSH pessoal para recuperar uma senha de administrador; você deve usar um par de EC2 chaves.

A seguir, descrevemos como usar o EC2 console para recuperar uma senha de administrador e fazer login em uma instância. Se você preferir as ferramentas da linha de comando, também pode usar o comando `get-password-data` da CLI da AWS para recuperar a senha.

Para fazer login como administrador

1. Verifique se você especificou um par de EC2 chaves para a instância. Você pode especificar um par de chaves padrão para todas as instâncias da pilha ao criá-la, ou pode especificar um par de chaves para uma determinada instância ao criar essa instância.
2. Abra o EC2 console, configure-o para a região da pilha e escolha Instâncias no painel de navegação.
3. Selecione a instância, escolha Connect e selecione Get Password.
4. Forneça um caminho para a EC2 chave privada do par de chaves em sua estação de trabalho e escolha Descriptografar senha. Copie a senha descriptografada para uso posterior.
5. Abra o cliente de Conexão de Área de Trabalho Remota do Windows, escolha Show Options e forneça as seguintes informações:

- Computador: o nome DNS público ou endereço IP público da instância, que podem ser obtidos na página de detalhes da instância.
- Nome de usuário: `Administrator`.

6. Quando o cliente solicitar suas credenciais, forneça a senha descriptografada da Etapa 4.

Apps

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Um aplicativo AWS OpsWorks Stacks representa o código que você deseja executar em um servidor de aplicativos. O próprio código reside em um repositório, como um arquivo morto do Amazon S3; o aplicativo contém as informações necessárias para implantar o código nas instâncias apropriadas do servidor de aplicativos.

Quando você implanta um aplicativo, o AWS OpsWorks Stacks aciona um evento Deploy, que executa as receitas Deploy de cada camada. AWS OpsWorks O Stacks também instala a configuração da pilha e os atributos de implantação que contêm todas as informações necessárias para implantar o aplicativo, como o repositório do aplicativo e os dados de conexão do banco de dados.

Você deve implementar as receitas personalizadas que recuperam os dados de implantação do aplicativo dos atributos de configuração e implantação da pilha e lidam com as tarefas de implantação.

Tópicos

- Adição de aplicativos
- Implementação de aplicativos
- Editar aplicativos
- Conectando-se a um aplicativo para um servidor de banco de dados

- Usar variáveis de ambiente do
- Transmissão de dados para aplicativos
- Utilização de chaves SSH de repositório Git
- Usando domínios predefinidos
- Uso de SSL

Adição de aplicativos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A primeira etapa na implantação de um aplicativo nos seus servidores de aplicativos é a adição do aplicativo na pilha. O aplicativo representa o programa e contém uma variedade de metadados, como o nome e o tipo do aplicativo e as informações necessárias para implantar o aplicativo nas instâncias do servidor, como o URL do repositório. Você precisa ter a permissão Manage para adicionar um aplicativo a uma pilha. Para obter mais informações, consulte Gerenciamento de permissões de usuário.

> **Note**
>
> O procedimento descrito nesta seção se aplica a pilhas do Chef 12 e mais recentes. Para obter mais informações sobre como adicionar aplicativos a camadas nas pilhas do Chef 11, consulte Etapa 2.4: Criar e implantar um aplicativo - Chef 11.

Para adicionar um aplicativo a uma pilha

1. Coloque o código no repositório de sua preferência: um arquivo compactado do Amazon S3, um repositório Git, um repositório Subversion ou um arquivo compactado HTTP. Para obter mais informações, consulte Origem do aplicativo.

2. No painel de navegação, clique em Apps. Na página Apps, clique em Add an app no seu primeiro aplicativo. Para os aplicativos subsequentes, clique em +App.
3. Use a página App New para configurar o aplicativo, como descreve a seção a seguir.

Configuração do aplicativo

A página Add App consiste nas seguintes seções: Settings, Application source, Data Sources, Environment Variables, Add Domains e SSL Settings.

Tópicos

- Configurações
- Origem do aplicativo
- Fontes de dados
- Variáveis de ambiente
- Configurações de Domínio e SSL

Configurações

Name

O nome do aplicativo, que é usado para representar o aplicativo na interface do usuário. AWS OpsWorks O Stacks também usa esse nome para gerar um nome curto para o aplicativo usado internamente e para identificar o aplicativo nos atributos de configuração e implantação da pilha. Depois de adicionar o aplicativo à pilha, você pode ver o nome abreviado clicando em Apps no painel de navegação e, em seguida, clicando no nome do aplicativo para abrir a página de detalhes.

Document root

AWS OpsWorks Stacks atribui a configuração raiz do documento ao `[:document_root]`atributo nos atributos do `deploy` aplicativo. O valor padrão é `null`. Suas receitas de implantação podem obter esse valor dos atributos `deploy` usando a sintaxe do nó padrão do Chef e implantar o código especificado no local apropriado do servidor. Para obter mais informações sobre como implantar aplicativos, consulte Receitas de implantação.

Origem do aplicativo

Você pode implantar aplicativos dos seguintes tipos de repositório: Git, pacote do Amazon S3, pacote de HTTP e outros. Todos os tipos de repositórios exigem que você especifique o tipo e o URL do repositório. Tipos de repositórios individuais têm seus próprios requisitos, como explicamos a seguir.

> **Note**
>
> AWS OpsWorks O Stacks implanta automaticamente aplicativos dos repositórios padrão nas camadas integradas do servidor. Se você usa o tipo Outro repositório, que é a única opção para pilhas do Windows, o AWS OpsWorks Stacks coloca as informações do repositório nos `deploy`atributos do aplicativo, mas você deve implementar receitas personalizadas para lidar com as tarefas de implantação.

Tópicos

- Arquivo HTTP
- Arquivo do Amazon S3
- Repositório Git
- Outros repositórios

Arquivo HTTP

Para usar um servidor HTTP de acesso público como um repositório:

1. Crie um arquivo compactado do tipo zip, gzip, bzip2, Java WAR ou tarball com o conteúdo da pasta que contém o código do aplicativo e quaisquer arquivos associados.

   > **Note**
   >
   > AWS OpsWorks O Stacks não suporta arquivos tar não compactados.

2. Carregue o arquivo compactado no servidor.
3. Para especificar o repositório no console, selecione HTTP Archive como tipo do repositório e insira o URL.

Se o arquivo compactado é protegido por senha, especifique as credenciais de entrada em Fonte do aplicativo.

Arquivo do Amazon S3

Crie um bucket do Amazon Simple Storage Service como um repositório:

1. Crie um bucket público ou privado no Amazon S3. Para mais informações, consulte a documentação do Amazon S3.
2. Para que o AWS OpsWorks Stacks acesse buckets privados, você deve ser um usuário com pelo menos direitos de somente leitura no bucket do Amazon S3 e precisará do ID da chave de acesso e da chave de acesso secreta. Para obter mais informações, consulte a Documentação do AWS Identity and Access Management.
3. Coloque o código e quaisquer arquivos associados em uma pasta e armazene a pasta em um arquivo compactado – zip, gzip, bzip2, Java WAR ou tarball.

   > **Note**
   >
   > AWS OpsWorks O Stacks não suporta arquivos tar não compactados.

4. Carregue o arquivo compactado no bucket do Amazon S3 e anote o URL.
5. Para especificar o repositório no console AWS OpsWorks Stacks, defina o tipo de repositório como S3 Archive e insira a URL do arquivo. Para arquivos compactados privados, você também deve fornecer um access key ID e a chave de acesso secreta da AWS cuja política concede as permissões para acessar o bucket. Deixe essas configurações em branco para arquivos compactados públicos.

Repositório Git

Um repositório Git fornece controle de origem e controle de versão. AWS OpsWorks O Stacks oferece suporte a sites de repositórios hospedados publicamente, como GitHubo Bitbucket, bem como a servidores Git hospedados de forma privada. Para ambos os aplicativos e os submódulos do Git, o formato que você usa para especificar o URL do repositório em Application Source depende se o repositório é público ou privado:

Repositório público: use os protocolos HTTPS ou Git somente de leitura. Por exemplo, Conceitos básicos das pilhas Linux do Chef 11 usa um GitHub repositório público que pode ser acessado por qualquer um dos seguintes formatos de URL:

- Somente leitura do Git: **`git://github.com/amazonwebservices/opsworks-demo-php-simple-app.git`**
- HTTPS: **`https://github.com/amazonwebservices/opsworks-demo-php-simple-app.git`**

Repositório privado: use o formato de leitura/gravação do SSH como mostram os exemplos a seguir:

- Repositórios Github: **`git@github.com:`*`project`*`/`*`repository`***.
- Repositórios em um servidor Git: ***`user`*`@`*`server`*`:`*`project`*`/`*`repository`***

Ao selecionar Git em Source Control, duas configurações opcionais adicionais são exibidas:

Chave SSH de repositório

Você deve especificar uma chave SSH de implantação para acessar os repositórios Git privados. Esse campo requer a chave privada; a chave pública é atribuída ao seu repositório Git. Para submódulos do Git, a chave especificada deve ter acesso a esses submódulos. Para obter mais informações, consulte Utilização de chaves SSH de repositório Git.

> **Important**
>
> A chave SSH de implantação não pode exigir uma senha; o AWS OpsWorks Stacks não tem como passá-la.

Ramificação/Revisão

Se o repositório tiver várias ramificações, o AWS OpsWorks Stacks baixa a ramificação master por padrão. Para especificar uma ramificação específica, insira o nome da ramificação, o SHA1 hash ou o nome da tag. Para especificar uma determinada confirmação, insira o identificador completo de confirmação de 40 dígitos hexadecimais.

Outros repositórios

Se os repositórios padrão não atendem aos seus requisitos, você pode usar outros repositórios, como o Bazaar. No entanto, o AWS OpsWorks Stacks não implanta automaticamente aplicativos desses repositórios. Você deve implementar receitas personalizadas para lidar com o processo de implantação e atribuí-las aos eventos de implantação das camadas apropriadas. Para ver um exemplo de como implementar o Implantar receitas, consulte Receitas de implantação.

Fontes de dados

Esta seção mostra como anexar um banco de dados no aplicativo. Você tem as seguintes opções:

- RDS: anexe uma das camadas de serviço do Amazon RDS da pilha.
- Nenhum: não anexe um servidor de banco de dados.

Se você selecionar RDS, precisará especificar o seguinte.

Instância do banco de dados

A lista inclui todas as camadas de serviço do Amazon RDS. Você também pode selecionar um dos seguintes:

(Obrigatório) especifique o servidor de banco de dados a ser anexado ao aplicativo. O conteúdo da lista depende da fonte de dados.

- RDS: uma lista das camadas de serviço do Amazon RDS da pilha.

Nome do banco de dados

(Opcional) especifique um nome de banco de dados.

- Camada do Amazon RDS: insira o nome do banco de dados que você especificou para a instância do Amazon RDS.

  Você pode obter o nome do banco de dados do console do Amazon RDS.

Quando você implanta um aplicativo com um banco de dados anexado, o AWS OpsWorks Stacks adiciona a conexão da instância do banco de dados aos `deploy`atributos do aplicativo.

Você pode criar uma receita personalizada para recuperar as informações dos atributos `deploy` e colocá-la no arquivo que pode ser acessado pelo aplicativo. Esta é a única opção para fornecer informações de conexão de banco de dados ao tipo de aplicativo Outros.

Para obter mais informações sobre como lidar com conexões de banco de dados, consulte Conectar-se a um banco de dados.

Para desanexar um servidor de banco de dados de um aplicativo, edite a configuração do aplicativo para especificar um servidor de banco de dados diferente ou nenhum servidor.

Variáveis de ambiente

Você pode especificar um conjunto de variáveis de ambiente para cada aplicativo, que são específicas para o aplicativo. Por exemplo, se você tiver dois aplicativos, as variáveis de ambiente que você define para o primeiro aplicativo não são disponibilizadas para o segundo aplicativo e vice-versa. Você também pode definir a mesma variável de ambiente para vários aplicativos e atribuir a ela um valor diferente para cada aplicativo.

> **Note**
>
> Não há um limite específico para o número de variáveis de ambiente. No entanto, o tamanho da estrutura de dados associada, que inclui os nomes e valores das variáveis e os valores de sinalização protegidos, não pode ultrapassar 20 KB. Esse limite deve acomodar a maioria dos casos de uso. Se o limite for ultrapassado, isso causará um erro de serviço (console) ou exceção (API) com a mensagem, "Ambiente: é muito grande (o tamanho máximo é 20 KB)."

AWS OpsWorks O Stacks armazena as variáveis como atributos nos `deploy`atributos do aplicativo. Você pode fazer com que suas receitas personalizadas recuperem esses valores usando a sintaxe padrão do nó do Chef. Para obter exemplos de como acessar as variáveis de ambiente de um aplicativo, consulte Usar variáveis de ambiente do .

Chave

O nome da variável. Ele pode conter até 64 letras maiúsculas e minúsculas, números e caracteres sublinhados (_), mas deve começar com uma letra ou sublinhado.

Valor

O valor da variável. Ele pode conter até 256 caracteres, que devem ser todos imprimíveis.

Valor protegido

Determina se o valor é protegido. Essa configuração permite que você oculte informações confidenciais, como senhas. Se você definir uma variável como Protected value depois de criar o aplicativo:

- A página de detalhes do aplicativo exibirá somente o nome da variável, e não o valor.
- Se você tiver permissão para editar o aplicativo, pode clicar em Update value para especificar um novo valor, mas não poderá ver ou editar o valor antigo.

> **Note**
>
> Às vezes, os logs de implantação do Chef podem incluir variáveis de ambiente. Isso significa que as variáveis protegidas podem ser exibidas no console. Para evitar que isso ocorra, recomendamos que você use buckets do Amazon S3 como armazenamento para variáveis protegidas que você não deseja que sejam exibidas no console. Um exemplo de como usar um bucket do S3 para essa finalidade está disponível em Usar um bucket do Amazon S3 neste guia.

Configurações de Domínio e SSL

Para o tipo de aplicativo Outro, o AWS OpsWorks Stacks adiciona as configurações aos `deploy` atributos do aplicativo. Suas receitas podem recuperar os dados a partir desses atributos e configurar o servidor, conforme necessário.

Configurações de domínio

Esta seção tem um campo opcional Add Domains para especificar domínios. Para obter mais informações, consulte Usando domínios predefinidos.

Configurações de SSL

Esta seção tem uma opção de alternância de SSL Support que você pode usar para habilitar ou desabilitar o SSL. Se você clicar em Yes, precisará fornecer as informações do certificado SSL. Para obter mais informações, consulte Uso de SSL.

Implementação de aplicativos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

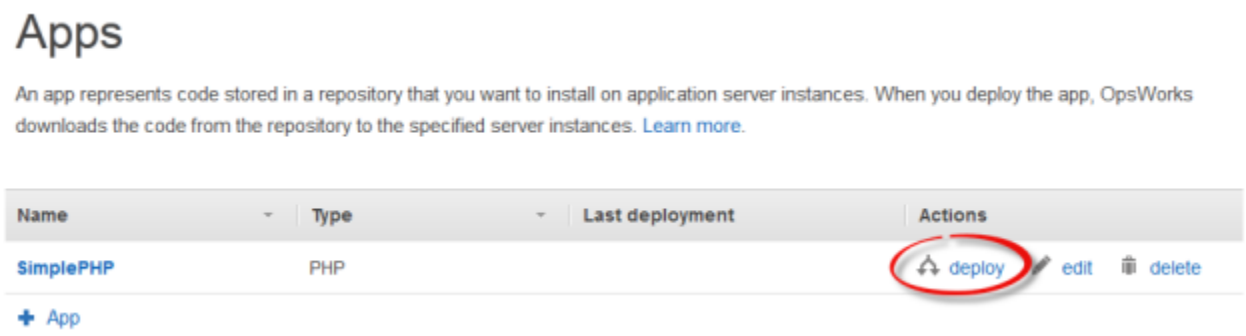
dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O objetivo principal da implementação é implementar código de aplicativo e arquivos relacionados em instâncias do servidor de aplicativos. A operação de implementação é processada pelas receitas de Implementação de cada Instância, determinadas pela camada da instância.

Quando você inicia uma instância, após a conclusão das receitas de configuração, o AWS OpsWorks Stacks executa automaticamente as receitas de implantação da instância. No entanto, ao adicionar ou modificar um aplicativo, você deve implementá-lo manualmente em quaisquer instâncias online. Você deve ter as permissões Gerenciar ou Implementar para implementar um aplicativo. Para obter mais informações, consulte Gerenciamento de permissões de usuário.

Para implementar um aplicativo

1. Na página Apps, clique na ação deploy do aplicativo.



> **Note**
>
> Você também pode implementar um aplicativo clicando em Deployments no painel de navegação. Na página Deployments & Commands, clique em Deploy an app. Ao fazer isso, você também pode escolher qual aplicativo implementar.

2. Especifique o seguinte:
   - (Obrigatório) Defina Command: como deploy, se ainda não estiver selecionado.
   - (Opcional) Inclua um comentário.
3. Clique em Avançado >> para especificar o JSON personalizado. AWS OpsWorks O Stacks adiciona um conjunto de atributos de configuração e implantação da pilha ao objeto do nó. Os

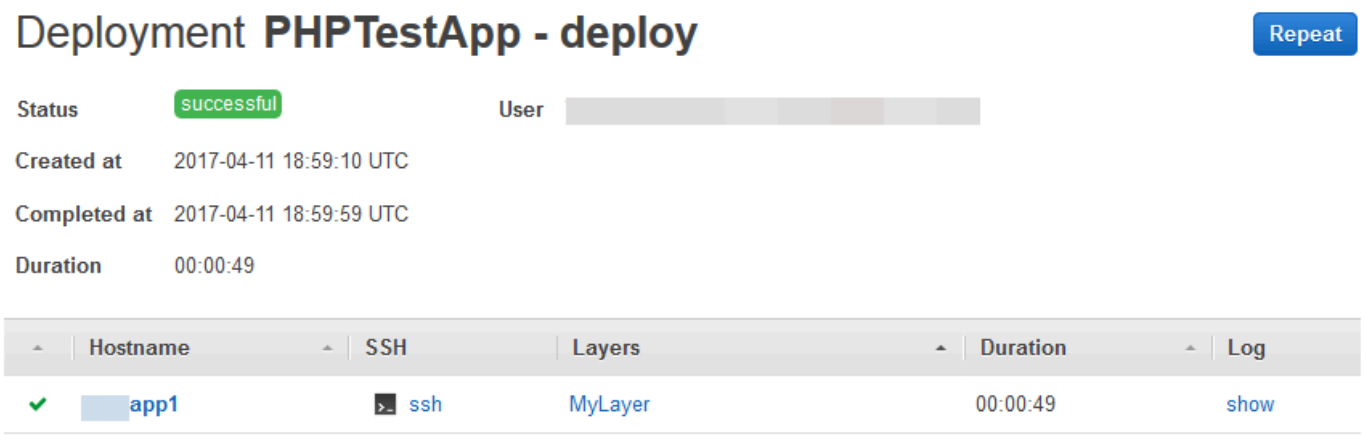
atributos `deploy` contêm os detalhes de implementação e podem ser usados pelas receitas de Implementação para cuidar da instalação e configuração. Nas pilhas do Linux, você pode usar o campo JSON personalizado para substituir as configurações padrão das AWS OpsWorks pilhas ou passar as configurações personalizadas para suas receitas personalizadas. Para obter mais informações sobre como usar JSON personalizado, consulte Usar JSON personalizado.

> **Note**
>
> Se você especificar JSON personalizado aqui, ele é adicionado aos atributos de implementação e configuração da pilha apenas para essa implementação. Se você deseja adicionar JSON personalizado permanentemente, você deve adicioná-lo à pilha. JSON personalizado é limitado a 120 KB. Caso precise de mais capacidade, recomendamos armazenar alguns dados no Amazon S3. Suas receitas personalizadas podem então usar a CLI da AWS ou o AWS SDK para Ruby para fazer download dos dados do bucket para sua instância. Para obter um exemplo, consulte Usar o SDK for Ruby.

4. Em Instances, clique em Advanced **>>** e especifique em quais instâncias executar o comando de implementação.

   O comando de implementação aciona um evento Implementar, que executa as receitas de implementação nas instâncias selecionadas. As receitas de implementação do servidor de aplicativo associado baixa o código e os arquivos relacionados do repositório e os instala na instância, portanto você normalmente seleciona todas as instâncias do servidor de aplicativo associado. No entanto, outros tipos de instâncias podem exigir algumas alterações na configuração para acomodar o novo aplicativo, portanto geralmente é útil executar as receitas de implementação nessas instâncias também. Essas receitas atualizam a configuração conforme necessário, mas não instalam os arquivos do aplicativo. Para obter mais informações sobre receitas, consulte Livros de receitas e receitas.

5. Clique em Deploy para executar as receitas de implementação nas instâncias especificadas, o que irá exibir a página Implementação. Quando o processo estiver concluído, o AWS OpsWorks Stacks marca o aplicativo com uma marca verde para indicar a implantação bem-sucedida. Se a implantação falhar, o AWS OpsWorks Stacks marca o aplicativo com um X vermelho. Nesse caso, você pode acessar a página Implantações e examinar o registro de implantação para obter mais informações.



> **Note**
>
> Quando você implanta uma atualização em um aplicativo JSP, o Tomcat pode não reconhecer a atualização e, em vez disso, continuar a executar a versão do aplicativo existente. Isso pode acontecer, por exemplo, se você implantar seu aplicativo como um arquivo.zip que contém apenas uma página JSP. Para garantir que o Tomcat execute a última versão implementada, o diretório raiz do projeto deve incluir um diretório WEB-INF que contenha um arquivo `web.xml`. Um arquivo `web.xml` pode conter uma variedade de conteúdos, mas o conteúdo a seguir é suficiente para garantir que o Tomcat reconheça as atualizações e execute a última versão implementada do aplicativo. Você não precisa alterar a versão para cada atualização. O Tomcat reconhecerá a atualização mesmo se a versão não tiver sido alterada.
>
> ```
> <context-param>
>  <param-name>appVersion</param-name>
>  <param-value>0.1</param-value>
> </context-param>
> ```

Outros comandos de implementação

A página Deploy app inclui vários outros comandos para gerenciar seus aplicativos e os servidores associados. Dos comandos a seguir, somente Undeploy está disponível para aplicativos nas pilhas do Chef 12.

Desfazer a Implementação

Aciona um evento do ciclo de vida para Desfazer a Implementação, que executa as receitas apropriadas para remover todas as versões do aplicativo das instâncias especificadas.

Reversão

Restaura a versão do aplicativo implantada anteriormente. Por exemplo, se você implementou o aplicativo três vezes e, em seguida, executou Rollback, o servidor irá servir o aplicativo da segunda implementação. Se você executar Rollback novamente, o servidor irá servir o aplicativo na primeira implantação. Por padrão, o AWS OpsWorks Stacks armazena as cinco implantações mais recentes, o que permite reverter até quatro versões. Caso exceda o número de versões armazenadas, o comando falha e deixa a versão mais antiga em vigor. Este comando não está disponível nas pilhas do Chef 12.

Iniciar o Servidor Web

Executa as receitas que iniciam o servidor do aplicativo nas instâncias especificadas. Este comando não está disponível nas pilhas do Chef 12.

Parar o Servidor Web

Executa as receitas que param o servidor do aplicativo nas instâncias especificadas. Este comando não está disponível nas pilhas do Chef 12.

Reiniciar o Servidor Web

Executa as receitas que reiniciam o servidor do aplicativo nas instâncias especificadas. Este comando não está disponível nas pilhas do Chef 12.

> **Important**
>
> Start Web Server, Stop Web Server, Restart Web Server e Rollback são versões personalizadas do comando de pilha Executar receitas. Eles executam um conjunto de receitas que realizam a tarefa nas instâncias especificadas.
>
> - Esses comandos não acionam um evento de ciclo de vida, portanto você não pode conectá-los para executar o código personalizado.
> - Esses comandos funcionam apenas para as camadas do servidor de aplicativo integradas.
>
> Especificamente, esses comandos não afetam as camadas personalizadas, mesmo que elas ofereçam suporte ao servidor de aplicativo. Para iniciar, parar ou reiniciar servidores

em uma camada personalizada, você deve implementar receitas personalizadas para realizar essas tarefas e usar o comando de pilha para Executar Receitas para executá-las. Para obter mais informações sobre como implementar e instalar receitas personalizadas, consulte Livros de receitas e receitas.

Editar aplicativos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode modificar a configuração de um aplicativo editando-o. Por exemplo, se você estiver pronto para implantar uma nova versão, poderá editar as configurações de AWS OpsWorks pilhas do aplicativo para usar a nova ramificação do repositório. É necessário ter as permissões Gerenciar ou Implementar para editar a configuração de um aplicativo. Para obter mais informações, consulte Gerenciamento de permissões de usuário.

Para editar um aplicativo

1. Na página Apps, clique no nome do aplicativo para abrir sua página de detalhes.
2. Clique em Edit para alterar a configuração do aplicativo.
   - Se você modificar o nome do aplicativo, o AWS OpsWorks Stacks usará o novo nome para identificar o aplicativo no console.

     Alterar o nome não altera o nome curto associado. O nome curto é definido ao adicionar o aplicativo na pilha e não pode ser modificado posteriormente.
   - Se você especificou uma variável de ambiente protegida, não é possível visualizar ou editar o valor. No entanto, você pode especificar um novo valor clicando em Update value.
3. Clique em Save para salvar a nova configuração e, em seguida, Deploy App para implementar o aplicativo.

A edição de um aplicativo altera as configurações com as AWS OpsWorks pilhas, mas não afeta as instâncias da pilha. Na primeira vez que um aplicativo é implementado, as receitas de Implementação baixam o código e os arquivos relacionados para as instâncias de servidor do aplicativo, que então executam a cópia local. Se você modificar o aplicativo no repositório ou alterar qualquer outra configuração, deverá implantar o aplicativo para instalar as atualizações nas instâncias do seu servidor de aplicativos, da seguinte maneira. AWS OpsWorks O Stacks implanta automaticamente a versão atual do aplicativo em novas instâncias quando elas são iniciadas. Entretanto, para as instâncias atuais a situação é diferente:

- AWS OpsWorks O Stacks implanta automaticamente a versão atual do aplicativo em novas instâncias quando elas são iniciadas.
- AWS OpsWorks O Stacks implanta automaticamente a versão mais recente do aplicativo em instâncias off-line, incluindo instâncias baseadas em carga e em tempo, quando elas são reiniciadas.
- Você deve implementar manualmente o aplicativo atualizado em instâncias online.

Para obter mais informações sobre como implementar aplicativos, consulte Implementação de aplicativos

Conectando-se a um aplicativo para um servidor de banco de dados

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode associar um servidor de banco de dados do Amazon RDS; a um aplicativo ao criar o aplicativo, ou, posteriormente, editando a configuração do aplicativo. O aplicativo pode então usar as informações de conexão de banco de dados, como nome de usuário, senha etc., para se conectar ao servidor de banco de dados. Quando você implanta um aplicativo, o AWS OpsWorks Stacks fornece essas informações aos aplicativos de duas maneiras:

- Para pilhas do Linux, o AWS OpsWorks Stacks cria um arquivo em cada uma das instâncias do servidor de aplicativos integrados que contêm os dados de conexão que o aplicativo pode usar para se conectar ao servidor do banco de dados.
- AWS OpsWorks As pilhas incluem as informações de conexão na configuração da pilha e nos atributos de implantação que são instalados em cada instância.

  Você pode implantar uma receita personalizada para extrair as informações de conexão desses atributos e colocá-la em um arquivo no formato de sua preferência. Para obter mais informações, consulte Transmissão de dados para aplicativos.

> **Important**
>
> Para pilhas do Linux, se você desejar associar uma camada de serviço Amazon RDS com seu aplicativo, você deve adicionar o driver adequado para o pacote camada de servidor do aplicativo associado, da seguinte forma:
>
> 1. Clique em Layers no painel de navegação e abra a guia Recipes do servidor do aplicativo.
> 2. Clique em Edit e adicione o driver adequado a OS Packages. Por exemplo, você deve especificar `mysql` se a camada contém as instâncias do Amazon Linux e `mysql-client` se a camada contém as instâncias Ubuntu.
> 3. Salve as alterações e reimplante o aplicativo.

Usando uma receita predefinida

Pode ser implementada uma receita predefinida que extrai os dados de conexão dos `deploy` atributos do aplicativo e os salva em um formato que o aplicativo pode ler, como um arquivo YAML.

Um servidor poderá ser anexado ao banco de dados de um aplicativo quando o aplicativo for criado ou posteriormente, editando o aplicativo. Quando você implanta o aplicativo, o AWS OpsWorks Stacks instala uma configuração de pilha e atributos de implantação em cada instância que incluem as informações de conexão do banco de dados. Dessa forma, seu aplicativo pode recuperar os atributos apropriados. Os detalhes dependem se você estiver usando uma pilha do Linux ou Windows.

Conectando-se a um servidor de banco de dados para uma pilha do Linux

Para pilhas do Linux, o namespace de atributos de implantação e configuração de pilha `deploy` incluem um atributo para cada aplicativo implantado, identificado pelo nome abreviado do aplicativo. Quando você anexa um servidor de banco de dados a um aplicativo, o AWS OpsWorks Stacks preenche o `[:database]` atributo do aplicativo com as informações de conexão e o instala nas instâncias da pilha para cada implantação subsequente. Os valores de atributo são fornecidos pelo usuário ou gerados pelo AWS OpsWorks Stacks.

> **Note**
>
> AWS OpsWorks O Stacks permite que você conecte um servidor de banco de dados a vários aplicativos, mas cada aplicativo pode ter somente um servidor de banco de dados conectado. Se desejar conectar um aplicativo para mais de um servidor de banco de dados, anexe um dos servidores ao aplicativo, e use as informações de atributos `deploy` do aplicativo para se conectar a esse servidor. Use o JSON predefinido para passar as informações de conexão aos outros servidores de banco de dados para o aplicativo. Para obter mais informações, consulte Transmissão de dados para aplicativos.

Um aplicativo pode usar as informações de conexão dos `deploy` atributos da instância para se conectar a um banco de dados. No entanto, os aplicativos não podem acessar diretamente essas informações, somente receitas podem acessar os atributos `deploy`. Você pode resolver esse problema, implementando uma receita predefinida que extrai as informações de conexão dos `deploy` atributos e os coloca em um arquivo que pode ser lido pelo aplicativo.

Usar variáveis de ambiente do

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> As recomendações neste tópico se aplicam ao Chef 11.10 e a versões anteriores do Chef. Para obter as variáveis de ambiente no Chef 12 ou em versões mais recentes, você deverá usar a Data bag do aplicativo. Para obter mais informações, consulte AWS OpsWorks Data Bag Reference e App Data Bag (aws_opsworks_app).

Quando você especifica variáveis de ambiente para um aplicativo, o AWS OpsWorks Stacks adiciona as definições das variáveis aos `deploy`atributos do aplicativo.

Camadas personalizadas podem usar uma receita para recuperar o valor de uma variável, usando a sintaxe de nó padrão, e armazená-lo em um formato que possa ser acessado pelos aplicativos da camada.

Você deve implementar uma receita personalizada que obtém os valores das variáveis de ambiente dos atributos `deploy` da instância. A receita poderá então armazenar os dados na instância em um formato que possa ser acessado pelo aplicativo, como um arquivo YAML. As definições das variáveis de ambiente de um aplicativo são armazenadas nos atributos `deploy`, nas `environment_variables` do aplicativo. O exemplo a seguir mostra a localização desses atributos para um aplicativo chamado `simplephpapp`, usando JSON para representar a estrutura do atributo.

```
{
  ...
  "ssh_users": {
  },
  "deploy": {
    "simplephpapp": {
      "application": "simplephpapp",
      "application_type": "php",
      "environment_variables": {
        "USER_ID": "168424",
        "USER_KEY": "somepassword"
      },
    ...
  }
}
```

Uma receita pode obter os valores das variáveis, usando a sintaxe de nó padrão. O exemplo a seguir mostra como obter o valor `USER_ID` do JSON anterior e colocá-lo no log do Chef.

```
Chef::Log.info("USER_ID: #{node[:deploy]['simplephpapp'][:environment_variables]
[:USER_ID]}")
```

Para obter uma descrição mais detalhada de como recuperar informações do JSON de configuração e implantação da pilha e armazená-las na instância, consulte Transmissão de dados para aplicativos.

Transmissão de dados para aplicativos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Muitas vezes, é útil transmitir dados como pares de chave-valor a um aplicativo no servidor. Para fazer isso, use um JSON personalizado para adicionar os dados à pilha. AWS OpsWorks O Stacks adiciona os dados ao objeto de nó de cada instância para cada evento do ciclo de vida.

Observe, no entanto, que embora as receitas personalizadas possam obter os dados do JSON personalizado no objeto de nó usando atributos do Chef, os aplicativos não conseguem fazer o mesmo. Uma abordagem para a obtenção de dados do JSON personalizado para um ou mais aplicativos é implementar uma receita personalizada que extraia os dados do objeto `node` e grave-os em um arquivo que o aplicativo possa ler. O exemplo neste tópico mostra como gravar os dados em um arquivo YAML, mas você pode usar a mesma abordagem básica para outros formatos, como JSON ou XML.

Para transmitir dados de chave-valor para instâncias da pilha, adicione um JSON personalizado, como o mostrado a seguir, à pilha. Para obter mais informações sobre como adicionar um JSON personalizado a uma pilha, consulte Usar JSON personalizado.

```
{
```

```
  "my_app_data": {
    "app1": {
      "key1": "value1",
      "key2": "value2",
      "key3": "value3"
    },
    "app2": {
      "key1": "value1",
      "key2": "value2",
      "key3": "value3"
    }
  }
}
```

O exemplo pressupõe que você tenha dois aplicativos cujos nomes curtos são `app1` e `app2`, sendo que cada um deles tem três valores de dados. A receita associada pressupõe que você use os nomes curtos dos aplicativos para identificar os dados associados; outros nomes são arbitrários. Para obter mais informações sobre os nomes abreviados de aplicativos, consulte Configurações.

A receita no exemplo a seguir mostra como extrair os dados para cada aplicativo dos atributos `deploy` e inseri-los em um arquivo `.yml`. A receita pressupõe que seu JSON personalizado contenha dados para cada aplicativo.

```
node[:deploy].each do |app, deploy|
  file File.join(deploy[:deploy_to], 'shared', 'config', 'app_data.yml') do
    content YAML.dump(node[:my_app_data][app].to_hash)
  end
end
```

Os atributos `deploy` contêm um atributo para cada aplicativo, chamado pelo nome abreviado do aplicativo. Cada atributo do aplicativo contém um conjunto de atributos que representam várias informações sobre o aplicativo. Este exemplo usa o diretório de implementação do aplicativo, que é representado pelo atributo `[:deploy][:app_short_name][:deploy_to]`. Para obter mais informações sobre `[:deploy]`, consulte Atributos deploy.

Para cada aplicativo em `deploy`, a receita faz o seguinte:

1. Cria um arquivo chamado `app_data.yml` no subdiretório `shared/config` do diretório `[:deploy_to]` do aplicativo.

Para obter mais informações sobre como o AWS OpsWorks Stacks instala aplicativos, consulte. Receitas de implantação

2. Converte os valores do JSON personalizado para YAML e grava os dados formatados em `app_data.yml`.

Para transmitir dados para um aplicativo

1. Adicione um aplicativo à pilha e anote o nome abreviado dele. Para obter mais informações, consulte Adição de aplicativos.
2. Adicione um JSON personalizado com os dados do aplicativo aos atributos `deploy`, conforme descrito anteriormente. Para obter mais informações sobre como adicionar um JSON personalizado a uma pilha, consulte Usar JSON personalizado.
3. Crie um livro de receitas e adicione uma receita a ele com código baseado no exemplo anterior, modificado conforme necessário em relação aos nomes dos atributos que você usou no JSON personalizado. Para obter mais informações sobre como criar livros de receitas e receitas, consulte Livros de receitas e receitas. Se você já tem receitas personalizadas para esta pilha, também pode adicionar a receita a um livro de receitas, ou até mesmo adicionar o código a uma receita Implantar existente.
4. Instale o livro de receitas em sua pilha. Para obter mais informações, consulte Instalação de livros de receitas personalizados.
5. Atribua a receita ao evento Deploy lifecycle da camada do servidor de aplicativos. AWS OpsWorks As pilhas então executarão a receita em cada nova instância, após a inicialização. Para obter mais informações, consulte Execução de receitas.
6. Implante o aplicativo, o que também instala os atributos de configuração e implantação da pilha, que agora contém seus dados.

> **Note**
>
> Se for necessário implantar os arquivos de dados antes da implantação do aplicativo, também é possível associar a receita ao evento de ciclo de vida Configuração da camada, o que ocorre uma vez, logo após a conclusão da inicialização da instância. No entanto, o AWS OpsWorks Stacks ainda não criou os diretórios de implantação, portanto, sua receita deve criar os diretórios necessários explicitamente antes de criar o arquivo de dados. O exemplo a

seguir cria explicitamente o diretório `/shared/config` do aplicativo e, em seguida, cria um arquivo de dados nesse diretório.

```
node[:deploy].each do |app, deploy|

 directory "#{deploy[:deploy_to]}/shared/config" do
      owner "deploy"
      group "www-data"
      mode 0774
      recursive true
      action :create
    end

  file File.join(deploy[:deploy_to], 'shared', 'config', 'app_data.yml') do
    content YAML.dump(node[:my_app_data][app].to_hash)
  end
end
```

Para carregar os dados, você pode usar algo como o seguinte código do Sinatra:

```
#!/usr/bin/env ruby
# encoding: UTF-8
require 'sinatra'
require 'yaml'

get '/' do
  YAML.load(File.read(File.join('..', '..', 'shared', 'config', 'app_data.yml')))
End
```

Você pode atualizar os valores dos dados do aplicativo a qualquer momento atualizando o JSON personalizado, como mostrado a seguir.

Para atualizar os dados do aplicativo

1. Edite o JSON personalizado para atualizar os valores de dados.
2. Implante o aplicativo novamente, o que faz com que as AWS OpsWorks pilhas executem as receitas de implantação nas instâncias da pilha. As receitas usarão os atributos atualizados de

configuração e implantação da pilha, portanto, sua receita atualizará os arquivos de dados com os valores atuais.

Utilização de chaves SSH de repositório Git

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Uma chave SSH de repositório Git, às vezes chamada de chave SSH de implantação, é uma chave SSH sem senha que fornece acesso a um repositório privado do Git. O ideal é que ela não pertença a um desenvolvedor específico. Seu objetivo é permitir que o AWS OpsWorks Stacks implante aplicativos ou livros de receitas de forma assíncrona de um repositório Git sem precisar de mais informações de você.

A tabela a seguir descreve o procedimento básico para a criação de uma chave SSH de repositório. Para obter detalhes, consulte a documentação do seu repositório. Por exemplo, Gerenciar chaves de implantação descreve como criar uma chave SSH de repositório para um GitHub repositório, e Chaves de implantação no Bitbucket descreve como criar uma chave SSH de repositório para um repositório Bitbucket. Observe que alguns documentos descrevem a criação de uma chave em um servidor. Para AWS OpsWorks Stacks, basta substituir “servidor” por “estação de trabalho” nas instruções.

Para criar uma chave SSH de repositório

1. Crie um par de chaves SSH de implantação para o repositório Git da sua estação de trabalho usando um programa como o `ssh-keygen`.

   > **Important**
   >
   > AWS OpsWorks O Stacks não oferece suporte a frases secretas de chave SSH.

2. Atribua a chave pública ao repositório e armazene a chave privada em sua estação de trabalho.

3. Insira a chave privada na caixa Repository SSH Key quando adicionar um aplicativo ou especificar um repositório de livros de receitas. Para obter mais informações, consulte Adição de aplicativos.

AWS OpsWorks O Stacks passa a chave SSH do repositório para cada instância, e as receitas integradas usam a chave para se conectar ao repositório e baixar o código. A chave é armazenadas nos atributos de `deploy` como `node[:deploy]['appshortname'][:scm][:ssh_key]`, e pode ser acessada apenas pelo usuário raiz.

Usando domínios predefinidos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Se você hospedar um nome de domínio com terceiros, você pode mapear esse nome de domínio para um aplicativo. O procedimento básico é como se segue:

1. Crie um subdomínio com o registrador do DNS e mapeie-o para o seu endereço IP elástico de load balancer ou para o seu endereço IP público do servidor de aplicativos.
2. Atualize a configuração do aplicativo para indicar o subdomínio e reimplantar o aplicativo.

> **Note**
>
> Certifique-se de encaminhar o seu nome de domínio não qualificado (como myapp1.example.com) para o seu nome de domínio completo (como www.myapp1.example.com) para que ambos sejam mapeados para seu aplicativo.

Quando você configurar um domínio para um aplicativo, ele é listado como um apelido do servidor no arquivo de configuração do servidor. Se estiver usando um load balancer, o load balancer verifica o

nome de domínio na URL, à medida que as solicitações chegam e redirecionam o tráfego com base no domínio.

Para mapear um subdomínio para um endereço IP

1. Se estiver usando um load balancer, na página Instances, clique no load balancer da instância para abrir sua página de detalhes e obter o endereço Elastic IP da instância. Caso contrário, obtenha o endereço IP público a partir da página de detalhes da instância do servidor de aplicativos.
2. Siga as instruções fornecidas pelo seu registrador DNS para criar e mapear seu subdomínio para o endereço IP a partir da Etapa 1.

> **Note**
>
> Se o load balancer da instância for encerrada em algum momento, você será atribuído a um novo endereço IP elástico. Você precisa atualizar o registrador DNS para mapear configurações para o novo endereço IP elástico.

AWS OpsWorks O Stacks simplesmente adiciona as configurações do domínio aos `deploy`atributos do aplicativo. Implemente uma receita personalizada para recuperar as informações do objeto do nó e configurar o servidor adequadamente. Para obter mais informações, consulte Livros de receitas e receitas.

Executando vários aplicativos no mesmo servidor de aplicações

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> As informações neste tópico não se aplicam aos aplicativos do Node.js.

Se tiver vários aplicativos do mesmo tipo, às vezes é mais econômico executá-los no mesmo servidor do aplicativo de instâncias.

Para executar vários aplicativos no mesmo servidor

1. Adicione um aplicativo à pilha para cada aplicativo.
2. Obtenha um subdomínio separado para cada aplicativo e mapeie os subdomínios para o servidor de aplicativo ou o endereço IP do load balancer.
3. Edite cada configuração do aplicativo para especificar o subdomínio apropriado.

Para obter mais informações sobre como executar essas tarefas, consulte Usando domínios predefinidos.

> **Note**
>
> Caso seu aplicativo esteja executando vários aplicativos HTTP, o Elastic Load Balancing poderá ser usado para balanceamento de carga. Para vários aplicativos HTTPS, você deve encerrar a conexão SSL no load balancer ou criar uma pilha separada para cada aplicativo. As solicitações de HTTPS são criptografadas, o que significa que, se você encerrar a conexão SSL em servidores, o load balancer não poderá verificar o nome de domínio para determinar qual aplicativo deverá lidar com a solicitação.

Uso de SSL

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para usar o SSL com seu aplicativo, consiga um certificado digital de servidor de uma autoridade de certificação. Para simplificar, essa demonstração cria um certificado e, em seguida, autoassina ele. Os certificados autoassinados são úteis para fins de aprendizagem e teste, mas sempre use um certificado assinado por uma autoridade de certificação para pilhas de produção.

Nessa demonstração, siga as seguintes instruções:

1. Instale e configure o OpenSSL.
2. Crie uma chave privada.
3. Crie uma solicitação de assinatura do certificado.
4. Gere um certificado autoassinado.
5. Edite o aplicativo com as informações do certificado.

> **Important**
>
> Se seu aplicativo usa SSL, recomendamos que você desative, se possível SSLv3, nas camadas do servidor de aplicativos para resolver as vulnerabilidades descritas em CVE-2014-3566. Caso sua pilha inclua uma camada Ganglia, desative o SSL v3 para essa camada também. Os detalhes dependem da camada em questão. Para obter mais informações, consulte o seguinte.
>
> - Camada de AWS OpsWorks pilhas do servidor de aplicativos Java
> - Camada de AWS OpsWorks pilhas do servidor de aplicativos Node.js
> - Camada de AWS OpsWorks pilhas do servidor de aplicativos PHP
> - Camada de pilhas do servidor de aplicativos AWS OpsWorks Rails
> - Camada estática de AWS OpsWorks pilhas de servidores Web
> - Camada Ganglia

Tópicos

- Etapa 1: instalar e configurar o OpenSSL.
- Etapa 2: criar uma chave privada
- Etapa 3: criar uma solicitação de assinatura do certificado
- Etapa 4: Enviar a CSR para a autoridade de certificação
- Etapa 5: editar o aplicativo

Etapa 1: instalar e configurar o OpenSSL.

A criação e a atualização de certificados de servidores exigem uma ferramenta compatível com os protocolos SSL e TLS. OpenSSL é uma ferramenta de código aberto que fornece as funções básicas de criptografia necessárias para criar um token RSA e assinar com sua chave privada.

O procedimento a seguir supõe que seu computados ainda não tem o OpenSSL instalado.

Para instalar o OpenSSL no Linux e Unix

1. Acesse OpenSSL: Source, Tarballs.
2. Faça o download da fonte mais recente.
3. Construa o pacote.

Para instalar o OpenSSL no Windows

1. Se o Pacote redistribuível do Microsoft Visual C++ 2008 não estiver instalado no seu sistema, faça o download do pacote.
2. Execute o instalador e siga as instruções fornecidas pelo assistente de configuração do Microsoft Visual C++ 2008 Redistributable para instalar o redistributable.
3. Acesse OpenSSL: distribuições binárias, clique na versão adequada dos binários do OpenSSL para o seu ambiente e salve o instalador localmente.
4. Execute o instalador e siga as instruções no OpenSSL Setup Wizard para instalar os binários.

Crie uma variável de ambiente que aponta para o ponto de instalação do OpenSSL abrindo o terminal ou a janela de comando e usando as seguintes linhas de comando.

- No Linux e Unix

```
export OpenSSL_HOME=path_to_your_OpenSSL_installation
```

- No Windows

```
set OpenSSL_HOME=path_to_your_OpenSSL_installation
```

Adicione o caminho dos binários do OpenSSL na variável de caminho do seu computador abrindo o terminal ou a janela de comando e usando as seguintes linhas de comando.

- No Linux e Unix

```
export PATH=$PATH:$OpenSSL_HOME/bin
```

- No Windows

```
set Path=OpenSSL_HOME\bin;%Path%
```

> **Note**
>
> Qualquer alteração feira nas variáveis do ambiente usando essas linhas de comando são válidas apenas para a seção atual de linha de comando.

Etapa 2: criar uma chave privada

Será necessário uma chave privada exclusiva para criar sua solicitação de assinatura de certificado (CSR). Crie a chave usando a seguinte linha de comando:

```
openssl genrsa 2048 > privatekey.pem
```

Etapa 3: criar uma solicitação de assinatura do certificado

Uma solicitação de assinatura do certificado (CSR) é um arquivo enviado para uma autoridade de certificação (CA) para solicitar um certificado digital de servidor. Crie a CSR usando a seguinte linha de comando.

```
openssl req -new -key privatekey.pem -out csr.pem
```

A saída do comando será semelhante à seguinte:

```
You are about to be asked to enter information that will be incorporated
 into your certificate request.
 What you are about to enter is what is called a Distinguished Name or a DN.
 There are quite a few fields but you can leave some blank
 For some fields there will be a default value,
 If you enter '.', the field will be left blank.
```

A tabela a seguir pode ajudar você a criar sua solicitação de certificado.

Dados da solicitação de certificado

Nome	Descrição	Exemplo
Nome do país	A abreviação ISO de duas letras para seu país.	US = Estados Unidos
Estado	O nome do estado ou província onde sua organizaç ão está localizada. Este nome não pode ser abreviado.	Washington
Nome da localidade	O nome da cidade onde sua organização está localizada.	Seattle
Nome da organização	A razão social completa da sua organização. Não abrevie o nome de sua organização.	CorporationX
Unidade Organizacional	(Opcional) Para informações adicionais da sua organizaç ão.	Marketing
Nome comum	O nome do domínio completamente qualifica do para seu CNAME. Você receberá um aviso de verificaç ão do nome do certificado se não houver correspondência.	www.exemplo.com
Endereço de e-mail	O endereço de e-mail do administrador do servidor	someone@example.com

> **Note**
>
> O campo do nome comum geralmente é mal-interpretado e completado incorretamente. O nome comum geralmente é o seu servidor mais o nome do domínio. Será semelhante a

"www.example.com" ou "example.com". Será necessário criar uma CSR usando o nome comum correto.

Etapa 4: Enviar a CSR para a autoridade de certificação

Para uso na produção, é preciso obter um certificado de servidor enviando sua CSR para uma autoridade de certificação (CA), que pode exigir outras credenciais ou comprovantes de identidade. Se sua solicitação for bem-sucedida, a CA envia de volta um certificado de identidade assinado digitalmente e, possivelmente, um arquivo de cadeia do certificado. AWS não recomenda um CA específica. Para obter uma lista parcial dos disponíveis CAs, consulte Autoridade Certificadora - Provedores na Wikipedia.

Além disso, é possível gerar um certificado autoassinado, que pode ser usado apenas para fins de teste. Para esse exemplo, use a seguinte linha de comando para gerar um certificado autoassinado.

```
openssl x509 -req -days 365 -in csr.pem -signkey privatekey.pem -out server.crt
```

A saída será semelhante à seguinte:

```
Loading 'screen' into random state - done
Signature ok
subject=/C=us/ST=washington/L=seattle/O=corporationx/OU=marketing/CN=example.com/
emailAddress=someone@example.com
Getting Private key
```

Etapa 5: editar o aplicativo

Após gerar o seu certificado e assiná-lo, atualize seu aplicativo para ativar o SSL e forneça as informações do seu certificado. Na página Apps (Aplicativos), escolha um aplicativo para abrir a página de detalhes e clique em Edit App (Editar aplicativo). Para ativar o suporte ao SSL, defina Enable SSL (Habilitar SSL) como Yes (Sim), que exibe as seguintes opções de configuração.

SSL Certificate (Certificado SSL)

Cole o conteúdo do arquivo do certificado da chave pública (.crt) na caixa. O certificado deve ser semelhante ao seguinte:

```
-----BEGIN CERTIFICATE-----
MIICuTCCAiICCQCtqFKItVQJpzANBgkqhkiG9w0BAQUFADCBoDELMAkGA1UEBhMC
dXMxEzARBgNVBAgMCndhc2hpbmd0b24xEDAOBgNVBAcMB3NlYXR0bGUxDzANBgNV
BAoMBmFtYXpvbjEWMBQGA1UECwwNRGV2IGFuZCBUb29sczEdMBsGA1UEAwwUc3Rl
cGhhbmllYXBpZXJjZS5jb20xIjAgBgkqhkiG9w0BCQEWE3NhcGllcmNlQGFtYXpv
...
-----END CERTIFICATE-----
```

> **Note**
>
> Caso esteja usando o Nginx e tenha um arquivo de cadeia do certificado, acrescente o conteúdo no arquivo do certificado da chave pública.

Se estiver atualizando um certificado existente, siga as seguintes instruções:

- Escolha Update SSL certificate (Atualizar certificado SSL) para atualizar o certificado.
- Caso o novo certificado não corresponda à chave privada existente, escolha Update SSL certificate key (Atualizar chave de certificado SSL).
- Caso o novo certificado não corresponda à cadeia de certificado existente, escolha Update SSL certificates (Atualizar certificados SSL).

SSL Certificate Key (Chave de certificado SSL)

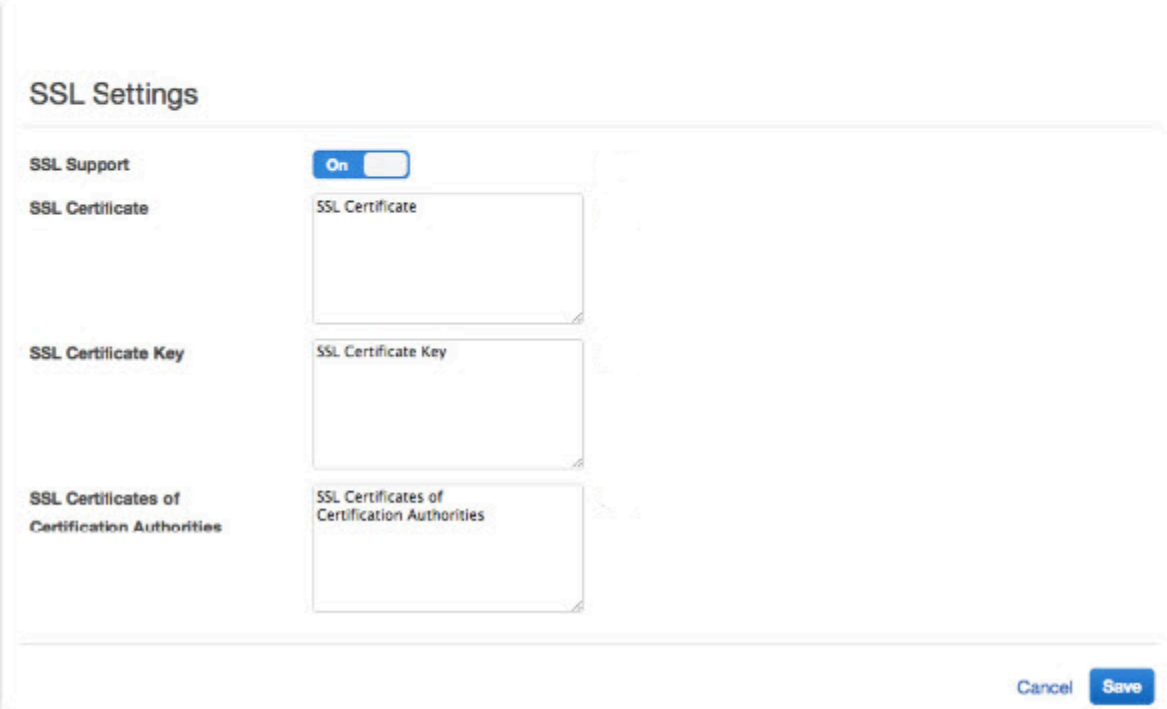
Cole o conteúdo do arquivo do certificado da chave privada (.pem) na caixa. Ela deve ser parecida com a seguinte:

```
----BEGIN RSA PRIVATE KEY-----
MIICXQIBAAKBgQC0CYklJY5r4vV2NHQYEpwtsLuMMBhylMrgBShKq+HHVLYQQCL6
+wGIiRq5qXqZlRXje3GM5Jvcm6q0R71MfRIl1FuzKyqDtneZaAIEYniZibHiUnmO
/UNqpFDosw/6hY3ONk0fSBlU4ivD0Gjpf6J80jL3DJ4R23Ed0sdL4pRT3QIDAQAB
AoGBAKmMfWrNRqYVtGKgnWB6Tji9QrKQLMXjmHeGg95mppdJELiXHhpMvrHtpIyK
...
-----END RSA PRIVATE KEY-----
```

SSL certificates of Certification Authorities

Se tiver um arquivo de cadeia do certificado, cole o conteúdo na caixa.

> **Note**
>
> Se estiver usando Nginx, deixe a caixa em branco. Se tiver um arquivo de cadeia do certificado, acrescente-o no arquivo do certificado da chave pública em SSL Certificate Key (Chave de certificado SSL).



Depois de clicar em Save, refaça a implantação do aplicativo para atualizar suas instâncias online.

Para as camadas integradas do servidor de aplicativos, o AWS OpsWorks Stacks atualiza automaticamente a configuração do servidor. Após o término da implantação, verifique se a instalação do OpenSSL funcionou da seguinte maneira.

Para verificar a instalação de um OpenSSL

1. Acesse a página Instances.
2. Execute o aplicativo clicando no endereço de IP da instância do servidor do aplicativo ou, se estiver usando um load balancer, o endereço de IP do load balancer.
3. Altere o prefixo do endereço de IP de **`http://`** para **`https://`** e atualize o navegador para verificar se a página carrega corretamente com o SSL.

Os usuários que têm aplicativos configurados para serem executados no Mozilla Firefox às vezes recebem o seguinte erro no certificado: `SEC_ERROR_UNKNOWN_ISSUER`. Esse erro pode ser causado pela funcionalidade de substituição do certificado nos programas antivírus e antimalware de sua organização, por alguns tipos de monitoramento de tráfego de rede e software de filtragem ou por malware. Para obter mais informações sobre como solucionar esse erro, consulte Como solucionar problemas de códigos de erro de segurança em sites seguros no site de suporte do Mozilla Firefox.

Para todas as outras camadas, incluindo as personalizadas, o AWS OpsWorks Stacks simplesmente adiciona as configurações do SSL aos atributos `deploy` do aplicativo. Implemente uma receita personalizada para recuperar as informações do objeto do nó e configurar o servidor adequadamente.

Livros de receitas e receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks usa livros de receitas do Chef para gerenciar tarefas como a instalação e configuração de pacotes e a implantação de aplicações. Esta seção descreve como usar livros de receitas com o AWS OpsWorks Stacks. Para obter mais informações, consulte Chef.

> **Note**
>
> AWS OpsWorks O Stacks atualmente oferece suporte ao Chef versões 12, 11.10.4, 11.4.4 e 0.9.15.5. Porém, o Chef 0.9.15.5 está obsoleto e não é recomendável usá-lo para novas pilhas. Para sua conveniência, essas versões são geralmente chamadas apenas pelos números de versão maior e menor. As pilhas que executam o Chef 0.9 ou 11.4 usam o Chef Solo e as pilhas que executam o Chef 12 ou 11.10 usam o Chef Client no modo local. Para pilhas do Linux, você pode usar o Gerenciador de configuração para especificar a versão do Chef que deseja utilizar quando cria uma pilha. As pilhas do Windows devem usar Chef 12.2. Para obter mais informações, incluindo as orientações necessárias para migrar pilhas para versões mais recentes do Chef, consulte Versões do Chef.

Tópicos

- Repositórios de livro de receitas
- Versões do Chef
- Versões do Ruby
- Instalação de livros de receitas personalizados
- Atualizar livros de receitas personalizadas
- Execução de receitas

Repositórios de livro de receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Seus livros de receita personalizados devem ser armazenados em um repositório online, seja um arquivo como um .zip ou um gerente de controle de fonte como Git. Uma pilha pode ter apenas um repositório de livro de receitas personalizado, mas o repositório pode conter qualquer número de

livros de receita. Quando você instala ou atualiza os livros de receitas, o AWS OpsWorks Stacks instala o repositório inteiro em um cache local em cada uma das instâncias da pilha. Quando uma instância precisa, por exemplo, executar uma ou mais receitas, ela usa o código do cache local.

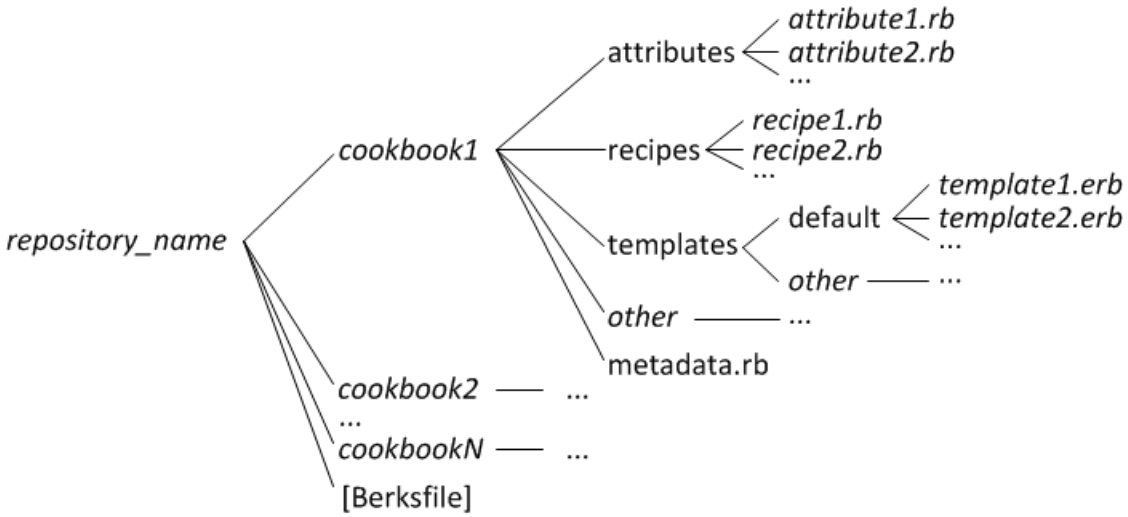
A seguir veja a descrição de como estruturar seu repositório de livro de receitas, que depende do tipo. O texto em itálico nas ilustrações representa nomes de diretório e arquivo definidos pelo usuário, incluindo o nome do repositório ou arquivo.

Gerente do controle de fonte

AWS OpsWorks O Stacks oferece suporte aos seguintes gerenciadores de controle de origem:

- Pilhas do Linux: Git e Subversion
- Pilhas do Windows: Git

Veja a seguir a estrutura de diretório e arquivo necessários:



- Os diretórios dos livros de receita devem ser todos de excelente nível.

Arquivo

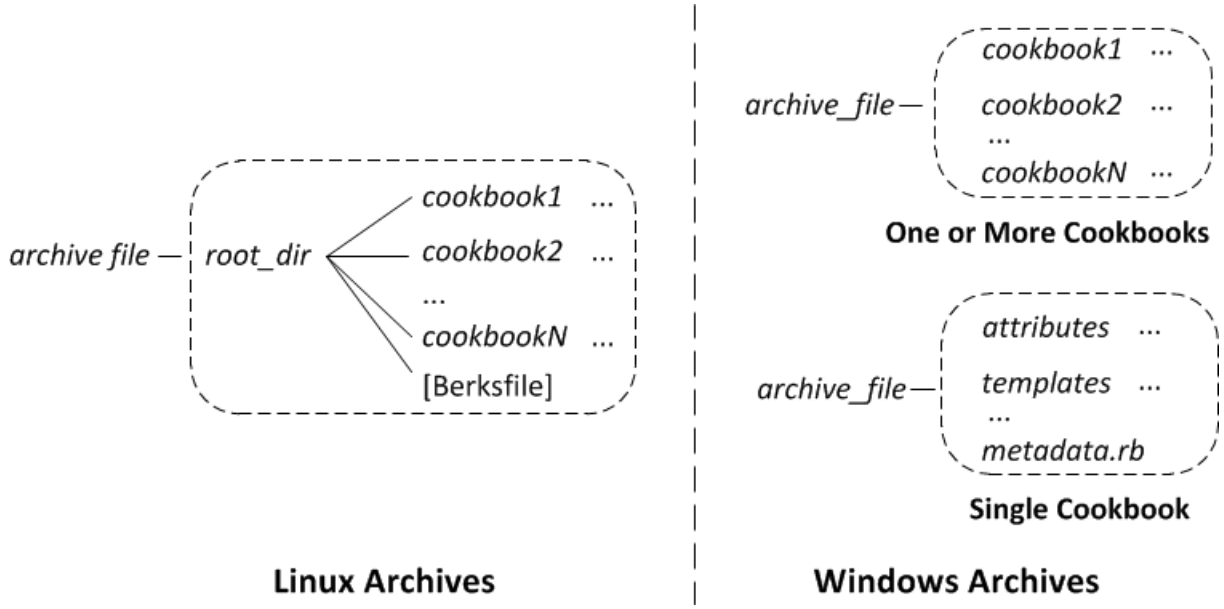
AWS OpsWorks O Stacks oferece suporte aos seguintes arquivos:

- Pilhas do Linux: arquivos zip, gzip, bzip2 ou tarball, armazenados no Amazon S3 ou em um site (arquivamento HTTP).

  AWS OpsWorks O Stacks não é compatível com pacotes não compactados.

- Pilhas do Windows: arquivos zip e tgz (tar compactados em gzip), armazenados no Amazon S3.

Veja a seguir o diretório necessário e estrutura de arquivo, que dependem se você está executando uma pilha do Linux ou Windows. A estrutura do livro de receitas é a mesma para os repositórios de SCM, então é representada por três-pontos (...).



- Pilhas do Linux: os diretórios dos livros de receitas devem estar contidos em um diretório raiz.
- Pilhas do Windows: os livros de receita devem estar no topo de nível do arquivo.

  Se tiver apenas um livro de receitas, você pode omitir opcionalmente o diretório do livro de receitas e colocar os seus arquivos em nível superior. Nesse caso, o AWS OpsWorks Stacks obtém o nome do livro de receitas do metadata.rb.

Cada diretório de livro de receitas tem pelo menos um e geralmente todos desses diretórios e arquivos padrão a seguir, que devem usar nomes padrão:

- `attributes`: os arquivos de atributos do livro de receitas.
- `recipes`: os arquivos de receita do livro de receitas.
- `templates`: os arquivos modelo do livro de receitas.
- *`other`*: diretórios opcionais definidos pelo usuário que contêm outros tipos de arquivos, como definições ou especificações.
- `metadata.rb`: os metadados do livro de receitas.

  Para Chef 11.10 e posterior, se suas receitas dependerem de outros livros de receitas, você deve incluir declarações de `depends` correspondentes no arquivo de seu livro de receitas de `metadata.rb`. Por exemplo, se o seu livro de receitas inclui uma receita com a instrução

`include_recipe anothercookbook::somerecipe`, o arquivo `metadata.rb` do livro de receitas deve incluir a seguinte linha: `depends "anothercookbook"`. Para obter mais informações, consulte Sobre metadados do livro de receitas.

Os modelos devem estar em um subdiretório do diretório `templates`, que contém pelo menos um e opcionalmente múltiplos subdiretórios. Esses subdiretórios podem, opcionalmente, ter subdiretórios também.

- Os modelos geralmente têm um subdiretório `default`, que contém os arquivos de modelo que Chef usa por padrão.
- other representa subdiretórios opcionais que podem ser usados para modelos específicos de sistemas operacionais.
- Chef usa automaticamente o modelo do subdiretório apropriado, com base em convenções de nome que são descritas em Especificidade do arquivo. Por exemplo, para os sistemas operacionais Linux e , você pode colocar modelos específicos de sistema operacional em subdiretórios nomeados `amazon`amazon ou `ubuntu`ubuntu, respectivamente.

Os detalhes de como lidar com livros de receita personalizados dependem de seu tipo de repositório preferido.

Para usar um arquivo

1. Implemente seu livro de receitas usando a estrutura de pasta mostrada na seção anterior.
2. Crie um arquivo comprimido e carregue-o para um bucket Amazon S3 ou site.

   Se você atualizar seus livros de receita, deve criar e carregar um novo arquivo de armazenamento. O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Para usar um SCM

1. Configure um repositório de Git ou Subversion usando a estrutura mostrada anteriormente.
2. Opcionalmente, use os recursos de controle da versão para implementar diversas ramificações ou versões.

Se você atualizar seus livros de receitas, poderá fazer isso em uma nova ramificação e apenas OpsWorks direcionar para usar a nova versão. Também é possível especificar versões marcadas em particular. Para obter detalhes, consulte Especificação de um repositório de livro de receitas personalizado.

Instalação de livros de receitas personalizadosdescreve como fazer com que o AWS OpsWorks Stacks instale seu repositório de livros de receitas nas instâncias da pilha.

> **Important**
>
> Depois de atualizar os livros de receitas existentes no repositório, você deve executar o comando `update_cookbooks` stack para direcionar o AWS OpsWorks Stacks a atualizar o cache local de cada instância online. Para obter mais informações, consulte Executa comandos de pilha.

Versões do Chef

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks suporta várias versões do Chef. Você seleciona a versão ao criar a pilha. AWS OpsWorks Em seguida, o Stacks instala a versão escolhida do Chef em todas as instâncias da pilha juntamente com um conjunto de receitas integradas que são compatíveis com essa versão. Se você instalar uma ou mais receitas personalizadas, elas devem ser compatíveis com a versão do Chef usada pela pilha.

AWS OpsWorks O Stacks atualmente oferece suporte ao Chef versões 12, 11.10, 11.4 e 0.9 para pilhas do Linux e ao Chef 12.2 (atualmente Chef 12.22) para pilhas do Windows. Para sua conveniência, essas versões são geralmente chamadas apenas pelos números de versão maior e menor. Para pilhas do Linux, você pode usar o Gerenciador de configuração para especificar a

versão do Chef que deseja utilizar quando cria uma pilha. As pilhas do Windows devem usar Chef 12.2. Para obter mais informações, incluindo as orientações necessárias para migrar pilhas para versões mais recentes do Chef, consulte Versões do Chef. Para obter informações completas sobre as versões, consulte AWS OpsWorks Sistemas operacionais de pilha.

Chef 12.2

O suporte ao Chef 12.2 foi introduzido em maio de 2015 e é usado somente por pilhas do Windows. A versão atual do Chef nas pilhas do Windows é o Chef 12.22. Ela é executada com o Ruby 2.3.6 e usa o cliente do Chef no modo local que executa um servidor do Chef na memória local chamado chef-zero. A presença desse servidor permite que as receitas utilizem a pesquisa e os recipientes de dados do Chef. O suporte apresenta algumas limitações que são descritas em Implementação de receitas: Chef 12.2, mas você pode executar vários livros de receitas da comunidade, sem alterações.

Chef 12

O suporte ao Chef 12 foi introduzido em dezembro de 2015 e é usado somente por pilhas do Linux. Ele é executado com o Ruby 2.1.6 ou 2.2.3 e usa o cliente do Chef no modo local, o que permite que as receitas utilizem a pesquisa e os recipientes de dados do Chef. Para obter mais informações, consulte AWS OpsWorks Sistemas operacionais de pilha.

Chef 11.10

O suporte ao Chef 11.10 foi introduzido em março de 2014, e é usado somente por pilhas do Linux. Ele é executado com o Ruby 2.0.0 e usa o cliente do Chef no modo local, o que permite que as receitas utilizem a pesquisa e os recipientes de dados do Chef. O suporte apresenta algumas limitações que são descritas em Implementação de receitas: Chef 11.10, mas você pode executar vários livros de receitas da comunidade, sem alterações. Você também pode usar o Berkshelf para gerenciar suas dependências dos livros de receitas. As versões compatíveis do Berkshelf dependem do sistema operacional. Para obter mais informações, consulte AWS OpsWorks Sistemas operacionais de pilha. Não é possível criar pilhas do CentOS que usam o Chef 11.10.

Chef 11.4

O suporte ao Chef 11.4 foi introduzido em julho de 2013 e é usado somente por pilhas do Linux. Ele é executado com o Ruby 1.8.7 e usa o chef-solo, que não oferece suporte à pesquisa ou aos recipientes de dados do Chef. Muitas vezes, você pode usar livros de receitas da comunidade que dependem desses recursos com o AWS OpsWorks Stacks, mas deve modificá-los conforme descrito em. Migração para uma nova versão do Chef Não é possível criar pilhas do CentOS que

usam o Chef 11.4. As pilhas do Chef 11.4 não são compatíveis com endpoints regionais fora do Leste dos EUA (N. da Virgínia).

Chef 0.9

O Chef 0.9 é usado somente por pilhas do Linux e não é mais compatível. Observe os seguintes detalhes:

- Não é possível usar o console para criar uma nova pilha do Chef 0.9.

  Você deve usar a CLI ou API, ou criar uma pilha usando uma versão diferente do Chef e, em seguida, editar a configuração da pilha.
- Os novos recursos de AWS OpsWorks pilhas não estão disponíveis para pilhas do Chef 0.9.
- As novas versões de sistemas operacionais fornecerão apenas suporte limitado para pilhas do Chef 0.9.

  O Amazon Linux 2014.09 e suas versões posteriores não são compatíveis com as pilhas do Chef 0.9 com camadas Rails App Server que dependem do Ruby 1.8.7.
- As novas regiões da AWS, incluindo a Europa (Frankfurt), não são compatíveis com as pilhas do Chef 0.9.

> **Note**
>
> Não é recomendável usar o Chef 0.9 para novas pilhas. Você deve migrar as pilhas existentes para a última versão do Chef o mais rápido possível.

Se você quiser usar livros de receitas da comunidade com AWS OpsWorks Stacks, recomendamos que você especifique o Chef 12 para novas pilhas Linux e migre suas pilhas Linux existentes para o Chef 12. Você pode usar o console, a API ou a CLI do AWS OpsWorks Stacks para migrar suas pilhas existentes para uma versão mais recente do Chef. Para obter mais informações, consulte Migração para uma nova versão do Chef.

Tópicos

- Implementação de receitas em pilhas do Chef 12.2
- Implementação de receitas em pilhas do Chef 12
- Implementação de receitas em pilhas do Chef 11.10
- Implementação de receitas em pilhas do Chef 11.4

- Migração de uma pilha existente do Linux para uma nova versão do Chef

Implementação de receitas em pilhas do Chef 12.2

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Chef 12.2 (atualmente Chef 12.22) está disponível somente em pilhas do Windows, o qual deve executar essa versão do Chef.

- As receitas devem usar atributos e recursos específicos do Windows para algumas finalidades.

  Para obter mais informações, consulte Chef para Microsoft Windows.
- As execuções do Chef usam o Ruby 2.3.6 e, portanto, suas receitam podem usar a nova sintaxe do Ruby.
- As receitas podem usar a pesquisa e os recipientes de dados do Chef.

  As pilhas do Chef 12.2 podem usar muitos livros de receitas da comunidade sem alterá-los. Para obter mais informações, consulte Utilização da pesquisa do Chef e Utilização de recipientes de dados.
- A maioria dos atributos de configuração e implantação de pilha descritos em AWS OpsWorks Referência do Stacks Data Bag e Atributos de livros de receitas integrados estão disponíveis nas receitas do Windows.

  Você pode usar a pesquisa do Chef para obter os valores desses atributos. Para obter um exemplo, consulte Obter valores de atributo com a pesquisa do Chef. Para obter uma lista de atributos, consulte AWS OpsWorks Referência do Stacks Data Bag.

Implementação de receitas em pilhas do Chef 12

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As pilhas do Chef 12 apresentam as seguintes vantagens em relação às pilhas do Chef 11.10:

- As execuções do Chef usam o Ruby 2.1.6 e, portanto, suas receitam podem usar a nova sintaxe do Ruby.
- As pilhas do Chef 12 podem usar ainda mais livros de receitas da comunidade sem alterá-los. Com a ausência dos livros de receitas integrados, não há mais chance de conflito de nome entre os livros de receitas integrados e os livros de receitas personalizados.
- Você não está mais limitado às versões da Berkshelf para as quais o AWS OpsWorks Stacks forneceu pacotes pré-construídos. O Berkshelf não está mais instalado nas instâncias do AWS OpsWorks Stacks no Chef 12. Em vez disso, você pode usar qualquer versão do Berkshelf na sua estação de trabalho local.
- Agora, há uma separação clara entre os livros de receitas integrados que o AWS OpsWorks Stacks fornece com o Chef 12 (Elastic Load Balancing, Amazon RDS, e Amazon ECS) e livros de receitas personalizados. Isso torna mais fácil a solução de problemas em execuções com falha do Chef.

Implementação de receitas em pilhas do Chef 11.10

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As pilhas do Chef 11.10 apresentam as seguintes vantagens em relação às stacks do Chef 11.4:

- As execuções do Chef usam o Ruby 2.0.0 e, portanto, suas receitam podem usar a nova sintaxe do Ruby.
- As receitas podem usar a pesquisa e os recipientes de dados do Chef.

  As pilhas do Chef 11.10 podem usar muitos livros de receitas da comunidade sem alterá-los.
- Você pode usar o Berkshelf para gerenciar os livros de receitas.

  O Berkshelf oferece uma maneira muito mais flexível de gerenciar os seus livros de receitas personalizados e usar os livros de receitas da comunidade em uma pilha.
- Os livros de receitas devem ter suas dependências declaradas no `metadata.rb`.

  Se o seu livro de receitas depende de outro livro de receitas, você deve incluir essa dependência no arquivo `metadata.rb` do livro de receitas. Por exemplo, se o seu livro de receitas inclui uma receita com a instrução `include_recipe anothercookbook::somerecipe`, o arquivo `metadata.rb` do livro de receitas deve incluir a seguinte linha: `depends "anothercookbook"`.
- AWS OpsWorks O Stacks só instalará um cliente MySQL nas instâncias de uma pilha se a pilha incluir uma camada MySQL.
- AWS OpsWorks O Stacks só instalará um cliente Ganglia nas instâncias de uma pilha se a pilha incluir uma camada Ganglia.
- Se uma implantação executar a `bundle install` e a instalação falhar, a implantação também falha.

> **Important**
>
> Não reutilize nomes de livros de receitas integrados para livros de receitas personalizados ou da comunidade. Livros de receitas personalizados com o mesmo nome de livros de receitas integrados podem falhar. Para obter uma lista completa dos livros de receitas integrados que estão disponíveis com as pilhas Chef 11.10, 11.4 e 0.9, consulte o repositório opsworks-cookbooks em. GitHub
>
> Livros de receitas com caracteres não ASCII que são executados com êxito nas pilhas do Chef 0.9 e 11.4 podem falhar quando executados em uma pilha do Chef 11.10. Isso ocorre porque as pilhas do Chef 11.10 usam o Ruby 2.0.0 para execuções do Chef, que é muito mais rigoroso em relação à codificação do que o Ruby 1.8.7. Para garantir que os livros de receitas executem com êxito nas pilhas do Chef 11.10, todo arquivo que utiliza caracteres

não ASCII deve ter um comentário na parte superior informando sobre a codificação. Por exemplo, para a codificação UTF-8, o comentário seria `# encoding: UTF-8`. Para obter mais informações sobre a codificação no Ruby 2.0.0, consulte Codificação.

Tópicos

- Instalação e precedência em livros de receitas
- Utilização da pesquisa do Chef
- Utilização de recipientes de dados
- Utilização do Berkshelf

Instalação e precedência em livros de receitas

O procedimento para instalar os livros de receitas do AWS OpsWorks Stacks funciona de forma um pouco diferente para as pilhas do Chef 11.10 do que para as versões anteriores do Chef. Para pilhas do Chef 11.10, depois que o AWS OpsWorks Stacks instala os livros de receitas integrados, personalizados e Berkshelf, ele os mescla em um diretório comum na seguinte ordem:

1. Livros de receitas integrados.
2. Livros de receitas do Berkshelf, se houver.
3. Livros de receitas personalizados, se houver.

Quando o AWS OpsWorks Stacks realiza essa mesclagem, ele copia todo o conteúdo dos diretórios, incluindo receitas. Se houver duplicatas, as seguintes regras se aplicam:

- O conteúdo dos livros de receitas do Berkshelf tem preferência sobre os livros de receitas integrados.
- O conteúdo dos livros de receitas personalizados tem preferência sobre os livros de receitas do Berkshelf.

Para ilustrar como esse processo funciona, considere o seguinte cenário, onde todos os três diretórios de livros de receitas contêm um livro de receitas chamado `mycookbook`:

- Livros de receitas integrados: `mycookbook` inclui um arquivo de atributos chamado `someattributes.rb`, um arquivo de modelo chamado `sometemplate.erb` e uma receita chamada `somerecipe.rb`.

- Livros de receitas do Berkshelf: `mycookbook` inclui `sometemplate.erb` e `somerecipe.rb`.
- Livros de receitas personalizados: `mycookbook` inclui `somerecipe.rb`.

O livro de receitas mesclado contém o seguinte:

- `someattributes.rb` do livro de receitas integrado.
- `sometemplate.erb` do livro de receitas do Berkshelf.
- `somerecipe.rb` do livro de receitas personalizado.

> **Important**
>
> Você não deve personalizar sua pilha do Chef 11.10 copiando um livro de receitas integrado inteiro para o repositório e, em seguida, modificando partes dele. Se fizer isso, você estará substituindo todo o livro de receitas integrado, incluindo as receitas. Se o AWS OpsWorks Stacks atualizar esse livro de receitas, sua pilha não obterá os benefícios dessas atualizações, a menos que você atualize manualmente sua cópia privada. Para obter mais informações sobre como personalizar as pilhas, consulte Personalizando pilhas AWS OpsWorks.

Utilização da pesquisa do Chef

Você pode utilizar o método `search` do Chef em suas receitas para consultar os dados da pilha. Você usa a mesma sintaxe que usaria para o Chef Server, mas o AWS OpsWorks Stacks obtém os dados do objeto do nó local em vez de consultar um servidor Chef. Esses dados incluem:

- Os atributos de configuração e implantação da pilha da instância.
- Os atributos dos arquivos de atributos dos livros de receitas integrados e personalizados da instância.
- Os dados do sistema coletados pelo Ohai.

Os atributos de configuração e implantação da pilha contêm a maioria das informações que as receitas obtém normalmente por meio de pesquisa, incluindo dados como nomes de host e endereços IP, para cada instância on-line na pilha. AWS OpsWorks O Stacks atualiza esses atributos em cada evento do ciclo de vida, o que garante que eles representem com precisão o estado atual da pilha. Isso significa que você pode, muitas vezes, usar receitas da comunidade que sejam

dependentes de pesquisa em sua pilha sem alterá-las. O método de pesquisa ainda retorna os dados apropriados; o que ocorre é que eles estão vindo dos atributos de configuração e implantação da pilha, em vez de um servidor.

A principal limitação da pesquisa de AWS OpsWorks pilhas é que manipula somente os dados no objeto do nó local, em particular a configuração da pilha e os atributos de implantação. Por esse motivo, os seguintes tipos de dados podem não estar disponíveis por meio de pesquisa:

- Atributos definidos localmente em outras instâncias.

  Se uma receita define um atributo localmente, essas informações não são reportadas ao serviço AWS OpsWorks Stacks, então você não pode acessar esses dados de outras instâncias usando a pesquisa.
- Atributos `deploy` personalizados.

  Você pode especificar um JSON personalizado quando implanta um aplicativo e os atributos correspondentes são instalados nas instâncias da pilha para aquela implantação. No entanto, se você implantar apenas em instâncias selecionadas, os atributos são instalados apenas nessas instâncias. As consultas sobre esses atributos JSON personalizados falharão em todas as outras instâncias. Além disso, os atributos personalizados são incluídos no JSON de configuração e implantação da pilha apenas para essa implantação específica. Eles estarão acessíveis apenas até que o próximo evento de ciclo de vida instale um novo conjunto de atributos de configuração e implantação da pilha. Observe que quando você especifica um JSON personalizado para a pilha, os atributos são instalados em cada instância para cada evento de ciclo de vida e estão sempre acessíveis por meio de pesquisa.
- Dados do Ohai de outras instâncias.

  A ferramenta Ohai do Chef obtém uma variedade de dados de sistema de uma instância e os adiciona ao objeto de nó. Esses dados são armazenados localmente e não são passados de volta ao serviço do AWS OpsWorks Stacks, portanto a pesquisa não pode acessar os dados do Ohai a partir de outras instâncias. No entanto, alguns desses dados podem ser incluídos nos atributos de configuração e implantação da pilha.
- Instâncias off-line.

  Os atributos de configuração e implantação da pilha somente contêm dados de instâncias online.

O trecho da receita a seguir mostra como obter o endereço IP privado de uma instância da camada de PHP usando a pesquisa.

```
appserver = search(:node, "role:php-app").first
Chef::Log.info("The private IP is '#{appserver[:private_ip]}'")
```

> **Note**
>
> Quando o AWS OpsWorks Stacks adiciona os atributos de configuração e implantação da pilha ao objeto do nó, ele realmente cria dois conjuntos de atributos de camada, cada um com os mesmos dados. Um conjunto está no `layers` namespace, que é como o AWS OpsWorks Stacks armazena os dados. O outro conjunto fica no namespace `role`, que é como o servidor Chef armazena os dados equivalentes. O objetivo do `role` namespace é permitir que o código de pesquisa que foi implementado para o Chef Server seja executado em uma instância do AWS OpsWorks Stacks. Se você estiver escrevendo código especificamente para AWS OpsWorks Stacks, poderá usar um `layers:php-app` ou `role:php-app` o exemplo anterior e `search` retornará o mesmo resultado.

Utilização de recipientes de dados

Você pode usar o método `data_bag_item` do Chef nas suas receitas para consultar informações em um recipiente de dados. A sintaxe usada é a mesma que você usaria para o servidor Chef, mas o AWS OpsWorks Stacks obtém os dados dos atributos de configuração e implantação da pilha da instância. No entanto, o AWS OpsWorks Stacks atualmente não suporta ambientes Chef, então `node.chef_environment` sempre retorna`_default`.

Você pode criar um recipiente de dados usando um JSON personalizado para adicionar um ou mais atributos ao atributo `[:opsworks][:data_bags]`. O exemplo a seguir mostra o formato geral de criação de um recipiente de dados em um JSON personalizado.

> **Note**
>
> Não é possível criar um recipiente de dados adicionando-o ao repositório de livros de receitas. Você deve usar um JSON personalizado.

```
{
  "opsworks": {
```

```
    "data_bags": {
      "bag_name1": {
        "item_name1: {
          "key1" : “value1”,
          "key2" : “value2”,
          ...
        }
      },
      "bag_name2": {
        "item_name1": {
          "key1" : “value1”,
          "key2" : “value2”,
          ...
        }
      },
      ...
    }
  }
}
```

Normalmente, você especifica um JSON personalizado para a pilha, que instala os atributos personalizados em cada instância para cada evento de ciclo de vida subsequente. Você também pode especificar um JSON personalizado quando implanta um aplicativo, mas os atributos são instalados apenas para essa implantação e só podem ser instalados para um conjunto de instâncias selecionado. Para obter mais informações, consulte Implementação de aplicativos.

O seguinte exemplo de JSON personalizado cria um recipiente de dados chamado `myapp`. Ele possui um item `mysql` com dois pares de chave e valor.

```
{ "opsworks": {
    "data_bags": {
      "myapp": {
        "mysql": {
          "username": "default-user",
          "password": "default-pass"
        }
      }
    }
  }
}
```

Para usar os dados em sua receita, você pode chamar o `data_bag_item` e passar para ele os nomes do recipiente de dados e do valor, como mostra o trecho a seguir.

```
mything = data_bag_item("myapp", "mysql")
Chef::Log.info("The username is '#{mything['username']}' ")
```

Para modificar os dados no recipiente de dados, basta modificar o JSON personalizado e ele será instalado nas instâncias da pilha para o próximo evento de ciclo de vida.

Utilização do Berkshelf

Em pilhas do Chef 0.9 e Chef 11.4, você pode instalar apenas um repositório de livros de receitas personalizados. Em pilhas do Chef 11.10, você pode usar o Berkshelf para gerenciar os livros de receitas e suas dependências, o que permite a você instalar livros de receitas de vários repositórios. (Para ter mais informações, consulte Empacotar dependências do livro de receitas localmente.) Em particular, com o Berkshelf, você pode instalar livros de receitas da comunidade AWS OpsWorks compatíveis com Stacks diretamente de seus repositórios, em vez de precisar copiá-los para seu repositório de livros de receitas personalizado. As versões compatíveis do Berkshelf dependem do sistema operacional. Para obter mais informações, consulte AWS OpsWorks Sistemas operacionais de pilha.

Para usar o Berkshelf, você deve habilitá-lo de forma explícita, conforme descrito em Instalação de livros de receitas personalizados. Em seguida, inclua um arquivo `Berksfile` no diretório raiz do repositório do seu livro de receitas para especificar os livros de receitas a serem instalados.

Para especificar a origem externa de um livros de receitas em um Berksfile, inclua um atributo de origem na parte superior do arquivo que especifica o URL do repositório padrão. A Berkshelf procurará os livros de receitas na fonte, a URLs menos que você especifique explicitamente um repositório. Em seguida, inclua uma linha para cada livro de receitas que você deseja instalar no seguinte formato:

```
cookbook 'cookbook_name', ['>= cookbook_version'], [cookbook_options]
```

Os campos que vêm logo após `cookbook` especificam o livro de receitas específico.

- *`cookbook_name`*— (Obrigatório) Especifica o nome do livro de receitas.

Se você não incluir nenhum outro campo, a Berkshelf instala o livro de receitas da fonte especificada. URLs

- *`cookbook_version`*— (Opcional) Especifica a versão ou versões do livro de receitas.

  Você pode usar um prefixo como = ou >= para especificar uma versão específica ou um intervalo de versões aceitas. Se você não especificar uma versão, o Berkshelf instala a mais recente.

- *`cookbook_options`*: (Opcional) O campo final é um hash contendo um ou mais pares de chave e valor que especificam opções como o local do repositório.

  Por exemplo, você pode incluir uma chave `git` para designar um repositório Git específico e uma chave `tag` para designar uma ramificação específica do repositório. Especificar a ramificação do repositório é geralmente a melhor forma de garantir que você está instalando o livro de receitas de sua preferência.

> **Important**
>
> Não faça a declaração de livros de receitas incluindo uma linha de `metadata` em seu Berksfile ou incluindo a declaração das dependências do livro de receitas no `metadata.rb`. Para que isso funcione corretamente, ambos os arquivos devem estar no mesmo diretório. Com o AWS OpsWorks Stacks, o Berksfile deve estar no diretório raiz do repositório, mas os `metadata.rb` arquivos devem estar em seus respectivos diretórios do livro de receitas. Em vez disso, você deve incluir as declarações explícitas dos livros de receitas externos no Berksfile.

Veja a seguir o exemplo de um Berksfile que mostra diferentes maneiras para especificar livros de receitas. Para obter mais informações sobre como criar um Berksfile, consulte Berkshelf.

```
source "https://supermarket.chef.io"

cookbook 'apt'
cookbook 'bluepill', '>= 2.3.1'
cookbook 'ark', git: 'git://github.com/opscode-cookbooks/ark.git'
cookbook 'build-essential', '>= 1.4.2', git: 'git://github.com/opscode-cookbooks/build-
essential.git', tag: 'v1.4.2'
```

Este arquivo instala os seguintes livros de receitas:

- A versão mais recente do `apt` do repositório de livros de receitas da comunidade.
- A versão mais recente do `bluepill` dos livros de receitas da comunidade, desde que seja a versão 2.3.1 ou posterior.
- A versão mais recente do `ark` de um repositório especificado.

  A URL deste exemplo é para um repositório público de livros de receitas da comunidade em GitHub, mas você pode instalar livros de receitas de outros repositórios, incluindo repositórios privados. Para obter mais informações, consulte Berkshelf.
- O livro de receitas `build-essential` da ramificação v1.4.2 do repositório especificado.

Um repositório de livros de receitas personalizados pode conter livros de receitas personalizados, além de um Berksfile. Nesse caso, o AWS OpsWorks Stacks instala os dois conjuntos de livros de receitas, o que significa que uma instância pode ter até três repositórios de livros de receitas.

- Os livros de receitas integrados são instalados em `/opt/aws/opsworks/current/cookbooks`.
- Se o seu repositório de livros de receitas personalizados contém livros de receitas, eles estão instalados em `/opt/aws/opsworks/current/site-cookbooks`.
- Se você habilitou o Berkshelf e seu repositório de livros de receitas personalizados contém um Berksfile, os livros de receitas especificados são instalados em `/opt/aws/opsworks/current/berkshelf-cookbooks`.

Os livros de receitas integrados e seus livros de receitas personalizados são instalados em cada instância durante a instalação e não são atualizados posteriormente, a menos que você execute manualmente o comando da pilha Update Custom Cookbooks. AWS OpsWorks O Stacks funciona `berks install` em cada execução do Chef e, portanto, seus livros de receitas do Berkshelf são atualizados em cada evento do ciclo de vida, de acordo com as seguintes regras:

- Se você tiver uma nova versão de um livro de receitas no repositório, esta operação atualiza o livro de receitas do repositório.
- Caso contrário, essa operação atualiza os livros de receitas do Berkshelf de um cache local.

> **Note**
>
> A operação sobrescreve os livros de receitas do Berkshelf e, portanto, se você tiver modificado as cópias locais de quaisquer livros de receitas, essas alterações serão sobrescritas. Para obter mais informações, consulte Berkshelf

Você também pode atualizar seus livros de receitas do Berkshelf executando o comando da pilhaUpdate Custom Cookbooks, que atualiza ambos os livros de receitas do Berkshelf e personalizados.

Implementação de receitas em pilhas do Chef 11.4

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Important**
>
> Não reutilize nomes de livros de receitas integrados para livros de receitas personalizados ou da comunidade. Livros de receitas personalizados com o mesmo nome de livros de receitas integrados podem falhar. Para obter uma lista completa dos livros de receitas integrados que estão disponíveis com as pilhas Chef 11.10, 11.4 e 0.9, consulte o repositório opsworks-cookbooks em. GitHub

A principal limitação das pilhas do Chef 11.4 é que as receitas não podem usar a pesquisa ou os recipientes de dados do Chef. No entanto, o AWS OpsWorks Stacks instala os atributos de configuração e implantação da pilha em cada instância que contêm muitas das informações que você obteria com a pesquisa, incluindo as seguintes:

- Dados definidos pelo usuário do console, como nomes de hosts ou de aplicativos.

- Dados de configuração da pilha gerados pelo serviço AWS OpsWorks Stacks, como camadas, aplicativos e instâncias da pilha, e detalhes sobre cada instância, como o endereço IP.
- Atributos de JSON personalizados que contêm dados fornecidos pelo usuário e podem servir ao mesmo propósito que os recipientes de dados.

AWS OpsWorks O Stacks instala uma versão atual dos atributos de configuração e implantação da pilha em cada instância para cada evento do ciclo de vida, antes de iniciar a execução do Chef do evento. Os dados estão disponíveis para as receitas com a sintaxe padrão `node[:attribute][:child_attribute][...]`. Por exemplo, os atributos de configuração e implantação da pilha incluem o nome da pilha, `node[:opsworks][:stack][:name]`.

O trecho a seguir, extraído de uma das receitas integradas, obtém o nome da pilha e utiliza-o para criar um arquivo de configuração.

```
template '/etc/ganglia/gmetad.conf' do
  source 'gmetad.conf.erb'
  mode '0644'
  variables :stack_name => node[:opsworks][:stack][:name]
  notifies :restart, "service[gmetad]"
end
```

Muitos dos valores de atributos de configuração e implantação da pilha contêm vários atributos. Você deve fazer a iteração nesses atributos para obter as informações de que precisa. O exemplo a seguir mostra um trecho com atributos de configuração e implantação da pilha, que são representados como objetos JSON por conveniência. Ele contém um atributo de nível superior `deploy`, que contém um atributo para cada um dos aplicativos da pilha, designados pelo nome abreviado do aplicativo.

```
{
  ...
  "deploy": {
    "app1_shortname": {
      "document_root": "app1_root",
      "deploy_to": "deploy_directory",
      "application_type": "php",
      ...
    },
    "app2_shortname": {
```

```
      "document_root": "app2_root",
      ...
    }
  },
  ...
}
```

Cada atributo de aplicativo contém um conjunto de atributos que caracterizam o aplicativo. Por exemplo, o atributo `deploy_to` representa o diretório de implantação aplicativo. O trecho a seguir define o usuário, o grupo e o caminho para o diretório de implantação de cada aplicativo.

```
node[:deploy].each do |application, deploy|
  opsworks_deploy_dir do
    user deploy[:user]
    group deploy[:group]
    path deploy[:deploy_to]
  end
  ...
end
```

Para obter mais informações sobre os atributos de configuração e implantação da pilha, consulte Personalizando pilhas AWS OpsWorks. Para obter mais informações sobre a implantação de diretórios, consulte Receitas de implantação.

As pilhas do Chef 11.4 não são compatíveis com os recipientes de dados, mas você pode adicionar dados arbitrários aos atributos de configuração e implantação da stack especificando um JSON personalizado. Suas receitas podem, dessa forma, acessar os dados usando a sintaxe padrão do nó do Chef. Para obter mais informações, consulte Usar JSON personalizado.

Se você precisa da funcionalidade de um recipiente de dados criptografado, uma opção é armazenar os atributos confidenciais em um local seguro, como um bucket privado do Amazon S3. Suas receitas podem então usar o SDK do AWS Ruby, que vem instalado em todas as instâncias do AWS OpsWorks Stacks, para fazer download dos dados do bucket.

> **Note**
>
> Cada instância do AWS OpsWorks Stacks tem um perfil de instância. O perfil do IAM associado especifica quais recursos do AWS podem ser acessados por aplicativos que estão sendo executados na instância. Para que suas receitas possam acessar um bucket

do Amazon S3, a política de atribuição do perfil deve incluir uma instrução semelhante à seguinte, que concede permissão para a recuperação de arquivos de um bucket especificado.

```
"Action": ["s3:GetObject"],
"Effect": "Allow",
"Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*",
```

Para obter mais informações sobre os perfis de instância, consulte Especificação de permissões para aplicativos executados em instâncias EC2 .

Migração de uma pilha existente do Linux para uma nova versão do Chef

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode usar o console, a API ou a CLI do AWS OpsWorks Stacks para migrar suas pilhas do Linux para uma versão mais recente do Chef. No entanto, pode ser necessário modificar suas receitas para que sejam compatíveis com a versão mais recente. Durante a preparação para a migração de uma pilha, considere o seguinte.

- Você não pode alterar as versões da AWS OpsWorks pilha de pilhas do Chef 11 para o Chef 12 editando ou clonando a pilha. A atualização de uma versão principal do Chef não pode ser executada usando os procedimentos desta seção. Para obter mais informações sobre a transição do Chef 11.10 para o Chef 12, consulte Implementação de receitas: Chef 12.
- A transição de uma versão do Chef para outra envolve uma série de alterações, e algumas delas são alterações interruptivas.

  Para obter mais informações sobre a transição do Chef 0.9 para o Chef 11.4, consulte Migração para uma nova versão do Chef. Para obter mais informações sobre a transição do Chef 11.4 para

o Chef 11.10, consulte Implementação de receitas: Chef 11.10. Para obter mais informações sobre a transição do Chef 11.10 para o Chef 12, consulte Implementação de receitas: Chef 12.

- As execuções do Chef usam versões diferentes do Ruby nas pilhas do Chef 0.9 e Chef 11.4 (Ruby 1.8.7), do Chef 11.10 (Ruby 2.0.0) e do Chef 12 (Ruby 2.1.6).

  Para obter mais informações, consulte Versões do Ruby.

- As pilhas do Chef 11.10 lidam com a instalação de livros de receitas de forma diferente das pilhas do Chef 0.9 ou do Chef 11.4.

  Essa diferença pode causar problemas durante a migração de pilhas que usam livros de receitas personalizados para o Chef 11.10. Para obter mais informações, consulte Instalação e precedência em livros de receitas.

Veja a seguir as diretrizes recomendadas para a migração de uma pilha do Chef para uma versão mais recente do Chef:

Para migrar uma pilha para uma versão mais recente do Chef

1. Faça a clonagem da sua pilha de produção. Na página Clone Stack, clique em Advanced>> para exibir a seção Configuration Management, e mude Chef version para a próxima versão mais recente.

   > **Note**
   >
   > Se você estiver começando de uma pilha do Chef 0.9, não poderá fazer upgrade diretamente para o Chef 11.10. Primeiro, será necessário fazer upgrade para o Chef 11.4. Se quiser migrar sua pilha para o Chef 11.10 antes de testar suas receitas, aguarde 20 minutos para que a atualização seja executada e, em seguida, atualize a pilha de 11.4 para 11.10.

2. Adicione instâncias às camadas e teste os aplicativos e livros de receitas da pilha clonada em um sistema de teste ou de preparação. Para mais informações, consulte Tudo sobre o Chef ....
3. Quando os resultados dos testes forem satisfatórios, faça o seguinte:

   - Se esta for a versão desejada do Chef, você pode usar a pilha clonada como sua pilha de produção ou redefinir a versão do Chef em sua pilha de produção.

- Se você estiver fazendo a migração do Chef 0.9 para o Chef 11.10 em duas etapas, repita o processo para migrar a pilha do Chef 11.4 para o Chef 11.10.

> **Note**
>
> Quando você está testando receitas, pode usar o SSH para se conectar à instância e, em seguida, usar o comando da CLI do agente da instância run_command para executar as receitas associadas aos vários eventos de ciclo de vida. A CLI do agente é especialmente útil para testar o Configurar receitas, pois você pode usá-la até mesmo quando a configuração falha e a instância não atinge o estado online. Você também pode usar o comando Configurar pilha para executar novamente o comando Configurar receitas, mas esse comando só estará disponível se a configuração tiver êxito e a instância estiver online.

É possível atualizar uma pilha em execução para uma nova versão do Chef.

Para atualizar uma pilha em execução para uma nova versão do Chef

1. Edite a pilha para alterar a configuração Chef version dela.
2. Salve as novas configurações e espere que as AWS OpsWorks pilhas atualizem as instâncias, o que normalmente leva de 15 a 20 minutos.

> **Important**
>
> AWS OpsWorks O Stacks não sincroniza a atualização da versão do Chef com os eventos do ciclo de vida. Se você deseja atualizar a versão do Chef em uma pilha de produção, deve tomar cuidado para garantir que a atualização seja concluída antes que o próximo evento de ciclo de vida ocorra. Quando ocorre um evento, normalmente um evento de configuração ou implantação, o agente da instância atualiza os livros de receitas personalizados e executa as receitas atribuídas a esse evento, esteja a atualização de versão concluída ou não. Não há uma maneira direta de se determinar quando uma atualização de versão está concluída, mas os logs de implantação incluem a versão do Chef.

Versões do Ruby

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Todas as instâncias em uma pilha do Linux tem o Ruby instalado. AWS OpsWorks O Stacks instala um pacote do Ruby em cada instância, que é usado para executar as receitas do Chef e o agente da instância. AWS OpsWorks O Stacks determina a versão do Ruby com base em qual versão do Chef a pilha está sendo executada. Não tente alterar essa versão, do contrário, o agente da instância pode ser desativado.

AWS OpsWorks O Stacks não instala um aplicativo executável Ruby nas pilhas do Windows. O cliente Chef 12.2 vem com o Ruby 2.0.0 p451, porém o executável Ruby não é acrescentado à variável de ambiente PATH das instâncias. Caso queira usar este executável para executar o código Ruby, ele está localizado em `\opscode\chef\embedded\bin\ruby.exe` no drive do Windows.

A tabela a seguir resume as versões do AWS OpsWorks Stacks Ruby. As versões do Ruby dos aplicativos disponíveis também dependem dos sistemas operacionais da instância. Para obter mais informações, incluindo as versões de patch disponíveis, consulte AWS OpsWorks Sistemas operacionais de pilha.

Versão do Chef	Versão Ruby do Chef	Versões Ruby dos aplicativos disponíveis
0.9 (c)	1.8.7	1.8.7(a), 1.9.3(e), 2.0.0
11.4 (c)	1.8.7	1.8.7(a), 1.9.3(e), 2.0.0, 2.1, 2.2.0, 2.3
11.10	2.0.0-p481	1.9.3(c, e), 2.0.0, 2.1, 2.2.0, 2.3, 2.6.1
12 (b)	2.1.6, 2.2.3	Nenhum
12.22 (d)	2.3.6	Nenhum

(a) Não disponível para Amazon Linux 2014.09 e posterior, Red Hat Enterprise Linux (RHEL) ou Ubuntu 14.04 LTS.

(b) Disponíveis apenas em stacks do Linux.

(c) Não disponível para RHEL.

(d) Disponíveis apenas em stacks do Windows. A versão principal é 12.2. A versão atual secundária é 12.22.

Substituição (e) concluída; o suporte terminou.

Os locais de instalação dependem da versão do Chef:

- Os aplicativos usam o executável `/usr/local/bin/ruby` para todas as versões do Chef.
- Para o Chef 0.9 e 11.4, o agente de instância e as receitas do Chef usam o executável `/usr/bin/ruby`.
- Para o Chef 11.10, o agente de instância e as receitas do Chef usam o executável `/opt/aws/opsworks/local/bin/ruby`.

Instalação de livros de receitas personalizados

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

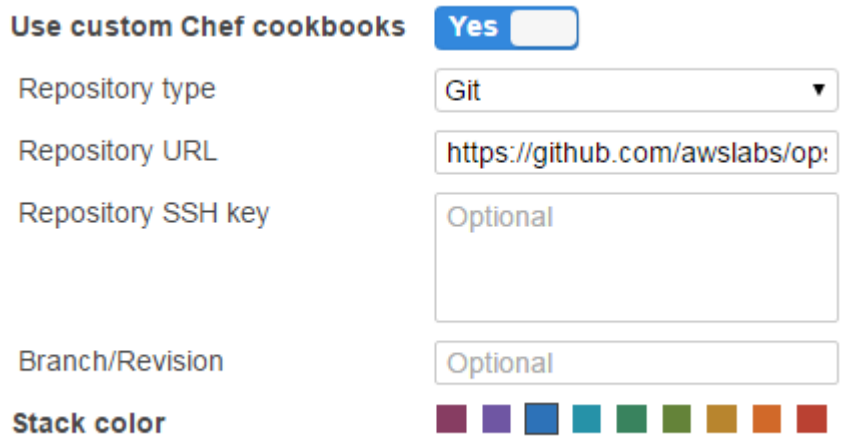
Para fazer com que uma pilha instale e use livros de receitas personalizados, você deve configurar a pilha para permitir os livros de receitas personalizados, se isso ainda não estiver configurado. Você deve fornecer o URL do repositório e quaisquer informações relacionadas, como uma senha.

> **Important**
>
> Depois de configurar a pilha para oferecer suporte a livros de receitas personalizados, o AWS OpsWorks Stacks instala automaticamente seus livros de receitas em todas as novas

instâncias na inicialização. No entanto, você deve orientar explicitamente o AWS OpsWorks Stacks a instalar livros de receitas novos ou atualizados em qualquer instância existente executando o comando Atualizar pilha de livros de receitas personalizados. Para obter mais informações, consulte Atualizar livros de receitas personalizadas. Antes de permitir Use custom Chef cookbooks (Usar livros de receitas do Chef personalizados) na sua pilha, certifique-se de que os livros de receitas personalizados e de comunidade que você executa têm suporte para a versão do Chef que sua pilha utiliza.

Para configurar uma pilha para livros de receitas personalizados

1. Na página da pilha, clique em Stack Settings para exibir a página Settings. Clique em Edit para editar as configurações.
2. Alterne Use custom Chef cookbooks para Yes.



3. Configuração dos livros de receitas.

Quando terminar, clique em Save para salvar a pilha atualizada.

Especificação de um repositório de livro de receitas personalizado

As pilhas do Linux podem instalar livros de receitas personalizados de qualquer um dos seguintes tipos de repositório:

- Arquivos HTTP ou Amazon S3.

  Eles podem ser públicos ou privados, mas o Amazon S3 costuma ser a opção preferida para um arquivo privado.

- Os repositórios Git e Subversion fornecem controle de origem e a capacidade de ter várias versões.

As pilhas do Windows podem instalar livros de receitas personalizados de arquivos do Amazon S3 e de repositórios Git.

Todos os tipos de repositório têm os seguintes campos obrigatórios.

- Tipo de repositório: o tipo de repositório
- URL de repositório: o URL do repositório

AWS OpsWorks O Stacks fornece suporte para sites de repositórios Git hospedados publicamente, como o Bitbucket, GitHubbem como para servidores Git hospedados privadamente. Para os repositórios Git, você deve usar um dos seguintes formatos de URL, dependendo se o repositório é público ou privado. Siga as mesmas diretrizes de URL para os submódulos do Git.

Para um repositório público Git, use os protocolos HTTPS ou somente leitura do Git:

- Somente leitura do Git: `git://github.com/amazonwebservices/opsworks-example-cookbooks.git`.
- HTTPS: `https://github.com/amazonwebservices/opsworks-example-cookbooks.git`.

Para um repositório privado Git, você deve usar o formato de leitura/gravação SSH, como mostrado nos exemplos a seguir:

- Repositórios Github: `git@github.com:`*`project`*`/`*`repository`*.
- Repositórios em um servidor Git: *`user`*`@`*`server`*`:`*`project`*`/`*`repository`*

As configurações restantes variam de acordo com o tipo de repositório e estão descritas nas seções a seguir.

Arquivo HTTP

Selecionar Http Archive para Repository type exibe duas configurações adicionais, que você deve completar se o arquivo for protegido por senha.

- Nome de usuário: seu nome de usuário

- Senha: sua senha

Arquivo do Amazon S3

Selecionar S3 Archive (Arquivamento do S3) para Repository type (Tipo de repositório) exibe as seguintes configurações adicionais e opcionais. AWS OpsWorks O Stacks pode acessar seu repositório usando EC2 funções da Amazon (autenticação do administrador do sistema operacional do host), se você usar a API ou o console do AWS OpsWorks Stacks.

- ID da chave de acesso: uma ID da chave de acesso da AWS, comoAKIAIOSFODNN7EXAMPLE.
- Chave de acesso secreta: a chave de acesso secreta da AWS correspondente, comowJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY.

Repositório Git

Ao selecionar Git em Source Control, estas configurações opcionais adicionais são exibidas:

Chave SSH de repositório

Você deve especificar uma chave SSH de implantação para acessar os repositórios Git privados. Para submódulos do Git, a chave especificada deve ter acesso a esses submódulos. Para obter mais informações, consulte Utilização de chaves SSH de repositório Git.

> **Important**
>
> A chave SSH de implantação não pode exigir uma senha; o AWS OpsWorks Stacks não tem como passá-la.

Ramificação/Revisão

Se o repositório tiver várias ramificações, o AWS OpsWorks Stacks baixa a ramificação master por padrão. Para especificar uma ramificação específica, insira o nome da ramificação, o SHA1 hash ou o nome da tag. Para especificar uma determinada confirmação, insira o ID completo de confirmação com 40 dígitos hexadecimais.

Repositório Subversion

Ao selecionar Subversion em Source Control, estas configurações adicionais são exibidas:

- Nome de usuário: seu nome de usuário, para repositórios privados.
- Senha: sua senha, para repositórios privados.
- Revisão: [opcional] o nome da revisão, se você tiver várias revisões.

  Para especificar uma ramificação ou uma tag, você deve modificar o URL do repositório, por exemplo: **`http://repository_domain/repos/myapp/branches/my-apps-branch`** ou **`http://repository_domain_name/repos/calc/myapp/my-apps-tag`**.

Atualizar livros de receitas personalizadas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Ao fornecer livros de receitas personalizadas para o AWS OpsWorks Stacks, a opção Configurar receitas internas cria um cache local em uma instância recém-criada e faz download dos livros de receitas para o cache. AWS OpsWorks Em seguida, o Stacks executa as receitas do cache e não do repositório. Se você modificar livros de receitas personalizadas no repositório, é necessário garantir que os livros de receitas atualizados estejam instalados nos caches locais das instâncias. AWS OpsWorks O Stacks implementa automaticamente os livros de receitas mais recente em novas instâncias quando são iniciadas. Entretanto, para as instâncias atuais a situação é diferente:

- Você deve implantar manualmente livros de receitas personalizadas atualizados para instâncias online.
- Não é necessário implantar livros de receitas personalizadas atualizados com base em instâncias com armazenamento em instâncias offline, incluindo instâncias com base em carga e tempo.

  AWS OpsWorks O Stacks implanta automaticamente os livros de receitas atuais quando as instâncias são reiniciadas.
- Você deve iniciar instâncias 24 horas por dia, 7 dias por semana offline baseadas no EBS, sem base em carga ou tempo.

- Você não pode iniciar instâncias offline baseadas no EBS com base em carga e tempo, então a abordagem mais simples é excluir as instâncias offline e adicionar novas instâncias para substitui-las.

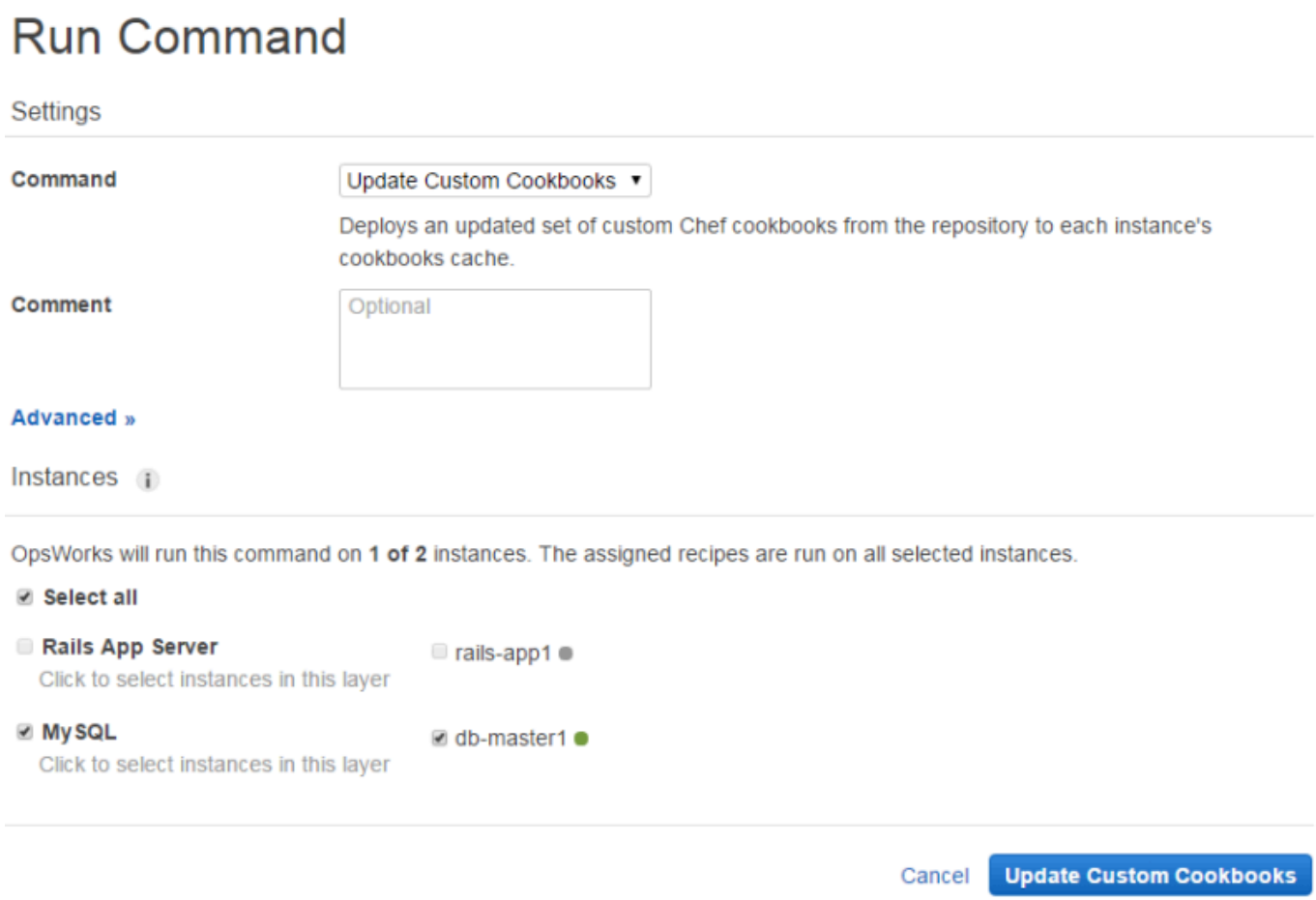
  Como agora são novas instâncias, o AWS OpsWorks Stacks implanta automaticamente os livros de receitas personalizados atuais quando as instâncias são iniciadas.

Para atualizar manualmente os livros de receitas personalizados

1. Atualize o repositório com os livros de receitas modificados. AWS OpsWorks O Stacks usa o URL em cache fornecido durante a instalação original dos livros de receitas, portanto, o nome do arquivo raiz do livro de receitas, o local do repositório e os direitos de acesso não devem mudar.

   - Para os repositórios Amazon S3 ou HTTP, substitua o arquivo .zip original pelo novo com o mesmo nome.
   - Para repositórios Git ou de Subversão, edite as configurações de pilha a fim de alterar o campo Branch/Revision para a nova versão.

2. Na página de pilha, clique em Run Command e selecione o comando Update Custom Cookbooks.



3. Adicione um comentário, se desejar.
4. Opcionalmente, especifique um objeto JSON personalizado para o comando para adicionar atributos personalizados à configuração da pilha e aos atributos de implantação que o AWS OpsWorks Stacks instala nas instâncias. Para obter mais informações, consulte Usar JSON personalizado e Sobrepor atributos.
5. Por padrão, o AWS OpsWorks Stacks atualiza os livros de receitas em todas as instâncias. Para especificar quais instâncias devem ser atualizadas, selecione as instâncias adequadas da lista no fim da página. Para selecionar cada instância em uma camada, marque a caixa de seleção da camada adequada na coluna esquerda.
6. Clique em Atualizar livros de receitas personalizados para instalar os livros de receita atualizados. AWS OpsWorks O Stacks exclui os livros de receitas personalizados em cache nas instâncias especificadas e instala os novos livros de receitas do repositório.

> **Note**
>
> Esse procedimento é necessário somente para instâncias atuais com versões antigas dos livros de receitas em cache. Se você posteriormente adicionar instâncias a uma camada, o AWS OpsWorks Stacks implanta os livros de receitas que estão atualmente no repositório para que eles obtenham automaticamente a versão mais recente.

Execução de receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode executar receitas de duas formas:

- Automaticamente, atribuindo receitas ao evento de ciclo de vida apropriado da camada.
- Manualmente, executando o comando de pilha Execute Recipes ou usando a CLI de agente.

Tópicos

- AWS OpsWorks Eventos de ciclo de vida de pilhas
- Execução automática de receitas
- Execução manual de receitas

AWS OpsWorks Eventos de ciclo de vida de pilhas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Cada camada tem um conjunto de cinco eventos de ciclo de vida, sendo que cada um tem um conjunto associado de receitas específicas para a camada. Quando ocorre um evento na instância de uma camada, o AWS OpsWorks Stacks executa automaticamente o conjunto de receitas adequado. Para fornecer uma resposta personalizada a esses eventos, implemente receitas personalizadas e as atribua aos eventos apropriados para cada camada. AWS OpsWorks O Stacks executa essas receitas após as receitas integradas do evento.

Setup

Esse evento ocorre após uma instância iniciada concluir a inicialização. Você também pode acionar manualmente o Setup evento usando o comando Configurar pilha. AWS OpsWorks O Stacks executa as receitas que definem a instância de acordo com sua camada. Por exemplo, se a instância for membro do camada do Rails App Server, as receitas Setup vão instalar Apache, Ruby Enterprise Edition, Passenger e Ruby on Rails.

> **Note**
>
> Um evento Setup pega uma instância fora de serviço. Como uma instância não está no estado Online quando o evento de ciclo de vida Setup é executado, as instâncias em que você executa eventos Setup são removidas de um load balancer.

Configure

Esse evento ocorre em todas as instâncias da pilha quando um dos seguintes ocorre:

- Uma instância entra ou sai do estado online.
- Você associa um endereço IP elástico a uma instância ou desassocia um de uma instância.
- Você anexa um balanceador de carga do Elastic Load Balancing a uma camada ou o desanexa de uma camada.

Por exemplo, suponha que sua pilha tenha instâncias A, B e C e você inicia uma nova instância, D. Depois que D termina de executar suas receitas de instalação, o AWS OpsWorks Stacks

aciona o Configure evento em A, B, C e D. Se você interrompe A posteriormente, o AWS OpsWorks Stacks aciona o Configure evento em B, C e D. O AWS OpsWorks Stacks responde ao Configure evento executando as Configure receitas de cada camada, que atualizam as configurações das instâncias para refletir o perfil atual conjunto de instâncias on-line. O evento Configure é, portanto, um bom momento para gerar novamente os arquivos de configuração. Por exemplo, as HAProxy Configure receitas reconfiguram o balanceador de carga para acomodar quaisquer alterações no conjunto de instâncias do servidor de aplicativos on-line.

Você também pode acionar manualmente o evento Configure usando o comando de pilha Configure.

Deploy

Esse evento ocorre quando você executa um comando Deploy, normalmente para implantar um aplicativo a um conjunto de instâncias do servidor de aplicativos. As instâncias executam receitas que implantam a aplicação e quaisquer arquivos relacionados de seu repositório para as instâncias da camada. Por exemplo, para instâncias de um servidor de aplicativos do Rails, as receitas Deploy verificam um aplicativo Ruby determinado e falam para o Phusion Passenger recarregá-lo. Você também pode executar Deploy em outras instâncias para que elas possam, por exemplo, atualizar a configuração delas para acomodar a aplicação recém-implantada.

> **Note**
>
> Setup inclui Deploy; ele executa as receitas de implantação após a conclusão da instalação.

Undeploy

Esse evento ocorre quando você exclui um aplicativo ou executa um comando Undeploy para remover um aplicativo de um conjunto de instâncias do servidor de aplicativos. As instâncias especificadas executam receitas para remover todas as versões da aplicação e executar qualquer limpeza necessária.

Shutdown

Esse evento ocorre depois que você direciona o AWS OpsWorks Stacks para desligar uma instância, mas antes que a EC2 instância da Amazon associada seja realmente encerrada. AWS OpsWorks O Stacks executa receitas para executar tarefas de limpeza, como desligamento de serviços.

Se você tiver anexado um balanceador de carga do Elastic Load Balancing à camada e habilitado o suporte para drenagem da conexão, o AWS OpsWorks Stacks aguardará a conclusão da diminuição da conexão antes de acionar o evento. Shutdown

Depois de acionar um Shutdown evento, o AWS OpsWorks Stacks dá às Shutdown receitas uma quantidade de tempo determinada para a realização de suas tarefas e, em seguida, interrompe ou encerra a instância da Amazon. EC2 O valor de tempo limite de Shutdown padrão é 120 segundos. Se suas receitas Shutdown exigirem mais tempo, você poderá editar a configuração da camada para alterar o valor do tempo limite. Para obter mais informações sobre instâncias Shutdown, consulte Interromper uma instância.

> **Note**
>
> A reinicialização de uma instância não aciona eventos de ciclo de vida.

Para mais discussões sobre os comandos de aplicativo Deploy e Undeploy, consulte Implementação de aplicativos.

Após uma instância iniciada concluir a inicialização, a sequência de inicialização restante é a seguinte:

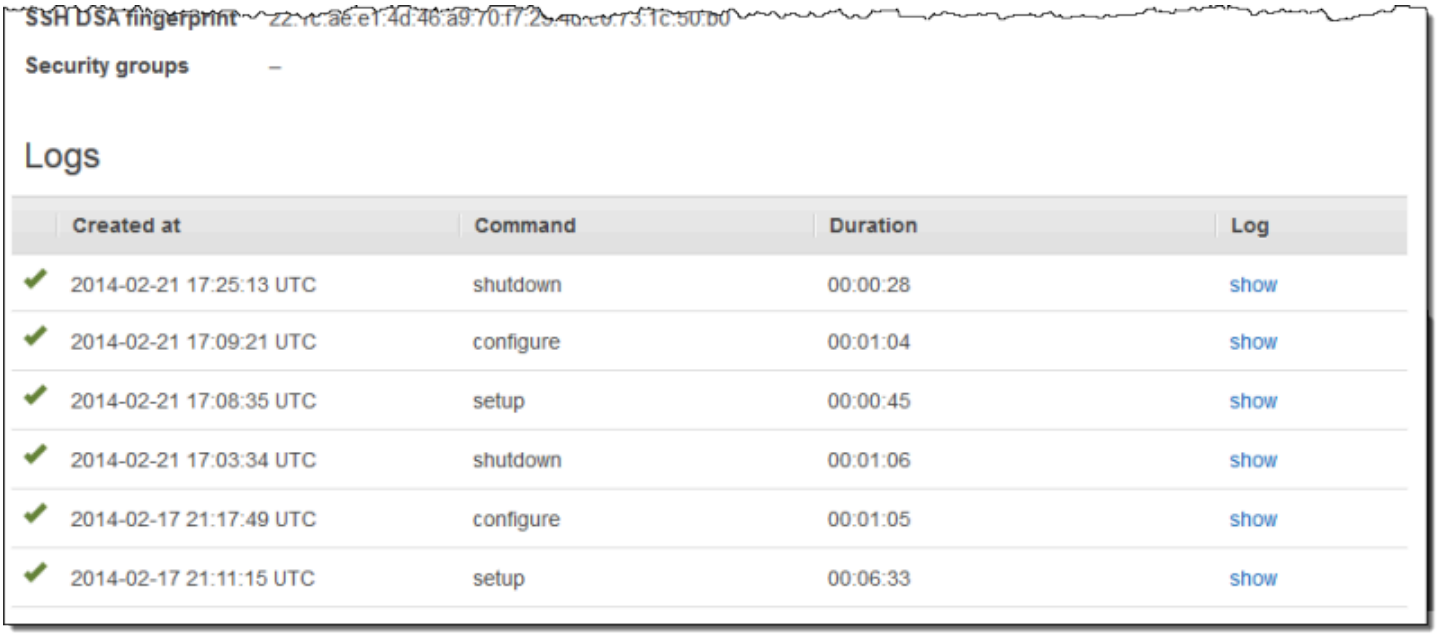
1. AWS OpsWorks O Stacks executa as Setup receitas integradas da instância, seguidas por qualquer Setup receita personalizada.
2. AWS OpsWorks O Stacks executa as Deploy receitas integradas da instância, seguidas por qualquer Deploy receita personalizada.

   A instância agora está online.
3. AWS OpsWorks As pilhas acionam um Configure evento em todas as instâncias da pilha, incluindo a instância recém-iniciada.

   AWS OpsWorks O Stacks executa as Configure receitas integradas das instâncias, seguidas por qualquer receita personalizada. Configure

> **Note**
>
> Para ver os eventos de ciclo de vida que ocorreram em uma instância específica, vá para a página Instances e clique no nome da instância para abrir sua página de detalhes. A lista de

eventos fica na seção Logs, na parte inferior da página. Você pode clicar em show na coluna Log para examinar o log do Chef para um evento. Isso fornece informações detalhadas sobre como o evento foi tratado, incluindo quais receitas foram executadas. Para obter mais informações sobre como interpretar os logs do Chef, consulte Logs do Chef.



Para cada evento do ciclo de vida, o AWS OpsWorks Stacks instala um conjunto de atributos de configuração e implantação da pilha em cada instância que contém o estado atual da pilha e, para Deploy eventos, informações sobre a implantação. Os atributos incluem informações sobre quais instâncias estão disponíveis, seus endereços IP, e assim por diante. Para obter mais informações, consulte Configuração de pilha e atributos de implantação.

> **Note**
>
> Iniciar ou interromper um grande número de instâncias ao mesmo tempo pode gerar rapidamente um grande número de eventos Configure. Para evitar processamento desnecessário, o AWS OpsWorks Stacks responde somente ao último evento. Os atributos de implantação e configuração de stack desse evento contêm todas as informações necessárias para atualizar as instâncias da pilha para todo o conjunto de alterações. Isso elimina a necessidade de também processar os Configure eventos anteriores. AWS OpsWorks As pilhas rotulam os Configure eventos não processados como substituídos.

Execução automática de receitas

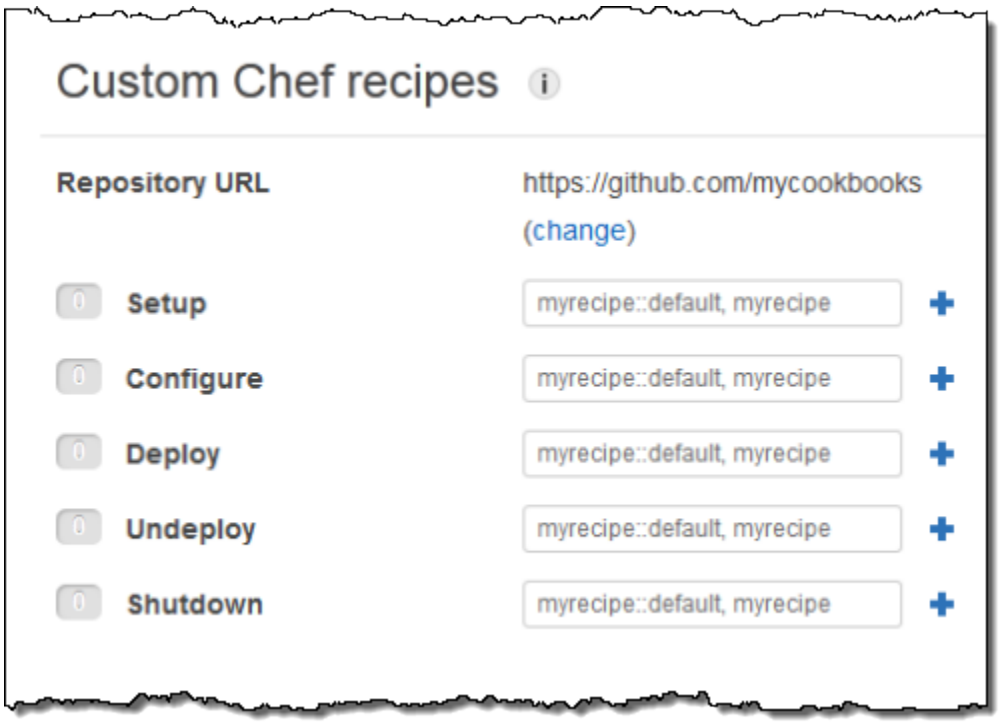
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Cada camada tem um conjunto de receitas integradas atribuídas a cada evento de ciclo de vida, embora algumas camadas não tenham receitas Undeploy. Quando ocorre um evento de ciclo de vida em uma instância, o AWS OpsWorks Stacks executa o conjunto apropriado de receitas para a camada associada.

Se você instalou livros de receitas personalizados, pode fazer com que o AWS OpsWorks Stacks execute algumas ou todas as receitas automaticamente atribuindo cada receita ao evento do ciclo de vida de uma camada. Após a ocorrência de um evento, o AWS OpsWorks Stacks executa as receitas personalizadas especificadas após as receitas integradas da camada.

Para atribuir receitas personalizadas a eventos de camada

1. Na página Layers, para a camada apropriada, clique em Recipes e, em seguida, em Edit. Se você ainda não tiver habilitado livros de receitas personalizados, clique em configurar cookbooks para abrir a página Settings da pilha. Alterne Use custom Chef Cookbooks para Yes, e forneça as informações do repositório do livro de receitas. Em seguida, clique em Save e navegue até a página de edição da guia Recipes. Para obter mais informações, consulte Instalação de livros de receitas personalizados.
2. Na guia Recipes, insira cada receita personalizada no campo de evento adequado e clique em + para adicioná-la à lista. Especifique uma receita da seguinte forma*`cookbook`*::: *`somerecipe`* (omita a `.rb` extensão).



Quando você inicia uma nova instância, o AWS OpsWorks Stacks executa automaticamente as receitas personalizadas para cada evento, depois de executar as receitas padrão.

> **Note**
>
> As receitas personalizadas são executadas na ordem que você as insere no console. Uma maneira alternativa de controlar a ordem de execução é implementando uma meta receita que executa as receitas na ordem correta.

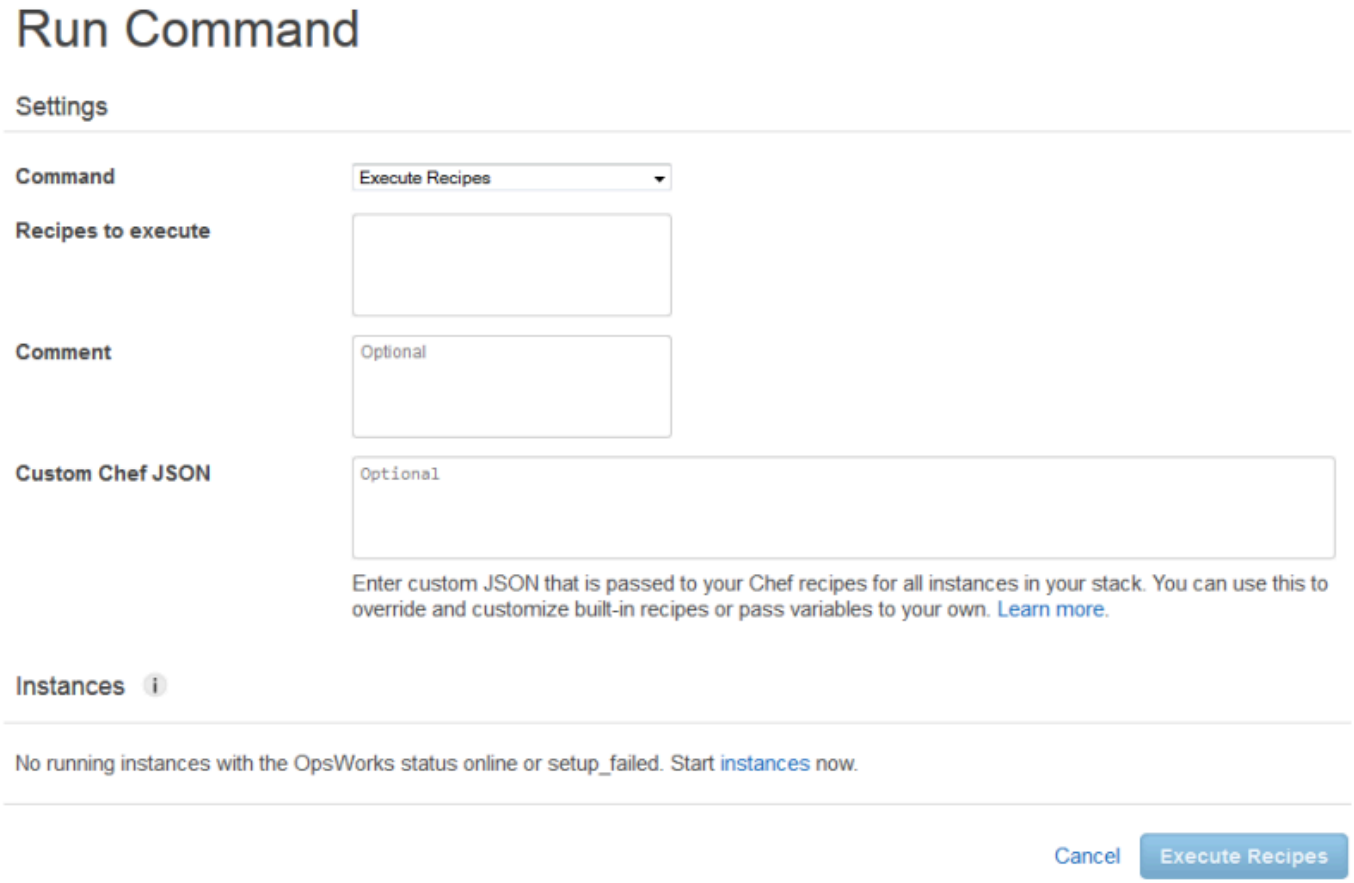
Execução manual de receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Embora as receitas costumem ser executadas automaticamente em resposta a eventos de ciclo de vida, você pode executá-las manualmente a qualquer momento em uma ou todas as instâncias da pilha. Esse recurso costuma ser usado para tarefas que não mapeiam bem a um evento de ciclo de vida, como o backup de instâncias. Para executar manualmente uma receita personalizada, ela deve estar em um de seus livros de receitas personalizados, mas não precisa ser atribuída a um evento de ciclo de vida. Quando você executa uma receita manualmente, o AWS OpsWorks Stacks instala os mesmos `deploy` atributos de um evento Deploy.

Para executar manualmente as receitas em instâncias da pilha

1. Na página Stack, clique em Run command. Para Command, selecione Execute Recipes.



2. Insira as receitas a serem executadas na caixa Receitas a serem executadas usando o *`recipename`* formato padrão*`cookbookname`*::. Use vírgulas para separar várias receitas, que serão executadas na ordem em que são listadas.
3. Como opção, use a caixa Custom Chef JSON para adicionar um objeto JSON personalizado que define atributos personalizados que serão incorporados aos atributos de implantação e configuração de pilha que são instalados nas instâncias. Para obter mais informações sobre o uso de objetos JSON personalizados, consulte Usar JSON personalizado e Sobrepor atributos.

4. Em Instâncias, selecione as instâncias nas quais AWS OpsWorks as pilhas devem executar as receitas.

Quando ocorre um evento de ciclo de vida, o agente AWS OpsWorks Stacks recebe um comando para executar as receitas associadas. Você pode executar manualmente esses comandos em uma instância específica usando o comando de pilha adequado ou usando o comando run_command da CLI do agente. Para obter mais informações sobre como usar a CLI do agente, consulte AWS OpsWorks CLI do Stacks Agent.

Gerenciamento de recursos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A página Recursos permite que você use o endereço IP elástico da sua conta, o volume do Amazon EBS ou os recursos da instância do Amazon RDS em uma AWS OpsWorks pilha Stacks. Você pode usar Resources para fazer o seguinte:

- Registrar um recurso com uma pilha, o que permite que você anexe o recurso a uma das instâncias da pilha.
- Anexar um recurso a uma das instâncias da pilha.
- Mover um recurso de uma instância para outra.
- Desanexar um recurso de uma instância. O recurso permanece registrado e pode ser anexado a outra instância.
- Cancelar o registro de um recurso. Um recurso não registrado não pode ser usado pelo AWS OpsWorks Stacks, mas ele permanece na sua conta, a menos que você o exclua, e pode ser registrado com outra pilha.

Observe as seguintes restrições:

- Você não pode anexar volumes Amazon EBS registrados para instâncias do Windows.
- A página Recursos gerencia volumes do Amazon EBS padrão, de PIOPS, de disco rígido com throughput otimizado ou de uso geral (SSD), mas não de matrizes RAID.
- Os volumes Amazon EBS devem estar no formato xfs.

  AWS OpsWorks O Stacks não oferece suporte a outros formatos de arquivo, como ext4. Para obter mais informações sobre a preparação de volumes Amazon EBS, consulte Tornando um volume Amazon EBS disponível para uso.
- Você não pode anexar um volume Amazon EBS a, ou desanexá-lo de, uma instância em execução.

  Só é possível operar em instâncias offline. Por exemplo, é possível registrar um volume em uso com uma pilha e anexá-lo a uma instância offline, mas você deve interromper a instância original e desanexar o volume antes de iniciar a nova instância. Caso contrário, o processo de início falhará.
- Todos os recursos registrados são gerenciados exclusivamente em AWS OpsWorks. Isso pode substituir as propriedades do ciclo de vida do recurso, como `DeleteOnTermination` para volumes. EC2
- Você pode anexar e desanexar um endereço Elastic IP de uma instância em execução.

  Você pode operar em instâncias online ou offline. Por exemplo, você pode registrar um endereço em uso e atribuí-lo a uma instância em execução, e o AWS OpsWorks Stacks reatribuirá automaticamente o endereço.
- Para registrar e cancelar o registro de recursos, sua política IAM deve conceder permissões para as seguintes ações:

Volumes do Amazon EBS	Endereços Elastic IP	Instâncias Amazon RDS
RegisterVolume	RegisterElasticIp	RegisterRdsDbInstance
UpdateVolume	UpdateElasticIp	UpdateRdsDbInstance
DeregisterVolume	DeregisterElasticIp	DeregisterRdsDbInstance

O Gerenciar nível de permissões concede permissões para todas essas ações. Para evitar que um usuário com nível Gerenciar registre ou cancele o registro de recursos específicos, edite sua

política IAM para negar permissões para as ações apropriadas. Para obter mais informações, consulte Segurança e permissões.

Tópicos

- Como registrar recursos com uma pilha
- Como anexar e mover recursos
- Como desanexar recursos
- Como cancelar registros de recursos

Como registrar recursos com uma pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Volumes Amazon EBS ou endereços Elastic IP devem ser registrados com uma pilha antes que você possa anexá-los em instâncias. Quando o AWS OpsWorks Stacks cria recursos para uma pilha, eles são automaticamente registrados nessa pilha. Se quiser usar recursos criados externamente, você deve registrá-los explicitamente. Observe o seguinte:

- Só é possível registrar um recurso em apenas uma pilha por vez.
- Quando você exclui uma pilha, o AWS OpsWorks Stacks cancela o registro de todos os recursos.

Tópicos

- Como registrar volumes Amazon EBS com uma pilha
- Como registrar Endereços elastic IP com uma pilha
- Como registrar instâncias Amazon RDS com uma pilha

Como registrar volumes Amazon EBS com uma pilha

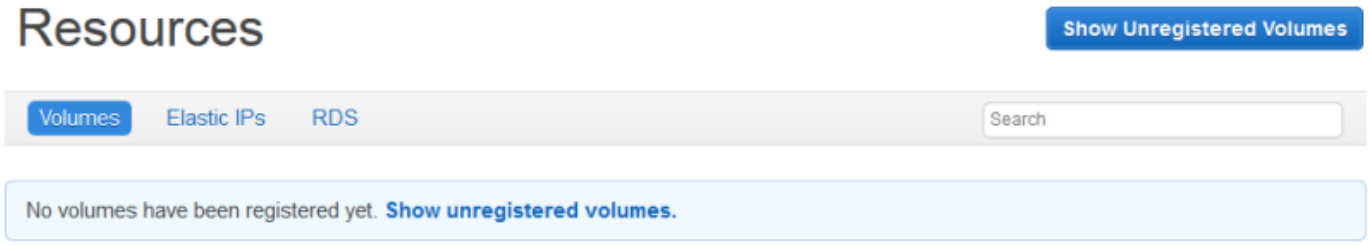
> **Note**
>
> Esse recurso pode ser usado apenas com pilhas do Linux. Embora seja possível registrar um volume Amazon EBS com uma pilha do Windows, não é possível anexá-lo a uma instância.

Você pode usar a página Recursos para registrar um volume Amazon EBS com uma pilha, de acordo com as seguintes restrições:

- Quando anexados, os volumes do Amazon EBS que não sejam raiz devem ser padrão, throughput, de disco rígido com taxa de transferência otimizada, de disco rígido frio, de PIOPS ou de uso geral (SSD), mas não de uma matriz RAID. Para obter informações sobre tamanhos máximos e mínimos de volumes, consulte Volumes do EBS neste guia.
- Volumes devem estar no formato XFS.
- O AWS OpsWorks Stacks não oferece suporte a outros formatos de arquivo, como o ext4 (fourth extended file system), para volumes Amazon EBS não raiz. Para obter mais informações sobre a preparação de volumes Amazon EBS, consulte Como tornar um volume Amazon EBS disponível para uso. Observe que o exemplo neste tópico descreve como criar um volume com base em ext4, mas você pode seguir as mesmas etapas para volumes baseados em XFS.

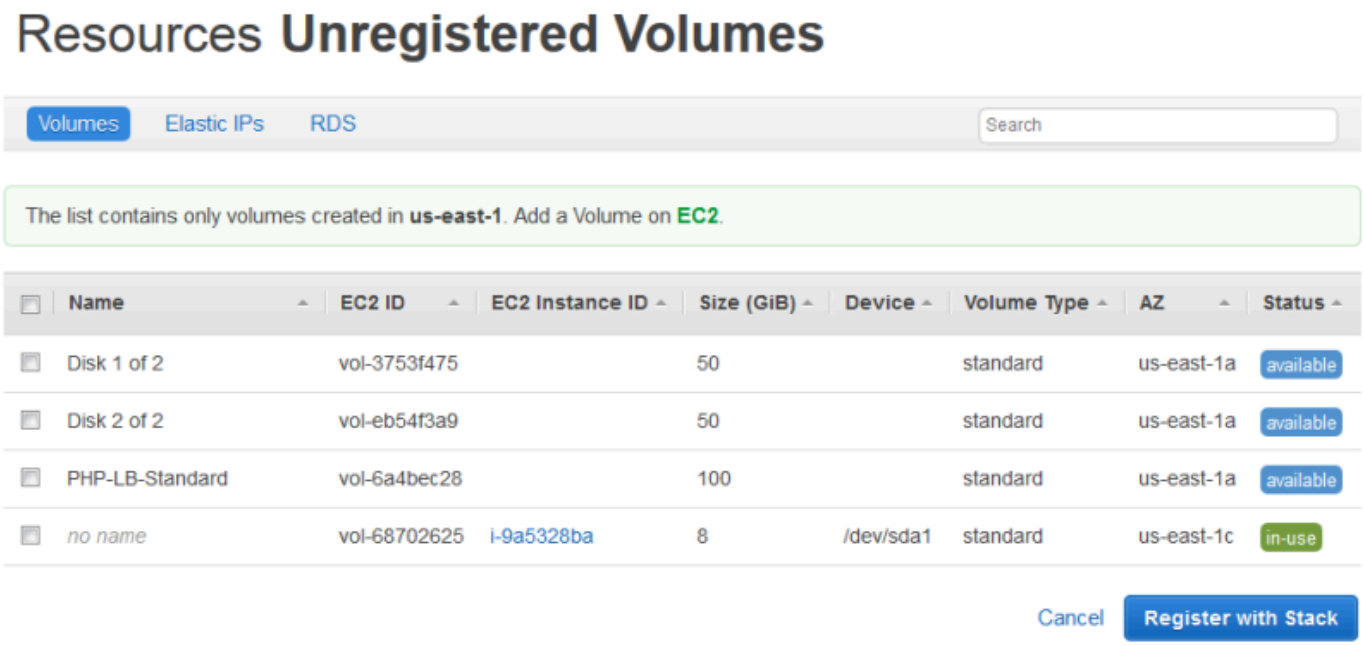
Registrar um de volume do Amazon EBS

1. Abra a pilha desejada e clique em Resources no painel de navegação.
2. Clique em Volumes para exibir os volumes Amazon EBS disponíveis. Inicialmente, a pilha não possui volumes registrados, como mostrado na ilustração a seguir.

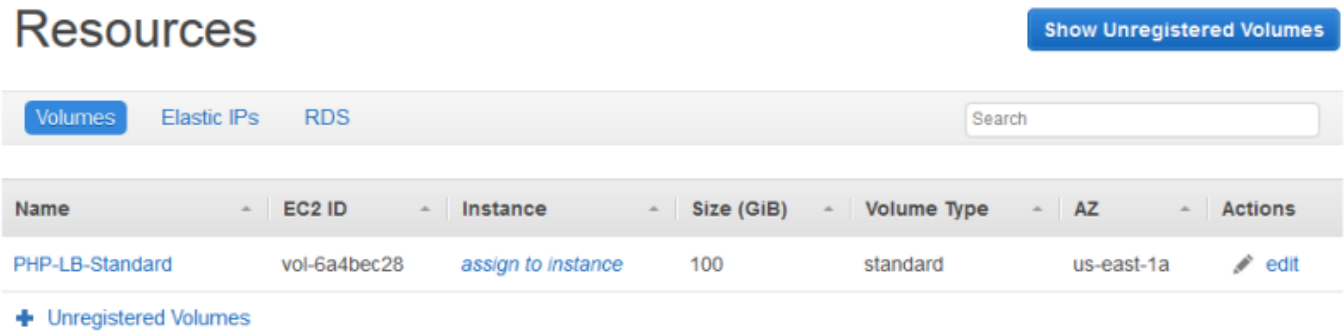


3. Clique em Mostrar volumes não registrados para exibir os volumes Amazon EBS em sua conta que estão na região da pilha e, se aplicável, a VPC da pilha. A coluna Status indica se os volumes estão disponíveis para uso. Volume Type (Tipo de volume) indica se o volume é padrão

(`standard`), de SSD de uso geral (`gp2`), de PIOPS (`io1`, seguido pela IOPS por valor de disco entre parênteses), de disco rígido com throughput otimizado (`st1`) ou de disco rígido frio (`sc1`).



4. Selecione os volumes apropriados e clique Register to Stack. A página Resources agora relaciona os volumes recém-registrados.



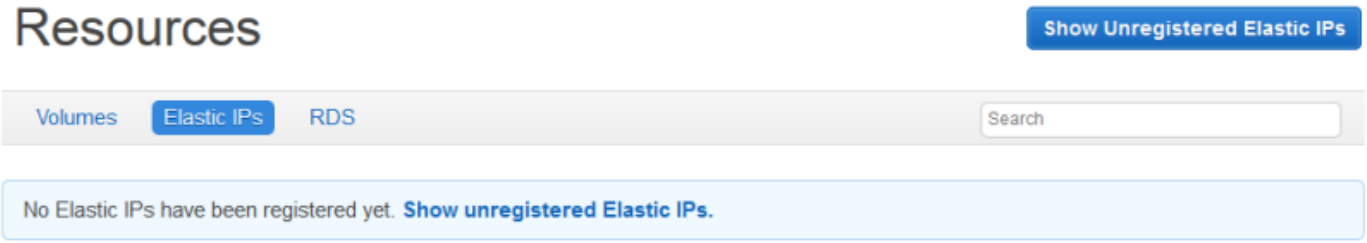
Para registrar volumes adicionais, clique em Show Unregistered Volumes ou + Unregistered Volumes e repita o procedimento.

Como registrar Endereços elastic IP com uma pilha

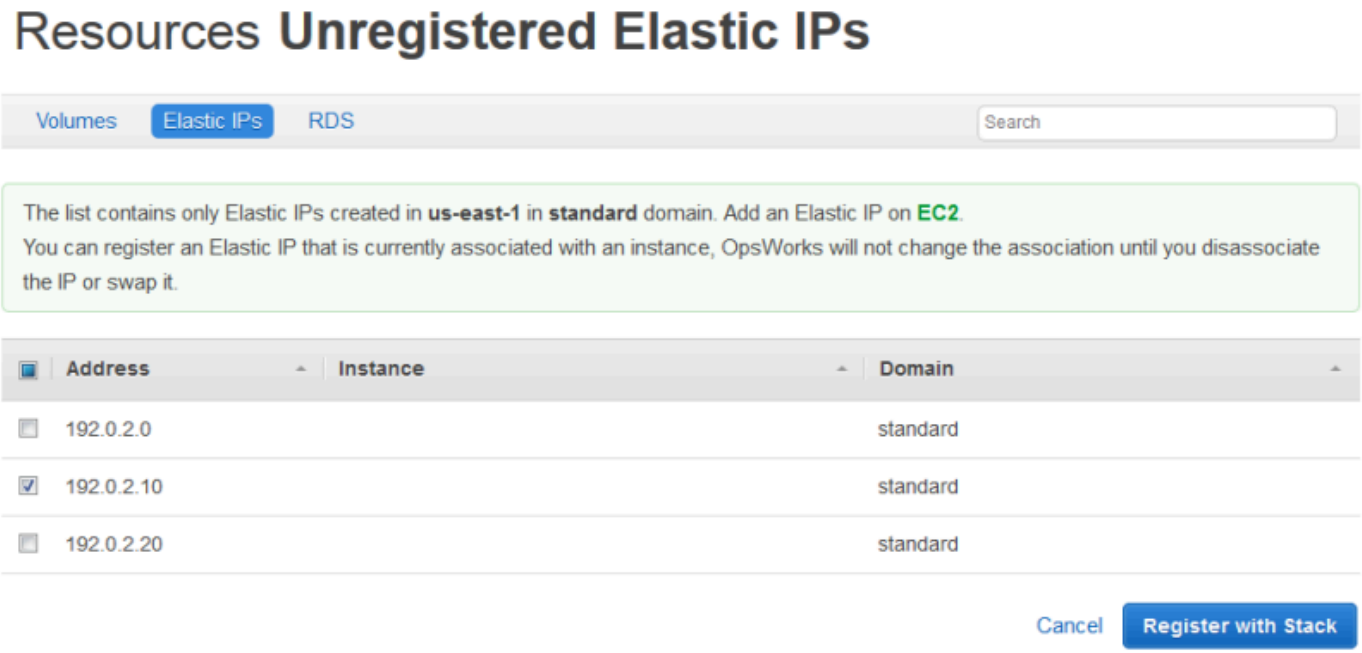
Use o procedimento a seguir para registrar endereços Elastic IP.

Para registrar um endereço Elastic IP

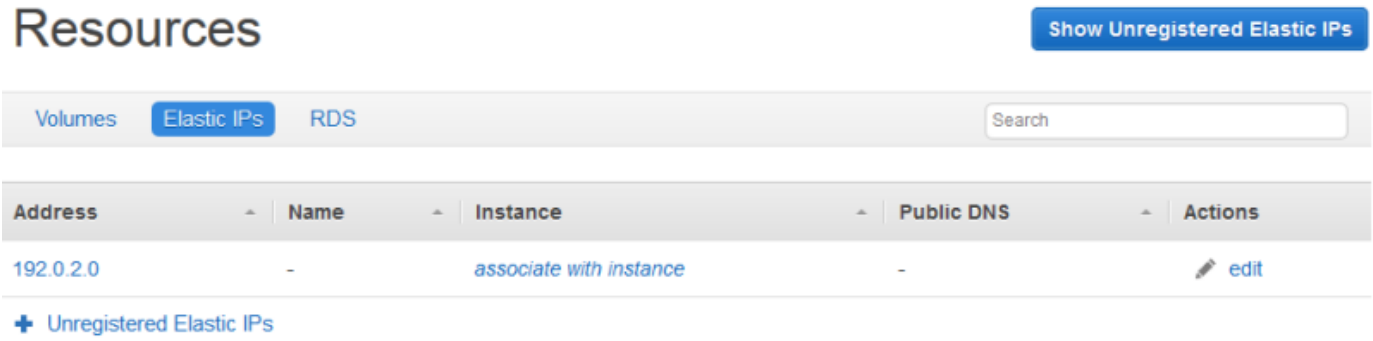
1. Abra a página de recursos da pilha e clique em Elastic IPs para exibir os endereços IP elásticos disponíveis. Inicialmente, a pilha não possui endereços registrados, como mostrado na ilustração a seguir.



2. Clique em Mostrar elástico não registrado IPs para exibir os endereços IP elásticos disponíveis em sua conta que estão na região da pilha.



3. Selecione os endereços apropriados e clique Register to Stack. Você retornará à página Resources, que agora lista os endereços recém-registrados.



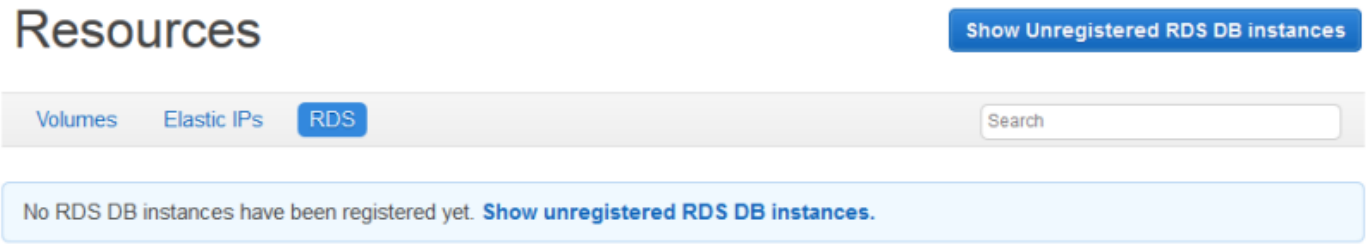
Para registrar endereços adicionais, clique em Mostrar elástico não registrado IPs ou + Elástico não registrado IPs e repita esse procedimento.

Como registrar instâncias Amazon RDS com uma pilha

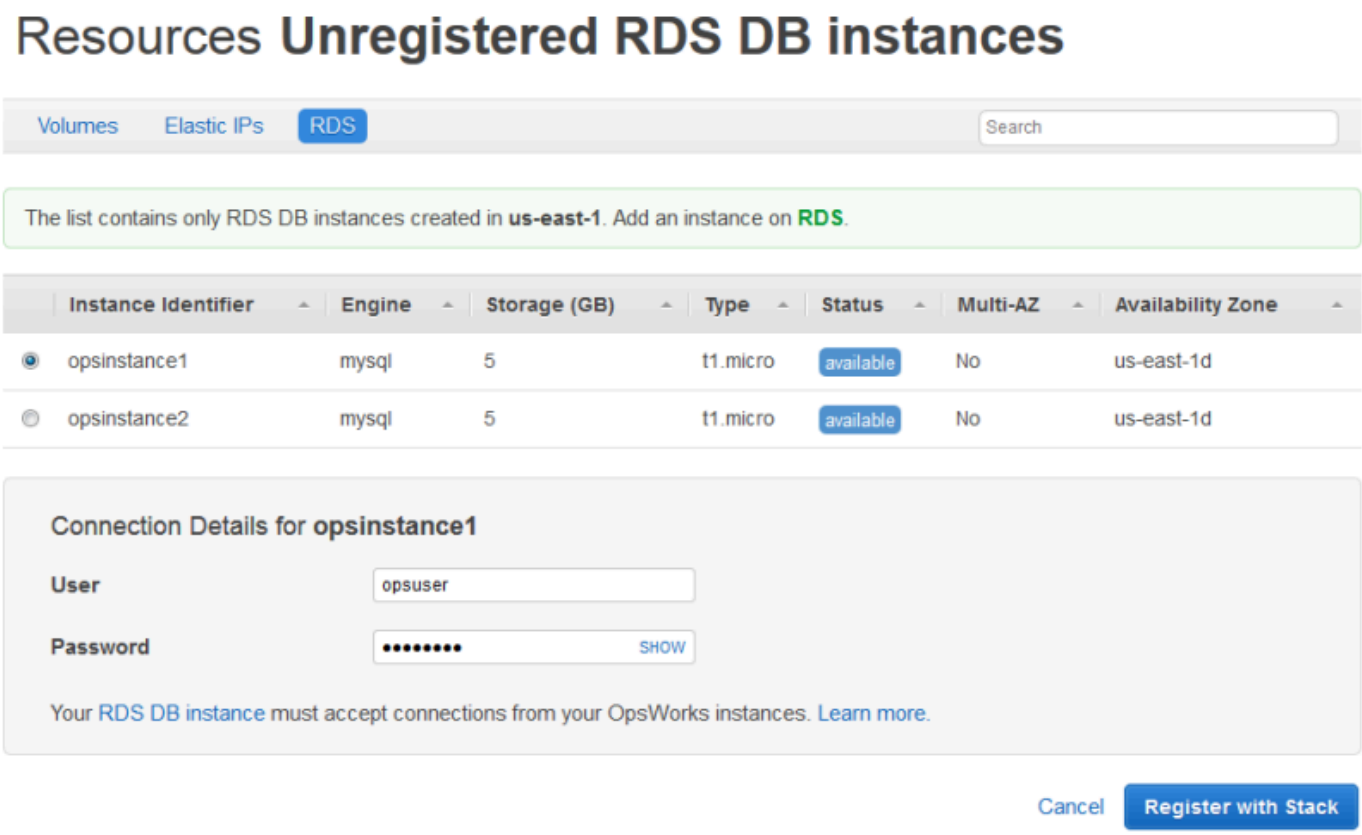
Use o procedimento a seguir para registrar instâncias Amazon RDS.

Registra uma instância do Amazon RDS

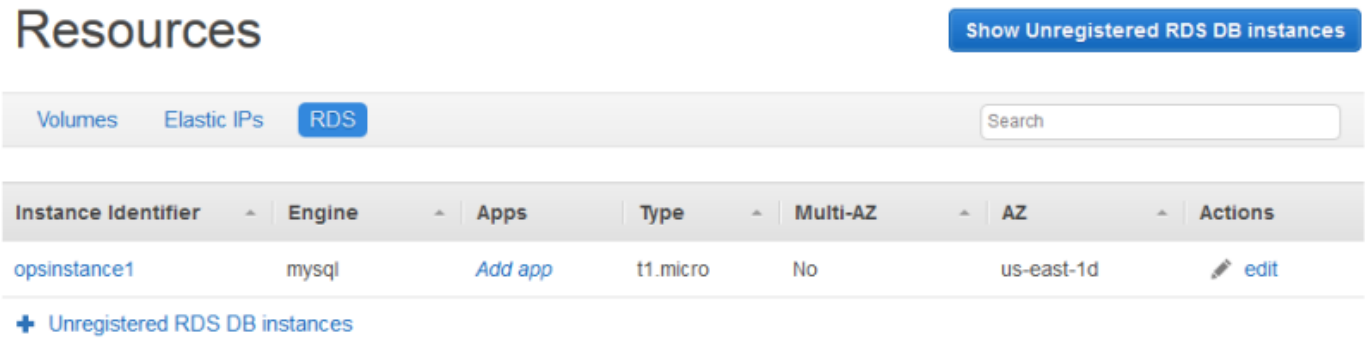
1. Abra a página Recursos da pilha e clique em RDS para exibir as instâncias Amazon RDS disponíveis. Inicialmente, a pilha não possui instâncias registradas, como mostrado na ilustração a seguir.



2. Clique em Mostrar instâncias de banco de dados do RDS não registradas para exibir as instâncias do Amazon RDS disponíveis em sua conta que estão na região da pilha.



3. Selecione a instância apropriada, insira seus valores de usuário mestre e senha mestre para User e Password. Em seguida, clique em Register to Stack. Você retornará à página Resources, que agora lista a instância recém-registrada.



> **Important**
>
> É necessário garantir que o usuário e a senha usados para registrar a instância Amazon RDS correspondem a um usuário e senha válidos. Caso não correspondam, seus aplicativos não serão capazes de se conectar à instância.

Para registrar endereços adicionais, clique em Show Unregistered RDS DB instances ou + Unregistered RDS DB instances e repita o procedimento. Para obter mais informações sobre como usar instâncias do Amazon RDS com AWS OpsWorks Stacks, consulte. Camada de serviços do Amazon RDS

> **Note**
>
> Você também pode registrar instâncias Amazon RDS através da página Camadas. Para obter mais informações, consulte Camada de serviços do Amazon RDS.

Como anexar e mover recursos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Após registrar um recurso com uma pilha, você pode anexá-lo a uma das instâncias da pilha. Também é possível mover um recurso anexado de uma instância para outra. Observe o seguinte:

- Ao anexar ou mover volumes Amazon EBS, as instâncias envolvidas na operação devem estar off-line. Se a instância de interesse não estiver na página Resources, vá até a página Instances e interrompa a instância. Após ser interrompida, você pode retornar à página Resources e anexar ou mover o recurso.
- Ao anexar ou mover endereços Elastic IP, as instâncias podem estar online ou offline.
- Se você excluir uma instância, os recursos anexados permanecem registrados na pilha. Você pode, em seguida, anexar o recurso em outra instância ou, caso não precise mais dele, cancelar o registro do recurso.

Tópicos

- Atribuição de volumes Amazon EBS a uma instância
- Como associar endereços elastic IP com uma instância
- Como anexar instâncias Amazon RDS a um aplicativo

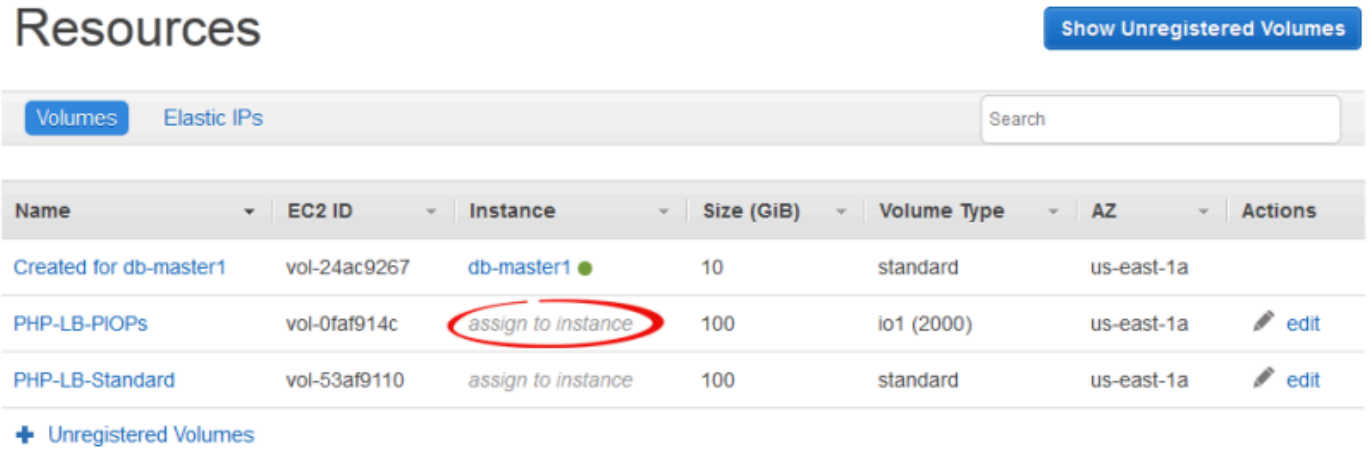
Atribuição de volumes Amazon EBS a uma instância

> **Note**
>
> Não é possível atribuir volumes Amazon EBS às instâncias do Windows.

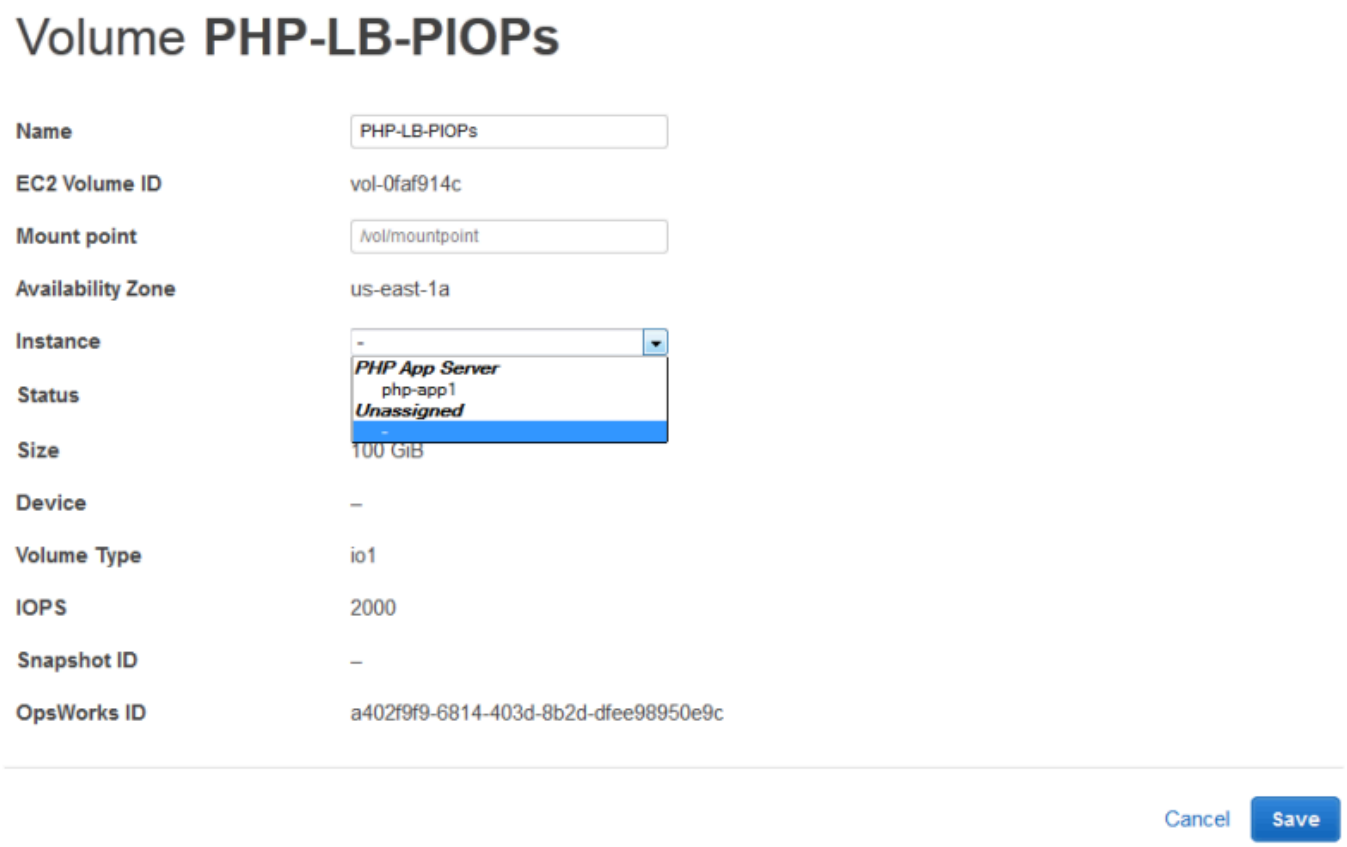
Você pode atribuir um volume Amazon EBS registrado a uma instância e movê-lo de uma instância para outra, no entanto ambas as instâncias devem estar offline.

Para atribuir um volume Amazon EBS a uma instância

1. Na página Recursos, clique em assign to instance na coluna Instance do volume apropriado.



2. Na página de detalhes do volume, selecione a instância apropriada, especifique o nome e o ponto de montagem do volume, e clique em Save para anexar o volume à instância.



> **Important**
>
> Se você atribuiu um volume externo em uso à sua instância, você deve usar o EC2 console, a API ou a CLI da Amazon para cancelar a atribuição da instância original ou o processo de início falhará.

Você também pode usar a página de detalhes para mover um volume Amazon EBS atribuído, para outra instância na pilha.

Para mover um volume Amazon EBS para outra instância

1. Verifique se ambas as instâncias estão no estado offline.
2. Na página Resources, clique em Volumes e, em seguida, clique em edit na coluna Actions do volume.
3. Execute um destes procedimentos:

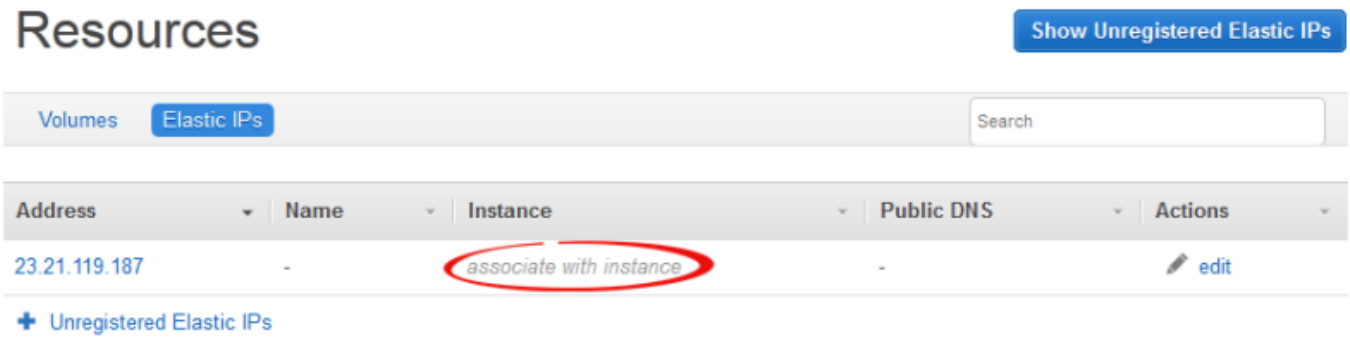
- Para mover o volume para outra instância na pilha, selecione a instância apropriada na lista Instance e clique em Save.
- Para mover o volume para uma instância em outra pilha, cancele o registro do volume, registre o volume na nova pilha, e anexe-o à nova instância.

Como associar endereços elastic IP com uma instância

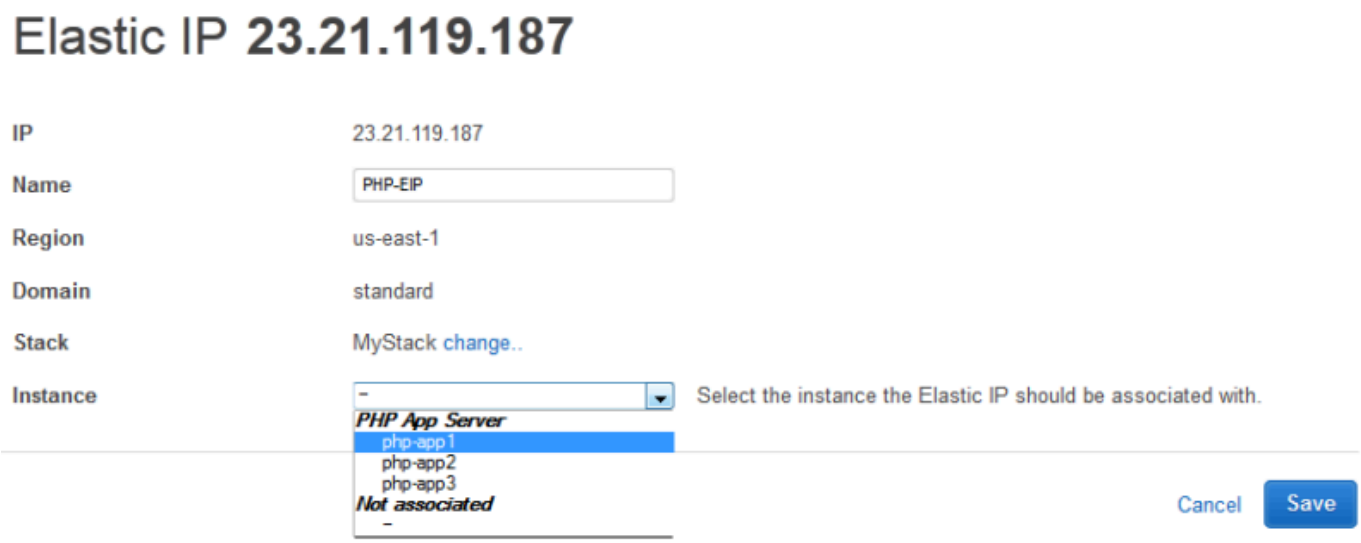
Você pode associar um endereço Elastic IP registrado com uma instância e movê-lo de uma instância para outra, incluindo instâncias em outras pilhas. As instâncias podem estar online ou offline.

Como associar um endereço IP elástico a uma instância

1. Na página Resources, clique em associate with instance na coluna Instance do volume apropriado.



2. Na página de detalhes do endereço, selecione a instância apropriada, especifique o nome do endereço, e clique em Save para associar o endereço à instância.

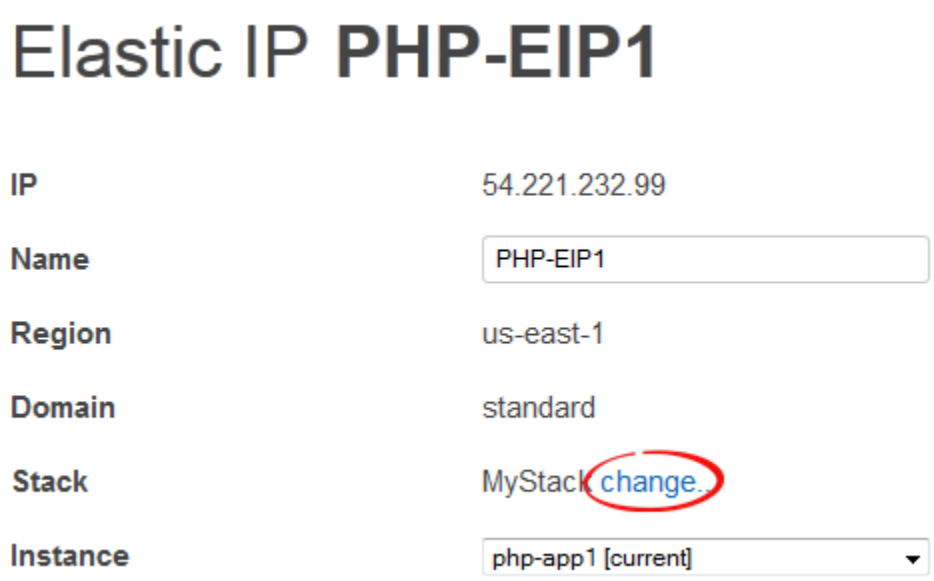


> **Note**
>
> Se o endereço IP elástico estiver atualmente associado a outra instância on-line, o AWS OpsWorks Stacks reatribuirá automaticamente o endereço à nova instância.

Você também pode usar a página de detalhes para mover um endereço Elastic IP associado para outra instância.

Para mover um endereço Elastic IP para outra instância

1. Na página Recursos, clique em Elastic IPs e clique em editar na coluna Ações do endereço.
2. Execute um destes procedimentos:
   - Para mover o endereço para outra instância na pilha, selecione a instância apropriada na lista Instance e clique em Save.
   - Para mover o endereço para uma instância em outra pilha, clique em alterar nas configurações da Stack para visualizar uma lista das pilhas disponíveis. Selecione uma pilha na lista Stack e uma instância na lista Instance. Em seguida, clique em Save.



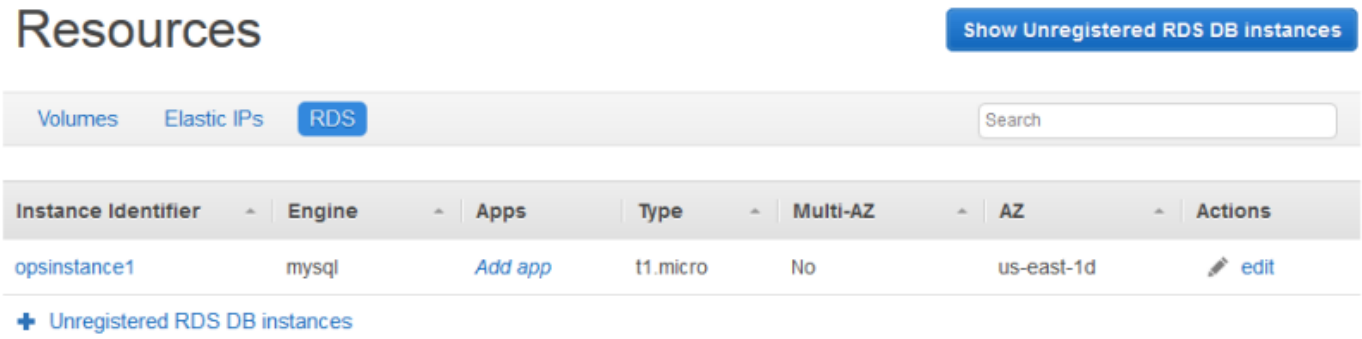
Depois de anexar ou mover um endereço, o AWS OpsWorks Stacks aciona um evento Configure lifecycle para notificar as instâncias da pilha sobre a alteração.

Como anexar instâncias Amazon RDS a um aplicativo

Você pode anexar uma instância Amazon RDS a um ou mais aplicativos.

Para anexar uma instância Amazon RDS a um aplicativo

1. Na página Resources, clique em Add app na coluna Apps da instância apropriada.



2. Use a página Adicionar aplicativo para anexar a instância do Amazon RDS. Para obter mais informações, consulte Adição de aplicativos.

Como um Amazon RDS pode ser anexado a vários aplicativos, não existe um procedimento especial para mover a instância de um aplicativo para outro. Basta editar o primeiro aplicativo para remover

a instância RDS ou editar o segundo aplicativo para adicionar a instância RDS. Para obter mais informações, consulte Editar aplicativos.

Como desanexar recursos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Quando um recurso anexado não é mais necessário, você pode desanexá-lo. Este recurso permanece registrado na pilha e pode ser anexado em outro lugar.

Tópicos

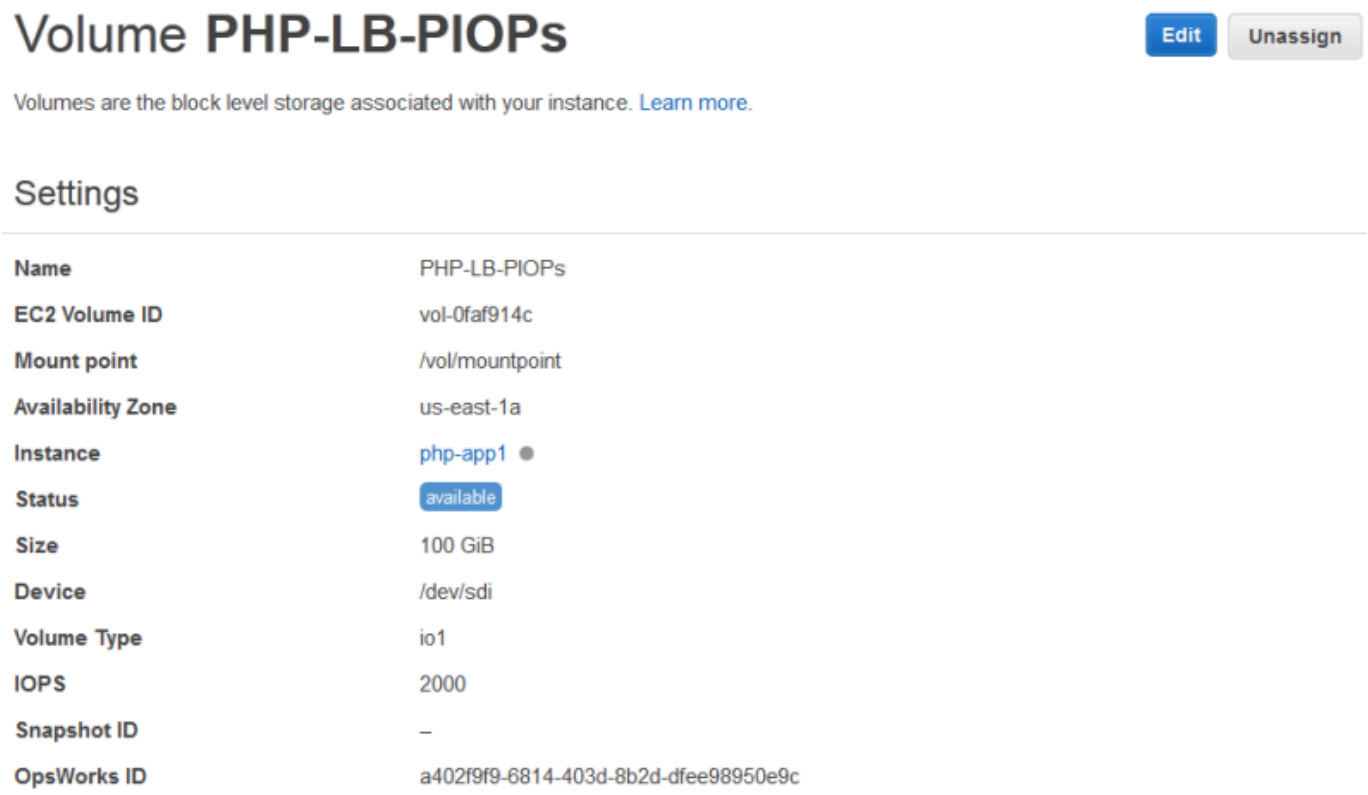
- Como cancelar a atribuição de volumes Amazon EBS
- Como desassociar endereços elastic IP
- Como desanexar instâncias Amazon RDS

Como cancelar a atribuição de volumes Amazon EBS

Use o procedimento a seguir para cancelar a atribuição de um volume Amazon EBS de sua instância.

Para cancelar a atribuição de um volume Amazon EBS

1. Verifique se a instância está no estado offline.
2. Na página Resources, clique em Volumes e clique em nome do volume.
3. Na página de detalhes do volume, clique em Unassign.

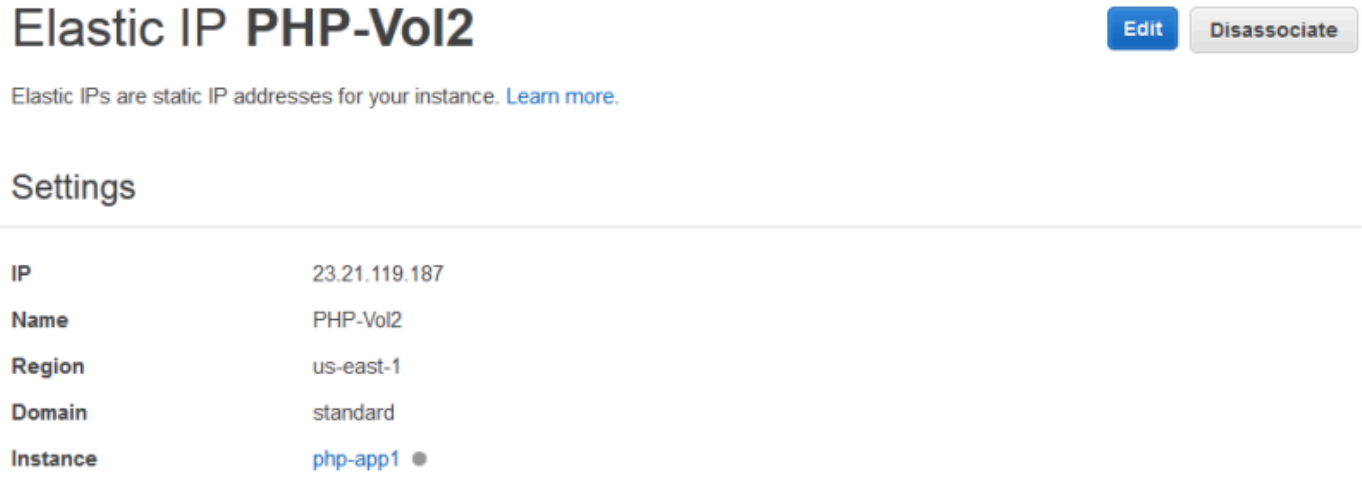


Como desassociar endereços elastic IP

Use o procedimento a seguir para desassociar um endereço Elastic IP de sua instância.

Para dissociar um endereço IP elástico

1. Na página Recursos, clique em Elastic IPs e clique em editar na coluna Ações do endereço.
2. Na página de detalhes do endereço, clique em Disassociate.



Depois de desassociar um endereço, o AWS OpsWorks Stacks aciona um evento Configure lifecycle para notificar as instâncias da pilha sobre a alteração.

Como desanexar instâncias Amazon RDS

Use o procedimento a seguir para desanexar um Amazon RDS de um aplicativo.

Para desanexar uma instância Amazon RDS

1. Na página Resources, clique em RDS e clique no aplicativo apropriado na coluna Apps.
2. Clique em Edit e edite a configuração do aplicativo para desanexar a instância. Para obter mais informações, consulte Editar aplicativos.

> **Note**
>
> Este procedimento desanexa um Amazon RDS de um único aplicativo. Se a instância estiver anexada a vários aplicativos, você deve repetir esse procedimento para cada aplicativo.

Como cancelar registros de recursos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Caso não seja mais necessário manter um recurso registrado em um pilha, você pode cancelar o registro dele. O cancelamento do registro não exclui o recurso da sua conta; ele permanece lá e pode ser registrado em outra pilha ou usado fora do Stacks. AWS OpsWorks Se você deseja excluir o recurso completamente, existem duas opções:

- Se um recurso Amazon EBS ou Elastic IP está anexado a uma instância, você pode excluir o recurso quando excluir a instância.

Acesse a página Instances, clique em delete na coluna Actions da instância e selecione Delete instance's EBS volumes ou Delete the instance's Elastic IP.

- Cancele o registro do recurso e use o console, a API ou a CLI da EC2 Amazon ou do Amazon RDS para excluí-lo.

Tópicos

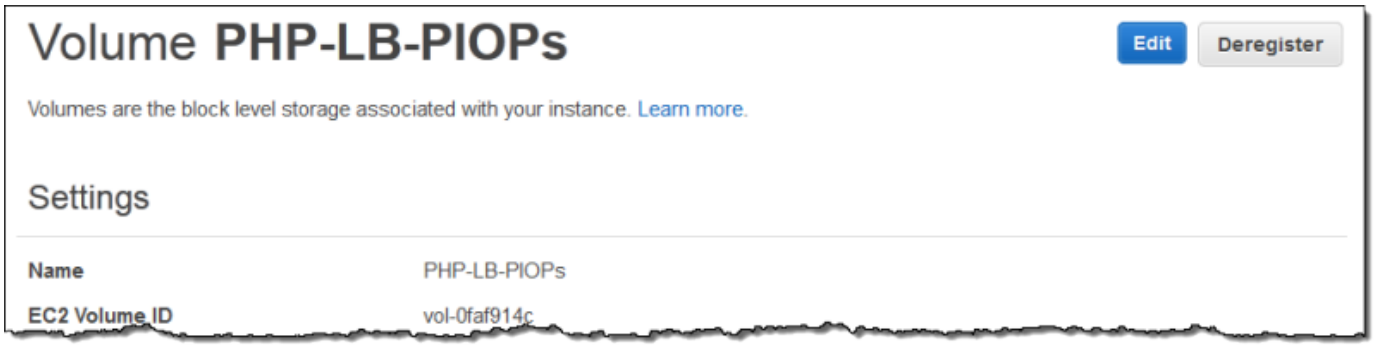
- Cancelando o registro de volumes do Amazon EBS
- Como cancelar o registro de endereços elastic IP
- Cancelando o registro de instâncias do Amazon RDS

Cancelando o registro de volumes do Amazon EBS

Use o procedimento a seguir para cancelar o registro de um volume Amazon EBS.

Cancelar o registro de volume do Amazon EBS

1. Se o volume está anexado a uma instância, cancele sua atribuição, conforme descrito em Como cancelar a atribuição de volumes Amazon EBS.
2. Na página Resources, clique no nome do volume na coluna Name.
3. Na página de detalhes do volume, clique em Deregister.

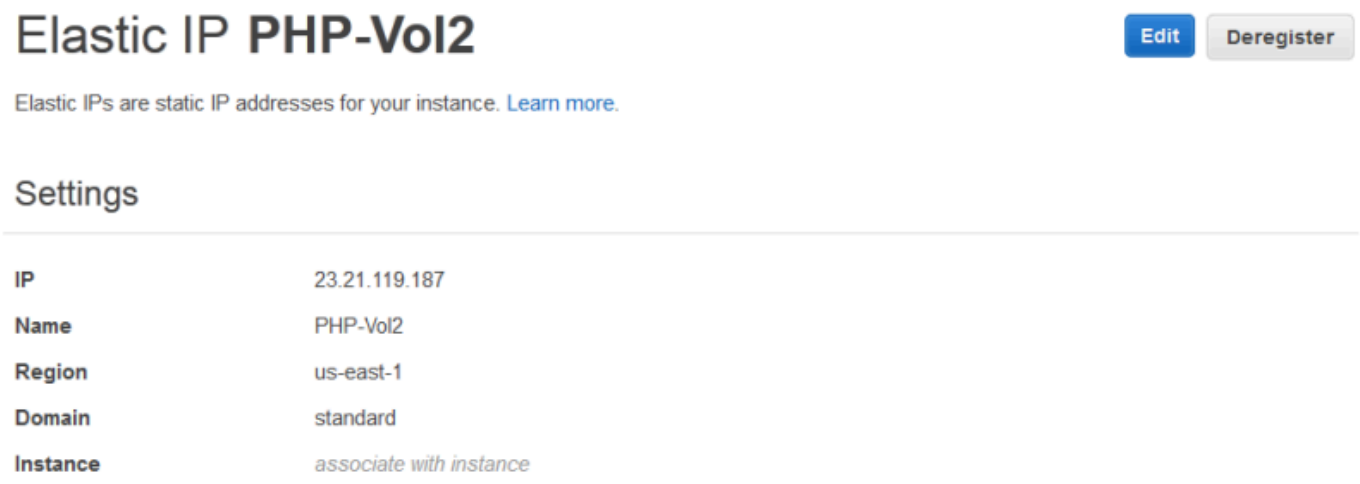


Como cancelar o registro de endereços elastic IP

Use o procedimento a seguir para cancelar o registro de um endereço Elastic IP.

Para cancelar o registro de um endereço Elastic IP

1. Se o endereço está associada a uma instância, desassocie-o, conforme descrito em Como desassociar endereços elastic IP.
2. Na página Recursos, clique em Elastic IPs e, em seguida, clique no endereço IP na coluna Endereço.
3. Na página de detalhes do endereço, clique em Deregister.



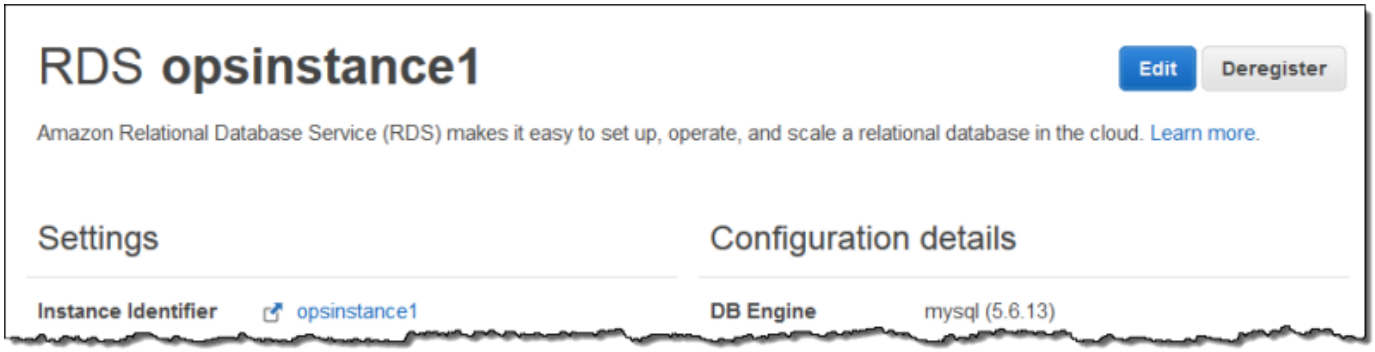
> **Note**
>
> Se você simplesmente deseja registrar um endereço Elastic IP em uma pilha diferente, você deve cancelar o registro na pilha atual e, em seguida, registrá-lo na nova pilha. No entanto, você pode mover um endereço Elastic IP anexado diretamente para uma instância em outra pilha. Para obter mais informações, consulte Como anexar e mover recursos.

Cancelando o registro de instâncias do Amazon RDS

Use o procedimento a seguir para cancelar o registro de uma instância Amazon RDS.

Cancelar o registro de uma instância do Amazon RDS

1. Se a instância estiver associada a um aplicativo, desanexe-a, conforme descrito em Como desanexar recursos.
2. Na página Resources, clique em RDS e, em seguida, no nome da instância.
3. Na página de detalhes da instância, clique em Deregister.



Tags

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Tags podem ajudá-lo a agrupar recursos nas pilhas do Chef 11.10, Chef 12 e Chef 12.2 e rastrear os custos de uso desses recursos no Gerenciamento de Faturamento e Custos da AWS.

Você pode aplicar tags no nível de camada e pilha. Quando você cria uma tag, está aplicando-a para cada recurso dentro da estrutura com a tag. Por exemplo, se você aplica uma tag a uma camada, está aplicando a tag a cada instância, volume do Amazon EBS (exceto o root) ou balanceador de carga do Elastic Load Balancing na camada. Atualmente, as tags não podem ser aplicadas à raiz, ou padrão, volume do EBS de uma instância.

Tags são pares de valores-chave que você atribui a pilhas ou camadas em Pilhas. AWS OpsWorks Depois de criar tags, abra o console do Faturamento e gerenciamento de custos para ativar tags definidas pelo usuário. Para obter mais informações sobre como ativar suas tags e usá-las para rastrear e gerenciar os custos dos recursos do AWS OpsWorks Stacks, consulte Usando tags de alocação de custos e ativando tags de alocação de custos definidas pelo usuário no Billing and Cost Management User Guide.

As tags funcionam de forma semelhante aos atributos personalizados no AWS OpsWorks Stacks. As tags que você aplica a uma pilha são herdadas por cada camada na pilha. No nível da camada, você

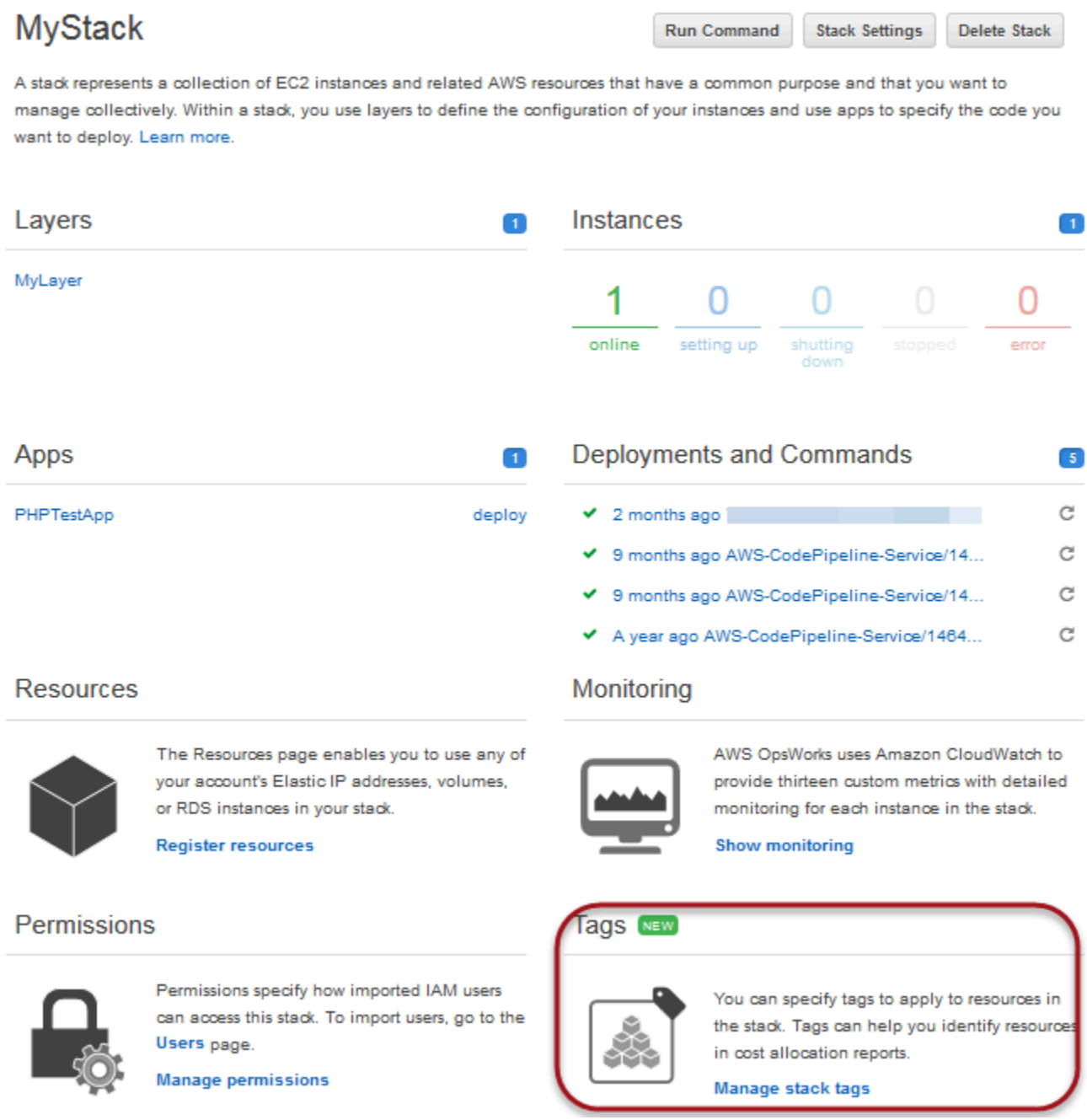
pode substituir os valores (mas não os nomes das chaves) das tags herdadas e adicionar novas tags específicas da camada. AWS OpsWorks aplica o conjunto de tags resultante a todos os recursos na camada. Conforme você cria novos ou atribui recursos existentes a uma camada, os novos recursos na camada são marcados com o mesmo grupo de tags.

Tópicos

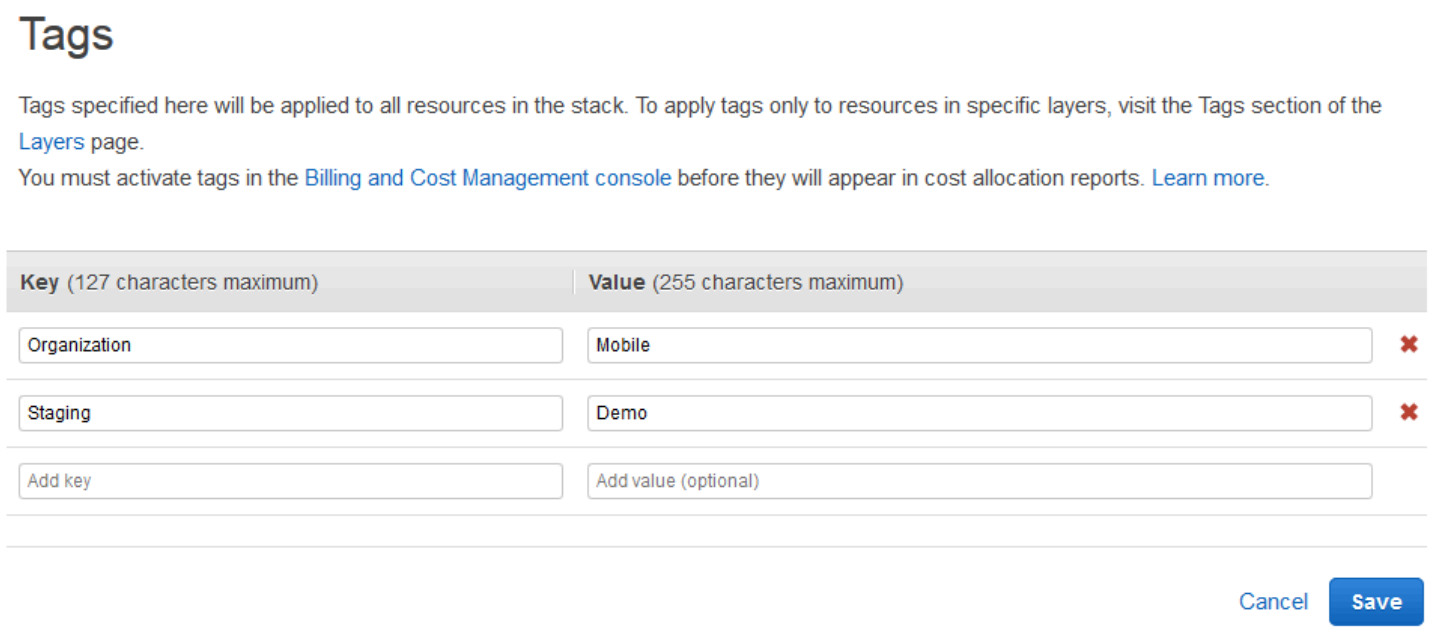
- Definição de tags no nível da pilha
- Definição de tags no nível da camada
- Gerenciando tags com o AWS CLI
- Limitações de tag

Definição de tags no nível da pilha

No nível da pilha, você pode adicionar e gerenciar tags escolhendo Tags na página inicial da pilha.

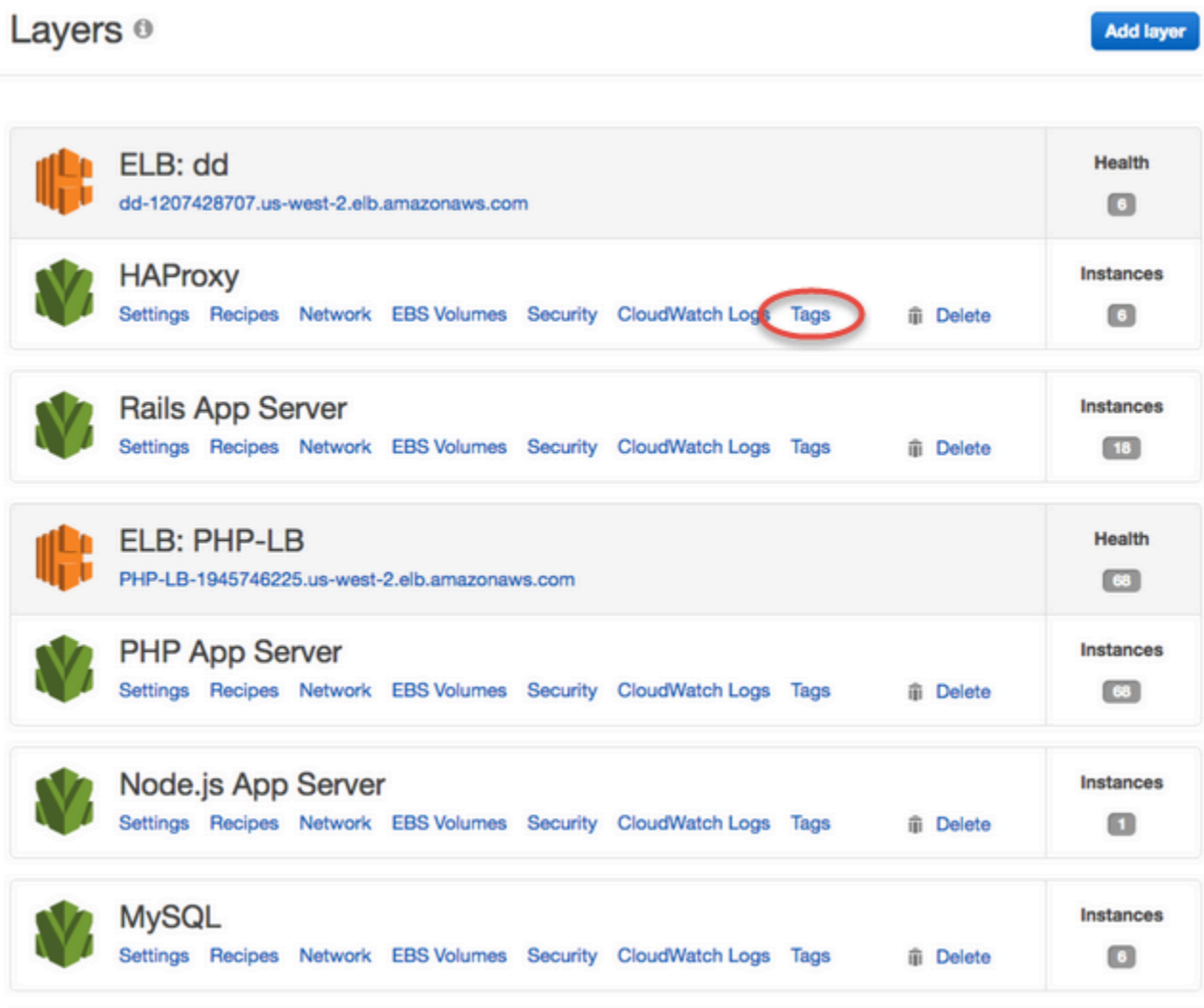


Na página Tags, adicione tags como pares chave-valor. A captura de tela a seguir mostra algumas tags de exemplo. Você pode excluir tags marcando o X em vermelho à direita do par chave-valor.

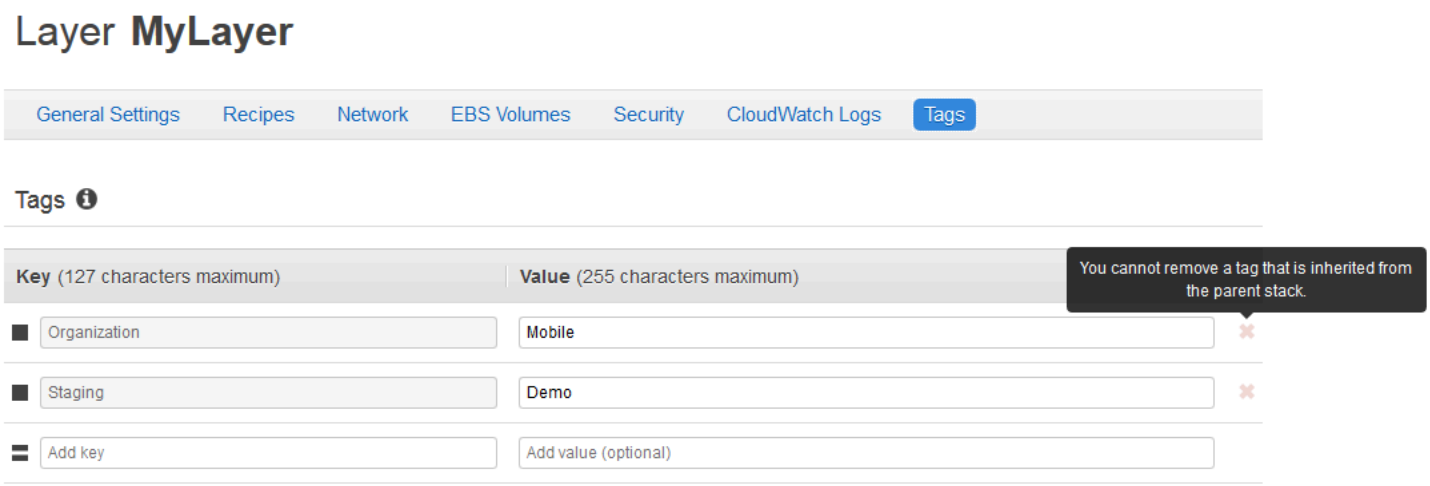


Definição de tags no nível da camada

No nível da camada, defina tags escolhendo a guia Tags. Você pode encontrar essa guia na página inicial Layers (Camadas), e na página inicial de cada camada individual.



Quando você muda ou adiciona tags no nível da camada, esteja ciente de que as tags adicionadas no nível da pilha parente são herdadas pela camada e seus recursos. Por mais que você possa mudar os valores das tags herdadas, não pode mudar os nomes-chave ou excluir tags herdadas. Alterar os nomes-chave ou excluir tags herdadas da pilha parente nas configurações de pilha. A captura de tela a seguir mostra exemplos de tags herdadas do nível da pilha. Tags herdadas estão desabilitadas.



Para obter mais informações sobre como adicionar tags a pilhas, consulte Criar uma nova pilha. Para obter mais informações sobre como adicionar tags a camadas, consulte Como editar a configuração de uma OpsWorks camada.

Gerenciando tags com o AWS CLI

Você também pode usar AWS CLI comandos para adicionar e remover tags no nível da pilha e da camada. Para obter mais informações sobre como baixar e instalar o AWS CLI, consulte Instalando a interface de linha de AWS comando. Lembre-se de adicionar o parâmetro `--region` ao seu comando se a pilha que você deseja marcar não está em sua região padrão. Atualmente, as camadas ARNs não aparecem no AWS Management Console. Para obter o ARN da camada, execute o comando describe-layers.

Para adicionar tags usando o AWS CLI

- No prompt de AWS CLI comando, digite o comando a seguir, substituindo ***stack_or_layer_ARN*** e especificando suas tags de pares de valores-chave e pressione Enter. Pontos de interrogação duplos são recuados com barras invertidas.

  ```
  aws opsworks tag-resource --resource-arn stack_or_layer_ARN --tags "{\"key\":
  \"value\",\"key\":\"value\"}"
  ```

  Veja um exemplo a seguir.

  ```
  aws opsworks tag-resource --resource-arn arn:aws:opsworks:us-
  east-2:800000000003:stack/500b99c0-ec00-4cgg-8a0d-1000000jjd1b --tags "{\"Stage\":
  \"Production\",\"Organization\":\"Mobile\"}"
  ```

Para remover tags usando o AWS CLI

- No prompt de AWS CLI comando, digite o seguinte e pressione Enter.

```
aws opsworks untag-resource --resource-arn stack_or_layer_ARN --tag-keys "[\"key\",
\"key\"]"
```

Para remover tags, você especifica apenas a chave da tag que deseja remover. Veja um exemplo a seguir.

```
aws opsworks untag-resource --resource-arn arn:aws:opsworks:us-
east-2:800000000003:stack/500b99c0-ec00-4cgg-8a0d-1000000jjd1b --tag-keys "[\"Stage
\",\"Organization\"]"
```

> **Note**
>
> Você não pode remover tags herdadas (tags que foram adicionadas ao nível de pilha parente) de uma camada. Em vez disso, remova tags herdadas da pilha.

Limitações de tag

Mantenha as seguintes limitações em mente ao criar tags.

- AWS OpsWorks As pilhas limitam o número de tags definidas pelo usuário no nível da pilha e da camada a 40, incluindo tags definidas pelo usuário herdadas de um nível principal. Isso deixa 10 slots disponíveis para tags padrão prefixadas com `opsworks:` e tags definidas por outros AWS processos. É permitido um máximo de 50 tags em um recurso, incluindo tags definidas pelo usuário e tags padrão criadas por AWS.
- Chaves de tags não podem começar com **`aws:`**, **`opsworks:`** ou **`rds:`**. Não use **`name`** ou **`Name`** como chave de tag, pois **`Name`** é reservada pelo AWS OpsWorks Stacks.
- Uma chave por ter, no máximo, 127 caracteres, e pode conter apenas letras de código único, números ou separadores, ou os caracteres especiais a seguir: `+ - = . _ : /` .
- Um valor pode ter, no máximo, 255 caracteres, e conter apenas letras de código único, números ou separadores, ou os caracteres especiais a seguir: `+ - = . _ : /` .

Monitoramento

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode monitorar suas pilhas das seguintes maneiras.

- AWS OpsWorks O Stacks usa CloudWatch a Amazon para fornecer treze métricas personalizadas com monitoramento detalhado para cada instância na pilha.
- AWS OpsWorks O Stacks se integra AWS CloudTrail para registrar todas as chamadas da API AWS OpsWorks Stacks e armazenar os dados em um bucket do Amazon S3.
- Você pode usar o Amazon CloudWatch Logs para monitorar o sistema, o aplicativo e os registros personalizados da sua pilha.

Tópicos

- Monitorando pilhas usando a Amazon CloudWatch
- Chamadas de API do Logging AWS OpsWorks Stacks com AWS CloudTrail
- Usando Amazon CloudWatch Logs com AWS OpsWorks pilhas
- Monitorando pilhas usando Amazon Events CloudWatch

Monitorando pilhas usando a Amazon CloudWatch

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks usa o Amazon CloudWatch (CloudWatch) para fornecer monitoramento para pilhas.

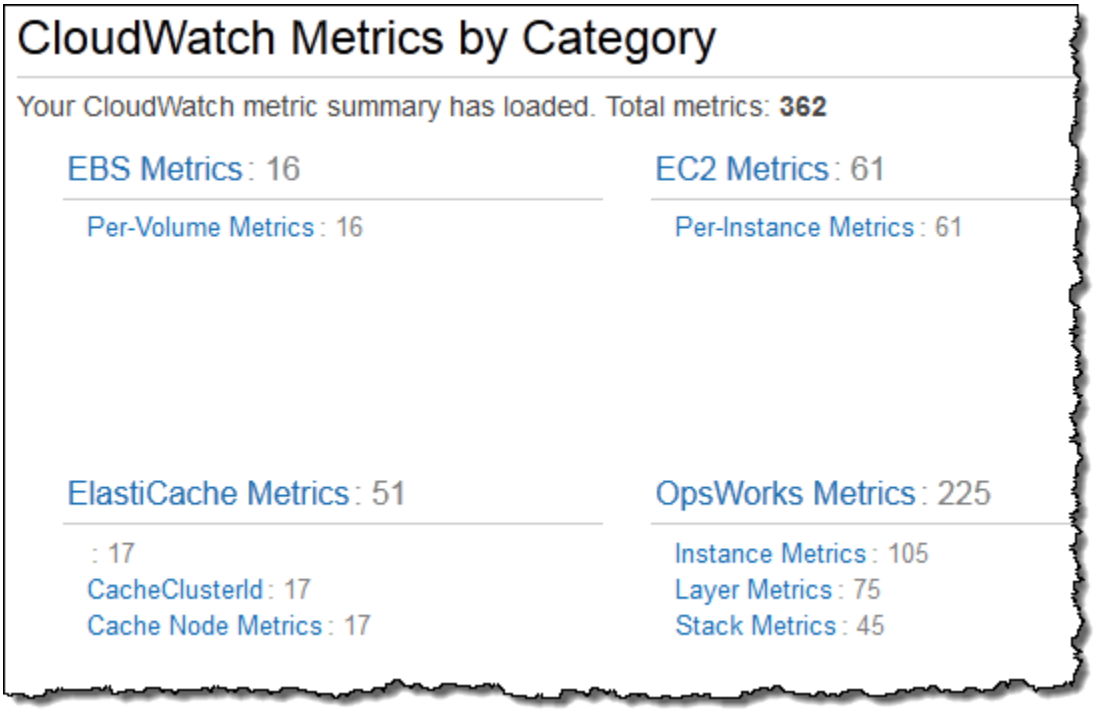
- Para pilhas Linux, o AWS OpsWorks Stacks suporta treze métricas personalizadas para fornecer monitoramento detalhado para cada instância na pilha e resume os dados para sua conveniência na página Monitoramento.
- Para pilhas do Windows, você pode monitorar EC2 as métricas padrão da Amazon para suas instâncias com o CloudWatch console.

  A página Monitoring (Monitoramento) não exibe os índices do Windows.

A página Monitoramento exibe métricas para uma pilha inteira, uma camada ou uma instância. AWS OpsWorks As métricas do Stacks são diferentes das EC2 métricas da Amazon. Você também pode ativar métricas adicionais por meio do CloudWatch console, mas elas normalmente exigem cobranças adicionais. Você também pode visualizar os dados subjacentes no CloudWatch console, da seguinte forma:

Para visualizar métricas OpsWorks personalizadas em CloudWatch

1. Abra o CloudWatch console em https://console.aws.amazon.com/cloudwatch/.
2. Na barra de navegação, selecione a região da pilha.
3. No painel de navegação, selecione Métricas.
4. Em OpsWorks Métricas, escolha Métricas de instância, Métricas de camada ou Métricas de pilha.



> **Note**
>
> AWS OpsWorks O Stacks coleta métricas executando um processo em cada instância (o agente da instância). Como CloudWatch coleta métricas de forma diferente, usando o hipervisor, os valores no CloudWatch console podem ser ligeiramente diferentes dos valores correspondentes na página Monitoramento no console AWS OpsWorks Stacks.

Você também pode usar o CloudWatch console para definir alarmes. Para obter mais informações sobre como criar alarmes, consulte Criação de alarmes na Amazon CloudWatch . Para obter uma lista de métricas CloudWatch personalizadas, consulte AWS OpsWorks Metrics and Dimensions. Para obter mais informações, consulte Amazon CloudWatch.

Tópicos

- AWS OpsWorks Métricas de pilhas
- Dimensões para AWS OpsWorks Stacks Metrics
- Índices da pilha
- Índices de camada
- Métricas de instância

AWS OpsWorks Métricas de pilhas

AWS OpsWorks O Stacks envia as seguintes métricas a CloudWatch cada cinco minutos.

Métricas de CPU

Métrica	Descrição
`cpu_idle`	O percentual de tempo em que a CPU está ociosa. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma
`cpu_nice`	A porcentagem de tempo em que a CPU está lidando com processos com um valor `nice` positivo, que têm uma prioridade de agendamento menor. Para obter mais informações sobre o que isso mede, consulte nice (Unix). Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma
`cpu_steal`	À medida que a AWS aloca recursos de CPU do hipervisor entre um número cada vez maior de instâncias, a carga de virtualização aumenta e pode afetar a frequência com que o hipervisor pode realizar o trabalho solicitado em uma instância. O `cpu_steal` mede a porcentagem de tempo em

Métrica	Descrição
	que uma instância espera que o hipervisor aloque recursos físicos da CPU. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma
`cpu_system`	O percentual de tempo que a CPU está lidando com operações do sistema. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma
`cpu_user`	O percentual de tempo que a CPU está lidando com operações do usuário. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma

Métrica	Descrição
`cpu_waitio`	O percentual de tempo que a CPU está aguardando operações de entrada/saída. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma

Métricas de memória

Métrica	Descrição
`memory_buffers`	A quantidade de memória armazenada em buffer. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma
`memory_cached`	A quantidade de memória em cache. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma

Métrica	Descrição
`memory_free`	A quantidade de memória livre. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma
`memory_swap`	A quantidade de espaço de troca. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma
`memory_total`	A quantidade total de memória. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma

Métrica	Descrição
`memory_used`	A quantidade de memória em uso. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma

Métricas de carga

Métrica	Descrição
`load_1`	A média da carga ao longo de uma janela de um minuto. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma
`load_5`	A carga média ao longo de uma janela de cinco minutos. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`.

Métrica	Descrição
	Unidade: nenhuma
`load_15`	A média da carga ao longo de uma janela de 15 minutos. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma

Métricas do processo

Métrica	Descrição
`procs`	O número de processos ativos. Dimensões válidas: os IDs recursos individuais para os quais você está visualizando métricas: StackId, LayerId, ou InstanceId. Estatísticas válidas: `Average`, `Minimum`, `Maximum`, `Sum`, ou `Data Samples`. Unidade: nenhuma

Dimensões para AWS OpsWorks Stacks Metrics

AWS OpsWorks As métricas de pilhas usam o namespace AWS OpsWorks Stacks e fornecem métricas para as seguintes dimensões:

Dimensão	Descrição
`StackId`	Valores médios de uma pilha.
`LayerId`	Valores médios de uma camada.
`InstanceId`	Valores médios de uma instância.

Índices da pilha

Para ver um resumo das métricas de uma pilha inteira, selecione uma pilha no Painel de AWS OpsWorks pilhas e clique em Monitoramento no painel de navegação. O exemplo a seguir é para uma pilha com um PHP e uma camada DB.

A exibição da pilha apresenta gráficos dos quatro tipos de índices para cada camada ao longo de um período de tempo especificado: 1 hora, 8 horas, 24 horas, 1 semana ou 2 semanas. Observe o seguinte:

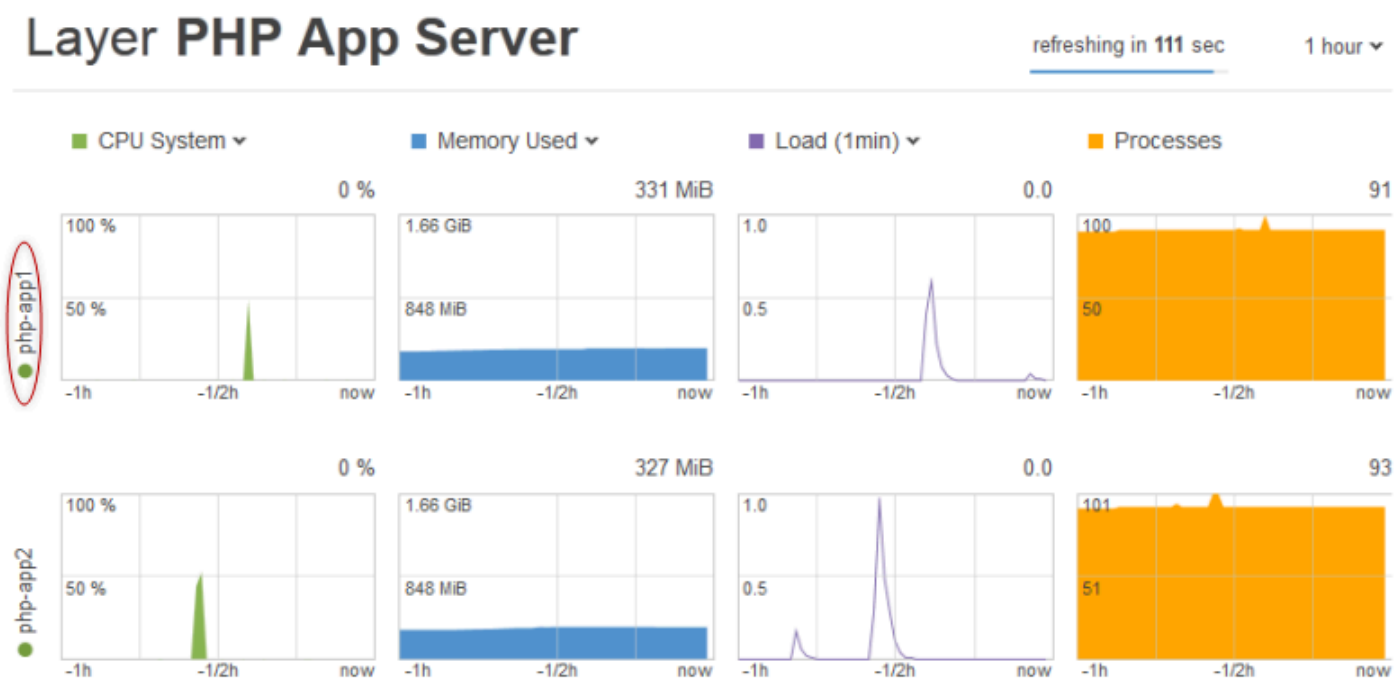
- AWS OpsWorks O Stacks atualiza periodicamente os gráficos; o cronômetro de contagem regressiva no canto superior direito indica o tempo restante até a próxima atualização,
- Se uma camada tem mais de uma instância, os gráficos exibem valores médios para a camada.

- Você pode especificar o período de tempo clicando na lista no canto superior direito e selecionando o valor desejado.

Para cada tipo de índice, você pode usar a lista na parte superior do gráfico para selecionar o índice específico que deseja visualizar.

Índices de camada

Para visualizar índices para uma camada específica, clique no nome da camada na exibição Monitoring Layers. O exemplo a seguir mostra os índices para a camada PHP, que possui duas instâncias.

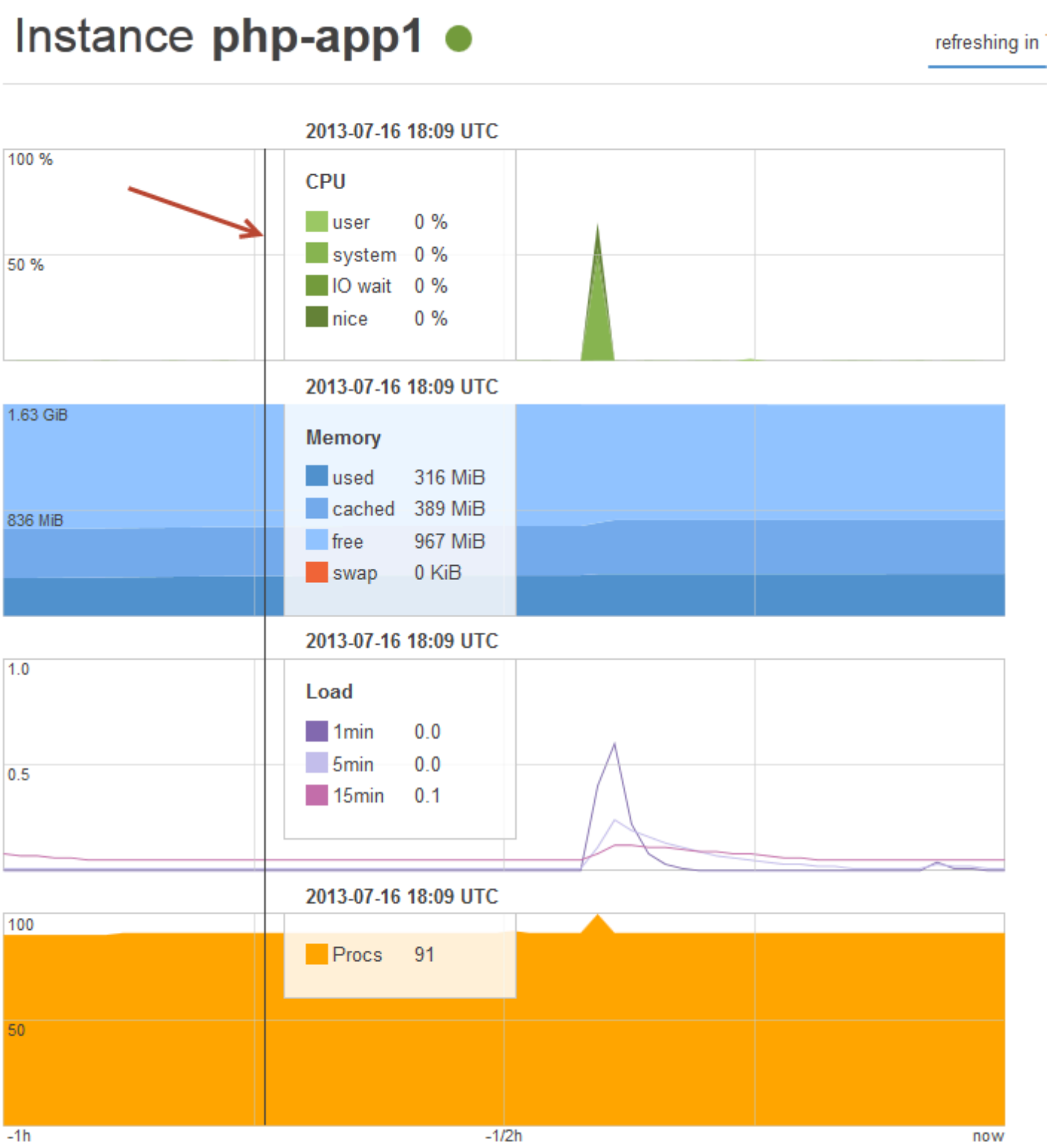


Os tipos de índices são iguais aos dos índices da pilha e, para cada tipo, você pode usar a lista na parte superior do gráfico para selecionar o índice específico que deseja visualizar.

> **Note**
>
> Você também pode exibir índices de camada acessando a página de detalhes da camada e, em seguida, clicando em Monitoring na parte superior direita.

Métricas de instância

Para visualizar métricas de uma instância específica, clique no nome da instância na exibição de monitoramento da camada. O exemplo a seguir mostra as métricas da instância php-app1 da camada PHP.


Instance php-app1
refreshing in
2013-07-16 18:09 UTC
100 %
50 %
CPU
user 0 %
system 0 %
IO wait 0 %
nice 0 %
2013-07-16 18:09 UTC
1.63 GiB
836 MiB
Memory
used 316 MiB
cached 389 MiB
free 967 MiB
swap 0 KiB
2013-07-16 18:09 UTC
1.0
0.5
Load
1min 0.0
5min 0.0
15min 0.1
2013-07-16 18:09 UTC
100
50
Procs 91
-1h
-1/2h
now

Os gráficos resumem todas as métricas disponíveis para cada tipo de métrica. Para obter os valores exatos de um determinado ponto no tempo, use o mouse para mover a barra deslizante (indicada pela seta vermelha na ilustração anterior) para a posição apropriada.

> **Note**
>
> Você também pode exibir os índices de instância acessando a página de detalhes da instância e selecionando Monitoring na parte superior direita.

Chamadas de API do Logging AWS OpsWorks Stacks com AWS CloudTrail

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks é integrado com AWS CloudTrail um serviço que fornece um registro das ações realizadas por uma identidade do IAM ou por um AWS serviço no AWS OpsWorks Stacks. CloudTrail captura todas as chamadas de API para AWS OpsWorks Stacks como eventos, incluindo chamadas do console AWS OpsWorks Stacks e de chamadas de código para o Stacks. AWS OpsWorks APIs Se você criar uma trilha, poderá habilitar a entrega contínua de CloudTrail eventos para um bucket do Amazon S3, incluindo eventos para AWS OpsWorks Stacks. Se você não configurar uma trilha, ainda poderá ver os eventos mais recentes no CloudTrail console no Histórico de eventos. Usando as informações coletadas por CloudTrail, você pode determinar a solicitação que foi feita ao AWS OpsWorks Stacks, o endereço IP do qual a solicitação foi feita, quem fez a solicitação, quando ela foi feita e detalhes adicionais.

Para saber mais sobre isso CloudTrail, consulte o Guia AWS CloudTrail do usuário.

AWS OpsWorks Acumula informações em CloudTrail

CloudTrail é ativado em sua AWS conta quando você cria a conta. Quando a atividade ocorre no AWS OpsWorks Stacks, essa atividade é registrada em um CloudTrail evento junto com outros

eventos de AWS serviço no histórico de eventos. Você pode visualizar, pesquisar e baixar eventos recentes em sua AWS conta. Para obter mais informações, consulte Visualização de eventos com histórico de CloudTrail eventos.

Para um registro contínuo dos eventos em sua AWS conta, incluindo eventos para AWS OpsWorks Stacks, crie uma trilha. Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Por padrão, ao criar uma trilha no console, ela é aplicada a todas as regiões da . A trilha registra eventos de todas as regiões na AWS partição e entrega os arquivos de log ao bucket do Amazon S3 que você especificar. Além disso, você pode configurar outros AWS serviços para analisar e agir com base nos dados de eventos coletados nos CloudTrail registros. Para obter mais informações, consulte:

- Visão geral da criação de uma trilha
- CloudTrail Serviços e integrações compatíveis
- Configurando notificações do Amazon SNS para CloudTrail
- Recebendo arquivos de CloudTrail log de várias regiões e recebendo arquivos de CloudTrail log de várias contas

Todas as ações do AWS OpsWorks Stacks são registradas CloudTrail e documentadas na Referência da API AWS OpsWorks Stacks. Por exemplo, chamadas para as `StartInstance` ações `CreateLayerDescribeInstances`, e geram entradas nos arquivos de CloudTrail log.

Cada entrada de log ou evento contém informações sobre quem gerou a solicitação. As informações de identidade ajudam a determinar:

- Se a solicitação foi feita com credenciais de usuário-raiz ou usuário do IAM.
- Se a solicitação foi feita com credenciais de segurança temporárias de uma função ou de um usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para obter mais informações, consulte Elemento userIdentity do CloudTrail .

Entendendo as entradas do arquivo de log do AWS OpsWorks Stacks

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log para um bucket do Amazon S3 que você especificar. CloudTrail os arquivos de log contêm uma ou mais

entradas de log. Um evento representa uma única solicitação de qualquer fonte e inclui informações sobre a ação solicitada, a data e a hora da ação, os parâmetros da solicitação e assim por diante. CloudTrail os arquivos de log não são um rastreamento de pilha ordenado das chamadas públicas de API, portanto, eles não aparecem em nenhuma ordem específica.

O exemplo a seguir mostra uma entrada de CloudTrail registro que demonstra a `CreateLayer` ação.

```
  {
"Records": [
    {
        "awsRegion": "us-west-2",
        "eventID": "342cd1ec-8214-4a0f-a68f-8e6352feb5af",
        "eventName": "CreateLayer",
        "eventSource": "opsworks.amazonaws.com",
        "eventTime": "2014-05-28T16:05:29Z",
        "eventVersion": "1.01"ed,
        "requestID": "e3952a2b-e681-11e3-aa71-81092480ee2e",
        "requestParameters": {
            "attributes": {},
            "customRecipes": {},
            "name": "2014-05-28 16:05:29 +0000 a073",
            "shortname": "customcf4571d5c0d6",
            "stackId": "a263312e-f937-4949-a91f-f32b6b641b2c",
            "type": "custom"
        },
        "responseElements": null,
        "sourceIPAddress": "198.51.100.0",
        "userAgent": "aws-sdk-ruby/2.0.0 ruby/2.1 x86_64-linux",
        "userIdentity": {
            "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
            "accountId": "111122223333",
            "arn": "arn:aws:iam::111122223333:user/A-User-Name",
            "principalId": "AKIAI44QH8DHBEXAMPLE",
            "type": "IAMUser",
            "userName": "A-User-Name"
        }
    },
    {
        "awsRegion": "us-west-2",
        "eventID": "a860d8f8-c1eb-449b-8f55-eafc373b49a4",
        "eventName": "DescribeInstances",
```

```
            "eventSource": "opsworks.amazonaws.com",
            "eventTime": "2014-05-28T16:05:31Z",
            "eventVersion": "1.01",
            "requestID": "e4691bfd-e681-11e3-aa71-81092480ee2e",
            "requestParameters": {
                "instanceIds": [
                    "218289c4-0492-473d-a990-3fbe1efa25f6"
                ]
            },
            "responseElements": null,
            "sourceIPAddress": "198.51.100.0",
            "userAgent": "aws-sdk-ruby/2.0.0 ruby/2.1x86_64-linux",
            "userIdentity": {
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "accountId": "111122223333",
                "arn": "arn:aws:iam::111122223333:user/A-User-Name",
                "principalId": "AKIAI44QH8DHBEXAMPLE",
                "type": "IAMUser",
                "userName": "A-User-Name"
            }
        }
    ]
}
```


Usando Amazon CloudWatch Logs com AWS OpsWorks pilhas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para simplificar o processo de monitoramento de registros em várias instâncias, o AWS OpsWorks Stacks oferece suporte ao Amazon CloudWatch Logs. Você ativa CloudWatch Logs no nível da camada em AWS OpsWorks Stacks. CloudWatch A integração de registros funciona com pilhas baseadas em Linux do Chef 11.10 e do Chef 12. Você incorre em cobranças adicionais ao ativar o CloudWatch Logs, então revise os CloudWatchpreços da Amazon antes de começar.

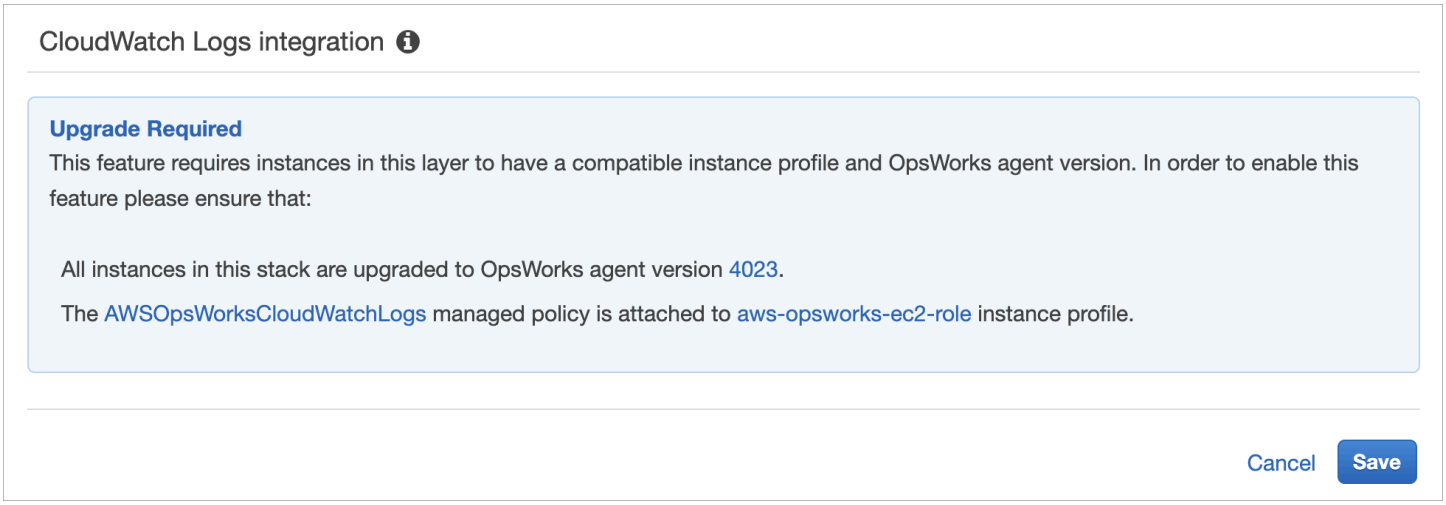
CloudWatch O Logs monitora os registros selecionados quanto à ocorrência de um padrão especificado pelo usuário. Por exemplo, você pode monitorar os logs da ocorrência de um termo literal, como `NullReferenceException`ou contabilizar o número dessas ocorrências. Depois de ativar o CloudWatch Logs in AWS OpsWorks Stacks, o agente do AWS OpsWorks Stacks envia os registros para CloudWatch o Logs. Para obter mais informações sobre CloudWatch registros, consulte Introdução aos CloudWatch registros.

Pré-requisitos

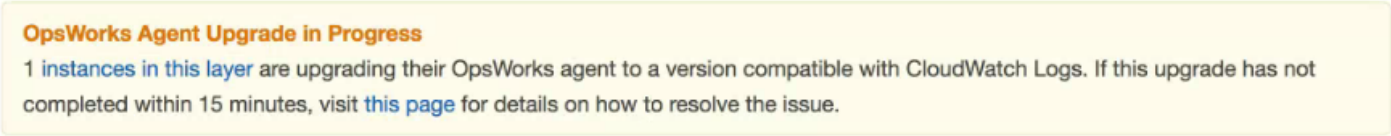
Antes de habilitar CloudWatch os Logs, suas instâncias devem estar executando a versão 3444 ou posterior do agente AWS OpsWorks Stacks nas pilhas do Chef 11.10 e 4023 ou posterior nas pilhas do Chef 12. Você também precisa usar um perfil de instância compatível para todas as instâncias que estiver monitorando usando o CloudWatch Logs.

Se você estiver usando um perfil de instância personalizado (um que o AWS OpsWorks Stacks não fornecia quando você criou a pilha), o AWS OpsWorks Stacks não pode atualizar automaticamente o perfil da instância. Você deve anexar manualmente a AWSOpsWorksCloudWatchLogspolítica ao seu perfil usando o IAM. Para obter mais informações, consulte Políticas gerenciadas pelo IAM no Manual do usuário do IAM.

Se você precisar atualizar a versão do agente ou o perfil da instância, o AWS OpsWorks Stacks exibirá um lembrete semelhante à captura de tela a seguir quando você abre a guia CloudWatch Registros na página Camada.



Atualizar o agente em todas as instâncias em uma camada pode levar algum tempo. Se você tentar ativar CloudWatch os registros em uma camada antes que a atualização do agente seja concluída, você verá uma mensagem semelhante à seguinte.



Habilitando CloudWatch registros

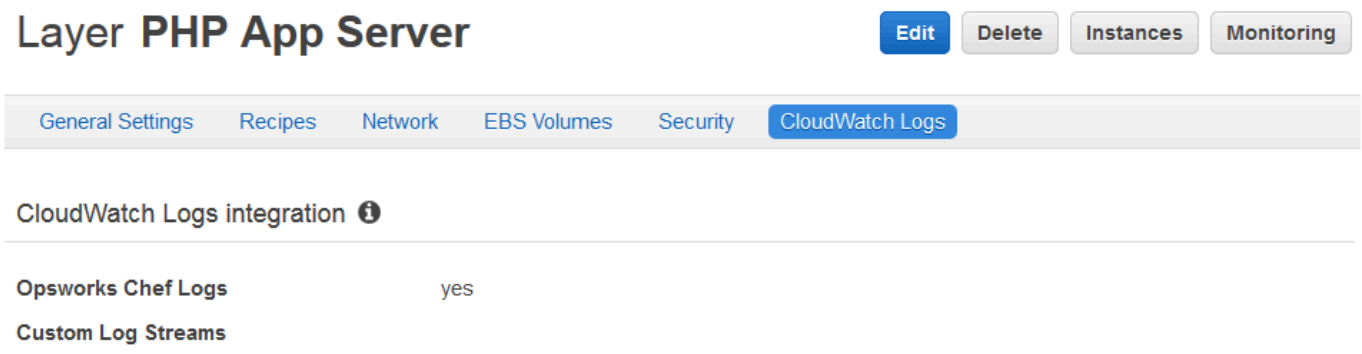
1. Depois que todas as atualizações necessárias do perfil do agente e da instância forem concluídas, você poderá ativar CloudWatch os registros definindo o controle deslizante na guia CloudWatch Registros como Ativado.

2. Para logs de comando de fluxo, alterne o botão de seleção Stream command logs para On. Isso envia registros das atividades do Chef e comandos iniciados pelo usuário nas instâncias da sua camada para o CloudWatch Logs.

   Os dados incluídos nesses registros são muito parecidos com os que você vê nos resultados de uma DescribeCommandsoperação quando você abre o destino da URL do registro. Isso inclui dados sobre `setup`, `configure`, `deploy`, `undeploy`, `start`, `stop` e comandos de execução de receita.

3. Para fazer o streaming de logs de atividades que são armazenados em um local personalizado nas instâncias da sua camada, como `/var/log/apache/myapp/mylog*`, digite o local personalizado na caixa de strings Stream custom logs e, em seguida, escolha Add (+).

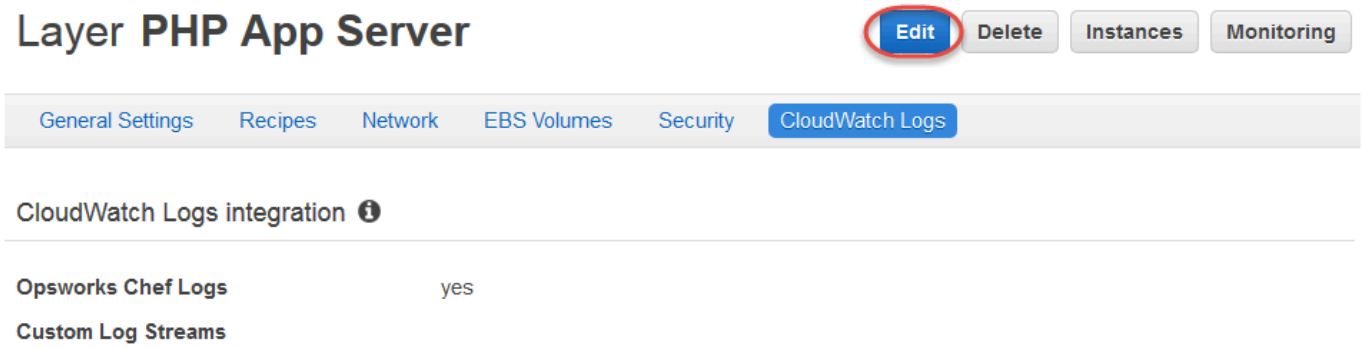
4. Escolha Salvar. Em alguns minutos, os streams de registros do AWS OpsWorks Stacks devem estar visíveis no console de CloudWatch registros.



Desativando CloudWatch registros

Para desativar CloudWatch os Registros, edite suas configurações de camada.

1. Na página de propriedades da camada, escolha Edit.



2. Na página de edição, escolha a guia CloudWatch Registros.
3. Na área CloudWatch Registros, desative os registros do comando Stream. Escolha X em logs personalizados para excluí-los dos fluxos de log, se aplicável.
4. Escolha Salvar.

Excluindo registros transmitidos dos registros CloudWatch

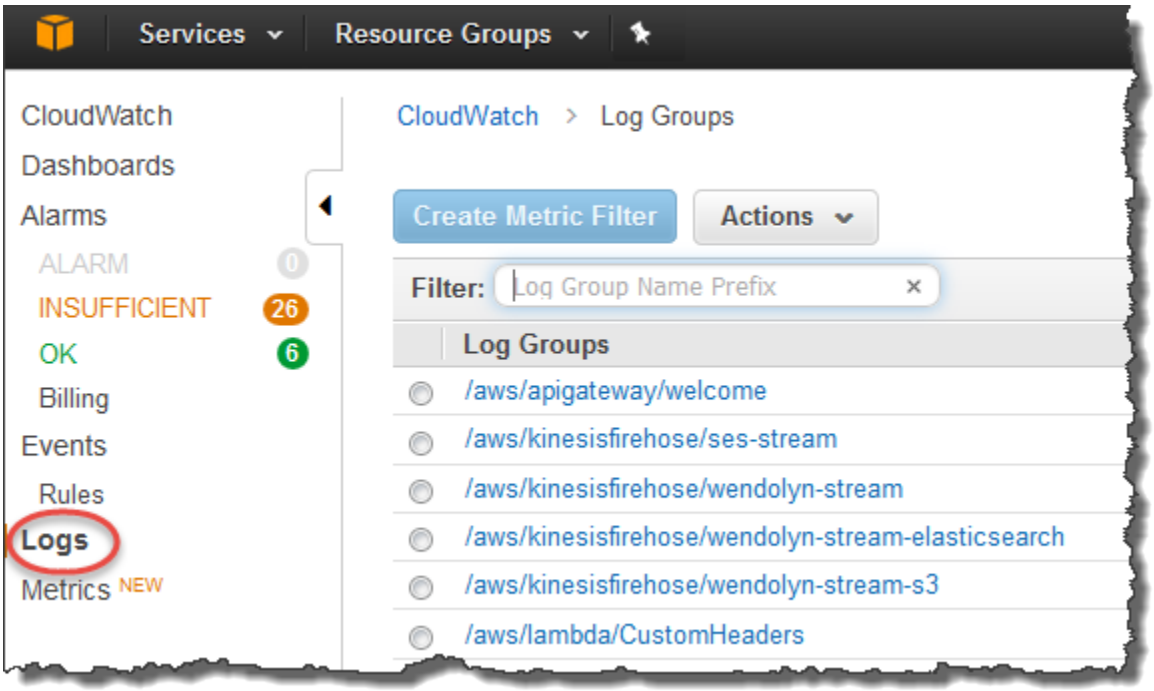
Depois de desativar o streaming de CloudWatch registros do AWS OpsWorks Stacks, os registros existentes ainda estarão disponíveis no console de gerenciamento de CloudWatch registros. Você ainda será cobrado pelos logs armazenados, a menos que os exporte para o Amazon S3 ou os exclua. Para obter mais informações sobre a exportação de logs para o S3, consulte Exportar dados de log para o Amazon S3.

Você pode excluir fluxos e grupos de CloudWatch registros no console de gerenciamento de registros ou executando os `delete-log-group` AWS CLI comandos `delete-log-stream`e. Para

obter mais informações sobre como alterar os períodos de retenção de registros, consulte Retenção de dados de registro de alterações em CloudWatch registros.

Gerenciando seus registros em CloudWatch registros

Os registros que você está transmitindo são gerenciados no console de CloudWatch registros.



AWS OpsWorks cria grupos de registros e fluxos de registros padrão automaticamente. Os grupos de log para dados do AWS OpsWorks Stacks têm nomes que correspondem ao padrão a seguir:

stack_name/*layer_name*/*chef_log_name*

Os logs personalizados têm nomes que correspondem ao padrão a seguir:

/stack_name/layer_short_name/file_path_name. O nome do caminho se torna mais legível por humanos com a remoção de caracteres especiais, como asteriscos (*).

Depois de localizar seus CloudWatch registros em Registros, você pode organizar os registros em grupos, pesquisar e filtrar registros criando filtros de métricas e criando alarmes personalizados.

Configurando camadas do Chef 12.2 Windows para usar registros CloudWatch

CloudWatch A integração automática de registros não é compatível com instâncias baseadas em Windows. A guia CloudWatch Registros não está disponível em camadas nas pilhas do Chef 12.2.

Para ativar manualmente o streaming para instâncias baseadas no CloudWatch Logs for Windows, faça o seguinte.

- Atualize o perfil da instância para instâncias baseadas no Windows para que o agente do CloudWatch Logs tenha as permissões apropriadas. A instrução da política AWSOpsWorksCloudWatchLogs mostra quais permissões são necessárias.

  Normalmente, você executa essa tarefa somente uma vez. Em seguida, você pode usar o perfil da instância atualizado para todas as instâncias do Windows em uma camada.
- Edite o seguinte arquivo de configuração JSON em cada instância. Esse arquivo inclui as preferências de streaming de log, como quais logs monitorar.

  ```
  %PROGRAMFILES%\Amazon\Ec2ConfigService\Settings
  \AWS.EC2.Windows.CloudWatch.json
  ```

Você pode automatizar as duas tarefas anteriores, criando receitas personalizadas para lidar com as tarefas necessárias e atribuindo-as aos eventos de Setup da camada do Chef 12.2. Cada vez que você inicia uma nova instância nessas camadas, o AWS OpsWorks Stacks executa automaticamente suas receitas depois que a instância termina de inicializar, ativando os Logs. CloudWatch

Para desativar CloudWatch os registros em instâncias baseadas no Windows, inverta o processo. Desmarque a caixa de seleção Ativar integração de CloudWatch registros na caixa de diálogo Propriedades do EC2 serviço, exclua as preferências de fluxo de log do `AWS.EC2.Windows.CloudWatch.json` arquivo e pare de executar qualquer receita do Chef que esteja atribuindo automaticamente permissões de CloudWatch registros a novas instâncias nas camadas do Chef 12.2.

Monitorando pilhas usando Amazon Events CloudWatch

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode configurar regras no Amazon CloudWatch Events para alertá-lo sobre mudanças nos recursos do AWS OpsWorks Stacks e direcionar os CloudWatch Eventos a realizar ações com base no conteúdo do evento. Para obter mais informações sobre como começar a usar CloudWatch eventos e configurar regras, consulte Introdução aos CloudWatch eventos no Guia do usuário de CloudWatch eventos.

Os seguintes tipos de eventos AWS OpsWorks Stacks são compatíveis com CloudWatch Eventos.

Alteração no estado de instância

Indica uma mudança no estado de uma instância do AWS OpsWorks Stacks.

Alteração no estado de comando

Indica que ocorreu uma alteração no estado de um comando AWS OpsWorks Stacks.

Alteração no estado de implantação

Indica que ocorreu uma alteração no estado de uma implantação do AWS OpsWorks Stacks.

Alertas

Indica que um erro de serviço do AWS OpsWorks Stacks foi gerado.

Para obter mais informações sobre os tipos de eventos do AWS OpsWorks Stacks que são compatíveis com os CloudWatch Eventos, consulte AWS OpsWorks Stacks Events no Guia do usuário do CloudWatch Events.

Segurança e permissões

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Cada um de seus usuários deve ter AWS as credenciais apropriadas para acessar os AWS recursos da sua conta. A forma recomendada de fornecer credenciais aos usuários é com AWS Identity and

Access Management(IAM). AWS OpsWorks O Stacks se integra ao IAM para permitir que você controle o seguinte:

- Como usuários individuais podem interagir com o AWS OpsWorks Stacks.

  Por exemplo, você pode permitir que alguns usuários implantem aplicativos em qualquer pilha, mas não modifiquem a pilha em si, e, ao mesmo tempo, conceder acesso total a outros usuários, mas apenas a determinadas pilhas, e assim por diante.
- Como AWS OpsWorks as pilhas podem agir em seu nome para acessar recursos da pilha, como EC2 instâncias da Amazon e buckets do Amazon S3.

  AWS OpsWorks O Stacks fornece uma função de serviço que concede permissões para essas tarefas.
- Como os aplicativos executados em EC2 instâncias da Amazon controladas pelo AWS OpsWorks Stacks podem acessar outros AWS recursos, como dados armazenados em buckets do Amazon S3.

  Você pode atribuir um perfil de instância às instâncias de uma camada que concede permissões aos aplicativos em execução nessas instâncias para acessar outros AWS recursos.
- Como gerenciar chaves SSH com base em usuários e usar o SSH ou o RDP para se conectar a instâncias.

  Para cada pilha, os usuários administrativos podem atribuir a cada usuário do uma chave SSH pessoal ou autorizar que usuários especifiquem a própria chave. Você também pode autorizar o acesso a SSH ou RDP e privilégios de administrador ou sudo nas instâncias da pilha para cada usuário.

Outros aspectos de segurança incluem o seguinte:

- Como gerenciar a atualização do sistema operacional das instâncias com os patches de segurança mais recentes.

  Para obter mais informações, consulte Gerenciamento de atualizações de segurança.
- Como configurar grupos de EC2 segurança da Amazon para controlar o tráfego de rede de e para suas instâncias.

  Como especificar grupos de segurança personalizados em vez dos grupos de segurança padrão do AWS OpsWorks Stacks. Para obter mais informações, consulte Usar grupos de segurança.

Tópicos

- Gerenciando permissões AWS OpsWorks de usuário do Stacks
- Permitindo que AWS OpsWorks as pilhas ajam em seu nome
- Prevenção confusa entre serviços em Stacks AWS OpsWorks
- Especificação de permissões para aplicativos executados em instâncias EC2
- Gerenciamento do acesso por SSH
- Gerenciamento de atualizações de segurança do Linux
- Usar grupos de segurança

Gerenciando permissões AWS OpsWorks de usuário do Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Como prática recomendada, restrinja os usuários do AWS OpsWorks Stacks a um conjunto específico de ações ou a um conjunto de recursos do stack. Você pode controlar as permissões de usuário do AWS OpsWorks Stacks de duas maneiras: usando a página de permissões do AWS OpsWorks Stacks e aplicando uma política apropriada do IAM.

A página OpsWorks Permissões, ou as ações equivalentes da CLI ou da API, permite que você controle as permissões do usuário em um ambiente multiusuário por pilha, atribuindo a cada usuário um dos vários níveis de permissão. Cada nível concede permissões para um conjunto padrão de ações para um determinado recurso de pilha. Usando página Permissions, você pode controlar o seguinte:

- Quem pode acessar cada pilha.
- Quais ações cada usuário tem permissão para executar em cada pilha.

  Por exemplo, você pode permitir que alguns usuários apenas visualizem a pilha, enquanto outros podem implantar aplicativos, adicionar instâncias e assim por diante.

- Quem pode gerenciar cada pilha.

  A gestão de cada pilha pode ser delegada para um ou mais usuários determinados.
- Quem tem acesso SSH em nível de usuário e privilégios sudo (Linux) ou acesso RDP e privilégios de administrador (Windows) nas instâncias da Amazon de cada pilha. EC2

  Você pode conceder ou remover essas permissões separadamente para cada usuário a qualquer momento.

> **Important**
>
> Negar o acesso SSH/RDP não necessariamente impede que um usuário faça login nas instâncias. Se você especificar um par de EC2 chaves da Amazon para uma instância, qualquer usuário com a chave privada correspondente poderá fazer login ou usar a chave para recuperar a senha de administrador do Windows. Para obter mais informações, consulte Gerenciamento do acesso por SSH.

Você pode usar a CLI, a API ou o console do IAM para adicionar políticas aos seus usuários que concedem permissões explícitas para os diversos recursos e ações do AWS OpsWorks Stacks.

- Usar uma política do IAM para especificar as permissões é mais flexível do que usar os níveis de permissões.
- Você pode configurar identidades do IAM (usuários, grupos de usuários e perfis), que concedem permissões às identidades do IAM, como usuários e grupos de usuários, ou definir perfis que podem ser associadas a usuários federados.
- Uma política do IAM é a única maneira de conceder permissões para determinadas ações-chave do AWS OpsWorks Stacks.

  Por exemplo, você deve usar o IAM para conceder permissões para `opsworks:CreateStack` e `opsworks:CloneStack`, que são usados para criar e clonar pilhas, respectivamente.

Embora não seja explicitamente possível importar usuários federados no console, um usuário federado pode criar implicitamente um perfil de usuário escolhendo Minhas configurações no canto superior direito do console AWS OpsWorks Stacks e, em seguida, escolhendo Usuários, também no canto superior direito. Na página Usuários, os usuários federados, cujas contas foram criadas

usando a API ou CLI, ou implicitamente por meio do console, podem gerenciar suas contas de maneira similar aos usuários não federados.

As duas abordagens não são mutuamente exclusivas e, às vezes, é útil combiná-las; o AWS OpsWorks Stacks, portanto, avalia os dois conjuntos de permissões. Por exemplo, suponha que você queira permitir aos usuários adicionar ou excluir instâncias, mas não adicionar ou excluir camadas. Nenhum dos níveis de permissão do AWS OpsWorks Stacks concede esse conjunto específico de permissões. No entanto, você pode usar a página Permissões para conceder aos usuários um nível de permissão Gerenciar, o que lhes permite executar a maioria das operações de pilha e, em seguida, aplicar uma política do IAM que nega permissões para adicionar ou remover camadas. Para obter mais informações, consulte Controlar o acesso a recursos da AWS usando políticas.

O seguinte é um modelo típico para o gerenciamento de permissões de usuários. Em cada caso, o leitor (você) é considerado um usuário administrativo.

1. Use o console do IAM para aplicar AWSOpsWorks_FullAccess políticas a um ou mais usuários administrativos.
2. Crie um usuário do para cada usuário não administrativo com uma política que não garanta nenhuma permissão do AWS OpsWorks Stacks.

   Se um usuário precisar acessar somente AWS OpsWorks as pilhas, talvez você nem precise aplicar uma política. Em vez disso, você pode gerenciar suas permissões com a página Permissões de AWS OpsWorks pilhas.
3. Use a página AWS OpsWorks Stacks Users para importar os usuários não administrativos para AWS OpsWorks o Stacks.
4. Para cada pilha, use a página Permissions da pilha para atribuir um nível de permissão a cada usuário.
5. Conforme necessário, personalize os níveis de permissão dos usuários aplicando uma política do IAM adequadamente configurada.

Para obter mais recomendações sobre o gerenciamento de usuários, consulte Melhores práticas: Gerenciamento de permissões.

Para obter mais informações sobre as práticas recomendadas do IAM, consulte Práticas recomendadas de segurança no IAM no Guia do usuário do IAM.

Tópicos

- Gerenciando AWS OpsWorks usuários do Stacks
- Concedendo permissões por AWS OpsWorks pilha aos usuários do Stacks
- Gerenciando permissões de AWS OpsWorks pilhas anexando uma política do IAM
- Exemplo de políticas
- AWS OpsWorks Níveis de permissões de pilhas

Gerenciando AWS OpsWorks usuários do Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Antes de importar usuários para o AWS OpsWorks Stacks e conceder permissões a eles, primeiro você precisa criar um usuário para cada indivíduo. Para criar usuários do IAM, comece fazendo login AWS como um usuário que recebeu as permissões definidas na política de IAMFull acesso. Em seguida, você usa o console do IAM para criar usuários do IAM para todos que precisam acessar o AWS OpsWorks Stacks. Em seguida, você pode importar esses usuários para o AWS OpsWorks Stacks e conceder permissões de usuário da seguinte forma:

Usuários regulares AWS OpsWorks do Stacks

Os usuários regulares não exigem uma política anexada. Se eles tiverem uma, ela normalmente não inclui nenhuma permissão do AWS OpsWorks Stacks. Em vez disso, use a página Permissões de AWS OpsWorks pilhas para atribuir um dos seguintes níveis de permissão a usuários comuns em uma stack-by-stack base.

- As permissões Show permitem que os usuários visualizem a pilha, mas não executem nenhuma operação.
- As permissões Deploy incluem as permissões Show e também permitem que os usuários implementem e atualizem aplicativos.
- As permissões de gerenciamento incluem as permissões de implantação e também permitem que os usuários realizem operações de gerenciamento de pilhas, como adicionar camadas

ou instâncias, usar a página Permissões para definir permissões de usuário e habilitar seus próprios SSH/RDP and sudo/admin privilégios.

- As permissões Deny negam acesso à pilha.

Se esses níveis de permissões não forem exatamente o que você deseja para um determinado usuário, você pode personalizar as permissões do usuário anexando uma política do IAM. Por exemplo, talvez você queira usar a página Permissões de AWS OpsWorks pilhas para atribuir o nível Gerenciar permissões a um usuário, o que lhe concede permissões para realizar todas as operações de gerenciamento de pilhas, mas não para criar ou clonar pilhas. Em seguida, você poderia aplicar uma política que restringe essas permissões, negando a ele a permissão para adicionar ou excluir camadas, ou que aumenta essas permissões, permitindo que ele crie ou clone pilhas. Para obter mais informações, consulte Gerenciando permissões de AWS OpsWorks pilhas anexando uma política do IAM.

AWS OpsWorks Usuários administrativos do Stacks

Os usuários administrativos são o proprietário da conta ou um usuário do IAM com as permissões definidas pela AWSOpsWorks_FullAccess política. Além das permissões concedidas a usuários Manage, essa política inclui permissões para ações que não podem ser concedidas por meio da página Permissions, tais como o seguinte:

- Importação de usuários para o Stacks AWS OpsWorks
- Criação e clonagem de pilhas

Para a política completa, consulte Exemplo de políticas. Para obter uma lista detalhada de permissões que podem ser concedidas a usuários apenas aplicando uma política do IAM, consulte AWS OpsWorks Níveis de permissões de pilhas.

Tópicos

- Usuários e regiões
- Criando um usuário administrativo do AWS OpsWorks Stacks
- Criação de usuários do IAM para AWS OpsWorks Stacks
- Importação de usuários para pilhas AWS OpsWorks
- Editando configurações AWS OpsWorks do usuário do Stacks

Usuários e regiões

AWS OpsWorks Os usuários do Stacks estão disponíveis no endpoint regional em que foram criados. Você pode criar usuários em qualquer uma das seguintes regiões.

- Região Leste dos EUA (Ohio)
- Região Leste dos EUA (Norte da Virgínia)
- Região Oeste dos EUA (Oregon)
- Região Oeste dos EUA (Norte da Califórnia).
- Região do Canadá (Central) (somente API); não disponível no AWS Management Console
- Região Ásia-Pacífico (Mumbai)
- Região Ásia-Pacífico (Singapura)
- Região Ásia-Pacífico (Sydney)
- Região Ásia-Pacífico (Tóquio)
- Região Ásia-Pacífico (Seul)
- Região Europa (Frankfurt)
- Região Europa (Irlanda)
- Região Europa (Londres)
- Região Europa (Paris)
- Região América do Sul (São Paulo)

Ao importar usuários para o AWS OpsWorks Stacks, você os importa para um dos endpoints regionais; se quiser que um usuário esteja disponível em mais de uma região, você deve importar o usuário para essa região. Você também pode importar usuários do AWS OpsWorks Stacks de uma região para outra; se você importar um usuário para uma região que já tenha um usuário com o mesmo nome, o usuário importado substituirá o usuário existente. Para obter mais informações sobre a importação de usuários, consulte Importação de usuários.

Criando um usuário administrativo do AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode criar um usuário administrativo do AWS OpsWorks Stacks adicionando a `AWSOpsWorks_FullAccess` política a um usuário, o que concede a esse usuário permissões de acesso total ao AWS OpsWorks Stacks. Para obter mais informações sobre a criação de um usuário administrativo, consulte Criar um usuário administrativo.

> **Note**
>
> A AWSOpsWorks_FullAccess política permite que os usuários criem e gerenciem AWS OpsWorks pilhas de pilhas, mas os usuários não podem criar uma função de serviço do IAM para a pilha; eles devem usar uma função existente. O primeiro usuário a criar uma pilha deve ter permissões do IAM adicionais, conforme descrito em Permissões administrativas. Quando esse usuário cria a primeira pilha, o AWS OpsWorks Stacks cria uma função de serviço do IAM com as permissões necessárias. Portanto, qualquer usuário com permissões `opsworks:CreateStack` pode usar essa função para criar pilhas adicionais. Para obter mais informações, consulte Permitindo que AWS OpsWorks as pilhas ajam em seu nome.

Ao criar um usuário, você pode adicionar outras políticas gerenciadas pelo cliente para ajustar as permissões do usuário, conforme necessário. Por exemplo, você pode querer que um usuário administrativo seja capaz de criar ou excluir pilhas, mas não importar novos usuários. Para obter mais informações, consulte Gerenciando permissões de AWS OpsWorks pilhas anexando uma política do IAM.

Se você tiver vários usuários administrativos, em vez de definir permissões separadamente para cada usuário, poderá adicionar a AWSOpsWorks_FullAccess política a um grupo do IAM e adicionar os usuários a esse grupo.

Para obter informações sobre a criação de um grupo, consulte Criando grupos de usuários no IAM. Ao criar o grupo, adicione a AWSOpsWorks_FullAccesspolítica. Você também pode adicionar a AdministratorAccesspolítica, que inclui as AWSOpsWorks_FullAccesspermissões.

Para obter informações sobre como adicionar permissões a um grupo existente, consulte Anexar uma política a um grupo de usuários do IAM.

Criação de usuários do IAM para AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Antes de importar usuários do IAM para o AWS OpsWorks Stacks, você precisa criá-los. Você pode fazer isso usando o console do IAM, a linha de comando ou a API. Para obter instruções completas, consulte Como criar um usuário do IAM em sua AWS conta.

Observe que, ao contrário dos usuários administrativos, você não precisa anexar uma política para definir permissões. Você pode definir permissões depois de importar os usuários para o AWS OpsWorks Stacks, conforme explicado em Gerenciamento de permissões de usuário.

Para obter mais informações sobre a criação de usuários e grupos do IAM, consulte Conceitos básicos do IAM.

Importação de usuários para pilhas AWS OpsWorks

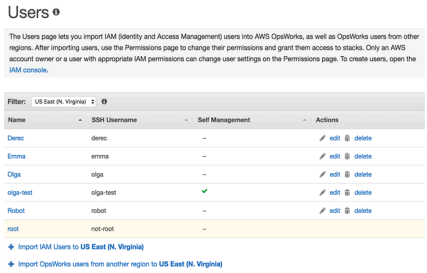
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Usuários administrativos podem importar usuários para o AWS OpsWorks Stacks; eles também podem importar usuários do AWS OpsWorks Stacks de um endpoint regional para outro. Ao importar usuários para o AWS OpsWorks Stacks, você os importa para um dos endpoints regionais do AWS OpsWorks Stacks. Se quiser que um usuário do fique disponível em mais de uma região, você deverá importar o usuário para essa região.

Embora não seja explicitamente possível importar usuários federados no console, um usuário federado pode criar implicitamente um perfil de usuário escolhendo Minhas configurações no canto superior direito do console AWS OpsWorks Stacks e, em seguida, escolhendo Usuários, também no canto superior direito. Na página Usuários, os usuários federados, cujas contas foram criadas usando a API ou CLI, ou implicitamente por meio do console, podem gerenciar suas contas de maneira similar aos usuários não federados.

Para importar usuários para o AWS OpsWorks Stacks

1. Faça login no AWS OpsWorks Stacks como usuário administrativo ou como proprietário da conta.
2. Selecione Users no canto superior direito para abrir a página Users.



3. Escolha Importar usuários do IAM para < ***region name*** > para exibir os usuários que estão disponíveis, mas que ainda não foram importados.

4. Preencha a caixa de seleção Select all ou selecione um ou mais usuários individuais. Ao terminar, escolha Importar para OpsWorks.

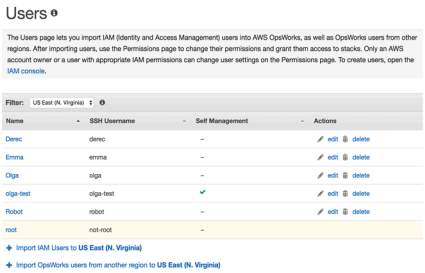
> **Note**
>
> Depois de importar um usuário para o AWS OpsWorks Stacks, se você usar o console do IAM ou a API para excluir o usuário da sua conta, o usuário não perderá automaticamente o acesso SSH que você concedeu por meio do AWS OpsWorks Stacks. Você também deve excluir o usuário do AWS OpsWorks Stacks abrindo a página Usuários e escolhendo excluir na coluna Ações do usuário.

Para importar usuários do AWS OpsWorks Stacks de uma região para outra

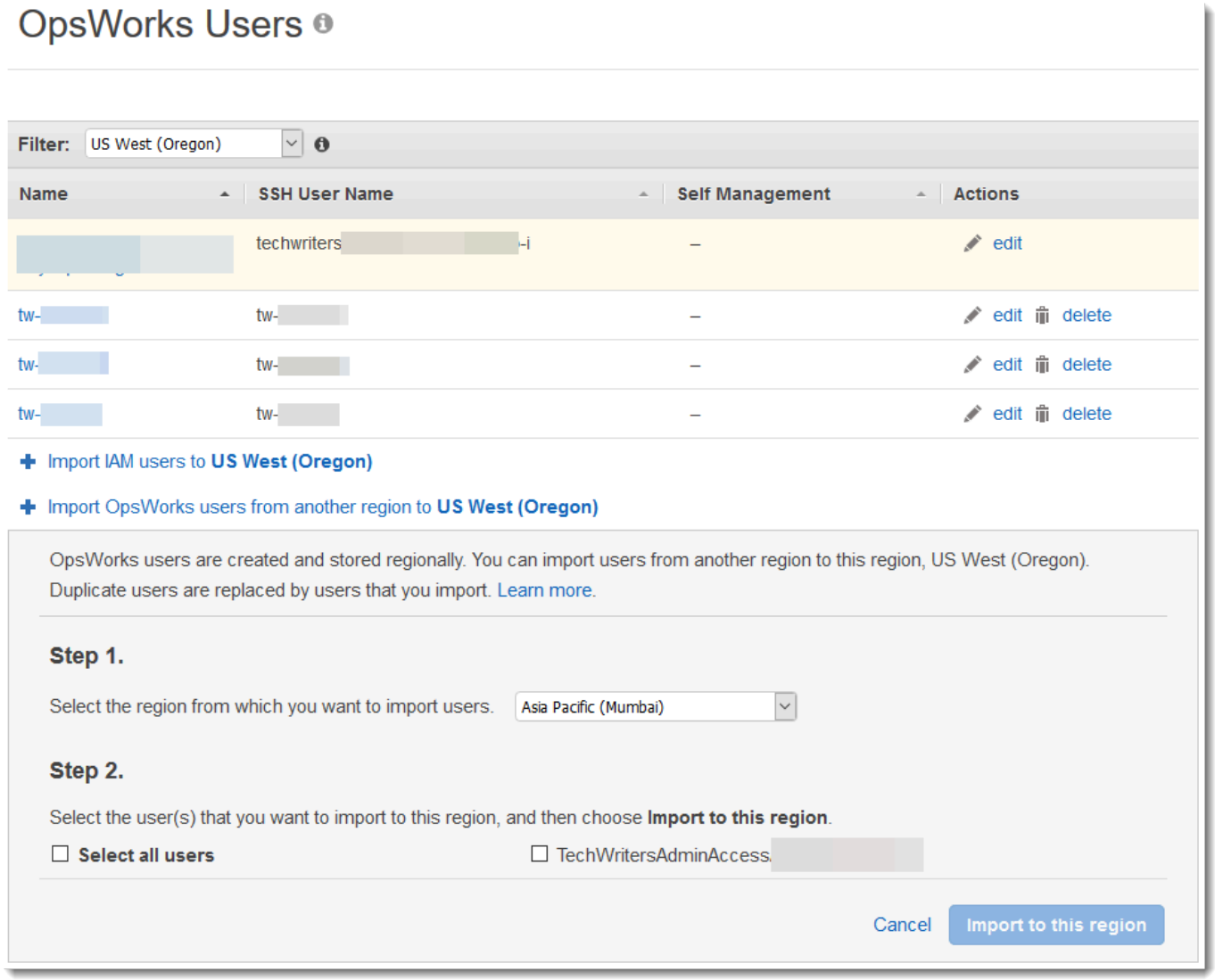
AWS OpsWorks Os usuários do Stacks estão disponíveis no endpoint regional em que foram criados. Você pode criar usuários nas regiões mostradas em Usuários e regiões.

Você pode importar usuários do AWS OpsWorks Stacks de uma região para a região para a qual sua lista de usuários está atualmente filtrada. Se você importar um usuário para uma região que já tem um usuário com o mesmo nome, o usuário importado substituirá o usuário existente.

1. Faça login no AWS OpsWorks Stacks como usuário administrativo ou como proprietário da conta.
2. Selecione Users no canto superior direito para abrir a página Users. Se você tiver usuários do AWS OpsWorks Stacks em mais de uma região, use o controle Filtro para filtrar a região para a qual você deseja importar usuários.



3. Escolha Importar usuários do AWS OpsWorks Stacks de outra região para **<** ***current region*** **>**.



4. Selecione a região da qual você deseja importar usuários do AWS OpsWorks Stacks.
5. Selecione um ou mais usuários para importar ou selecione todos os usuários e, em seguida, escolha Import to this region. Aguarde até que AWS OpsWorks as pilhas exibam os usuários importados na lista de usuários.

Unix IDs e usuários criaram pilhas externas AWS OpsWorks

AWS OpsWorks atribui aos usuários em instâncias do AWS OpsWorks Stacks valores de Unix ID (UID) entre 2000 e 4000. Como AWS OpsWorks reserva a faixa de 2.000 a 4.000 UIDs, os usuários que você cria fora AWS OpsWorks (usando receitas de livros de receitas ou importando usuários AWS OpsWorks do IAM, por exemplo) podem fazer com UIDs que sejam substituídos pelo AWS OpsWorks Stacks para outro usuário. Isso pode fazer com que os usuários que você criou fora do

AWS OpsWorks Stacks não apareçam nos resultados da pesquisa do data bag ou sejam excluídos da operação integrada `sync_remote_users` do AWS OpsWorks Stacks.

Os processos externos também podem criar usuários com os quais o UIDs AWS OpsWorks Stacks pode sobrescrever. Alguns pacotes de sistemas operacionais, por exemplo, podem criar um usuário como parte dos processos pós-instalação. Quando você ou um processo de software cria um usuário em um sistema operacional baseado em Linux sem especificar explicitamente um UID, que é o padrão, o UID atribuído pelo Stacks é + 1. AWS OpsWorks *<highest existing AWS OpsWorks UID>*

Como prática recomendada, crie usuários do AWS OpsWorks Stacks e gerencie seus acessos no console do AWS OpsWorks Stacks ou usando um AWS SDK. AWS CLI Se você criar usuários em instâncias do AWS OpsWorks Stacks fora da AWS OpsWorks, use *UnixID* valores maiores que 4000.

Editando configurações AWS OpsWorks do usuário do Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Depois de importar os usuários, você pode editar as configurações conforme o seguinte:

Para editar as configurações de usuário

1. Na página Users (Usuários), escolha edit (editar) na coluna Actions (Ações) do usuário.
2. Você pode especificar as seguintes configurações.

   Autogerenciamento

   Selecione Sim para permitir que o usuário use a MySettings página para especificar sua chave SSH pessoal.

> **Note**
>
> Você também pode ativar o autogerenciamento adicionando a política do IAM à identidade do IAM que concede permissões para as UpdateMyUserProfileações DescribeMyUserProfilee.

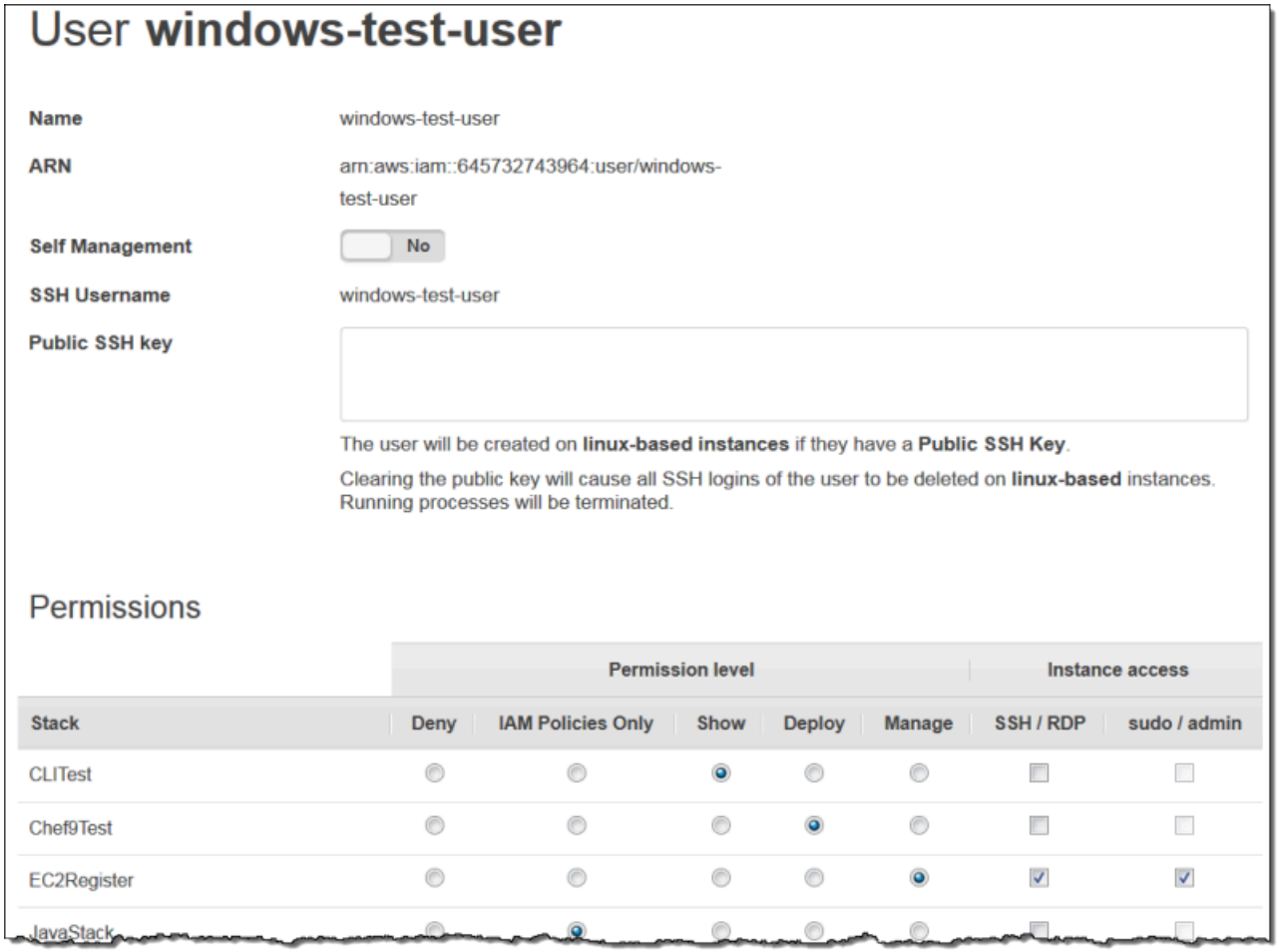
Chave SSH pública

(Opcional) Insira uma chave SSH pública para o usuário. Essa chave aparecerá na página My Settings do usuário. Se você permitir o autogerenciamento, o usuário poderá editar My Settings e especificar sua própria chave. Para obter mais informações, consulte Registro de uma chave SSH pública de um usuário .

AWS OpsWorks O Stacks instala essa chave em todas as instâncias do Linux; os usuários podem usar a chave privada associada para fazer login. Para obter mais informações, consulte Login com SSH. Você não pode usar essa chave com pilhas do Windows.

Permissões

(Opcional) Defina os níveis de permissões do usuário para cada pilha em um único lugar, em vez de configurá-los separadamente usando a página Permissions de cada pilha. Para obter mais informações sobre os níveis de permissões, consulte Concessão de permissões por pilha.



Concedendo permissões por AWS OpsWorks pilha aos usuários do Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A maneira mais simples de gerenciar as permissões de usuário do AWS OpsWorks Stacks é usando a página de Permissões de uma pilha. Cada pilha tem sua própria página, que concede permissões para esse pilha.

Você deve estar conectado como usuário administrativo ou usuário Manage para modificar qualquer uma das configurações de permissões. A lista mostra somente os usuários que foram importados para o AWS OpsWorks Stacks. Para obter informações sobre como criar e importar usuários, consulte Gerenciamento de usuários.

O nível de permissão padrão é Apenas políticas do IAM, que concede aos usuários apenas as permissões que estão em sua política do IAM.

- Quando você importa um usuário do IAM ou de outra região, o usuário é adicionado à lista para todas as pilhas existentes com um nível de permissão Somente políticas do IAM.
- Por padrão, um usuário que você acabou de importar de outra região não tem acesso a pilhas na região de destino. Se você importar usuários de outra região, para permitir que eles gerenciem pilhas na região de destino, deve atribuir a eles permissões para essas pilhas depois de importá-las.
- Quando você cria uma nova pilha, todos os usuários atuais são adicionados à lista com níveis de permissão IAM Policies Only.

Tópicos

- Configuração das permissões de um usuário
- Visualização das suas permissões
- Uso das chaves de condição do IAM para verificar credenciais temporárias

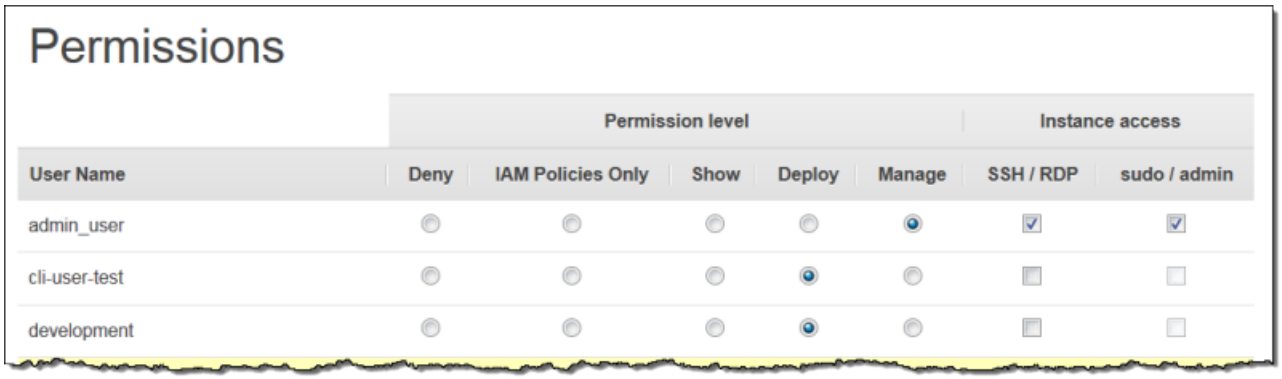
Configuração das permissões de um usuário

Para definir as permissões de um usuário

1. No painel de navegação, selecione Permissions (Permissões).
2. Na página Permissions (Permissões), escolha Edit (Editar).
3. Altere as configurações de Permission level (Nível de permissão) e Instance access (Acesso à instância):
    - Use as configurações de Permissions level para atribuir um dos níveis de permissão padrão para cada usuário, o que determina se o usuário pode acessar essa pilha e quais ações ela pode realizar. Se um usuário tiver uma política de IAM, o AWS OpsWorks Stacks avalia os dois conjuntos de permissões. Para ver um exemplo, consulte Exemplo de políticas.

- A configuração de Instance access SSH/RDP especifica se o usuário tem acesso SSH (Linux) ou RDP (Windows) às instâncias da pilha.

  Se você autorizar o acesso SSH/RDP, terá a opção de selecionar sudo/admin, que concede ao usuário privilégios de sudo (Linux) ou administrativos (Windows) nas instâncias da pilha.



Você pode atribuir cada usuário a um dos seguintes níveis de permissões. Para obter uma lista das ações que são permitidas por cada nível, consulte AWS OpsWorks Níveis de permissões de pilhas.

Negar

O usuário não pode realizar nenhuma ação do AWS OpsWorks Stacks na pilha, mesmo que tenha uma política do IAM que conceda às AWS OpsWorks Stacks permissões de acesso total. Você pode usar isso, por exemplo, para negar que alguns usuários acessem pilhas de produtos não lançados.

IAM Policies Only

O nível padrão, que é atribuído a todos os usuários recém-importados e a todos os usuários para pilhas recém-criadas. As permissões de usuário são determinadas pela política do IAM. Se um usuário não tiver uma política de IAM ou se sua política não tiver permissões explícitas de AWS OpsWorks pilhas, ele não poderá acessar a pilha. Os usuários administrativos costumam ser atribuídos a esse nível, pois suas políticas do IAM já concedem permissões de acesso total.

Show (Mostrar)

O usuário pode visualizar uma pilha, mas não pode executar nenhuma operação. Por exemplo, os gerentes podem querer monitorar as pilhas de uma conta, mas não precisariam implantar aplicativos ou modificar a pilha.

Implante

Inclui as permissões Show e também permite que o usuário implemente aplicativos. Por exemplo, um desenvolvedor de aplicativo talvez possa precisar implantar atualizações nas instâncias da pilha, mas não adicionar camadas ou instâncias à pilha.

Gerencie

Inclui as permissões Deploy e também permite que o usuário execute uma variedade de operações de gerenciamento de pilha, incluindo:

- Adição ou exclusão de camadas e instâncias.
- Uso da página Permissions da pilha para atribuir níveis de permissões a usuários.
- Registro ou cancelamento do registro de recursos.

Por exemplo, cada pilha pode ter um gerente que é responsável por assegurar que a pilha tenha um número e um tipo apropriado de instâncias, lidar com atualizações de pacote e sistema operacional, e assim por diante.

> **Note**
>
> O nível Manage não permite que os usuários criem ou clonem pilhas. Essas permissões devem ser concedidas por uma política do IAM. Para obter um exemplo, consulte Gerencie permissões.

Se o usuário também tiver uma política de IAM, o AWS OpsWorks Stacks avalia os dois conjuntos de permissões. Isso permite que você atribua um nível de permissão a um usuário e, em seguida, aplique uma política para restringir ou aumentar as ações permitidas do nível. Por exemplo, você poderia aplicar uma política que permite a um usuário Gerenciar criar ou clonar pilhas ou que nega a esse usuário a capacidade de registrar ou cancelar o registro de recursos. Para ver alguns exemplos de tais políticas, consulte Exemplo de políticas.

> **Note**
>
> Se a política do usuário permitir ações adicionais, o resultado poderá aparentar anular as configurações da página Permissions. Por exemplo, se um usuário tem uma política que permite a CreateLayeração, mas você usa a página Permissões para especificar permissões de implantação, o usuário ainda tem permissão para criar camadas. A exceção

a essa regra é a opção Negar, que nega o acesso à pilha até mesmo para usuários com AWSOpsWorks_FullAccess políticas. Para obter mais informações, consulte Controle do acesso aos AWS recursos usando políticas.

Visualização das suas permissões

Se o autogerenciamento estiver ativado, os usuários poderão ver um resumo dos seus níveis de permissão para cada pilha, escolhendo My Settings no canto superior direito. Os usuários também podem acessar Minhas configurações se suas políticas concederem permissões para as UpdateMyUserProfileações DescribeMyUserProfilee.

Uso das chaves de condição do IAM para verificar credenciais temporárias

AWS OpsWorks O Stacks tem uma camada de autorização integrada que oferece suporte a casos de autorização adicionais (como o gerenciamento simplificado do acesso somente de leitura ou leitura e gravação às pilhas para usuários individuais). Essa camada de autorização se baseia no uso de credenciais temporárias. Consequentemente, você não pode usar uma condição `aws:TokenIssueTime` para verificar se os usuários estão usando credenciais de longo prazo ou bloquear ações de usuários que estão usando credenciais temporárias, como descrito em referência a elementos da política JSON do IAM na documentação do IAM.

Gerenciando permissões de AWS OpsWorks pilhas anexando uma política do IAM

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode especificar as permissões do AWS OpsWorks Stacks de um usuário anexando uma política do IAM. Uma política anexada é necessária para algumas permissões:

- Permissões de usuário administrativas, como a importação de usuários.
- Permissões para algumas ações, como a criação ou a clonagem de uma pilha.

Para obter uma lista completa de ações que exigem uma política anexada, consulte AWS OpsWorks Níveis de permissões de pilhas.

Você também pode usar uma política para personalizar os níveis de permissão que foram concedidos por meio da página Permissões. Esta seção fornece um breve resumo de como aplicar uma política do IAM a um usuário para especificar as permissões do AWS OpsWorks Stacks. Para obter mais informações, consulte Gerenciamento de acesso para AWS recursos.

Uma política do IAM é um objeto JSON que contém uma ou mais instruções. Cada elemento de instrução tem uma lista de permissões que tem três elementos básicos próprios:

Ação

As ações que a permissão afeta. Você especifica ações AWS OpsWorks de pilhas comoopsworks:*action*. Uma `Action` pode ser definida para uma ação específica, como `opsworks:CreateStack`, que especifica se o usuário tem permissão para chamar `CreateStack`. Você também pode usar asteriscos para especificar grupos de ações. Por exemplo, `opsworks:Create*` especifica todas as ações de criação. Para ver uma lista completa das ações do AWS OpsWorks Stacks, consulte a Referência da AWS OpsWorks Stacks API.

Efeito

Se as ações especificadas são permitidas ou negadas.

Recurso

Os AWS recursos que a permissão afeta. AWS OpsWorks As pilhas têm um tipo de recurso, a pilha. Para especificar permissões para um determinado recurso de pilha, defina `Resource` como o Nome de região da Amazon (ARN), que tem o seguinte formato: `arn:aws:opsworks:`*region*`:`*account_id*`:stack/`*stack_id*`/`.

Você também pode usar caracteres curinga. Por exemplo, configurar `Resource` como `*` concede permissões para todos os recurso.

Por exemplo, a seguinte política nega ao usuário a capacidade de interromper instâncias na pilha cujo ID é `2860-2f18b4cb-4de5-4429-a149-ff7da9f0d8ee`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Action": "opsworks:StopInstance",
      "Effect": "Deny",
      "Resource": "arn:aws:opsworks:*:*:stack/2f18b4cb-4de5-4429-a149-ff7da9f0d8ee/"
    }
  ]
}
```

Para obter informações sobre como adicionar permissões a um usuário do IAM, consulte https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users_change-permissions.html#users_change_permissions-add-console.

Para obter mais informações sobre como criar ou modificar políticas do IAM, consulte Políticas e permissões no IAM. Para ver alguns exemplos de políticas do AWS OpsWorks Stacks, consulteExemplo de políticas.

Exemplo de políticas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve exemplos de políticas do IAM que podem ser aplicadas aos usuários do AWS OpsWorks Stacks.

- Permissões administrativas descreve duas políticas que podem ser usadas para conceder permissões a usuários administrativos.
- Gerencie permissões e Permissões Deploy mostram exemplos de políticas que podem ser aplicadas a um usuário para aumentar ou restringir os níveis de permissões Manage e Deploy.

  AWS OpsWorks O Stacks determina as permissões do usuário avaliando as permissões concedidas pelas políticas do IAM, bem como as permissões concedidas pela página de Permissões. Para obter mais informações, consulte Controle do acesso aos AWS recursos usando políticas. Para obter mais informações sobre as permissões da página Permissions, consulte AWS OpsWorks Níveis de permissões de pilhas.

Permissões administrativas

Use o console do IAM, https://console.aws.amazon.com/iam/, para acessar a AWSOpsWorks_FullAccess política. Anexe essa política a um usuário para conceder a ele permissões para realizar todas as ações do AWS OpsWorks Stacks. As permissões do IAM são necessárias, entre outras coisas, para permitir que um usuário administrativo importe usuários.

Você deve criar uma função do IAM que permita que o AWS OpsWorks Stacks atue em seu nome para acessar outros AWS recursos, como EC2 instâncias da Amazon. Normalmente, você realiza essa tarefa fazendo com que um usuário administrativo crie a primeira pilha e deixe que AWS OpsWorks as pilhas criem a função para você. Em seguida, você pode usar essa função para todos os pilhas subsequentes. Para obter mais informações, consulte Permitindo que AWS OpsWorks as pilhas ajam em seu nome.

O usuário administrativo que cria a primeira pilha deve ter permissões para algumas ações do IAM que não estão incluídas na AWSOpsWorks_FullAccess política. Adicione as seguintes permissões à seção `Actions` da política. Para obter uma sintaxe JSON adequada, adicione vírgulas entre as ações e remova a vírgula final no final da lista de ações.

```
"iam:PutRolePolicy",
"iam:AddRoleToInstanceProfile",
"iam:CreateInstanceProfile",
"iam:CreateRole"
```

Gerencie permissões

O nível de permissões Manage permite que um usuário execute uma variedade de ações de gerenciamento de pilha, incluindo adição e exclusão de camadas. Este tópico descreve várias políticas que você pode usar para Gerenciar usuários para aumentar ou restringir as permissões padrão.

Negar a um usuário Manage a capacidade de adicionar ou excluir camadas

Você pode restringir os níveis de permissão Gerenciar para permitir que um usuário execute todas as ações Gerenciar, exceto adicionar ou excluir camadas, usando a seguinte política do IAM. Substitua *`regionaccount_id`*,, e *`stack_id`* por valores apropriados à sua configuração.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "opsworks:CreateLayer",
        "opsworks:DeleteLayer"
      ],
      "Resource": "arn:aws:opsworks:region:account_id:stack/stack_id/"
    }
  ]
}
```

Permitir que um usuário Manage crie ou clone pilhas

O nível de permissões Gerenciar não permite que os usuários criem ou clonem pilhas. Você pode mudar as permissões Gerenciar para permitir que um usuário crie ou clone pilhas aplicando as seguintes políticas do IAM. *account_id*Substitua *region* e por valores apropriados à sua configuração.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetRolePolicy",
        "iam:ListRoles",
        "iam:ListInstanceProfiles",
        "iam:ListUsers",
        "opsworks:DescribeUserProfiles",
        "opsworks:CreateUserProfile",
        "opsworks:DeleteUserProfile"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": "arn:aws:opsworks::account_id:stack/*/",
      "Condition": {
        "StringEquals": {
```

```
          "iam:PassedToService": "opsworks.amazonaws.com"
        }
      }
    }
  ]
}
```

Negar a um usuário Manage a capacidade de registrar ou cancelar o registro de recursos

O nível de permissões Gerenciar permite que o usuário registre e cancele o registro do Amazon EBS e dos recursos de endereço IP elástico com a pilha. Você pode restringir as permissões Gerenciar para permitir que o usuário execute todas as ações Gerenciar, exceto registrar os recursos, aplicando a política a seguir.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "opsworks:RegisterVolume",
        "opsworks:RegisterElasticIp"
      ],
      "Resource": "*"
    }
  ]
}
```

Permitir que um usuário Manage importe usuários

O nível Gerenciar permissões não permite que os usuários importem usuários para o AWS OpsWorks Stacks. Você pode aumentar as permissões Gerenciar para permitir que um usuário importe e exclua usuários aplicando a seguinte política do IAM. *account_id*Substitua *region* e por valores apropriados à sua configuração.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iam:GetRolePolicy",
```

```
        "iam:ListRoles",
        "iam:ListInstanceProfiles",
        "iam:ListUsers",
        "iam:PassRole",
        "opsworks:DescribeUserProfiles",
        "opsworks:CreateUserProfile",
        "opsworks:DeleteUserProfile"
      ],
      "Resource": "arn:aws:iam:region:account_id:user/*",
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": "opsworks.amazonaws.com"
        }
      }
    }
  ]
}
```

Permissões Deploy

O nível de permissões Deploy não permite que os usuários criem ou excluam aplicativos. Você pode aumentar as permissões Implantar para permitir que um usuário crie e exclua aplicativos aplicando a seguinte política do IAM. Substitua *regionaccount_id*,, e *stack_id* por valores apropriados à sua configuração.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "opsworks:CreateApp",
        "opsworks:DeleteApp"
      ],
      "Resource": "arn:aws:opsworks:region:account_id:stack/stack_id/"
    }
  ]
}
```

AWS OpsWorks Níveis de permissões de pilhas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção lista as ações permitidas pelos níveis de permissões Mostrar, Implantar e Gerenciar na página AWS OpsWorks Stacks Permissions. Ela também inclui uma lista de ações às quais você pode conceder permissões apenas aplicando uma política do IAM ao usuário.

Show (Mostrar)

O nível Show permite comandos `Describe`*XYZ*, com as seguintes exceções:

```
DescribePermissions
DescribeUserProfiles
DescribeMyUserProfile
DescribeStackProvisioningParameters
```

Se um usuário administrativo permitiu o autogerenciamento para o usuário, os usuários Show também podem usar `DescribeMyUserProfile` e `UpdateMyUserProfile`. Para obter mais informações sobre o autogerenciamento, consulte Edição das configurações de usuário.

Implante

As ações a seguir são permitidas pelo nível Deploy, além das ações permitidas pelo nível Show.

```
CreateDeployment
UpdateApp
```

Gerencie

As ações a seguir são permitidas pelo nível Manage, além das ações permitidas pelos níveis Deploy e Show.

```
AssignInstance
```

```
AssignVolume
AssociateElasticIp
AttachElasticLoadBalancer
CreateApp
CreateInstance
CreateLayer
DeleteApp
DeleteInstance
DeleteLayer
DeleteStack
DeregisterElasticIp
DeregisterInstance
DeregisterRdsDbInstance
DeregisterVolume
DescribePermissions
DetachElasticLoadBalancer
DisassociateElasticIp
GrantAccess
GetHostnameSuggestion
RebootInstance
RegisterElasticIp
RegisterInstance
RegisterRdsDbInstance
RegisterVolume
SetLoadBasedAutoScaling
SetPermission
SetTimeBasedAutoScaling
StartInstance
StartStack
StopInstance
StopStack
UnassignVolume
UpdateElasticIp
UpdateInstance
UpdateLayer
UpdateRdsDbInstance
UpdateStack
UpdateVolume
```

Permissões que exigem uma política do IAM

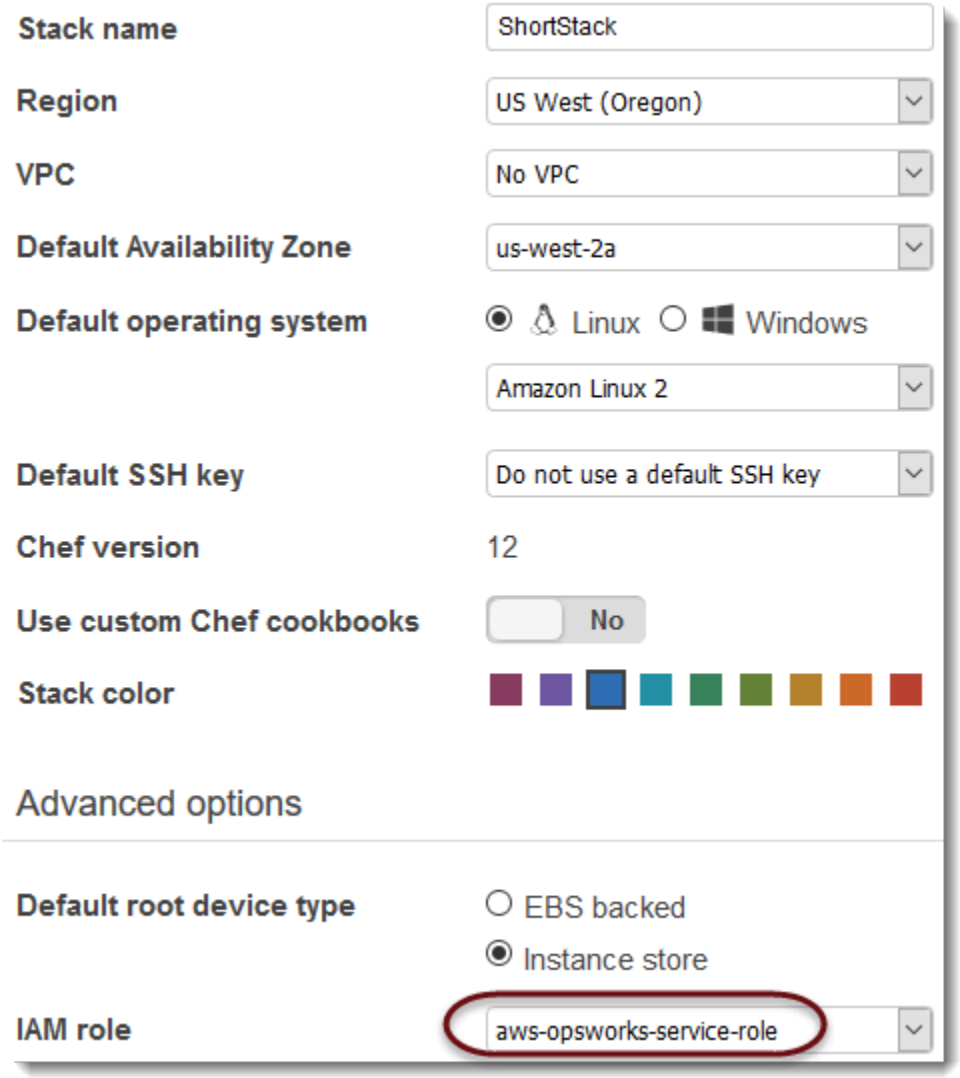
Você deve conceder permissões para as seguintes ações aplicando uma política do IAM adequada ao usuário. Para obter alguns exemplos, consulte Exemplo de políticas.

```
CloneStack
CreateStack
CreateUserProfile
DeleteUserProfile
DescribeUserProfiles
UpdateUserProfile
```

Permitindo que AWS OpsWorks as pilhas ajam em seu nome

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks precisa interagir com uma variedade de serviços da AWS em seu nome. Por exemplo, o AWS OpsWorks Stacks interage com EC2 a Amazon para criar instâncias e com a Amazon CloudWatch para obter estatísticas de monitoramento. Ao criar uma pilha, você especifica uma função do IAM, geralmente chamada de função de serviço, que concede às AWS OpsWorks pilhas as permissões apropriadas.



Quando você especifica uma função de serviço nova da pilha, poderá optar por fazer o seguinte:

- Especifique uma função de serviço padrão criada anteriormente.

  Geralmente, você pode criar um serviço padrão quando cria sua primeira pilha, e então usas essa função para todos pilhas subsequentes.
- Especifica um perfil de serviço personalizado criado usando o console IAM ou API.

  Essa abordagem é útil se você quiser conceder às AWS OpsWorks pilhas permissões mais limitadas do que a função de serviço padrão.

> **Note**
>
> Para criar sua primeira pilha, você precisa ter as permissões definidas no modelo de AdministratorAccesspolítica do IAM. Essas permissões permitem que o AWS OpsWorks Stacks crie uma nova função de serviço do IAM e permitem que você importe usuários , conforme descrito anteriormente . Para todas as pilhas subsequentes, os usuários podem selecionar a função de serviço criado para a primeira pilha; eles não precisam de permissões administrativas para criar uma pilha.

A função de serviço padrão oferece as seguintes permissões:

- Execute todas as EC2 ações da Amazon (`ec2:*`).
- Obtenha CloudWatch estatísticas (`cloudwatch:GetMetricStatistics`).
- Use o Elastic Load Balancing para distribuir tráfego entre os servidores (`elasticloadbalancing:*`).
- Use uma instância do Amazon RDS como um servidor de banco de dados (`rds:*`).
- Use funções do IAM (`iam:PassRole`) para fornecer comunicação segura entre AWS OpsWorks Stacks e suas EC2 instâncias da Amazon.

Se você criar uma função de serviço personalizada, deverá garantir que ela conceda todas as permissões que o AWS OpsWorks Stacks precisa para gerenciar sua pilha. O exemplo de JSON a seguir é a declaração de política para a função de serviço padrão. Uma função de serviço personalizada deve incluir pelo menos as permissões a seguir na declaração de política.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:*",
                "iam:PassRole",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:DescribeAlarms",
                "ecs:*",
                "elasticloadbalancing:*",
                "rds:*"
            ],
```

```
            "Effect": "Allow",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "ec2.amazonaws.com"
                }
            }
        }
    ]
}
```

Uma função de serviço também tem uma relação de confiança. As funções de serviço criadas pelo AWS OpsWorks Stacks têm a seguinte relação de confiança.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "StsAssumeRole",
      "Effect": "Allow",
      "Principal": {
        "Service": "opsworks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

A função de serviço deve ter essa relação de confiança para que a AWS OpsWorks Stacks atue em seu nome. Se você usar a função de serviço padrão, não modifique a relação de confiança. Se você estiver criando um perfil de serviço personalizado, especifique a relação de confiança fazendo uma das seguintes ações:

- Se você estiver usando o assistente Criar perfil no console do IAM, em Escolher um caso de uso, escolha Opsworks. Esse perfil tem a relação de confiança apropriada, mas nenhuma política está implicitamente vinculada. Para conceder permissões ao AWS OpsWorks Stacks para agir em seu nome, crie uma política gerenciada pelo cliente que contenha o seguinte, e anexe-a ao novo perfil.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:DescribeAlarms",
        "cloudwatch:GetMetricStatistics",
        "ec2:*",
        "ecs:*",
        "elasticloadbalancing:*",
        "iam:GetRolePolicy",
        "iam:ListInstanceProfiles",
        "iam:ListRoles",
        "iam:ListUsers",
        "rds:*"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": "ec2.amazonaws.com"
        }
      }
    }
  ]
}
```

- Se você estiver usando um AWS CloudFormation modelo, poderá adicionar algo como o seguinte à seção Recursos do seu modelo.

```
"Resources": {
  "OpsWorksServiceRole": {
      "Type": "AWS::IAM::Role",
```

```
      "Properties": {
        "AssumeRolePolicyDocument": {
            "Statement": [ {
              "Effect": "Allow",
              "Principal": {
                  "Service": [ "opsworks.amazonaws.com" ]
              },
              "Action": [ "sts:AssumeRole" ]
            } ]
        },
        "Path": "/",
        "Policies": [ {
            "PolicyName": "opsworks-service",
            "PolicyDocument": {
              ...
            } ]
        }
      },
   }
}
```

Prevenção confusa entre serviços em Stacks AWS OpsWorks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O problema de "confused deputy" é uma questão de segurança em que uma entidade que não tem permissão para executar uma ação pode coagir uma entidade mais privilegiada a executá-la. Em AWS, a falsificação de identidade entre serviços pode resultar em um problema confuso de delegado. A personificação entre serviços pode ocorrer quando um serviço (o serviço de chamada) chama outro serviço (o serviço chamado). O serviço de chamada pode ser manipulado de modo a usar suas permissões para atuar nos recursos de outro cliente de uma forma na qual ele não deveria ter permissão para acessar. Para evitar isso, a AWS fornece ferramentas que ajudam você

a proteger seus dados para todos os serviços com entidades principais de serviço que receberam acesso aos recursos em sua conta.

Recomendamos usar as chaves de contexto de condição `aws:SourceAccount`global `aws:SourceArn`e as chaves de contexto nas políticas de acesso às pilhas para limitar as permissões que o AWS OpsWorks Stacks concede a outro serviço às pilhas. Se o valor de `aws:SourceArn` não contém ID da conta, como um ARN do bucket do Amazon S3, você deve usar ambas as chaves de contexto de condição global para limitar as permissões. Se você usa ambas as chaves de contexto de condição global, e o valor `aws:SourceArn` contém o ID da conta, o valor `aws:SourceAccount` e a conta no valor `aws:SourceArn` deverão utilizar a mesma ID de conta quando na mesma declaração de política. Use `aws:SourceArn` se quiser que apenas uma pilha seja associada ao acesso entre serviços. Use `aws:SourceAccount` se quiser permitir que qualquer pilha nessa conta seja associada ao uso entre serviços.

O valor de `aws:SourceArn` deve ser o ARN de uma AWS OpsWorks pilha.

A maneira mais eficaz de se proteger contra o problema do substituto confuso é usar a chave de contexto de condição global `aws:SourceArn` da pilha do AWS OpsWorks Stacks com o ARN completo do recurso. Se você não souber o ARN completo ou se estiver especificando várias pilhas ARNs, use a chave de condição de contexto `aws:SourceArn` global com curingas (`*`) para as partes desconhecidas do ARN. Por exemplo, `arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`.

A seção a seguir mostra como você pode usar as chaves de contexto de condição `aws:SourceAccount` global `aws:SourceArn` e as chaves de contexto no AWS OpsWorks Stacks para evitar o confuso problema auxiliar.

Evite explorações confusas de delegados no Stacks AWS OpsWorks

Esta seção descreve como você pode ajudar a evitar explorações secundárias confusas no AWS OpsWorks Stacks e inclui exemplos de políticas de permissões que você pode anexar à função do IAM que você está usando para acessar AWS OpsWorks o Stacks. Como prática recomendada de segurança, sugerimos adicionar as chaves de condição `aws:SourceArn` e `aws:SourceAccount` às relações de confiança que seu perfil do IAM possui com outros serviços. As relações de confiança permitem que AWS OpsWorks as Stacks assumam a função de realizar ações em outros serviços que são necessárias para criar ou gerenciar suas pilhas de AWS OpsWorks Stacks.

Para editar relações de confiança para adicionar chaves de condição **`aws:SourceArn`** e **`aws:SourceAccount`**

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.

2. No painel de navegação à esquerda, selecione Perfis.
3. Na caixa Pesquisar, pesquise a função que você usa para acessar AWS OpsWorks as pilhas. A função AWS gerenciada é`aws-opsworks-service-role`.
4. Na página Resumo do perfil, escolha a guia Relações de confiança.
5. Na guia Relacionamentos de confiança, escolha Editar política de confiança.
6. Na página Editar política de confiança, adicione pelo menos uma das chaves de condição `aws:SourceArn` ou `aws:SourceAccount` à política. Use `aws:SourceArn` para restringir a relação de confiança entre serviços cruzados (como Amazon EC2) e AWS OpsWorks Stacks a pilhas específicas de AWS OpsWorks Stacks, o que é mais restritivo. Adicione `aws:SourceAccount` para restringir a relação de confiança entre serviços cruzados e AWS OpsWorks pilhas às pilhas em uma conta específica, o que é menos restritivo. Veja um exemplo a seguir. Observe que, se você usar as duas chaves de condição, a conta IDs deverá ser a mesma.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "opsworks.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnEquals": {
          "arn:aws:opsworks:us-east-2:123456789012:stack/
EXAMPLEd-5699-40a3-80c3-22c32EXAMPLE/"
        }
      }
    }
  ]
}
```

7. Quando terminar de adicionar as permissões à política, escolha Atualizar política.

Veja a seguir exemplos adicionais de funções que limitam o acesso às pilhas usando `aws:SourceArn` e `aws:SourceAccount`.

Tópicos

- Exemplo: acessando pilhas em uma região específica
- Exemplo: adicionar mais de um ARN de pilha ao aws:SourceArn

Exemplo: acessando pilhas em uma região específica

A seguinte declaração de relação de confiança e função acessa qualquer pilha de AWS OpsWorks Stacks na região Leste dos EUA (Ohio) (). `us-east-2` Observe que a região está especificada no valor ARN de `aws:SourceArn`, mas o valor do ID da pilha é um curinga (*).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "opsworks.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnEquals": {
          "aws:SourceArn": "arn:aws:opsworks:us-east-2:123456789012:stack/*"
        }
      }
    }
  ]
}
```

Exemplo: adicionar mais de um ARN de pilha ao **`aws:SourceArn`**

O exemplo a seguir limita o acesso a uma matriz de duas pilhas de AWS OpsWorks pilhas na ID da conta 123456789012.

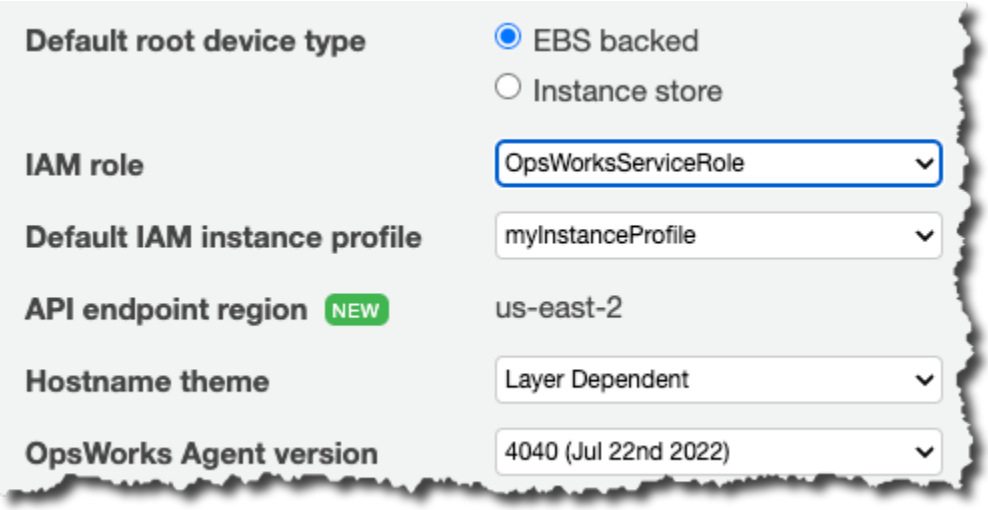
```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "opsworks.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnEquals": {
          "aws:SourceArn": [
             "arn:aws:opsworks:us-east-2:123456789012:stack/unique_ID1",
             "arn:aws:opsworks:us-east-2:123456789012:stack/unique_ID2"
           ]
       }
      }
    }
  ]
}
```

Especificação de permissões para aplicativos executados em instâncias EC2

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Se os aplicativos em execução nas EC2 instâncias da Amazon da sua pilha precisarem acessar outros recursos da AWS, como buckets do Amazon S3, eles devem ter as permissões apropriadas. Para outorgar essas permissões, deve-se utilizar um perfil de instância. Você pode especificar um perfil de instância para cada instância ao criar uma pilha de AWS OpsWorks pilhas.



Para especificar um perfil para instâncias de camada, edite as configurações de camada.

O perfil da instância especifica uma função da IAM. Os aplicativos em execução na instância podem assumir essa função para acessar os recursos da AWS, sujeitos a permissões concedidas pela política de atribuição. Para obter mais informações sobre como um aplicativo assume uma função, consulte Assumir a função usando uma chamada de API.

Um perfil de instância pode ser criado em qualquer uma das seguintes formas:

- Use o console de IAM ou a API para criar um perfil.

  Para obter mais informações, consulte Funções (delegação and federação).
- Use um AWS CloudFormation modelo para criar um perfil.

  Para alguns exemplos de como incluir recursos de IAM em um modelo, consulte Snippets de modelos do Identity and Access Management (IAM).

Um perfil de instância deve ter uma relação de confiança e uma política anexada que conceda permissões para acessar os recursos da AWS.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
```

```
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

O perfil da instância deve ter essa relação de confiança para que o AWS OpsWorks Stacks atue em seu nome. Se você usar a função de serviço padrão, não modifique a relação de confiança. Se você estiver criando uma função de serviço personalizado, especifique a relação de confiança como a seguir:

- Se você estiver usando o assistente Create Role no console do IAM, especifique o tipo de EC2 função da Amazon em AWS Service Roles na segunda página do assistente.
- Se você estiver usando um AWS CloudFormation modelo, poderá adicionar algo como o seguinte à seção Recursos do seu modelo.

```
"Resources": {
      "OpsWorksEC2Role": {
         "Type": "AWS::IAM::Role",
         "Properties": {
            "AssumeRolePolicyDocument": {
               "Statement": [ {
                  "Effect": "Allow",
                  "Principal": {
                     "Service": [ "ec2.amazonaws.com" ]
                  },
                  "Action": [ "sts:AssumeRole" ]
               } ]
            },
            "Path": "/"
         }
      },
      "RootInstanceProfile": {
         "Type": "AWS::IAM::InstanceProfile",
         "Properties": {
            "Path": "/",
            "Roles": [ {
               "Ref": "OpsWorksEC2Role"
            }
         ]
      }
```

```
    }
  }
```

Quando você criar seu perfil de instância, pode-se anexar uma política apropriada para a função do perfil nesse mesmo momento. Depois de criar a pilha, use o console do IAM ou a API para associar uma política apropriada à função do perfil. Por exemplo, a política a seguir concede acesso total a todos os objetos no bucket do Amazon S3 chamado amzn-s3-demo-bucket. Substitua *region* e amzn-s3-demo-bucket por valores apropriados à sua configuração.

```
{
  "Version": "2012-10-17",
  "Statement": [ {
    "Effect": "Allow",
    "Action": "s3:*",
    "Resource": "arn:aws:s3:region::amzn-s3-demo-bucket/*"
    }
  ]
}
```

Para obter um exemplo de como criar e usar um perfil de instância, consulte Usar um bucket do Amazon S3.

Se seu aplicativo usa um perfil de instância para chamar a API AWS OpsWorks Stacks a partir de uma EC2 instância, a política deve permitir a `iam:PassRole` ação, além das ações apropriadas para AWS OpsWorks Stacks e outros serviços da AWS. A permissão `iam:PassRole` autoriza o AWS OpsWorks Stacks; a assumir a função do serviço em seu nome. Para obter mais informações sobre a API AWS OpsWorks Stacks, consulte AWS OpsWorks API Reference.

Veja a seguir um exemplo de uma política do IAM que permite chamar qualquer ação do AWS OpsWorks Stacks de uma EC2 instância, bem como qualquer ação da Amazon EC2 ou do Amazon S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:*",
        "s3:*",
```

```
        "opsworks:*",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:ec2:region:account_id:instance/*",
      "Condition": {
        "StringEquals": {
          "iam:PassedToService": "opsworks.amazonaws.com"
        }
      }
    }
  ]
}
```

> **Note**
>
> Se você não permitir `iam:PassRole`, qualquer tentativa de chamar uma ação do AWS OpsWorks Stacks falhará com um erro como o seguinte:
>
> ```
> User: arn:aws:sts::123456789012:federated-user/Bob is not authorized
> to perform: iam:PassRole on resource:
> arn:aws:sts::123456789012:role/OpsWorksStackIamRole
> ```

Para obter mais informações sobre o uso de funções em uma EC2 instância para obter permissões, consulte Conceder acesso aos recursos da AWS a aplicativos executados em EC2 instâncias da Amazon no Guia do AWS Identity and Access Management usuário.

Gerenciamento do acesso por SSH

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks oferece suporte a chaves SSH para pilhas Linux e Windows.

- Em instâncias do Linux, você pode usar o SSH para fazer login em uma instância, por exemplo, para executar comandos da CLI do agente.

  Para obter mais informações, consulte Login com SSH.
- Em instâncias do Windows, você pode usar uma chave SSH para obter a senha do administrador da instância, a qual você pode usar para fazer login com o RDP.

  Para obter mais informações, consulte Login com RDP.

A autenticação é baseada em um par de chaves SSH, que consiste em uma chave pública e uma chave privada:

- Você instala a chave pública na instância.

  A localização depende do sistema operacional específico, mas o AWS OpsWorks Stacks cuida dos detalhes para você.
- Você armazena a chave privada localmente e a passa para um cliente SSH, como o `ssh.exe`, para acessar a instância.

  O cliente SSH usa a chave privada para conectar-se à instância.

Para fornecer acesso por SSH aos usuários de uma pilha, você precisa de uma maneira de criar os pares de chaves SSH, instalar as chaves públicas nas instâncias da pilha e gerenciar as chaves privadas com segurança.

EC2 A Amazon fornece uma maneira simples de instalar uma chave SSH pública em uma instância. Você pode usar o EC2 console ou a API da Amazon para criar um ou mais pares de chaves para cada região da AWS que você planeja usar. A Amazon EC2 armazena as chaves públicas na AWS e você armazena as chaves privadas localmente. Ao iniciar uma instância, você especifica um dos pares de chaves da região e a Amazon o instala EC2 automaticamente na instância. Em seguida, você pode usar a chave privada correspondente para fazer login na instância. Para obter mais informações, consulte Amazon EC2 Key Pairs.

Com o AWS OpsWorks Stacks, você pode especificar um dos pares de EC2 chaves da Amazon da região ao criar uma pilha e, opcionalmente, substituí-lo por um par de chaves diferente ao criar cada instância. Quando o AWS OpsWorks Stacks inicia a EC2 instância correspondente da Amazon, ele especifica o par de chaves e a Amazon EC2 instala a chave pública na instância. Em seguida, você

pode usar a chave privada para fazer login ou recuperar uma senha de administrador, assim como faria com uma EC2 instância padrão da Amazon. Para obter mais informações, consulte Instalando uma Amazon EC2 Key.

Usar um par de EC2 chaves da Amazon é conveniente, mas tem duas limitações significativas:

- Um par de EC2 chaves da Amazon está vinculado a uma região específica da AWS.

  Se você trabalha em várias regiões, deve gerenciar vários pares de chaves.
- Você pode instalar somente um par de EC2 chaves da Amazon em uma instância.

  Se você deseja permitir que vários usuários façam login, todos eles precisam ter uma cópia da chave privada, o que não é uma prática de segurança recomendada.

Para pilhas Linux, o AWS OpsWorks Stacks fornece uma maneira mais simples e flexível de gerenciar pares de chaves SSH.

- Cada usuário registra um par de chaves pessoal.

  Eles armazenam a chave privada localmente e registram a chave pública no AWS OpsWorks Stacks, conforme descrito emRegistro de uma chave SSH pública de um usuário .
- Quando você define permissões de usuário para uma stack, especifica quais usuários devem ter acesso por SSH às instâncias da pilha.

  AWS OpsWorks O Stacks cria automaticamente um usuário do sistema nas instâncias da pilha para cada usuário autorizado e instala sua chave pública. O usuário pode, em seguida, usar a chave privada correspondente para fazer login, conforme descrito em Login com SSH.

O uso de chaves SSH pessoais tem as seguintes vantagens.

- Não há necessidade de configurar manualmente as chaves nas instâncias; o AWS OpsWorks Stacks instala automaticamente as chaves públicas apropriadas em cada instância.
- AWS OpsWorks O Stacks instala somente as chaves públicas pessoais dos usuários autorizados.

  Os usuários não autorizados não podem usar suas chaves privadas pessoais para obter acesso a instâncias. Com os pares de EC2 chaves da Amazon, qualquer usuário com a chave privada correspondente pode fazer login, com ou sem acesso SSH autorizado.

- Se um usuário não precisar mais de acesso por SSH, você pode usar a página Permissions para revogar as permissões de SSH/RDP do usuário.

  AWS OpsWorks O Stacks desinstala imediatamente a chave pública das instâncias da pilha.
- Você pode usar a mesma chave para qualquer região da AWS.

  Os usuários gerenciam apenas uma chave privada.
- Não há necessidade de compartilhar chaves privadas.

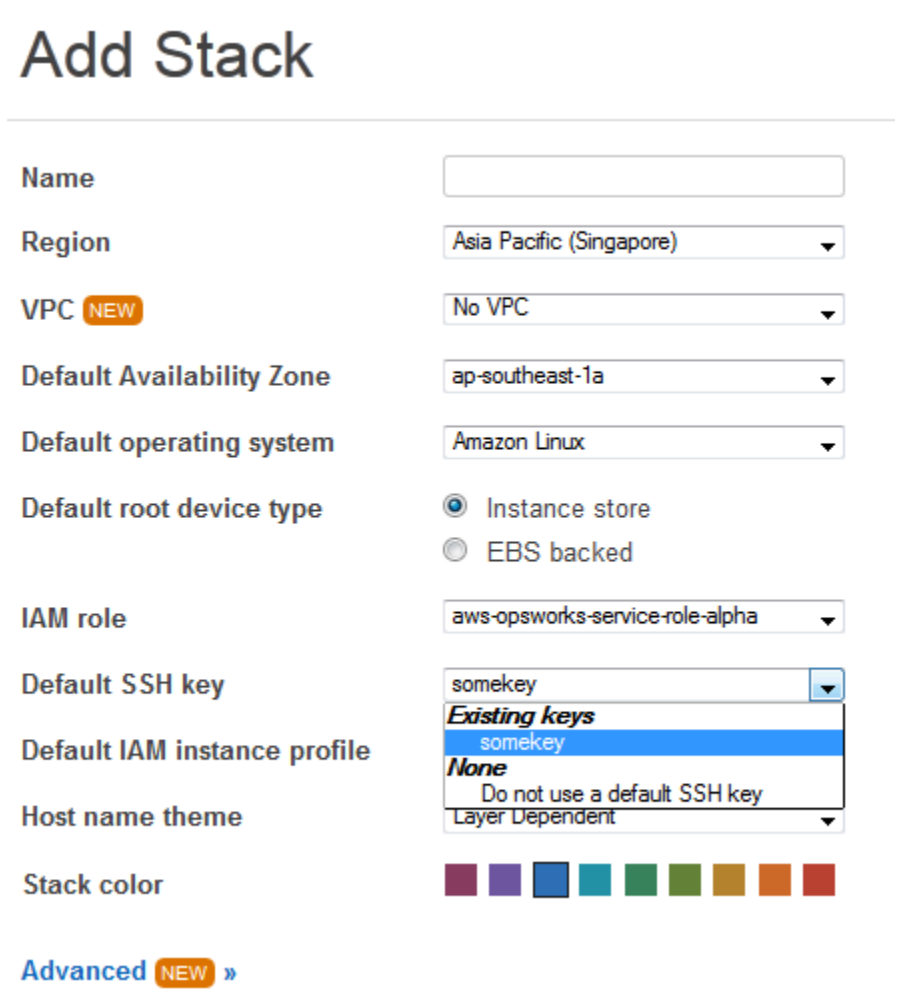
  Cada usuário tem sua própria chave privada.
- É fácil fazer a rotação das chaves.

  Você ou o usuário atualiza a chave pública em My Settings (Minhas configurações) e o AWS OpsWorks Stacks atualiza automaticamente as instâncias.

Instalando uma Amazon EC2 Key

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

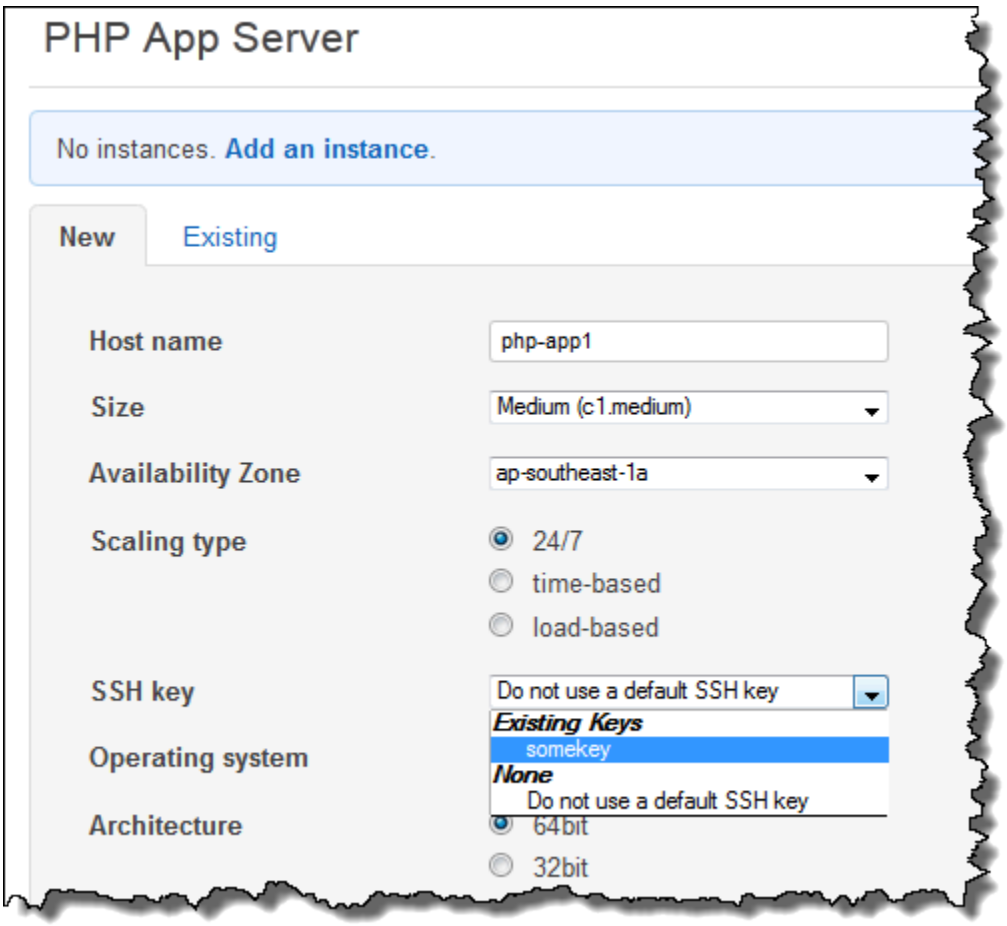
Ao criar uma pilha, você pode especificar uma chave EC2 SSH da Amazon que é instalada por padrão em cada instância na pilha.



A lista de chaves SSH padrão mostra a Amazon EC2keys da sua conta da AWS. Você pode executar uma das seguintes ações:

- Selecione a chave apropriada a partir da lista.
- Selecione Do not use a default SSH key para não especificar nenhuma chave.

Se você tiver selecionado Do not use a default SSH key, ou se deseja substituir a chave padrão da pilha, pode especificar uma chave quando criar uma instância.



Quando você inicia a instância, o AWS OpsWorks Stacks instala a chave pública no `authorized_keys` arquivo.

Registro de uma chave SSH pública de um usuário

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Há duas maneiras de registrar a chave SSH pública de um usuário:

- Um usuário administrativo pode atribuir uma chave SSH pública a um ou mais usuários e fornecer-lhes a chave privada correspondente.
- Um usuário administrativo pode habilitar o autogerenciamento para um ou mais usuários.

  Esses usuários podem especificar suas próprias chaves SSH públicas.

Para obter mais informações sobre como os usuários administrativos podem habilitar o autogerenciamento ou atribuir chaves públicas a usuários, consulte Edição das configurações de usuário.

A conexão com instâncias baseadas em Linux usando o SSH em um terminal PuTTY requer etapas adicionais. Para obter mais informações, consulte Conexão da sua instância Linux a partir do Windows usando PuTTY e Solução de problemas de conexão da sua instância na documentação da AWS.

A seção a seguir descreve como um usuário com o autogerenciamento habilitado pode especificar sua chave pública.

Para especificar sua chave SSH pública

1. Crie um par de chaves SSH.

   A abordagem mais simples é gerar o par de chaves localmente. Para obter mais informações, consulte Como gerar sua própria chave e importá-la para a Amazon EC2.

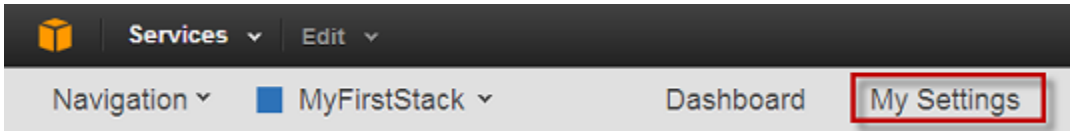
   > **Note**
   >
   > Se você usar Pu TTYgen para gerar seu par de chaves, copie a chave pública da caixa Chave pública para colar no arquivo OpenSSH authorized_keys. Clicar em Salvar chave pública salva a chave pública em um formato que não é suportado pelo MindTerm.

2. Faça login no console do AWS OpsWorks Stacks como um usuário do IAM com o autogerenciamento ativado.

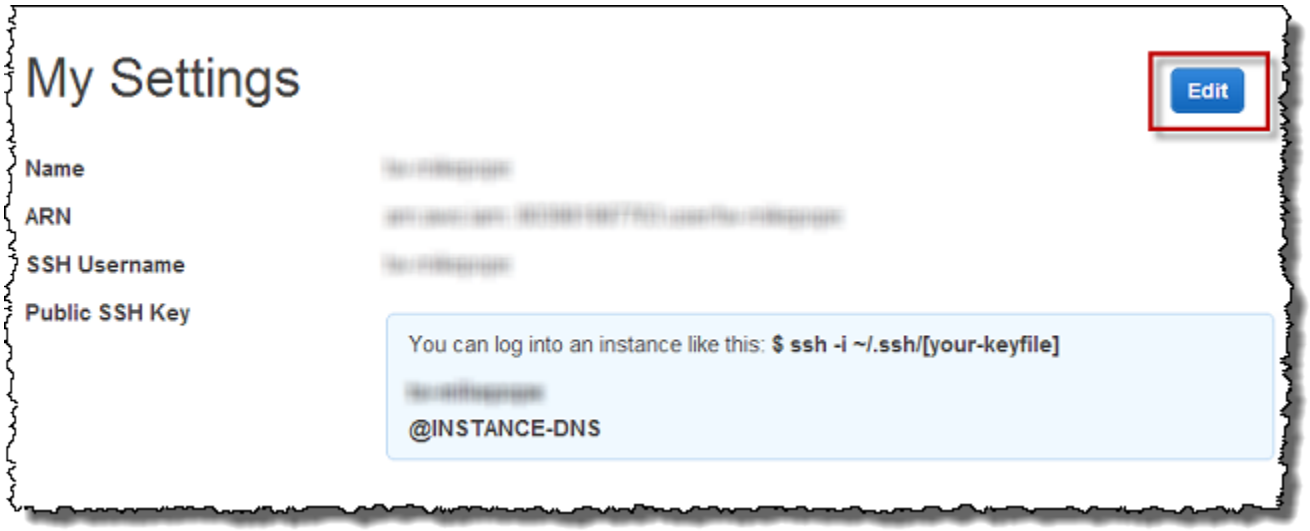
   > **Important**
   >
   > Se você fizer login como proprietário da conta ou como usuário do IAM sem autogerenciamento ativado, o AWS OpsWorks Stacks não exibirá Minhas configurações.

Se você é um usuário administrativo ou o proprietário da conta, pode especificar chaves SSH acessando a página Users e editando as configurações do usuário.

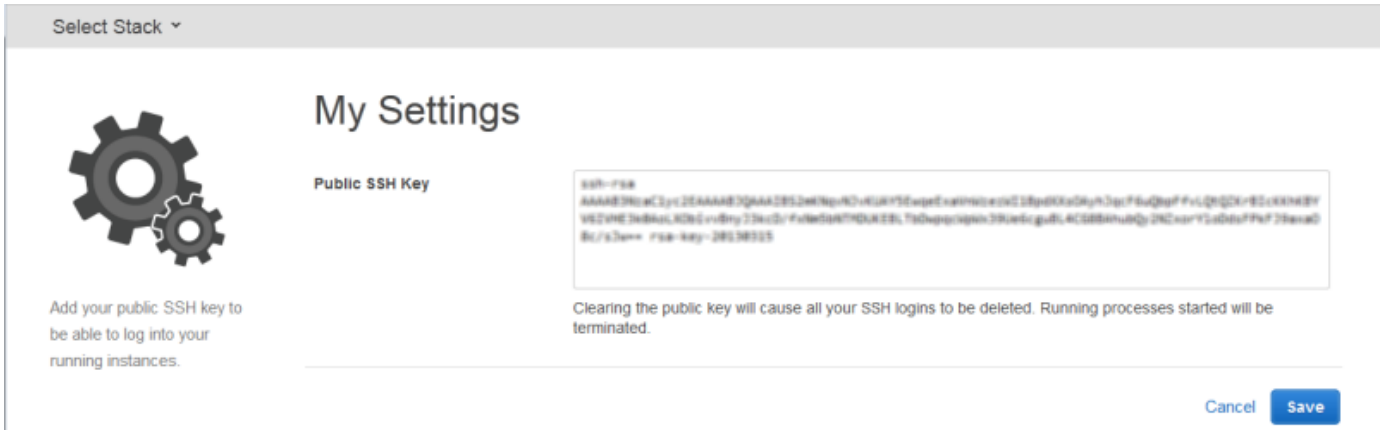
3. Selecione Minhas configurações, que exibe as configurações do usuário conectado.



4. Na página My Settings, clique em Edit.



5. Em Public SSH Key box, insira sua chave pública SSH e clique em Save.



Important

Para usar o cliente MindTerm SSH integrado para se conectar às EC2 instâncias da Amazon, um usuário deve estar conectado como usuário do IAM e ter uma chave SSH pública

registrada no Stacks. AWS OpsWorks Para obter mais informações, consulte Usando o cliente MindTerm SSH integrado.

Gerenciamento de atualizações de segurança do Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Atualizações de segurança

Os fornecedores do sistema operacional Linux fornecem atualizações regulares, a maioria das quais são patches de segurança do sistema operacional, mas também podem incluir atualizações aos pacotes instalados. Você deve ter certeza de que os sistemas operacionais das instâncias são atuais com os patches de segurança mais recentes.

Por padrão, o AWS OpsWorks Stacks instala automaticamente as atualizações mais recentes durante a configuração, após a conclusão da inicialização da instância. AWS OpsWorks O Stacks não instala atualizações automaticamente depois que uma instância está on-line, para evitar interrupções, como reiniciar servidores de aplicativos. Em vez disso, você gerencia atualizações para suas instâncias online você mesmo, para que possa minimizar quaisquer interrupções.

Recomendamos que você use um dos seguintes para atualizar suas instâncias online.

- Crie e inicie novas instâncias para substituir suas instâncias online atuais. Depois, exclua as instâncias atuais.

  As novas instâncias terão o último conjunto de patches de segurança instalados durante a configuração.
- Em instâncias baseadas em Linux no Chef 11,10 ou pilhas mais antigas, execute o comando de pilha Atualizar dependências, que instala o conjunto atual de patches de segurança e outras atualizações nas instâncias especificadas.

Para essas duas abordagens, o AWS OpsWorks Stacks executa a atualização executando `yum update` para Amazon Linux e Red Hat Enterprise Linux (RHEL) ou `apt-get update` para Ubuntu. Cada distribuição lida com as atualizações de forma diferente, então você deveria examinar as informações nos links associados para entender exatamente como uma atualização afetará suas instâncias:

- Amazon Linux: as atualizações do Amazon Linux instalam patches de segurança e também podem instalar atualizações de atributo, incluindo atualizações de pacotes.

  Para obter mais informações, consulte AMI do Amazon Linux FAQs.
- Ubuntu: as atualizações do Ubuntu são amplamente limitadas à instalação de patches de segurança, mas também podem instalar atualizações de pacotes para um número limitado de correções críticas.

  Para obter mais informações, consulte LTS - Ubuntu Wiki.
- CentOS: as atualizações do CentOS geralmente mantêm a compatibilidade binária com versões anteriores.
- RHEL: as atualizações do RHEL geralmente mantêm a compatibilidade binária com versões anteriores.

  Para obter mais informações, consulte Ciclo de vida do Red Hat Enterprise Linux.

Se você quiser ter mais controle sobre as atualizações, como especificar versões específicas do pacote, você pode desativar as atualizações automáticas usando as UpdateLayerações CreateInstance,, UpdateInstanceCreateLayer, ou — ou os métodos equivalentes do AWS SDK ou comandos da AWS CLI — para definir o parâmetro como. `InstallUpdatesOnBoot false` Os exemplos a seguir mostram como usar o CLI da AWS para desativar `InstallUpdatesOnBoot` como configuração padrão de uma camada existente.

```
aws opsworks update-layer --layer-id layer ID --no-install-updates-on-boot
```

Depois, você deve gerenciar as atualizações você mesmo. Por exemplo, você pode empregar uma dessas estratégias:

- Implementar uma receita personalizada que executa o comando shell apropriado para instalar suas atualizações preferidas.

Como as atualizações do sistema não são mapeadas naturalmente para um evento de ciclo de vida, inclua a receita em seus livros personalizados, mas execute-a manualmente. Para atualizações de pacote, você também pode usar os recursos yum_package (Amazon Linux) ou apt_package (Ubuntu) em vez do comando shell.

- Entre em cada instância com SSH e execute os comandos apropriados manualmente.

Usar grupos de segurança

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Grupos de segurança

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Cada EC2 instância da Amazon tem um ou mais grupos de segurança associados que controlam o tráfego de rede da instância, como um firewall. Um grupo de segurança tem uma ou mais regras, sendo que cada uma delas especifica uma determinada categoria de permissão de tráfego. Uma regra especifica o seguinte:

- O tipo de tráfego permitido, como SSH ou HTTP
- O protocolo de tráfego, como TCP ou UDP
- O intervalo de endereços IP do qual o tráfego pode ser proveniente

- O intervalo de porta permitido do tráfego

Os grupos de segurança têm dois tipos de regras:

- As regras de entrada regulam o tráfego de entrada da rede.

  Por exemplo, as instâncias de servidor de aplicativos normalmente têm uma regra de entrada que permite a entrada de tráfego HTTP de qualquer endereço IP para a porta 80 e outra regra de entrada que permite a entrada de tráfego SSH para a porta 22 de um conjunto de endereços IP especificado.
- As regras de saída controlam o tráfego de saída da rede.

  Uma prática comum é usar a configuração padrão, que permite qualquer tráfego de saída.

Para obter mais informações sobre grupos de segurança, consulte Amazon EC2 Security Groups.

Na primeira vez que você cria uma pilha em uma região, o AWS OpsWorks Stacks cria um grupo de segurança integrado para cada camada com um conjunto apropriado de regras. Todos os grupos têm regras de saída padrão que permitem todo o tráfego de saída. Em geral, as regras de entrada permitem o seguinte:

- Tráfego TCP, UDP e ICMP de entrada das camadas apropriadas do Stacks AWS OpsWorks
- Tráfego de entrada TCP na porta 22 (SSH login)

> **Warning**
>
> A configuração padrão do grupo de segurança abre o SSH (porta 22) para qualquer rede local (0.0.0.0/0). Isso permite que todos os endereços IP acessem sua instância usando o SSH. Para ambientes de produção, você deve usar uma configuração que só permite o acesso SSH de um endereço IP específico ou de um intervalo de endereços. Atualize os grupos de segurança padrão imediatamente após criá-los ou use grupos de segurança personalizados.

- Para camadas de servidor da Web, todo tráfego de entrada TCP e UDP vai para as portas 80 (HTTP) e 443 (HTTPS)

> **Note**
>
> O grupo de segurança `AWS-OpsWorks-RDP-Server` integrado é atribuído para todas as instâncias do Windows para permitir o acesso RDP. No entanto, por padrão, ele não tem regras. Se você estiver executando uma pilha do Windows e quiser usar o RDP para acessar instâncias, deve adicionar uma regra de entrada que permita o acesso RDP. Para obter mais informações, consulte Login com RDP.

Para ver os detalhes de cada grupo, acesse o EC2console da Amazon, selecione Security Groups no painel de navegação e selecione o grupo de segurança da camada apropriada. Por exemplo, AWS- OpsWorks -Default-Server é o grupo de segurança integrado padrão para todas as pilhas, e AWS OpsWorks - WebApp - é o grupo de segurança integrado padrão para a pilha de amostras do Chef 12.

> **Note**
>
> Se você excluir acidentalmente um grupo de segurança do AWS OpsWorks Stacks, a forma preferida de recriá-lo é fazer com que o AWS OpsWorks Stacks execute a tarefa para você. Basta criar uma nova pilha na mesma região da AWS — e na VPC, se AWS OpsWorks presente — e o Stacks recriará automaticamente todos os grupos de segurança integrados, incluindo aquele que você excluiu. Você pode então excluir a pilha se não tiver mais uso para ela; os security groups permanecerão. Se você quer recriar o grupo de segurança manualmente, ele deve ser uma duplicata exata do original, incluindo a capitalização do nome do grupo.
>
> Além disso, o AWS OpsWorks Stacks tentará recriar todos os grupos de segurança integrados se alguma das seguintes situações ocorrer:
>
> - Você faz qualquer alteração na página de configurações da pilha no console AWS OpsWorks Stacks.
> - Você iniciar uma das instâncias da pilha.
> - Você criar uma nova pilha.

Você pode usar uma das seguintes abordagens para especificar grupos de segurança. Você usa a configuração Usar grupos de OpsWorks segurança para especificar sua preferência ao criar uma pilha.

- Sim (configuração padrão) — O AWS OpsWorks Stacks associa automaticamente o grupo de segurança incorporado apropriado a cada camada.

  Você pode ajustar um grupo de segurança integrado de uma camada adicionando um grupo de segurança personalizado nas configurações de sua preferência. No entanto, quando a Amazon EC2 avalia vários grupos de segurança, ela usa as regras menos restritivas, portanto, você não pode usar essa abordagem para especificar regras mais restritivas do que o grupo incorporado.
- Não — O AWS OpsWorks Stacks não associa grupos de segurança integrados a camadas.

  Você deve criar grupos de segurança apropriados e associar pelo menos um a cada camada que criar. Use essa abordagem para especificar regras mais restritivas do que os grupos integrados. Note que ainda é possível associar um grupo de segurança integrado a uma camada se preferir. Os grupos de segurança personalizados são necessários apenas para as camadas que precisam de configurações personalizadas.

> **Important**
>
> Se você usar os grupos de segurança integrados, não pode criar regras mais restritivas modificando as configurações do grupo manualmente. Cada vez que você cria uma pilha, o AWS OpsWorks Stacks substitui as configurações dos grupos de segurança integrados, de modo que todas as alterações feitas serão perdidas na próxima vez que você criar uma pilha. Se uma camada exigir configurações de grupo de segurança mais restritivas do que o grupo de segurança incorporado, defina Usar grupos de OpsWorks segurança como Não, crie grupos de segurança personalizados com suas configurações preferidas e atribua-os às camadas na criação.

AWS OpsWorks Suporte Stacks para Chef 12 Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção fornece uma breve visão geral do AWS OpsWorks Stacks for Chef 12 Linux. Para obter informações sobre o Chef 12 no Windows, consulte Conceitos básicos: Windows. Para obter informações sobre as versões anteriores do Chef no Linux, consulte Chef 11.10 e versões anteriores para Linux.

Visão geral

AWS OpsWorks O Stacks suporta o Chef 12, a versão mais recente do Chef, para pilhas Linux. Para obter mais informações, consulte Learn Chef.

AWS OpsWorks O Stacks continua oferecendo suporte ao Chef 11.10 para pilhas Linux. No entanto, se você é um usuário avançado do Chef que deseja aproveitar a ampla seleção de livros de receitas da comunidade ou criar os próprios livros de receitas personalizados, recomendamos que use o Chef 12. As pilhas do Chef 12 fornecem as seguintes vantagens em relação ao Chef 11.10 e pilhas anteriores para Linux:

- Duas execuções separadas do Chef - Quando um comando é executado em uma instância, o agente AWS OpsWorks Stacks agora executa duas execuções isoladas do Chef: uma execução para tarefas que integram a instância a outros serviços da AWS, como AWS Identity and Access Management (IAM), e uma execução para seus livros de receitas personalizados. A primeira execução do Chef instala o agente AWS OpsWorks Stacks na instância e executa tarefas do sistema, como configuração e gerenciamento de usuários, configuração e configuração de volumes, configuração de CloudWatch métricas e assim por diante. A segunda execução é dedicada exclusivamente à inicialização dos livros de receitas para AWS OpsWorks Eventos de ciclo de vida de pilhas. Essa segunda execução permite que você use as próprias receitas do Chef ou receitas da comunidade.
- Resolução de conflitos de namespace – Antes do Chef 12, o AWS OpsWorks Stacks realizava tarefas do sistema e executava receitas incorporadas e personalizadas em um ambiente compartilhado. Isso resultou em conflitos de namespace e falta de clareza sobre quais receitas o AWS OpsWorks Stacks havia executado. As configurações padrão indesejadas teriam que ser substituídas manualmente, uma tarefa demorada e sujeita a erro. No Chef 12 para Linux, o AWS OpsWorks Stacks não oferece mais suporte aos livros de receitas integrados do Chef para ambientes de servidores de aplicativos como PHP, Node.js ou Rails. Ao eliminar receitas integradas, o AWS OpsWorks Stacks elimina o problema de colisões de nomes entre receitas integradas e personalizadas.
- Forte suporte para livros de receitas da comunidade Chef — O AWS OpsWorks Stacks Chef 12 Linux oferece maior compatibilidade e suporte para livros de receitas comunitários do

supermercado Chef. Agora você pode usar livros de receitas da comunidade que são superiores aos livros de receitas integrados fornecidos anteriormente pela AWS OpsWorks Stacks — livros de receitas projetados para uso com os ambientes e estruturas de servidores de aplicativos mais recentes. Você pode executar a maioria desses livros de receitas sem modificações no Chef 12 para Linux. Para obter mais informações, acesse Chef Supermarket no site do Learn Chef, no site do Chef Supermarket e no repositório Chef Cookbooks em. GitHub

- Atualizações oportunas do Chef 12 - O AWS OpsWorks Stacks atualizará seu ambiente Chef para a versão mais recente do Chef 12 logo após cada lançamento do Chef. Com o Chef 12, pequenas atualizações do Chef e novos lançamentos do agente AWS OpsWorks Stacks coincidirão. Assim, será possível testar novas versões do Chef diretamente. Além disso, seus aplicativos e suas receitas do Chef poderão aproveitar os recursos mais recentes do Chef.

Para obter mais informações sobre as versões do Chef compatíveis antes do Chef 12, consulte Chef 11.10 e versões anteriores para Linux.

A mudança para o Chef 12

As alterações do Key AWS OpsWorks Stacks para o Chef 12 Linux, em comparação com o suporte para as versões anteriores do Chef 11.10, 11.4 e 0.9, são as seguintes:

- As camadas integradas não são mais fornecidas ou compatíveis com o Chef 12 para pilhas do Linux. Como somente as receitas personalizadas são executadas, a remoção dessa compatibilidade proporciona transparência total em relação ao modo como a instância é configurada e facilita muito a criação e manutenção dos livros de receitas personalizados. Por exemplo, não é mais necessário sobrescrever os atributos das receitas integradas do AWS OpsWorks Stacks. A remoção das camadas integradas também permite que o AWS OpsWorks Stacks ofereça melhor suporte aos livros de receitas desenvolvidos e mantidos pela comunidade Chef, para que você possa tirar o máximo proveito deles. Os tipos de camada integrados que não estão mais disponíveis no Chef 12 para Linux são: AWS Flow (Ruby), Ganglia HAProxy, Java App Server, Memcached, MySQL, Node.js App Server, PHP AppServer, Rails App Server e Static Web Server.
  - Como o AWS OpsWorks Stacks está executando as receitas fornecidas por você, não há mais a necessidade de substituir os atributos integrados do AWS OpsWorks Stacks executando livros de receitas personalizados. Para substituir atributos nas próprias receitas ou em receitas da comunidade, siga as instruções e os exemplos em About Attributes na documentação do Chef 12.

- AWS OpsWorks As pilhas continuam fornecendo suporte para as seguintes camadas das pilhas do Chef 12 Linux:
  - Camadas personalizadas
  - Camada de serviços do Amazon RDS
  - Camadas de cluster do ECS
- Os recipientes de dados e a configuração de pilha para o Chef 12 Linux foram alterados para parecerem muito semelhantes aos correspondentes para o Chef 12.2 Windows. Isso facilita muito a consulta, análise e solução de problemas desses recipientes de dados, especialmente se você trabalha com pilhas com diferentes tipos de sistema operacional. Observe que o AWS OpsWorks Stacks não oferece suporte a pacotes de dados criptografados. Para armazenar dados confidenciais de maneira criptografada, como senhas ou certificados, recomendamos armazená-los em um bucket do S3 privado. Você pode então criar uma receita personalizada que usa o Amazon SDK para Ruby para recuperar os dados. Para ver um exemplo, consulte Usar o SDK for Ruby. Para mais informações, consulte AWS OpsWorks Referência do Stacks Data Bag.
- No Chef 12 Linux, o Berkshelf não está mais instalado nas instâncias de pilha. Em vez disso, recomendamos que você use o Berkshelf em uma máquina de desenvolvimento local para empacotar as dependências do livro de receitas localmente. Em seguida, envie o pacote, com as dependências incluídas, ao Amazon Simple Storage Service. Por fim, modifique a pilha do Chef 12 Linux para usar o pacote carregado como uma origem de livro de receitas. Para obter mais informações, consulte Empacotar dependências do livro de receitas localmente.
- As configurações de RAID para volumes do EBS não são mais compatíveis. Para aumentar o desempenho, você pode usar IOPS provisionadas para o Amazon Elastic Block Store (Amazon EBS).
- autofs não é mais compatível.
- Os repositórios Subversion não são mais compatíveis.
- As instalações de pacote do SO por camada devem ser realizadas com as receitas personalizadas. Para obter mais informações, consulte Instalações de pacotes por camada.

Sistemas operacionais compatíveis

O Chef 12 é compatível com os mesmos sistemas operacionais Linux que as versões anteriores do Chef. Para obter uma lista dos tipos e das versões de sistemas operacionais Linux que as pilhas do Chef 12 Linux podem usar, consulte Sistemas operacionais Linux.

Tipos de instâncias compatíveis

AWS OpsWorks O Stacks oferece suporte a todos os tipos de instância para pilhas Linux do Chef 12, exceto tipos de instância especializados, como computação em cluster de computação de alto desempenho (HPC), GPU de cluster e tipos de instância de cluster com alta memória.

Mais informações

Para saber mais sobre como trabalhar com o Chef 12 para pilhas do Linux, consulte o item a seguir:

- Conceitos básicos: exemplo

  Apresenta o AWS OpsWorks Stacks guiando você por meio de um breve exercício prático com o console do AWS OpsWorks Stacks para criar um ambiente de aplicativo Node.js.

- Conceitos básicos: Linux

  Apresenta o AWS OpsWorks Stacks e o Chef 12 Linux, guiando você por um exercício prático com o console AWS OpsWorks Stacks para criar uma pilha básica do Chef 12 Linux que contém uma camada simples com um aplicativo Node.js que fornece tráfego.

- Camadas personalizadas

  Oferece orientação para adicionar uma camada que contém livros de receitas e receitas para uma pilha do Chef 12 Linux. Você pode usar livros de receitas e receitas prontamente disponíveis que a comunidade do Chef fornece ou pode criar suas próprias.

- Movendo para recipientes de dados

  Compara e contrasta a instância JSON que é usada por pilhas do Linux executando o Chef 11 e versões anteriores com o Chef 12. Além disso, fornece indicadores para a documentação de referência do formato JSON da instância do Chef 12.

Movendo configurações de pilha de atributos para recipientes de dados

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks Stacks expõe uma grande variedade de configurações de pilha às suas receitas do Chef. Essas configurações de pilha incluem valores como:

- Fonte do livro de receitas Stack URLs
- Configurações de volume de camada
- Nomes do host da instância
- Nomes de DNS do Elastic Load Balancing
- Fonte do aplicativo URLs
- Nomes de usuário

Fazer referências de configurações de pilha de receitas torna o código da receita mais robusto e menos propenso a erros que as configurações de pilha em formato hardcoding diretamente nas receitas. Esse tópico descreve como acessar essas configurações de pilha, assim como o jeito de mover atributos no Chef 11.10 e versões anteriores para Linux de recipientes de dados no Chef 12 Linux.

No Chef 11.10 e em versões anteriores do Linux, configurações de pilha estão disponíveis como Atributos do Chef e são acessadas por meio do objeto do Chef `node` ou da busca do Chef. Esses atributos são armazenados em instâncias do AWS OpsWorks Stacks em um conjunto de arquivos JSON no `/var/lib/aws/opsworks/chef` diretório. Para obter mais informações, consulte Atributos de implantação e configuração de pilha: Linux.

No Chef 12 Linux, configurações de pilha estão disponíveis como Recipientes de dados da pilha e são acessados apenas por meio da busca do Chef. Os pacotes de dados são armazenados em instâncias do AWS OpsWorks Stacks em um conjunto de arquivos JSON no `/var/chef/runs/`*`run-ID`*`/data_bags` diretório, onde *`run-ID`* há um ID exclusivo que o AWS OpsWorks Stacks atribui a cada execução do Chef em uma instância. Configurações de pilha não estão mais disponíveis como atributos de Chef, então as configurações de pilha não podem mais ser acessadas por meio do objeto `node` do Chef. Para obter mais informações, consulte AWS OpsWorks Referência do Stacks Data Bag.

Por exemplo, no Chef 11.10 e em versões anteriores para Linux, o código de receita a seguir usa o objeto `node` do Chef para obter atributos representando um nome curto de aplicativo e URL fonte. Ele então usa o log do Chef para escrever esses dois valores de atributo:

```
Chef::Log.info ("********** The app's short name is '#{node['opsworks']
['applications'].first['slug_name']}' **********")
Chef::Log.info("********** The app's URL is '#{node['deploy']['simplephpapp']['scm']
['repository']}' **********")
```

No Chef 12 Linux, o código de receita a seguir usa o índice de busca `aws_opsworks_app` para obter conteúdos do primeiro recipiente de dados no recipiente de dados `aws_opsworks_app`. Em seguida, o código grava duas mensagens no log do Chef, uma com o conteúdo de data bag de nome curto do aplicativo e outra com o conteúdo de data bag de URL de origem do aplicativo:

```
app = search("aws_opsworks_app").first

Chef::Log.info("********** The app's short name is '#{app['shortname']}' **********")
Chef::Log.info("********** The app's URL is '#{app['app_source']['url']}' **********")
```

Para migrar seu código de receita para acessar as configurações de pilha do Chef 11.10 e versões anteriores do Linux para Chef 12 Linux, você deve revisar seu código para:

- Acessar recipientes de dados Chef em vez de atributos Chef.
- Use busca do Chef em vez do objeto `node` do Chef.
- Use nomes de pacotes de dados do AWS OpsWorks Stacks`aws_opsworks_app`, como, em vez de usar nomes de atributos do AWS OpsWorks Stacks, como e. `opsworks` `deploy`

Para obter mais informações, consulte AWS OpsWorks Referência do Stacks Data Bag.

Support para versões anteriores do Chef em AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção fornece uma breve visão geral da documentação do AWS OpsWorks Stacks para as versões anteriores do Chef.

Chef 11.10 e versões anteriores para Linux

Fornece documentação sobre o suporte do AWS OpsWorks Stacks para Chef 11.10, 11.4 e 0.9 para pilhas Linux.

Chef 11.10 e versões anteriores para Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção fornece uma breve visão geral da documentação do AWS OpsWorks Stacks para Chef 11.10, 11.4 e 0.9 para Linux.

Conceitos básicos das pilhas Linux do Chef 11

Fornece uma descrição que mostra como criar uma stack de servidor de aplicativos PHP simples, mas funcional.

Criação da sua primeira stack do Node.js

Descreve como criar uma pilha do Linux que oferece suporte a um servidor de aplicativos Node.js e como implantar um aplicativo simples.

Personalizando pilhas AWS OpsWorks

Descreve como personalizar AWS OpsWorks as pilhas para atender às suas necessidades específicas.

Introdução a livros de receitas

Descreve como implementar receitas para instâncias do AWS OpsWorks Stacks.

Balanceamento de carga de uma camada

Descreve como usar as opções de balanceamento de carga do AWS OpsWorks Stacks disponíveis.

Execução de uma Stack em uma VPC

Descreve como criar e executar uma stack em uma nuvem privada virtual.

Migração do Chef Server

Fornece diretrizes para migrar do Chef Server para o AWS OpsWorks Stacks.

AWS OpsWorks Referência de camadas de pilhas

Descreve as camadas integradas do AWS OpsWorks Stacks disponíveis.

Componentes do livro de receitas

Descreve os três componentes do livro de receitas padrão: atributos, modelos e receitas.

Atributos de implantação e configuração de pilha: Linux

Descreve os atributos de configuração e implantação da pilha para Linux.

Atributos de livros de receitas integrados

Descreve como usar atributos de receita integrados para controlar a configuração do software instalado.

Solução de problemas do Chef 11.10 e versões anteriores para Linux

Descreve abordagens para solucionar vários problemas no AWS OpsWorks Stacks.

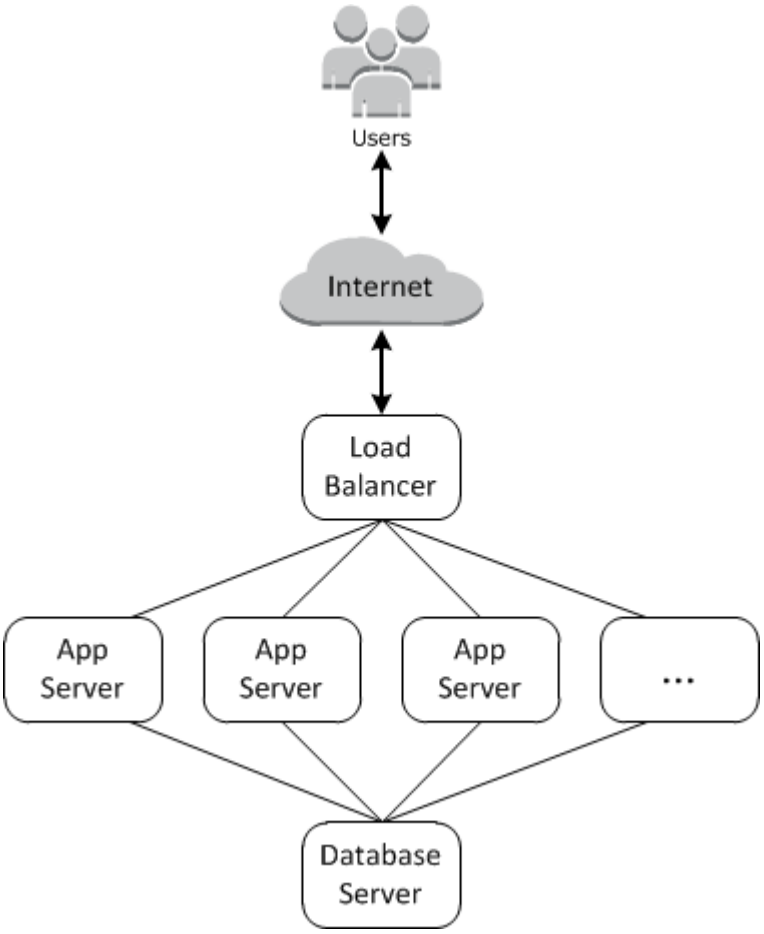
Conceitos básicos das pilhas Linux do Chef 11

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta seção descreve os conceitos básicos das pilhas Linux usando o Chef 11. Para obter informações sobre os conceitos básicos das pilhas Linux do Chef 12, consulte Conceitos básicos: Linux. Para obter informações sobre os conceitos básicos das pilhas Windows do Chef 12, consulte Conceitos básicos: Windows.

Aplicativos baseados na nuvem geralmente exigem um grupo de recursos relacionados (servidores de aplicativos, servidores de banco de dados e assim por diante) que devem ser criados e gerenciados coletivamente. Esta coleção de instâncias é chamada de pilha. Uma simples pilha de aplicativo pode ter a seguinte aparência.



A arquitetura básica contém o seguinte:

- Um load balancer para distribuir o tráfego de entrada dos usuários uniformemente em todos os servidores de aplicativos.
- Um conjunto de instâncias do servidor de aplicativos, a quantidade necessária para lidar com o tráfego.
- Um servidor de banco de dados para fornecer um armazenamento de dados de back-end aos servidores de aplicativos.

Além disso, normalmente é preciso ter uma maneira para distribuir os aplicativos para os servidores de aplicativos, monitorar a pilha, e assim por diante.

AWS OpsWorks O Stacks fornece uma maneira simples e direta de criar e gerenciar pilhas e seus aplicativos e recursos associados. Este capítulo apresenta os conceitos básicos do AWS OpsWorks Stacks, e alguns de seus recursos mais sofisticados, orientando você durante o processo de criação da pilha de servidor de aplicativos no diagrama. Ele usa um modelo de desenvolvimento incremental que o AWS OpsWorks Stacks facilita o acompanhamento: configure uma pilha básica e, quando estiver funcionando corretamente, adicione componentes até chegar a uma implementação completa.

- Etapa 1: concluir os pré-requisitos mostra como configurar a demonstração.
- Etapa 2: Criar uma pilha de servidor de aplicativos simples - Chef 11 mostra como criar uma pilha mínima que consiste em um único servidor de aplicativos.
- Etapa 3: Adição de um datastore de back-end mostra como adicionar um servidor de banco de dados e conectá-lo ao servidor de aplicativos.
- Etapa 4: Dimensionar MyStack mostra como dimensionar uma pilha para lidar com aumento de carga, adicionando mais servidores de aplicativos e um load balancer para distribuir o tráfego de entrada.

Tópicos

- Etapa 1: concluir os pré-requisitos
- Etapa 2: Criar uma pilha de servidor de aplicativos simples - Chef 11
- Etapa 3: Adição de um datastore de back-end
- Etapa 4: Dimensionar MyStack
- Etapa 5: Excluir MyStack

Etapa 1: concluir os pré-requisitos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Conclua as seguintes etapas de configuração antes de iniciar a demonstração. Essas etapas de configuração incluem se inscrever em uma AWS conta, criar um usuário administrativo e atribuir permissões de acesso ao AWS OpsWorks Stacks.

Se você já concluiu qualquer um das Começando a usar o AWS OpsWorks Stacks demonstrações, já cumpriu os pré-requisitos para esta demonstração e pode ir direto para Etapa 2: Criar uma pilha de servidor de aplicativos simples - Chef 11.

Tópicos

- Inscreva-se para um Conta da AWS
- Criar um usuário com acesso administrativo
- Atribua permissões de acesso de serviço para seu usuário do

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra a https://portal.aws.amazon.com/billing/inscrição.
2. Siga as instruções online.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática

recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, você pode visualizar a atividade atual da sua conta e gerenciar sua conta acessando https://aws.amazon.com/e escolhendo Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, insira a senha.

   Para obter ajuda ao fazer login usando o usuário-raiz, consulte Fazer login como usuário-raiz no Guia do usuário do Início de Sessão da AWS .

2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

   Para obter instruções, consulte Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz (console) no Guia do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilita o Centro de Identidade do IAM.

   Para obter instruções, consulte Habilitar o AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center .

2. No Centro de Identidade do IAM, conceda o acesso administrativo a um usuário.

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com o seu usuário do Centro de Identidade do IAM, use o URL de login enviado ao seu endereço de e-mail quando o usuário do Centro de Identidade do IAM foi criado.

  Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

   Para obter instruções, consulte Criar um conjunto de permissões no Guia do usuário do AWS IAM Identity Center .
2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Adicionar grupos no Guia do usuário do AWS IAM Identity Center .

Atribua permissões de acesso de serviço para seu usuário do

Ative o acesso ao serviço AWS OpsWorks Stacks (e aos serviços relacionados dos quais o AWS OpsWorks Stacks depende) adicionando as `AmazonS3FullAccess` permissões `AWSOpsWorks_FullAccess` e à sua função ou usuário.

Para obter mais informações sobre adicionar permissões, consulte Adicionar permissões de identidade do IAM (console).

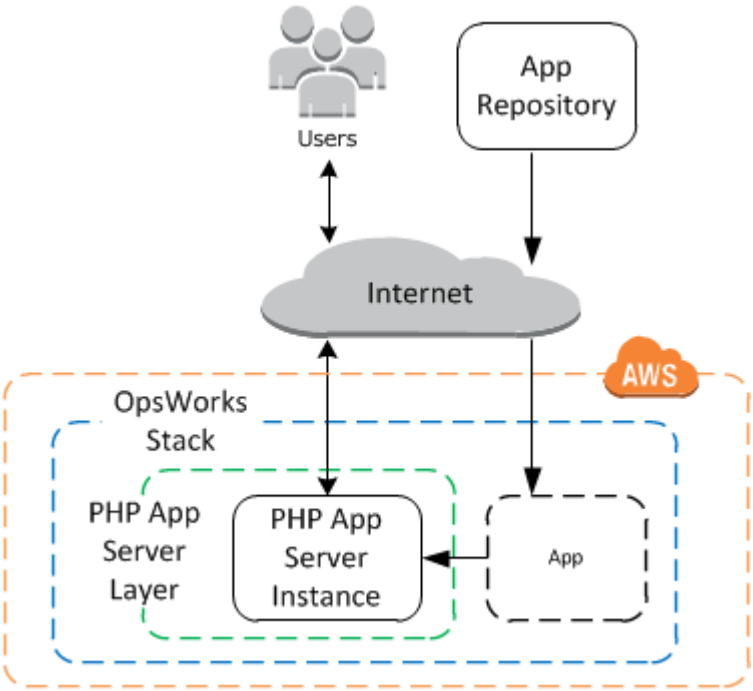
Agora você concluiu todas as etapas de configuração e pode iniciar esta demonstração.

Etapa 2: Criar uma pilha de servidor de aplicativos simples - Chef 11

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Uma pilha de servidor de aplicativo básico consiste em uma instância de servidor de aplicativo única com um endereço IP público para receber solicitações de usuário. Código de aplicativo e quaisquer arquivos relacionados são armazenados em um repositório separado e implantados de lá para o servidor. O diagrama a seguir ilustra essa pilha.



A pilha tem os seguintes componentes:

- Uma camada, que representa um grupo de instâncias e especifica como elas são configuradas.

  A camada neste exemplo representa um grupo de instâncias do PHP App Server.
- Uma instância, que representa uma EC2 instância da Amazon.

  Nesse caso, a instância é configurada para executar um servidor de aplicativo PHP. As camadas podem ter qualquer número de instâncias. AWS OpsWorks O Stacks também oferece suporte a vários outros servidores de aplicativos. Para obter mais informações, consulte Layers de servidor de aplicativos.
- Um aplicativo, que contém as informações necessárias para instalar um aplicativo no servidor do aplicativo.

O código é armazenado em um repositório remoto, como repositório Git ou um bucket Amazon S3.

As seções a seguir descrevem como usar o console AWS OpsWorks Stacks para criar a pilha e implantar o aplicativo. Você também pode usar um AWS CloudFormation modelo para provisionar uma pilha. Para ver um exemplo de modelo que provisiona a pilha descrita neste tópico, consulte AWS OpsWorks Snippets.

Tópicos

- Etapa 2.1: Criar uma pilha - Chef 11
- Etapa 2.2: adicionar uma camada de PHP App Server – Chef 11
- Etapa 2.3: adicionar uma instância à camada do PHP App Server – Chef 11
- Etapa 2.4: Criar e implantar um aplicativo - Chef 11

Etapa 2.1: Criar uma pilha - Chef 11

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você inicia um projeto AWS OpsWorks Stacks criando uma pilha, que atua como um contêiner para suas instâncias e outros recursos. A configuração da pilha especifica algumas configurações básicas, como a região da AWS e o sistema operacional padrão, que são compartilhados por todas as instâncias da pilha.

> **Note**
>
> Essa página ajuda você a criar pilhas do Chef 11. Para obter informações sobre come criar pilhas do Chef 12, consulte Criar uma pilha.

Essa página ajuda você a criar pilhas no Chef 11.

Para criar um nova pilha

1. Adicionar uma pilha

   Faça login no console do AWS OpsWorks Stacks. Se a conta não tiver pilhas existentes, você verá a OpsWorks página Bem-vindo à AWS; clique em Adicionar sua primeira pilha. Caso contrário, você verá o painel de AWS OpsWorks pilhas, que lista as pilhas da sua conta; clique em Adicionar pilha.

2. Configurar a pilha

   Na página Add Stack, escolha Chef 11 stack e, em seguida, especifique as seguintes configurações:

   Nome da pilha

   Insira um nome para sua pilha, que pode conter caracteres alfanuméricos (a–z, A–Z e 0–9) e hífens (-). A pilha de exemplo para esta demonstração se chama **`MyStack`**.

   Região

   Selecione Oeste dos EUA (Oregon) como a região da pilha.

   Aceite os valores padrão das outras configurações e, em seguida, clique em Add Stack. Para obter mais informações sobre as diversas configurações de pilha, consulte Criar uma nova pilha.

Etapa 2.2: adicionar uma camada de PHP App Server – Chef 11

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Apesar de uma pilha ser basicamente um contêiner de instâncias, você não adiciona instâncias diretamente à pilha. Você adiciona uma camada, que representa um grupo de instâncias relacionadas, e então adiciona instâncias à camada.

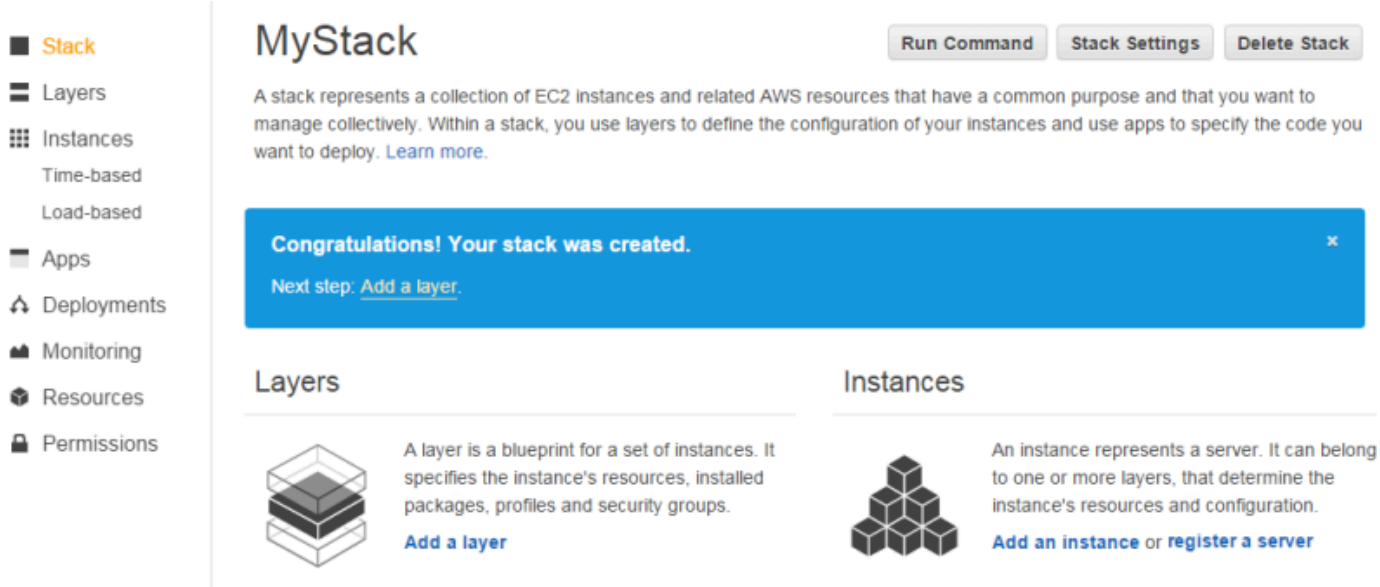
Uma camada é basicamente um modelo que o AWS OpsWorks Stacks usa para criar um conjunto de EC2 instâncias da Amazon com a mesma configuração. Você adiciona uma camada à pilha para cada grupo de instâncias relacionadas. AWS OpsWorks As pilhas incluem um conjunto de camadas integradas para representar grupos de instâncias executando pacotes de software padrão, como um servidor de banco de dados MySQL ou um servidor de aplicativos PHP. Além disso, você pode criar camadas personalizadas parcial ou completamente para se adequar a seus requisitos específicos. Para obter mais informações, consulte Personalizando pilhas AWS OpsWorks.

MyStack tem uma camada, a camada integrada do PHP App Server, que representa um grupo de instâncias que funcionam como servidores de aplicativos PHP. Para obter mais informações, incluindo descrições das camadas integradas, consulte Camadas.

Para adicionar uma camada do PHP App Server ao MyStack

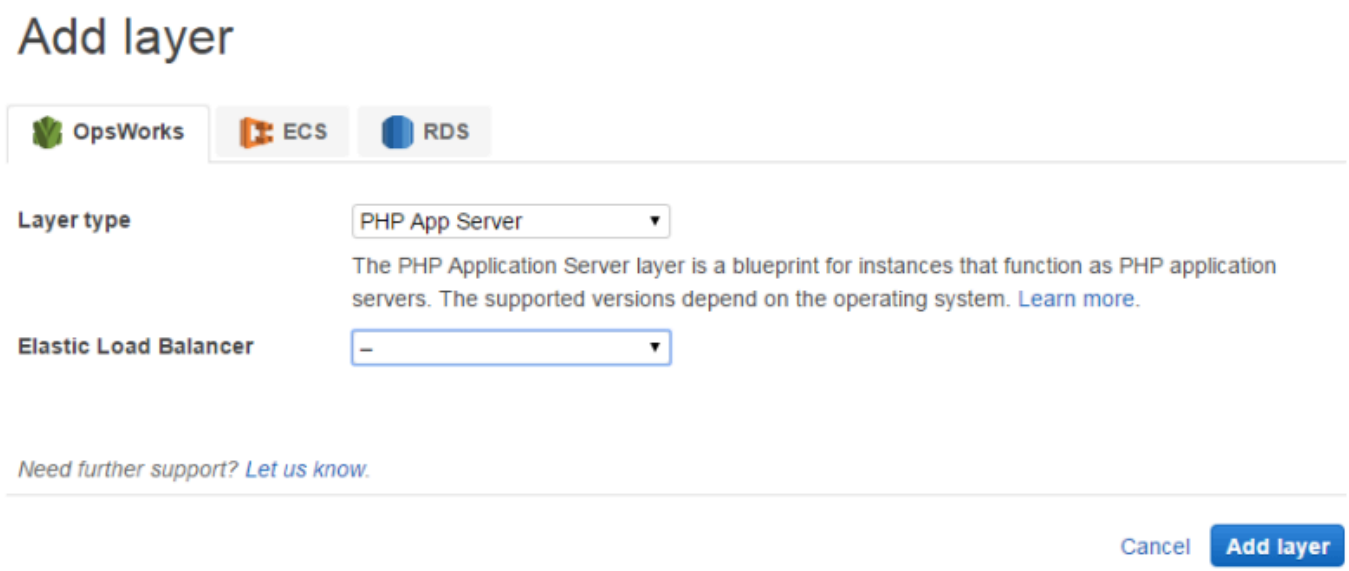
1. Abra a página Adicionar camada

   Depois de terminar de criar a pilha, o AWS OpsWorks Stacks exibe a página Stack. Clique em Add a layer para adicionar sua primeira layer.



2. Especifique um tipo de camada e configure-a

   Na caixa Tipo de camada, selecione PHP App Server, aceite a configuração padrão Balanceador de carga elástica e clique em Adicionar camada. Depois de criar a camada, você pode especificar outros atributos, como a configuração de volume de EBS editando a camada.



Etapa 2.3: adicionar uma instância à camada do PHP App Server – Chef 11

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Uma instância do AWS OpsWorks Stacks representa uma instância específica da Amazon EC2 :

- A configuração da instância especifica alguns princípios básicos, como o EC2operating sistema e o tamanho da Amazon; ela é executada, mas não faz muita coisa.
- A camada da instância adiciona funcionalidade à instância determinando quais pacotes serão instalados, se a instância tem um endereço IP elástico, e assim por diante.

AWS OpsWorks O Stacks instala um agente em cada instância que interage com o serviço. Para adicionar a funcionalidade de uma camada a uma instância, o AWS OpsWorks Stacks direciona o agente a executar pequenos aplicativos chamados receitas do Chef, que podem instalar aplicativos e pacotes, criar arquivos de configuração e assim por diante. AWS OpsWorks O Stacks executa receitas em pontos-chave do ciclo de vida da instância. Por exemplo, OpsWorks executa receitas de instalação após a conclusão da inicialização da instância para lidar com tarefas como instalação de software e executa receitas de implantação quando você implanta um aplicativo para instalar o código e os arquivos relacionados.

> **Note**
>
> Se você está curioso sobre como as receitas funcionam, todas as receitas integradas do AWS OpsWorks Stacks estão em um GitHub repositório público: OpsWorks livros de receitas. Você também pode criar suas próprias receitas e deixar que o AWS OpsWorks Stacks as executem, conforme descrito posteriormente.

Para adicionar um servidor de aplicativos PHP MyStack, adicione uma instância à camada do PHP App Server que você criou na etapa anterior.

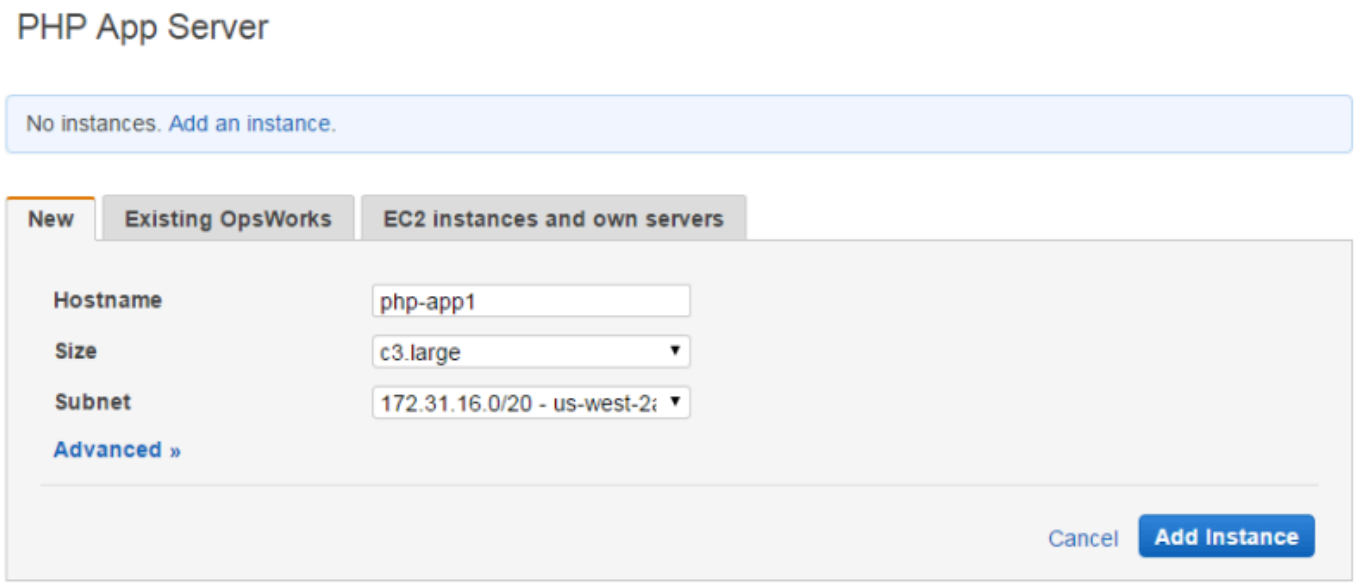
Para adicionar uma instância à camada do PHP App Server

1. Abra Adicionar uma instância

   Depois de terminar de adicionar a camada, AWS OpsWorks Stacks exibe a página Camadas. No painel de navegação, clique em Instâncias e sob PHP App Server, clique em Adicionar uma instância.

2. Configure a instância

   Cada instância tem um nome de host padrão que é gerado para você pelo AWS OpsWorks Stacks. Neste exemplo, AWS OpsWorks Stacks simplesmente adiciona um número ao nome curto da camada. Você pode configurar cada instância separadamente, incluindo sobreposição de algumas configurações padrão que você especificou ao criar a pilha, como a Zona de disponibilidade ou sistema operacional. Durante essa explicação, basta aceitar as configurações padrão e clicar em Add Instance para adicionar a instância à camada. Para obter mais informações, consulte Instâncias.

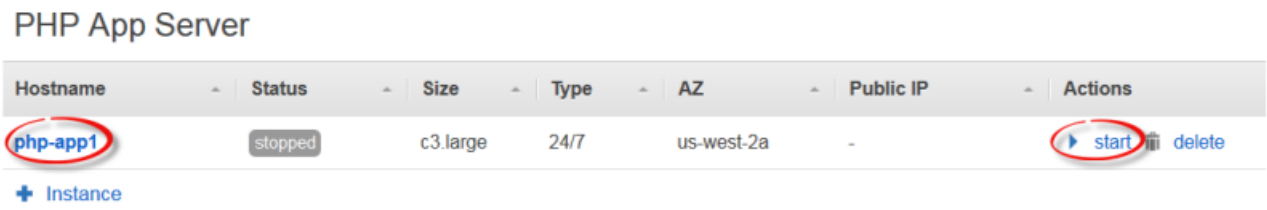


3. Inicie a instância

   Até agora, você especificou apenas a configuração da instância. Você precisa iniciar uma instância para criar uma EC2 instância da Amazon em execução. AWS OpsWorks Em seguida, o Stacks usa as configurações para iniciar uma EC2 instância da Amazon na zona de disponibilidade especificada. Os detalhes de como você inicia uma instância dependem do tipo de escalabilidade da instância. Na etapa anterior, você criou uma instância com o tipo de escalabilidade padrão, 24/7, que deve ser iniciado manualmente e então executado até que

seja interrompido manualmente. Você também pode criar tipos de escalabilidade baseados em tempo e em carga, cujos AWS OpsWorks Stacks iniciam e param automaticamente com base em uma programação ou na carga atual. Para obter mais informações, consulte Gerenciamento de carga com instâncias baseadas em tempo e carga.

Acesse php-app1 em PHP App Server e clique em Iniciar na coluna Ações da linha para iniciar a instância.

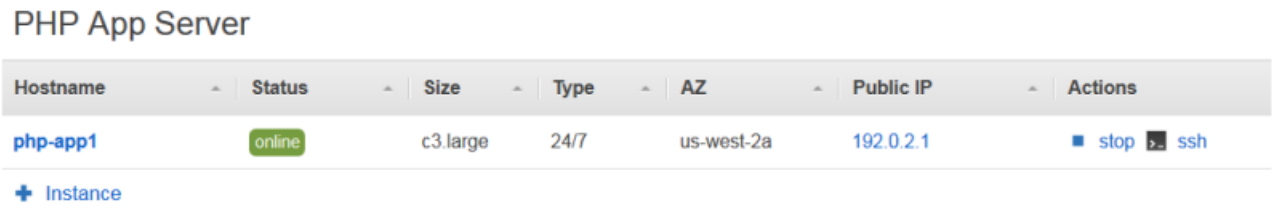


4. Monitore o status da instância durante o startup

Normalmente, são necessários alguns minutos para inicializar a EC2 instância da Amazon e instalar os pacotes. Conforme o startup progride, o campo Status da instância exibe as séries dos valores a seguir:

1. solicitado - AWS OpsWorks Stacks chamou o EC2 serviço da Amazon para criar a EC2 instância da Amazon.
2. pendente - O AWS OpsWorks Stacks está aguardando o início da EC2 instância da Amazon.
3. inicialização - A EC2 instância da Amazon está inicializando.
4. running_setup - O agente AWS OpsWorks Stacks está executando as receitas de configuração da camada, que lidam com tarefas como configuração e instalação de pacotes, e as receitas de implantação, que implantam qualquer aplicativo na instância.
5. online - A instância está pronta para uso.

Depois de php-app1 ficar online, a página Instances deve aparecer da seguinte forma:



A página começa com um pequeno resumo de todas suas instâncias de pilha. Agora mesmo, ela mostra uma instância online. Na coluna do php-app1 Actions, observe que stop, que interrompe a instância, substituiu start e delete.

Etapa 2.4: Criar e implantar um aplicativo - Chef 11

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para torná-lo MyStack mais útil, você precisa implantar um aplicativo na instância do PHP App Server. Você armazena um código de aplicativo e quaisquer arquivos relacionados em um repositório, como Git. Você precisa realizar duas etapas para obter esses arquivos em seus servidores de aplicativo:

> **Note**
>
> O procedimento descrito nesta seção se aplica às pilhas do Chef 11. Para obter mais informações sobre como adicionar aplicativos a camadas nas pilhas do Chef 12, consulte Adição de aplicativos.

1. Crie um aplicativo.

   Um aplicativo contém as informações que o AWS OpsWorks Stacks precisa para baixar o código e os arquivos relacionados do repositório. Você também pode especificar informações adicionais, como o domínio do aplicativo.

2. Implantar o aplicativo para seus servidores de aplicativo.

   Quando você implanta um aplicativo, o AWS OpsWorks Stacks aciona um evento de ciclo de vida do Deploy. Em seguida, o agente roda o Implantar receitas da instância, que baixa os arquivos

para o diretório apropriado juntamente com as tarefas relacionadas, como configuração do servidor, reinicialização do serviço e assim por diante.

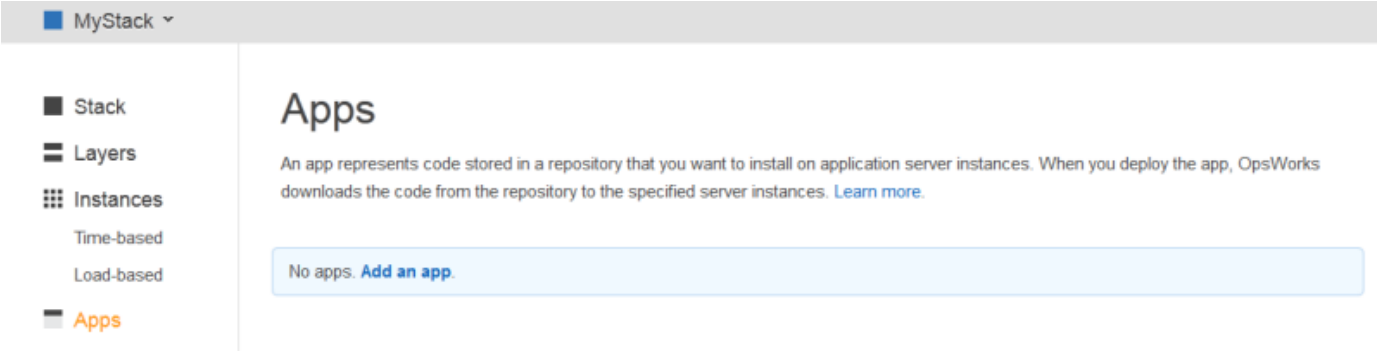
> **Note**
>
> Quando você cria uma nova instância, o AWS OpsWorks Stacks implanta automaticamente todos os aplicativos existentes na instância. No entanto, quando você cria um novo aplicativo ou atualiza um existente, você deve implantar manualmente o aplicativo ou atualizar a todas instâncias existentes.

Essa etapa mostra como implantar manualmente um aplicativo de exemplo de um repositório de Git público a um servidor de aplicativo. Se você quiser examinar o aplicativo, acesse https://github.com/amazonwebservices/opsworks-demo-php-simple-app. O aplicativo usado neste exemplo está na ramificação version1. AWS OpsWorks O Stacks também oferece suporte a vários outros tipos de repositórios. Para obter mais informações, consulte Origem do aplicativo.

Para criar e implantar um aplicativo

1. Abra a página Aplicativos

   No painel de navegação, clique em Apps e, na página Apps, clique em Add an app.



2. Configure o Aplicativo

   Na página App, especifique os seguintes valores:

   Name

   O nome do aplicativo, que o AWS OpsWorks Stacks usa para fins de exibição. O aplicativo de exemplo é chamado**`SimplePHPApp`**. AWS OpsWorks Stacks também gera um nome

curto — simplephpapp para este exemplo — que é usado internamente e pelas receitas do Deploy, conforme descrito posteriormente.

Tipo

O tipo do aplicativo, que determina onde implantar o aplicativo. O exemplo usa PHP, que implanta o aplicativo a instâncias do PHP App Server.

Tipo de fonte de dados

Um servidor de banco de dados associado. Por hora, selecione None; apresentaremos servidores de banco de dados em Etapa 3: Adição de um datastore de back-end.

Tipo de repositório

O tipo de repositório do aplicativo. O aplicativo de exemplo é armazenado em um repositório Git.

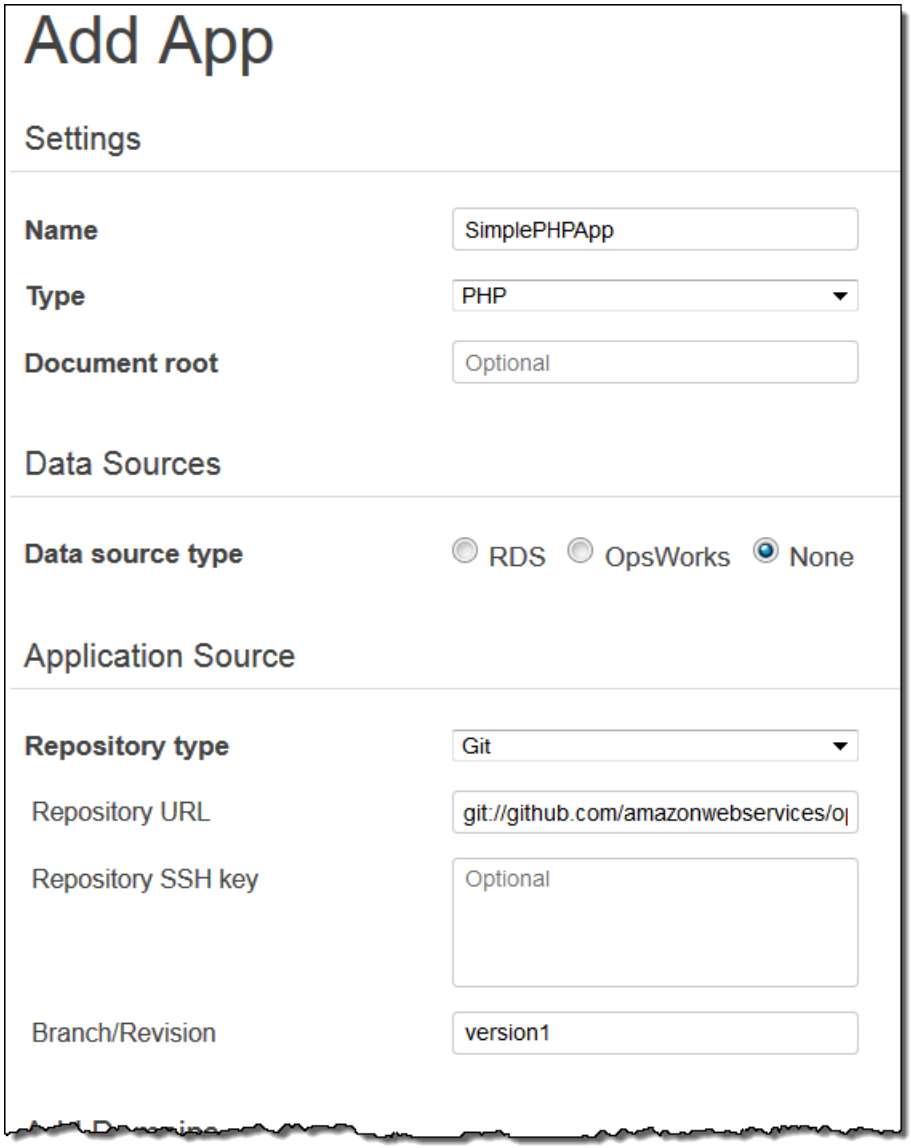
URL de repositório

URL de repositório do aplicativo. O URL do exemplo é: **`git://github.com/awslabs/opsworks-demo-php-simple-app.git`**

Ramificação/Revisão

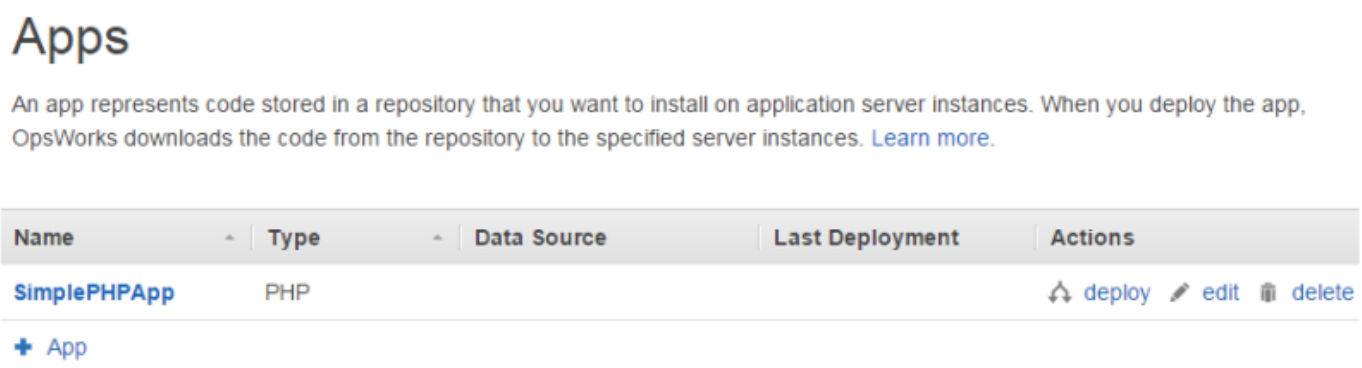
A ramificação do aplicativo ou versão. Esta parte da demonstração usa a ramificação **`version1`** .

Mantenha os valores padrão para as configurações restantes e clique em Add App. Para obter mais informações, consulte Adição de aplicativos.



3. Abra a página de implantação

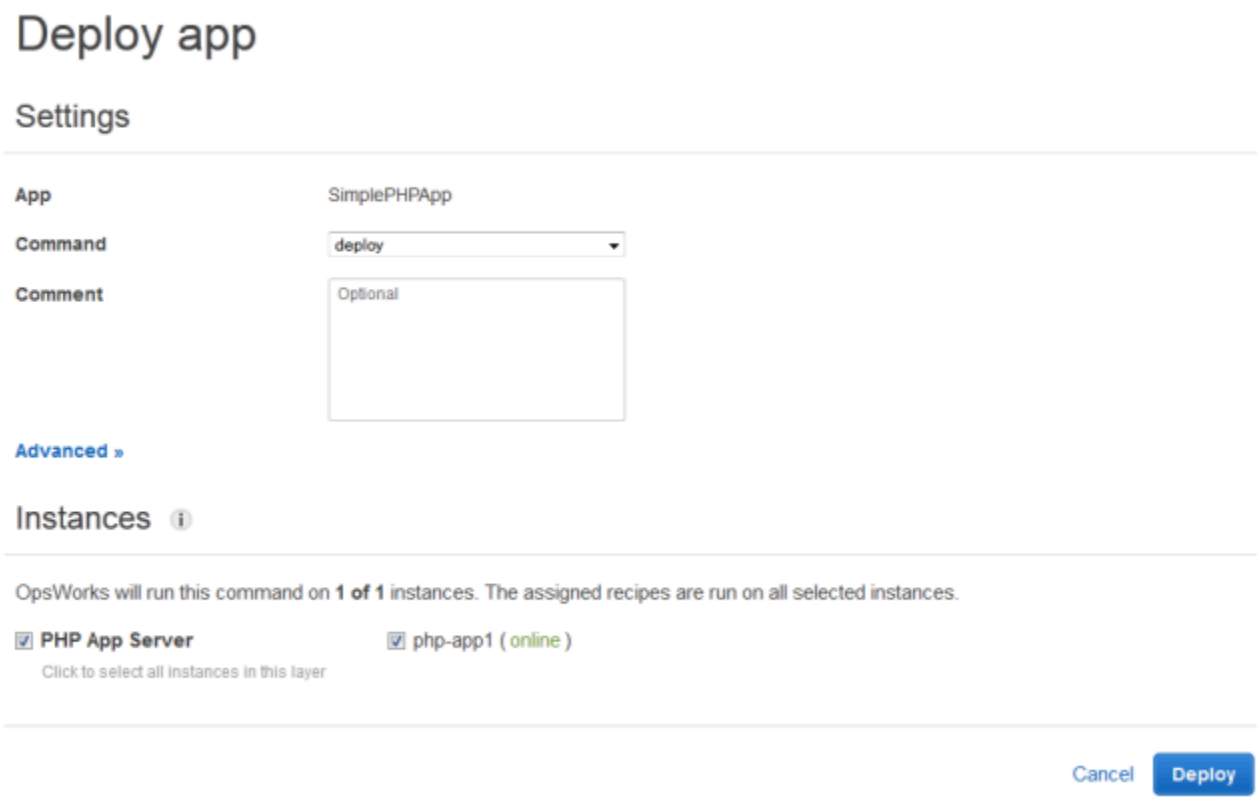
   Para instalar o código no servidor, você deve implantar o aplicativo. Para fazer isso, clique em implantar na coluna PHPApp Ações simples.



4. Implante o aplicativo

   Quando você implementa um aplicativo, o agente executa o Implantar receitas na instância do PHP App Server, que baixa e configura o aplicativo.

   Command já deve estar definido como deploy. Mantenha os valores padrão para as outras configurações e clique em Deploy para implantar o aplicativo.

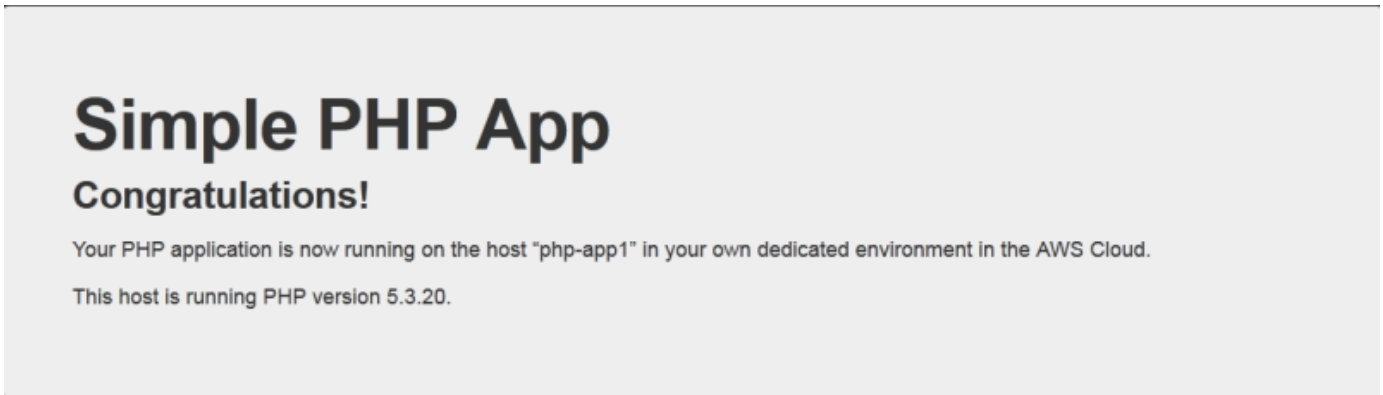


   Quando a implantação estiver completa, a página Deployment exibirá um Status de Successful, e php-app1 terá um marcador verde próximo a ele.

5. Execute de forma simples PHPApp

PHPApp O Simple agora está instalado e pronto para ser usado. Para executá-lo, clique em Instances, no painel de navegação, e acesse a página Instances. Depois, clique no endereço IP público da instância php-app1.

Você deve ver a página como a seguinte em seu navegador.

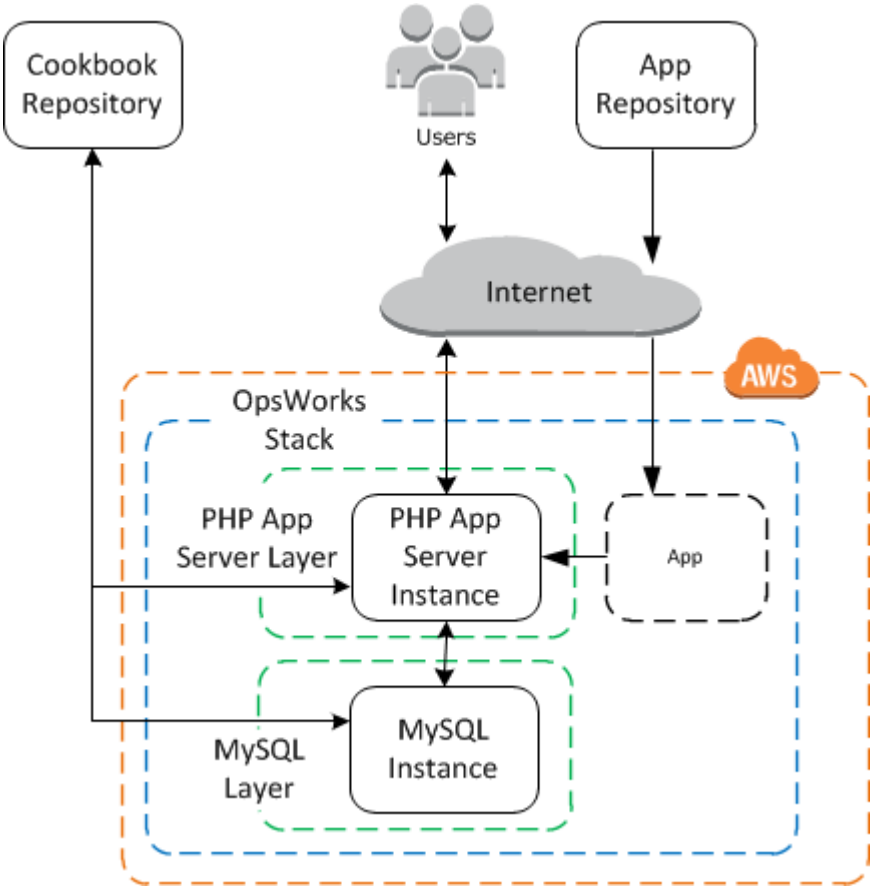


> **Note**
>
> Essa apresentação pressupõem que você irá para a próxima seção e finalmente completará a apresentação inteira em uma sessão. Se preferir, você pode parar a qualquer momento e continuar mais tarde entrando no AWS OpsWorks Stacks e abrindo o stack. No entanto, você é cobrado por qualquer recurso da AWS que utilizar, como instâncias online. Para evitar cobranças desnecessárias, você pode interromper sua instância, o que encerra a EC2 instância correspondente. Você pode iniciar as instâncias novamente quando estiver pronto para continuar.

Etapa 3: Adição de um datastore de back-end

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Etapa 2.1: Criar uma pilha - Chef 11 mostrou como criar uma pilha que foi usada por um aplicativo PHP. No entanto, aquele aplicativo era muito simples e fez pouco mais do que exibir um texto estático. Aplicativos de produção normalmente usam um datastore de back-end, resultando em uma configuração de pilha semelhante à da ilustração a seguir.



Esta seção mostra como estender MyStack para incluir um servidor de banco de dados MySQL back-end. Você precisa fazer mais do que simplesmente adicionar um servidor MySQL à pilha. Você também precisa configurar o aplicativo para se comunicar adequadamente com o servidor do banco

de dados. AWS OpsWorks O Stacks não faz isso por você; você precisará implementar algumas receitas personalizadas para lidar com essa tarefa.

Tópicos

- Etapa 3.1: Adição de um banco de dados de back-end
- Etapa 3.2: Atualização simples PHPApp
- Uma breve digressão: atributos de livros de receitas, receitas e pilhas AWS OpsWorks
- Etapa 3.3: Adicionar os livros de receitas personalizados ao MyStack
- Etapa 3.4: Execução de receitas
- Etapa 3.5: Implantar SimplePHPApp, versão 2
- Etapa 3.6: Executar o Simple PHPApp

Etapa 3.1: Adição de um banco de dados de back-end

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A nova versão do Simple PHPApp armazena seus dados em um banco de dados de back-end. AWS OpsWorks O Stacks oferece suporte a dois tipos de servidores de banco de dados:

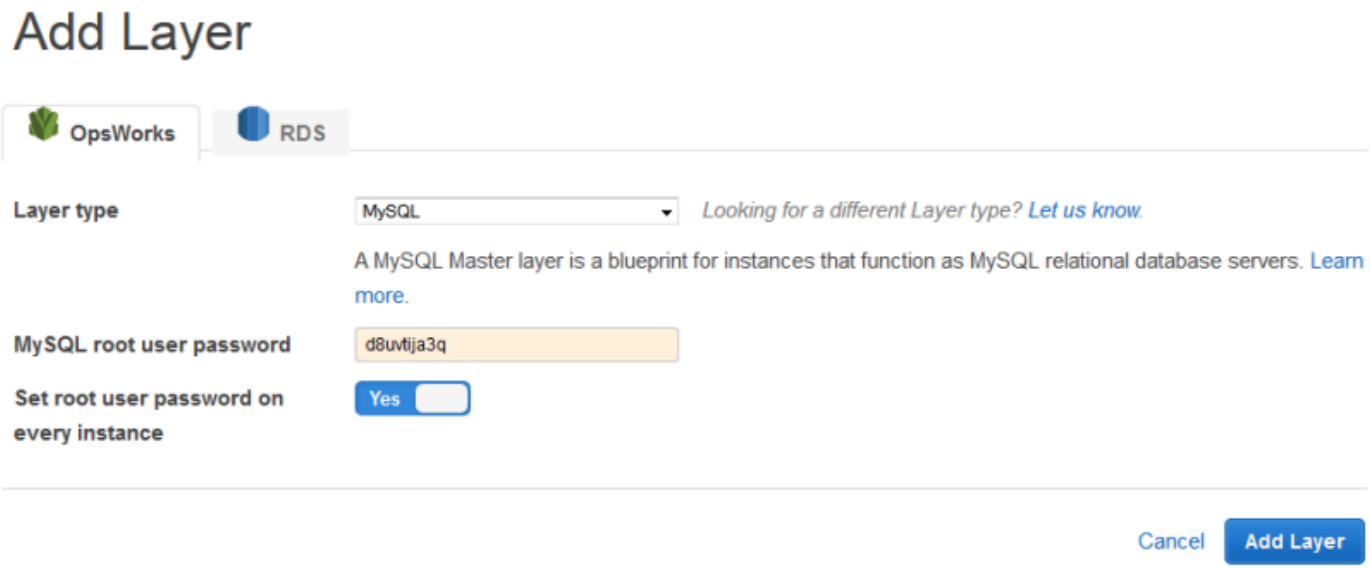
- A camada MySQL AWS OpsWorks Stacks é um modelo para criar EC2 instâncias da Amazon que hospedam um banco de dados MySQL mestre.
- A camada de serviço do Amazon RDS fornece uma forma de incorporar uma instância Amazon RDS em uma pilha.

Você também pode usar outros bancos de dados, como o Amazon DynamoDB ou criar uma camada personalizada para fornecer suporte a bancos de dados, como por exemplo, o MongoDB. Para obter mais informações, consulte the section called “Uso de um armazenamento de dados de back-end”.

Este exemplo usa uma camada do MySQL.

Para adicionar uma camada MySQL ao MyStack

1. Na página Layers, clique em + Layer.
2. Na página Add Layer, para Layer type, selecione MySQL, aceite as configurações padrão e clique em Add Layer.



Para adicionar uma instância à camada MySQL

1. Na linha MySQL da página Layers, clique em Add an instance.
2. Na página Instances, sob MySQL, clique em Add an instance.
3. Aceite os valores padrão e clique em Add instance, mas não a inicie ainda.

> **Note**
>
> AWS OpsWorks O Stacks cria automaticamente um banco de dados chamado usando o nome abreviado do aplicativo, simplephpapp neste exemplo. Você precisará desse nome se deseja usar as receitas do Chef para interagir com o banco de dados.

Etapa 3.2: Atualização simples PHPApp

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para começar, você precisa de uma nova versão do Simple PHPApp que use um armazenamento de dados de back-end. No AWS OpsWorks Stacks, é fácil atualizar um aplicativo. Se você usa um repositório Git ou Subversion, pode ter uma ramificação de repositório separada para cada versão do aplicativo. O aplicativo de exemplo armazena uma versão do aplicativo que usa um banco de dados de back-end na ramificação do repositório Git versão 2. Você só precisa atualizar a configuração do aplicativo para especificar a nova ramificação e reimplantar o aplicativo.

Para atualizar o Simple PHPApp

1. Abra a página de edição do aplicativo

   No painel de navegação, clique em Aplicativos e, em seguida, clique em editar na coluna Ações da PHPApp linha Simples.

2. Atualize a configuração do aplicativo

   Altere as seguintes configurações.

   Ramificação/Revisão

   Essa configuração indica a ramificação do repositório do aplicativo. A primeira versão do Simple PHPApp não se conectou a um banco de dados. Para usar uma versão do aplicativo habilitada para bancos de dados, defina esse valor como **`version2`**.

   Document root

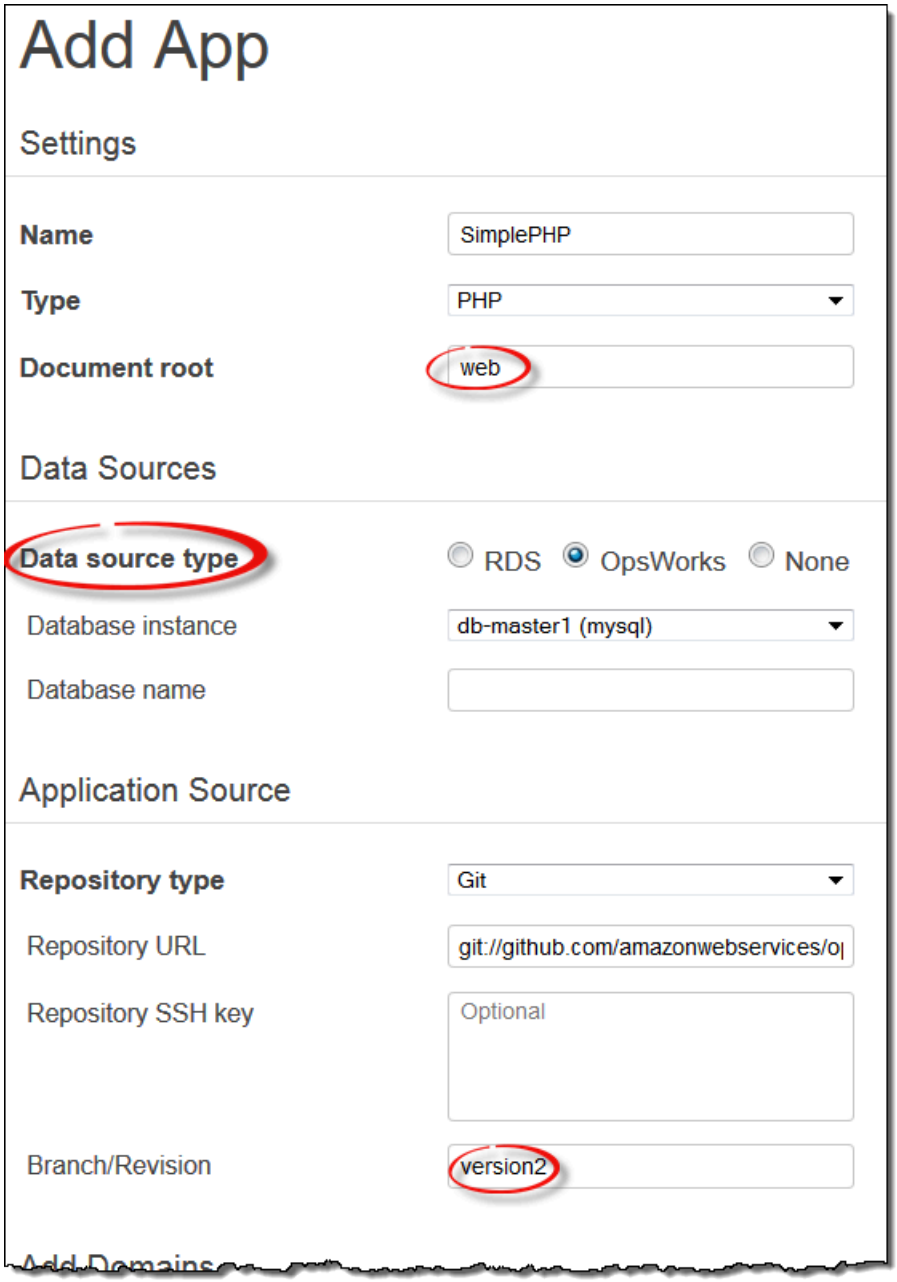
   Essa configuração especifica a pasta raiz do aplicativo. A primeira versão do Simple PHPApp usava a configuração padrão, que é instalada `index.php` na pasta raiz padrão do servidor (`/srv/www`para aplicativos PHP). Se você especificar uma subpasta aqui — apenas o nome,

sem “/' inicial —AWS OpsWorks Stacks a anexará ao caminho padrão da pasta. A versão 2 do Simple PHPApp deve entrar`/srv/www/web`, então defina a raiz do documento como**`web`**.

Tipo de fonte de dados

Essa configuração associa um servidor de banco de dados com o aplicativo. O exemplo usa a instância do MySQL que você criou na etapa anterior, então defina o tipo de fonte de dados como OpsWorks e a instância do banco de dados como a instância que você criou na etapa anterior, db-master1 (mysql). Deixe o nome do banco de dados vazio; o AWS OpsWorks Stacks criará um banco de dados no servidor com o nome abreviado do aplicativo, simplephpapp.

Em seguida, clique em Save para salvar a nova configuração.



3. Inicie a instância MySQL.

Depois de atualizar um aplicativo, o AWS OpsWorks Stacks implanta automaticamente a nova versão do aplicativo em qualquer nova instância do servidor de aplicativos quando você as inicia. No entanto, o AWS OpsWorks Stacks não implanta automaticamente a nova versão do aplicativo nas instâncias de servidor existentes; você deve fazer isso manualmente, conforme descrito emEtapa 2.4: Criar e implantar um aplicativo - Chef 11. Você poderia implantar o Simple atualizado PHPApp agora, mas neste exemplo, é melhor esperar um pouco.

Uma breve digressão: atributos de livros de receitas, receitas e pilhas AWS OpsWorks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você agora tem servidores de aplicativos e de banco de dados, mas eles ainda não estão prontos para uso. Você ainda precisa configurar o banco de dados e definir as configurações de conexão do aplicativo. AWS OpsWorks O Stacks não executa essas tarefas automaticamente, mas oferece suporte aos livros de receitas, receitas e atributos dinâmicos do Chef. Você pode implementar duas receitas, uma para configurar o banco de dados e outra para definir as configurações de conexão do aplicativo, e fazer com que o AWS OpsWorks Stacks as execute para você.

O livro de receitas phpapp, que contém as receitas necessárias, já está implementado e pronto para ser usado; você pode ir direto para Etapa 3.3: Adicionar os livros de receitas personalizados ao MyStack, se preferir. Se você deseja saber mais, esta seção fornece algumas informações sobre livros de receitas e receitas, e descreve como elas funcionam. Para ver o livro de receitas em si, acesse o livro de receitas phpapp.

Tópicos

- Receitas e atributos
- Configuração do banco de dados
- Conexão do aplicativo ao banco de dados

Receitas e atributos

Uma receita do Chef é basicamente um aplicativo Ruby especializado que executa tarefas em uma instância, como a instalação de pacotes, criação de arquivos de configuração e execução de comandos do shell, e assim por diante. Os grupos de receitas relacionadas são organizados em livros de receitas, que também contêm arquivos de suporte, como modelos para a criação de arquivos de configuração.

AWS OpsWorks Stacks tem um conjunto de livros de receitas que suportam as camadas integradas. Você também pode criar livros de receitas personalizados com suas próprias receitas para executar tarefas personalizadas em suas instâncias. Este tópico fornece uma breve introdução a receitas e mostra como usá-las para configurar o banco de dados e as definições de conexão do aplicativo. Para obter mais informações sobre livros de receitas e receitas, consulte Livros de receitas e receitas ou Personalizando pilhas AWS OpsWorks.

As receitas geralmente dependem dos atributos do Chef para a entrada de dados:

- Alguns desses atributos são definidos pelo Chef e fornecem informações básicas sobre a instância, como o sistema operacional.
- AWS OpsWorks As pilhas definem um conjunto de atributos que contêm informações sobre a pilha, como as configurações da camada, e sobre os aplicativos implantados, como o repositório do aplicativo.

  Você pode adicionar atributos personalizados a esse conjunto atribuindo um JSON personalizado à pilha ou à implantação.
- Seus livros de receitas também podem definir atributos específicos do livro de receitas.

  Os atributos do livro de receitas phpapp são definidos em `attributes/default.rb`.

Para obter uma lista completa dos atributos do AWS OpsWorks Stacks, consulte Atributos de implantação e configuração de pilha: Linux e. Atributos de livros de receitas integrados Para obter mais informações, consulte Sobrepor atributos.

Os atributos são organizados em uma estrutura hierárquica, que pode ser representada como um objeto JSON.

Você incorpora esses dados em seu aplicativo usando a sintaxe de nó do Chef, da seguinte forma:

```
[:deploy][:simplephpapp][:database][:username]
```

O nó `deploy` tem um único nó de aplicativo, `simplephpapp`, que contém informações sobre o banco de dados do aplicativo, repositório Git, e assim por diante. Esse exemplo representa o valor do nome de usuário do banco de dados, que resulta em `root`.

Configuração do banco de dados

O Configurar receitas integrado da camada do MySQL cria automaticamente um banco de dados para o aplicativo com o nome abreviado do aplicativo e, portanto, para este exemplo você já tem um banco de dados chamado simplephpapp. No entanto, você precisa concluir a configuração e criar uma tabela para o aplicativo a fim de armazenar seus dados. Você pode criar a tabela manualmente, mas uma abordagem melhor é implementar uma receita personalizada para lidar com a tarefa e fazer com que o AWS OpsWorks Stacks a execute para você. Esta seção descreve como a receita, `dbsetup.rb`, é implementada. O procedimento para fazer com que o AWS OpsWorks Stacks execute a receita é descrito posteriormente.

Para ver a receita no repositório, acesse dbsetup.rb. O exemplo a seguir mostra o código do `dbsetup.rb`.

O `execute` é um recurso do Chef que executa um comando especificado. Neste caso, é um comando MySQL que cria uma tabela. A diretiva `not_if` assegura que o comando não será executado se a tabela especificada já existir. Para obter mais informações sobre os recursos do Chef, consulte Sobre recursos e provedores.

A receita insere valores de atributos na string de comando usando a sintaxe de nó abordada anteriormente. Por exemplo, o comando a seguir insere o nome do usuário do banco de dados.

```
#{deploy[:database][:username]}
```

Vamos destrinchar esse código um tanto enigmático:

- Para cada iteração, o `deploy` é definido como o nó do aplicativo atual e tem como resultado `[:deploy][:app_name]`. Para este exemplo, o resultado é `[:deploy][:simplephpapp]`.
- Usando os valores de atributos de implantação mostrados anteriormente, o nó inteiro tem como resultado `root`.
- Você encapsula o nó em #{ } para inseri-lo em uma string.

A maioria dos outros nós são resolvidos de maneira semelhante. A exceção é o `#{node[:phpapp][:dbtable]}`, que é definido pelo arquivo de atributos dos livros de receitas personalizados e tem como resultado o nome da tabela `urler`. O comando que é efetivamente executado na instância do MySQL é, portanto:

```
"/usr/bin/mysql
    -uroot
    -pvjud1hw5v8
    simplephpapp
    -e'CREATE TABLE urler(
       id INT UNSIGNED NOT NULL AUTO_INCREMENT,
       author VARCHAR(63) NOT NULL,
       message TEXT,
       PRIMARY KEY (id))'
"
```

Este comando cria uma tabela chamada `urler` com id, autor e campos de mensagens usando as credenciais e o nome do banco de dados dos atributos de implantação.

Conexão do aplicativo ao banco de dados

A segunda peça do quebra-cabeça é o aplicativo, que precisa de informações de conexão, como a senha do banco de dados para acessar a tabela. O Simple PHPApp efetivamente tem apenas um arquivo funcional`app.`php; tudo `index.`php o que faz é carregar`app.`php.

O `app.`php inclui o `db-connect.`php, que lida com a conexão do banco de dados, mas esse arquivo não está no repositório. Não é possível criar o `db-connect.`php com antecedência porque ele define o banco de dados com base na instância específica. Em vez disso, a receita `appsetup.rb` gera o `db-connect.`php usando os dados de conexão dos atributos de implantação.

Para ver a receita no repositório, acesse appsetup.rb. O exemplo a seguir mostra o código do `appsetup.rb`.

Assim como ocorre com o `dbsetup.rb`, o `appsetup.rb` faz a iteração dos aplicativos do nó `deploy`, apenas simplephpapp novamente. Ele executa um bloco de código com um recurso `script` e um `template`.

O recurso `script` instala o Composer: um gerenciador de dependências para aplicativos PHP. Em seguida, ele executa o comando `install` do Composer para instalar as dependências para o aplicativo de amostra no diretório raiz do aplicativo.

O recurso `template` gera `db-connect.`php e o coloca que `/srv/www/simplephpapp/current`. Observe o seguinte:

- A receita usa uma instrução condicional para especificar o proprietário do arquivo que depende do sistema operacional da instância.

- A diretiva `only_if` determina ao Chef que gere o modelo somente se o diretório especificado existir.

O recurso `template` opera em um modelo que tem essencialmente o mesmo conteúdo e estrutura do arquivo associado, mas inclui espaços reservados para diversos valores de dados. O parâmetro `source` especifica o modelo `db-connect.php.erb`, que está no diretório `templates/default` do livro de receitas phpapp, e contém o seguinte:

Quando o Chef processa o modelo, ele substitui os espaços reservados `<%= =>` pelos valores das variáveis correspondentes no recurso do modelo, que são, por sua vez, extraídos dos atributos de implantação. O arquivo gerado é, portanto:

Etapa 3.3: Adicionar os livros de receitas personalizados ao MyStack

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Armazene os livros de receitas personalizados em um repositório, assim como você faz com aplicativos. Cada pilha pode ter um repositório que contém um conjunto de livros de receitas personalizados. Em seguida, você AWS OpsWorks orienta o Stacks a instalar seus livros de receitas personalizados nas instâncias da pilha.

1. Clique em Stack no painel de navegação para ver a página da pilha atual.
2. Clique em Stack Settings e, em seguida, Edit.
3. Modifique a configuração da pilha da seguinte forma:
   - Usar livros de receitas do Chef personalizados: Sim
   - Tipo de repositório: Git
   - URL do repositório: **`git://github.com/amazonwebservices/opsworks-example-cookbooks.git`**
4. Clique em Save para atualizar a configuração da pilha.

AWS OpsWorks Em seguida, o Stacks instala o conteúdo do seu repositório de livros de receitas em todas as instâncias da pilha. Se você criar novas instâncias, o AWS OpsWorks Stacks instalará automaticamente o repositório do livro de receitas.

> **Note**
>
> Se precisar atualizar qualquer um dos seus livros de receitas ou adicionar novos livros de receitas ao repositório, você pode fazer isso sem tocar nas configurações da pilha. AWS OpsWorks O Stacks instalará automaticamente os livros de receitas atualizados em todas as novas instâncias. No entanto, o AWS OpsWorks Stacks não instala automaticamente livros de receitas atualizados nas instâncias on-line da pilha. Você deve orientar explicitamente o AWS OpsWorks Stacks a atualizar os livros de receitas executando o `Update Cookbooks` comando stack. Para obter mais informações, consulte Executa comandos de pilha.

Etapa 3.4: Execução de receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Depois de obter seus livros de receitas personalizados, você precisa executar as receitas nas instâncias apropriadas. Você pode executá-las manualmente. No entanto, as receitas geralmente precisam ser executadas em pontos previsíveis do ciclo de vida de uma instância como, por

exemplo, depois que a instância é inicializada ou quando você implanta um aplicativo. Esta seção descreve uma abordagem muito mais simples: faça com que o AWS OpsWorks Stacks os execute automaticamente para você no momento apropriado.

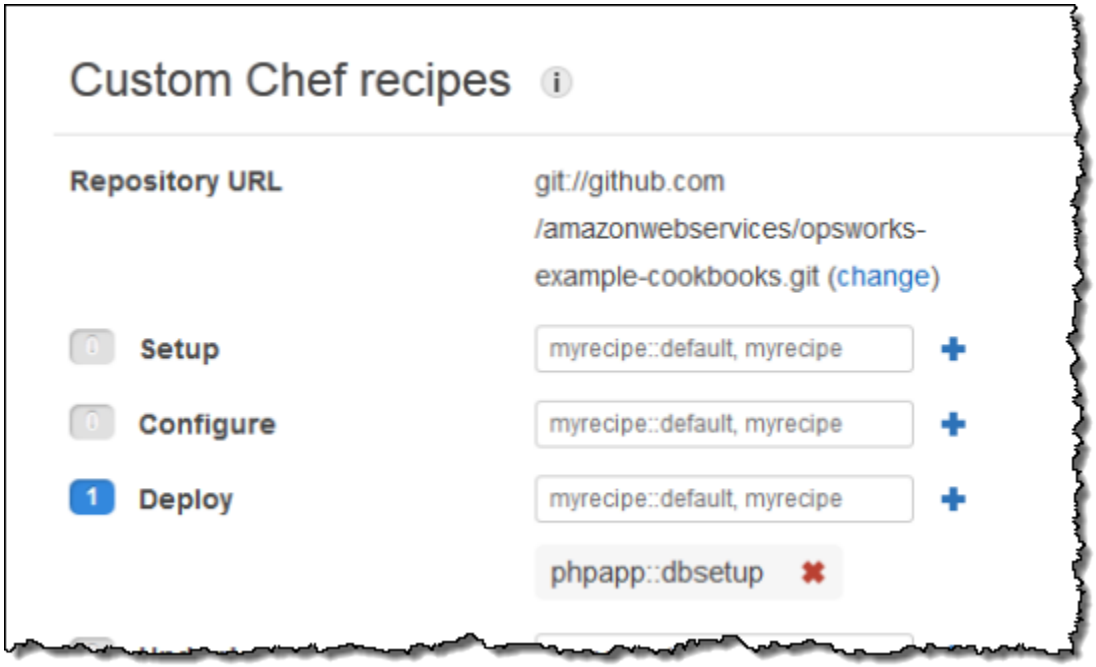
AWS OpsWorks O Stacks oferece suporte a um conjunto de eventos de ciclo de vida que simplificam a execução de receitas. Por exemplo, o evento Configurar ocorre depois que a inicialização da instância termina e o evento Implantar ocorre quando você implanta um aplicativo. Cada layer tem um conjunto de receitas integradas associadas a cada evento de ciclo de vida. Quando um evento de ciclo de vida ocorre em uma instância, o agente executa as receitas associadas em cada camada da instância. Para que o AWS OpsWorks Stacks execute uma receita personalizada automaticamente, adicione-a ao evento de ciclo de vida apropriado na camada apropriada e o agente executará a receita após a conclusão das receitas integradas.

Neste exemplo, você precisa executar duas receitas, `dbsetup.rb` na instância My SQLinstance e `appsetup.rb` na instância PHP App Server.

> **Note**
>
> Você especifica receitas no console usando o *`recipe_name`* formato*`cookbook_name`*::, onde *`recipe_name`* não inclui a extensão.rb. Por exemplo, a receita `dbsetup.rb` é especificada como phpapp::dbsetup**`phpapp::dbsetup`**.

Para atribuir receitas personalizadas a eventos de ciclo de vida

1. Na página Camadas, para MySQL, clique em Receitas e depois em Editar.
2. Na seção Custom Chef recipes (Receitas do Chef personalizadas), insira **`phpapp::dbsetup`** para Deploy (Implantar).



3. Clique no ícone + para atribuir a receita ao evento e clique em Save para salvar a nova configuração de layer.
4. Volte para a página Camadas e repita o procedimento para atribuir **`phpapp::appsetup`** ao evento Implantar da camada PHP App Server.

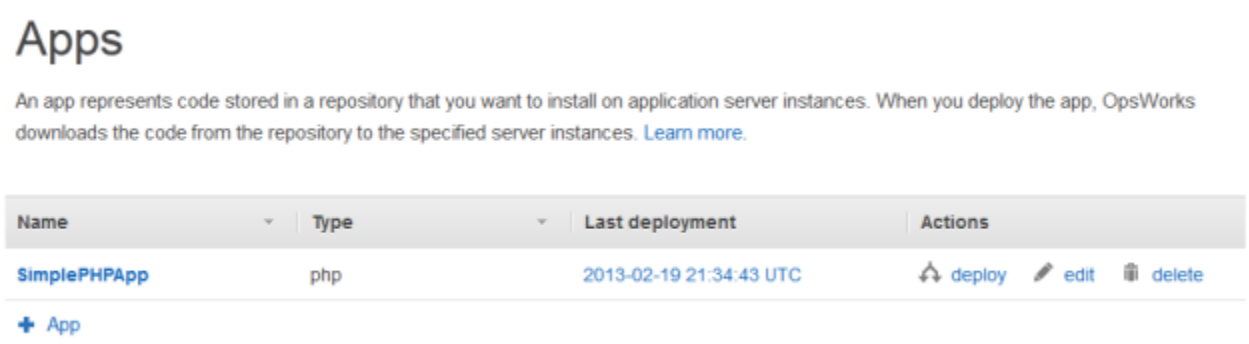
Etapa 3.5: Implantar SimplePHPApp, versão 2

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

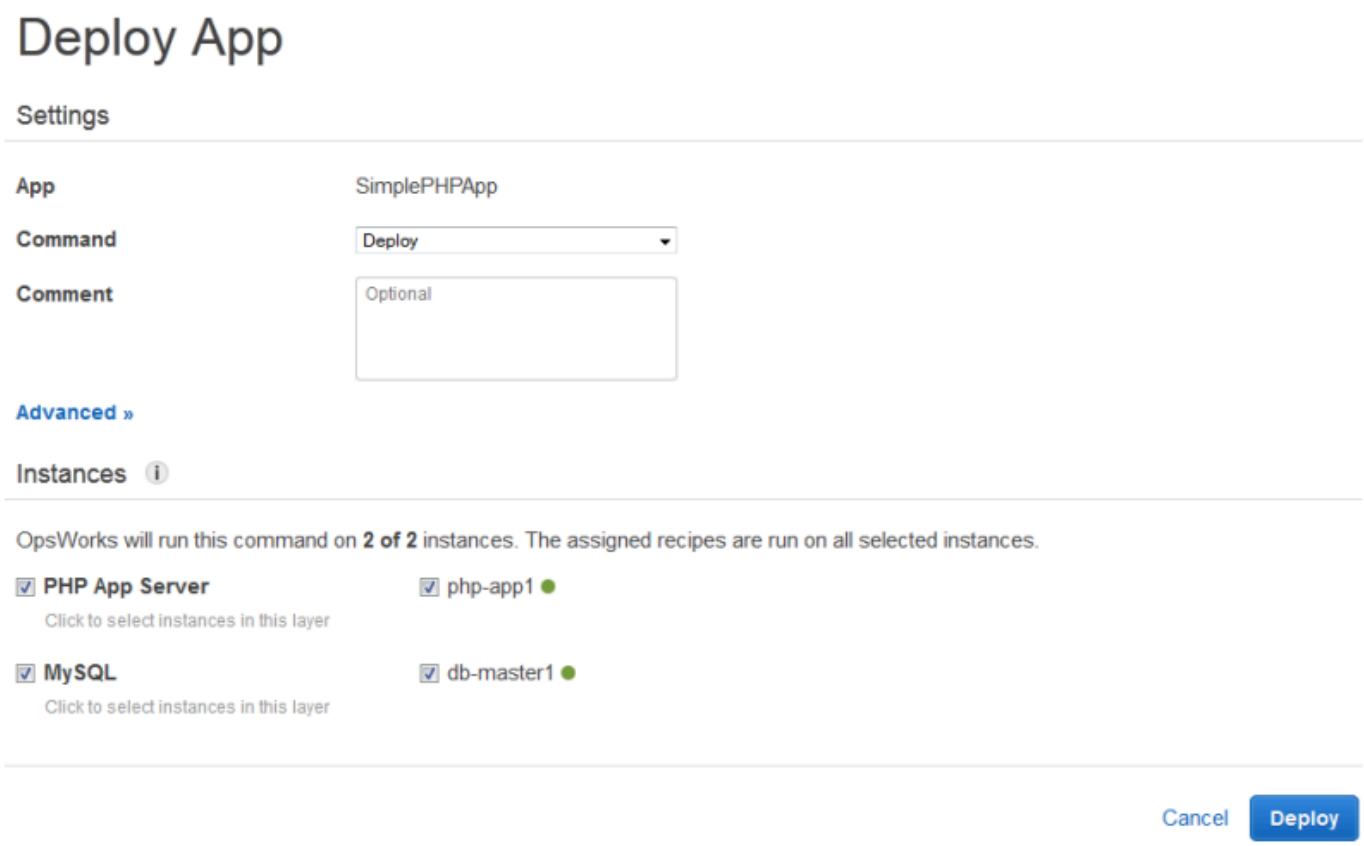
A etapa final é implantar a nova versão do SimplePHPApp.

Para implantar o Simple PHPApp

1. Na página Aplicativos, clique em implantar nas Ações do PHPApp aplicativo Simple.



2. Aceite os padrões e clique em Deploy.



Ao clicar em Deploy na página Deploy App, você aciona um evento de ciclo de vida Implantar, que notifica os agentes para executar o Implantar receitas. Por padrão, o evento é acionado em todas as instâncias da pilha. O Implantar receitas integrado implanta o aplicativo somente nas instâncias apropriadas para aquele tipo de aplicativo, neste caso, nas instâncias do PHP App Server. No entanto, muitas vezes é útil acionar o evento Implantar em outras instâncias, a fim de permitir que elas respondam à implantação do aplicativo. Neste caso, você também deve acionar o evento Implantar na instância do MySQL para configurar o banco de dados.

Observe o seguinte:

- O agente da instância do PHP App Server executa a receita integrada da camada e, em seguida, a receita `appsetup.rb`, que configura a conexão de banco de dados do aplicativo.
- O agente da instância do MySQL não instala nada, mas executa a `dbsetup.rb` para criar a tabela urler.

Quando a implantação for concluída, Status será alterado para successful na página Deployment.

Etapa 3.6: Executar o Simple PHPApp

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

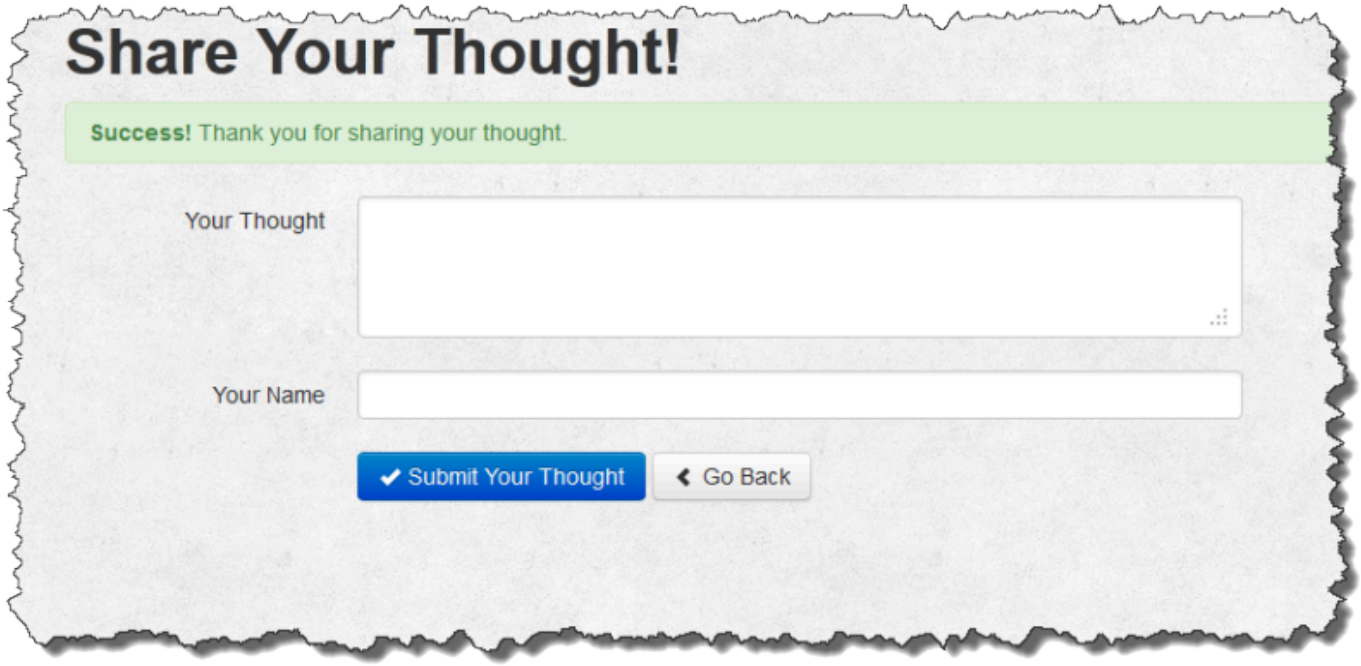
Depois que o status de implantação mudar para bem-sucedido, você poderá executar a nova PHPApp versão Simple, da seguinte maneira.

Para executar o Simple PHPApp

1. Na página Instances, clique no endereço IP público na linha php-app1.

   Você deve ver a página a seguir em seu navegador.

2. Clique em Share Your Thought (Compartilhar sua opinião) e digite algo como **`Hello world!`** para Your Thought (Sua opinião) e seu nome para Your Name (Seu nome). Em seguida, clique em Submit Your Thought para adicionar a mensagem ao banco de dados.

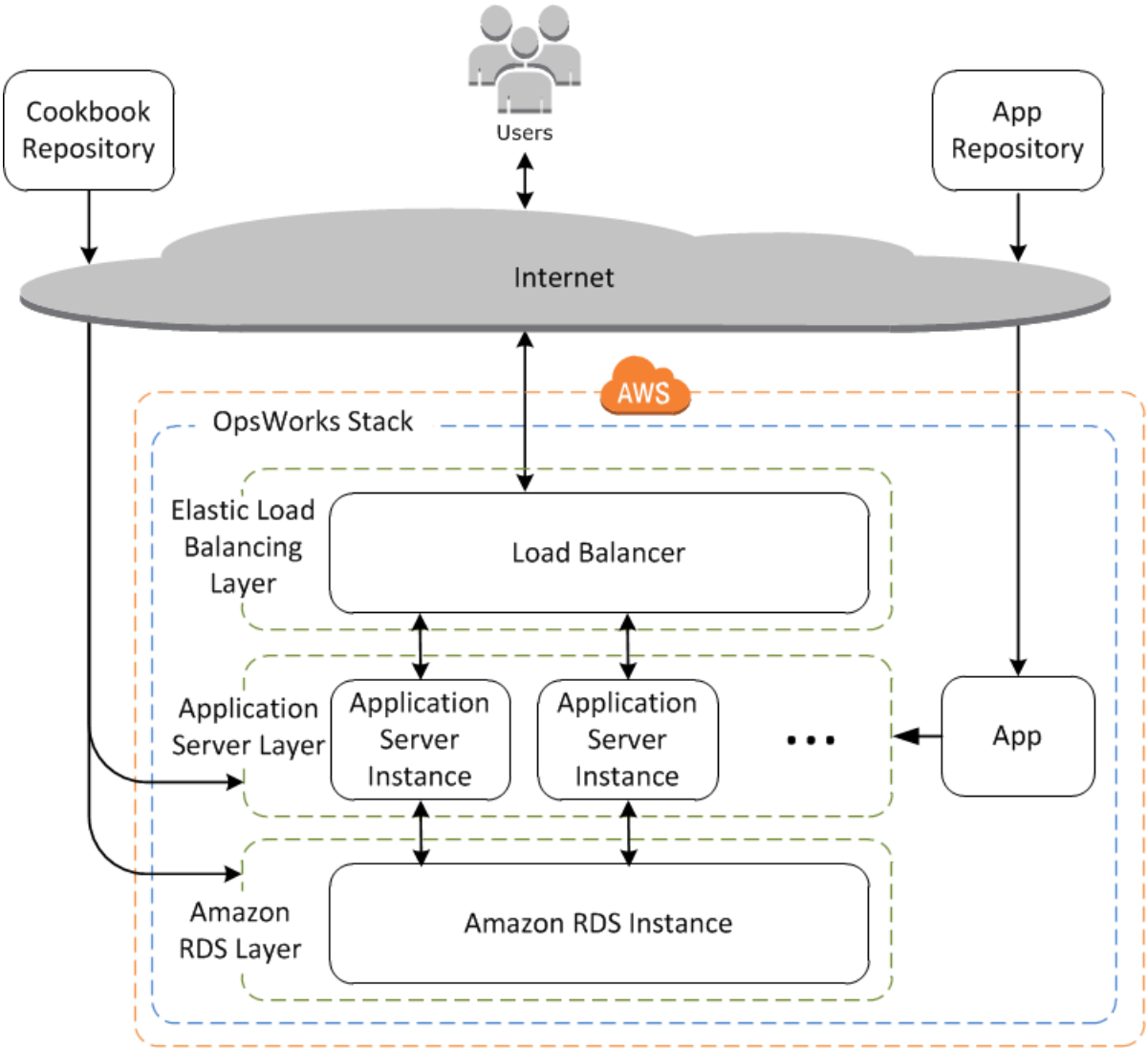


3. Clique em Go Back para visualizar todas as mensagens no banco de dados.

Etapa 4: Dimensionar MyStack

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

MyStack atualmente tem apenas um servidor de aplicativos. Uma pilha de produção provavelmente precisará de vários servidores de aplicativo para processar o tráfego de entrada e um load balancer para distribuir este tráfego de maneira uniforme para os servidores de aplicativos. A arquitetura terá a aparência a seguir:



AWS OpsWorks As pilhas facilitam a escalabilidade das pilhas. Esta seção descreve as noções básicas de como escalar uma pilha adicionando uma segunda instância do PHP App Server 24 horas por dia, 7 dias por semana, MyStack e colocando ambas as instâncias atrás de um load balancer do Elastic Load Balancing. Você pode estender facilmente o procedimento para adicionar um número arbitrário de instâncias 24 horas por dia, 7 dias por semana, ou pode usar instâncias baseadas em tempo ou em carga para que as pilhas escalem sua AWS OpsWorks pilha automaticamente. Para obter mais informações, consulte Gerenciamento de carga com instâncias baseadas em tempo e carga.

Etapa 4.1: adicionar um load balancer

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O Elastic Load Balancing é um serviço da AWS que distribui automaticamente o tráfego de entrada do aplicativo em várias instâncias da Amazon. EC2 Além de distribuir o tráfego, o Elastic Load Balancing faz o seguinte:

- Detecta instâncias insalubres da Amazon EC2 .

  Ele redireciona o tráfego para as instâncias íntegras restantes até que as instâncias com problemas sejam restauradas.
- Escalona automaticamente a capacidade de gerenciamento de solicitações em resposta ao tráfego de entrada

> **Note**
>
> Um load balancer pode servir a dois propósitos. O óbvio é equalizar a carga em seus servidores de aplicativos. Além disso, vários sites preferem isolar os servidores e bancos de dados dos aplicativos do acesso direto do usuário. Com o AWS OpsWorks Stacks, você pode fazer isso executando sua pilha em uma nuvem privada virtual (VPC) com uma sub-rede pública e privada, da seguinte maneira.
>
> - Coloque os servidores e o banco de dados do aplicativo na sub-rede privada, onde eles podem ser acessados por outras instâncias na VPC, mas não pelos usuários.
> - Direcione o tráfego de usuários para um load balancer na sub-rede pública, que, em seguida, encaminha o tráfego para os servidores do aplicativo em uma sub-rede privada e retorna as respostas para os usuários.

Para obter mais informações, consulte Execução de uma Stack em uma VPC. Para obter um AWS CloudFormation modelo que estende o exemplo desta explicação passo a passo para execução em uma VPC, baixe o arquivo. `OpsWorksVPCtemplates.zip`

Embora o Elastic Load Balancing normalmente seja mencionado como uma camada, ele funciona de uma forma um pouco diferente das outras camadas integradas. Em vez de criar uma camada e adicionar instâncias a ela, você cria um balanceador de carga do Elastic Load Balancing usando o EC2 console da Amazon e o anexa a uma de suas camadas existentes, geralmente uma camada de servidor de aplicativos. AWS OpsWorks Em seguida, o Stacks registra as instâncias existentes da camada com o serviço e adiciona automaticamente quaisquer novas instâncias. O procedimento a seguir descreve como adicionar um balanceador de carga à camada MyStack do PHP App Server.

> **Note**
>
> AWS OpsWorks O Stacks não é compatível com o Application Load Balancer. Você só pode usar o Classic Load Balancer com AWS OpsWorks pilhas.

Para anexar um balanceador de carga a uma camada PHP App Server

1. Use o EC2 console da Amazon para criar um novo balanceador de carga para MyStack. Os detalhes dependem se sua conta é compatível com o EC2 Classic. Para obter mais informações, consulte Conceitos básicos do Elastic Load Balancing. Quando você executar o assistente do Create Load Balancer, configure o load balancer como a seguir:

   Define Load Balancer

   Atribua ao balanceador de carga um nome facilmente reconhecível, como PHP-LB, para facilitar a localização no console do Stacks. AWS OpsWorks Em seguida, selecione Continue para aceitar os padrões para as outras configurações.

   Se você escolher uma VPC com uma ou mais sub-redes do menu Create LB Inside, será preciso selecionar uma sub-rede para cada zona de disponibilidade para onde o tráfego será enviado pelo load balancer.

Assign Security Groups

Se a sua conta for compatível com a VPC padrão, o assistente exibirá esta página para determinar o grupo de segurança do load balancer. Ela não exibe esta página do EC2 Classic.

Para este passo-a-passo, escolha default VPC security group.

Configure Security Settings

Se você escolher HTTPS como Load Balancer Protocol na página Define Load Balancer, defina as configurações de certificado, criptografia e protocolo SSL nesta página. Para este passo-a-passo, aceite os valores padrão e escolha Configure Health Check.

Configurar verificações de integridade

Defina o caminho de ping como **/** e aceite os padrões para as outras configurações.

Adicionar EC2 instâncias

Escolha Continuar; o AWS OpsWorks Stacks registra automaticamente as instâncias com o balanceador de carga.

Add Tags

Adicione tags a fim de ajudar a encontrar. Cada tag é um par chave-valor. Por exemplo, é possível especificar **Description**como a chave e **Test LB** como o valor para os fins dessa demonstração.

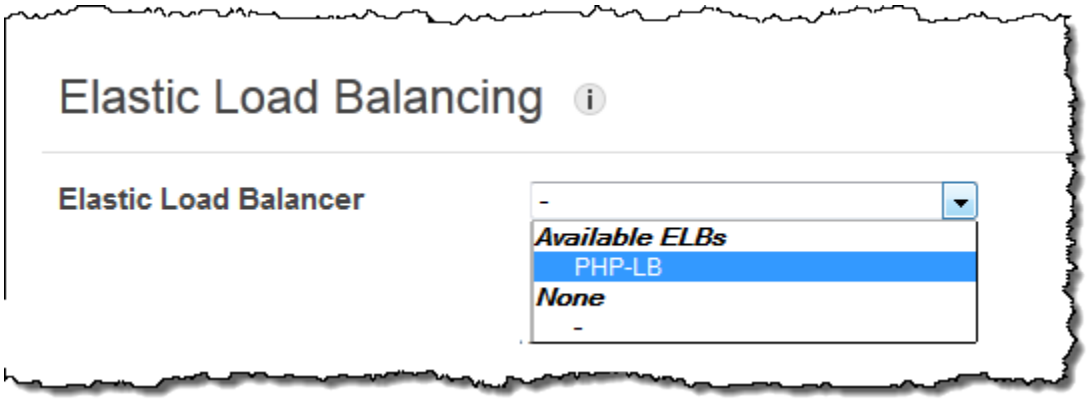
Revisar

Revise suas escolhas, selecione Create e, em seguida, escolha Close, que inicia o load balancer.

2. Se sua conta é compatível com a VPC padrão, após iniciar o load balancer, verifique se o grupo de segurança dela tem as regras de entrada adequadas. A regra padrão não aceita tráfego de entrada.

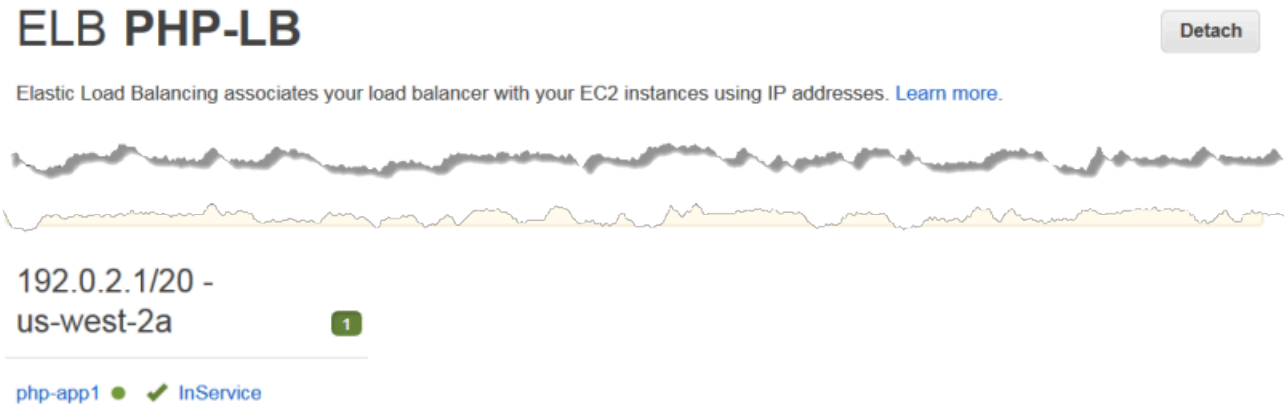
   1. Escolha Security Groups no painel de EC2 navegação da Amazon.
   2. Selecione default VPC security group
   3. Escolha Edit na guia Inbound.
   4. Para esta demonstração, defina Source como Anywhere, que direciona o load balancer a aceitar tráfego de entrada de qualquer endereço IP.

3. Volte para o console AWS OpsWorks Stacks. Na página Layers, escolha o link Network da camada e, em seguida, Edit.
4. Na opção Elastic Load Balancing, selecione o load balancer criado na Etapa 1 e, em seguida, selecione Save.



Depois de conectar o balanceador de carga à camada, o AWS OpsWorks Stacks registra automaticamente as instâncias atuais da camada e adiciona novas instâncias à medida que elas ficam on-line.

5. Na página Layers, clique no nome do load balancer para abrir sua página de detalhes. Quando o registro é concluído e a instância passa por uma verificação de saúde, o AWS OpsWorks Stacks mostra uma marca de seleção verde ao lado da instância na página do balanceador de carga.

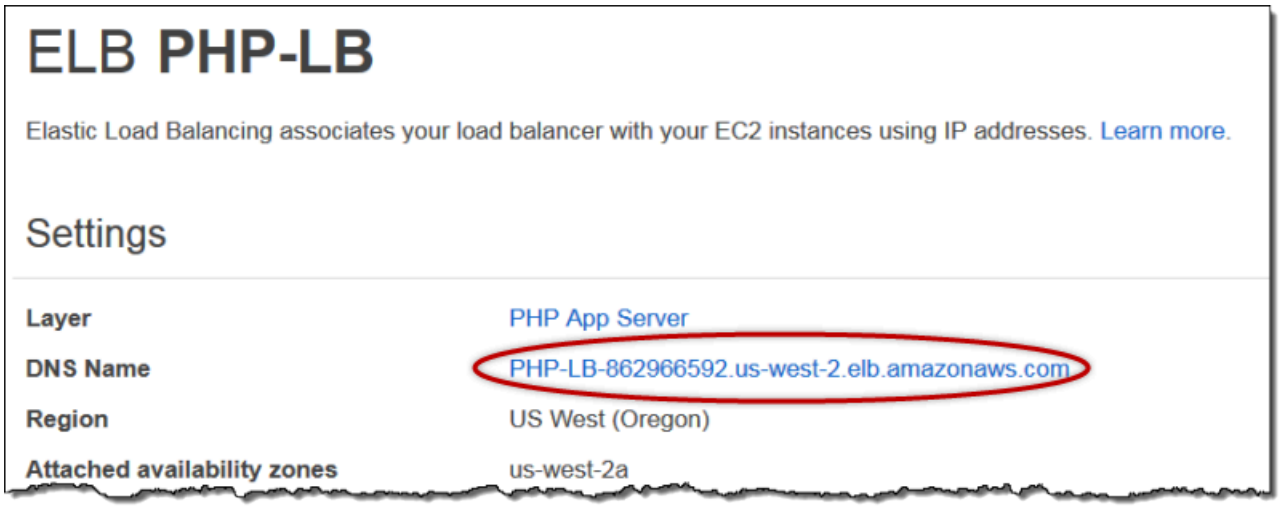


Agora você pode executar o Simple PHPApp enviando uma solicitação ao balanceador de carga.

Para executar o Simple PHPApp por meio do balanceador de carga

1. Abra a página de detalhes do load balancer novamente, caso ainda não estiver aberta.

2. Na página de propriedades, verifique o status de verificação de integridade da instância e clique no nome DNS do balanceador de carga para executar o Simple. PHPApp O balanceador de carga encaminha a solicitação para a instância PHP App Server e retorna a resposta, que deve ser exatamente igual à resposta obtida quando você clica no endereço IP público da instância PHP App Server.



> **Note**
>
> AWS OpsWorks O Stacks também oferece suporte ao balanceador de HAProxy carga, o que pode ter vantagens para alguns aplicativos. Para obter mais informações, consulte HAProxy AWS OpsWorks Camada de pilhas.

Etapa 4.2: adicionar instâncias do PHP App Server

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Agora que o balanceador de carga está posicionado, é possível aumentar a escala horizontalmente da pilha adicionando mais instâncias à camada PHP App Server. A partir da sua perspectiva, a

operação é integrada. Sempre que uma nova instância do PHP App Server fica on-line, o AWS OpsWorks Stacks a registra automaticamente no balanceador de carga e implanta o SimplePHPApp, para que o servidor possa começar imediatamente a lidar com o tráfego de entrada. Em resumo, esse tópico mostra como adicionar uma instância PHP App Server adicional, mas é possível usar a mesma abordagem para adicionar a quantidade que for necessária.

Para adicionar outra instância à camada PHP App Server

1. Na página Instâncias, clique em + Instância sob PHP App Server.
2. Aceite as configurações padrão e clique em Add Instance.
3. Clique em start para iniciar a instância.

Etapa 4.3: Monitorar MyStack

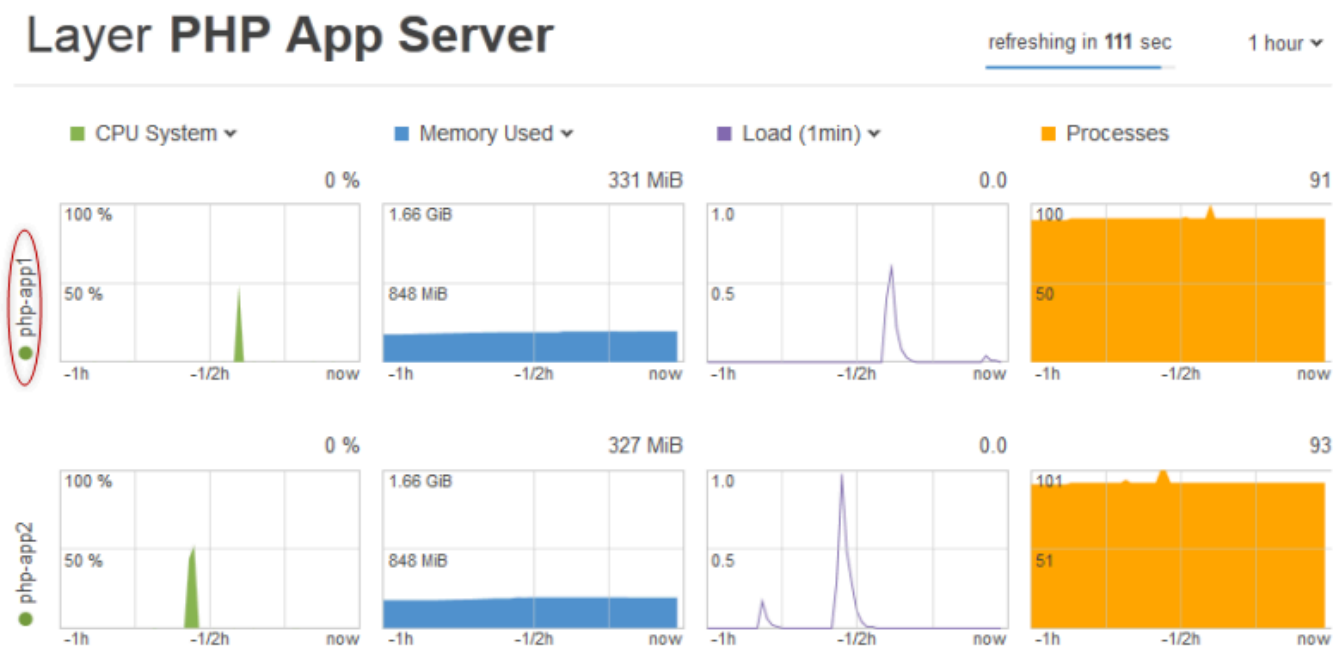
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks usa CloudWatch a Amazon para fornecer métricas para uma pilha e as resume para sua conveniência na página de monitoramento. É possível ver as métricas para toda a pilha, uma camada específica ou uma instância específica.

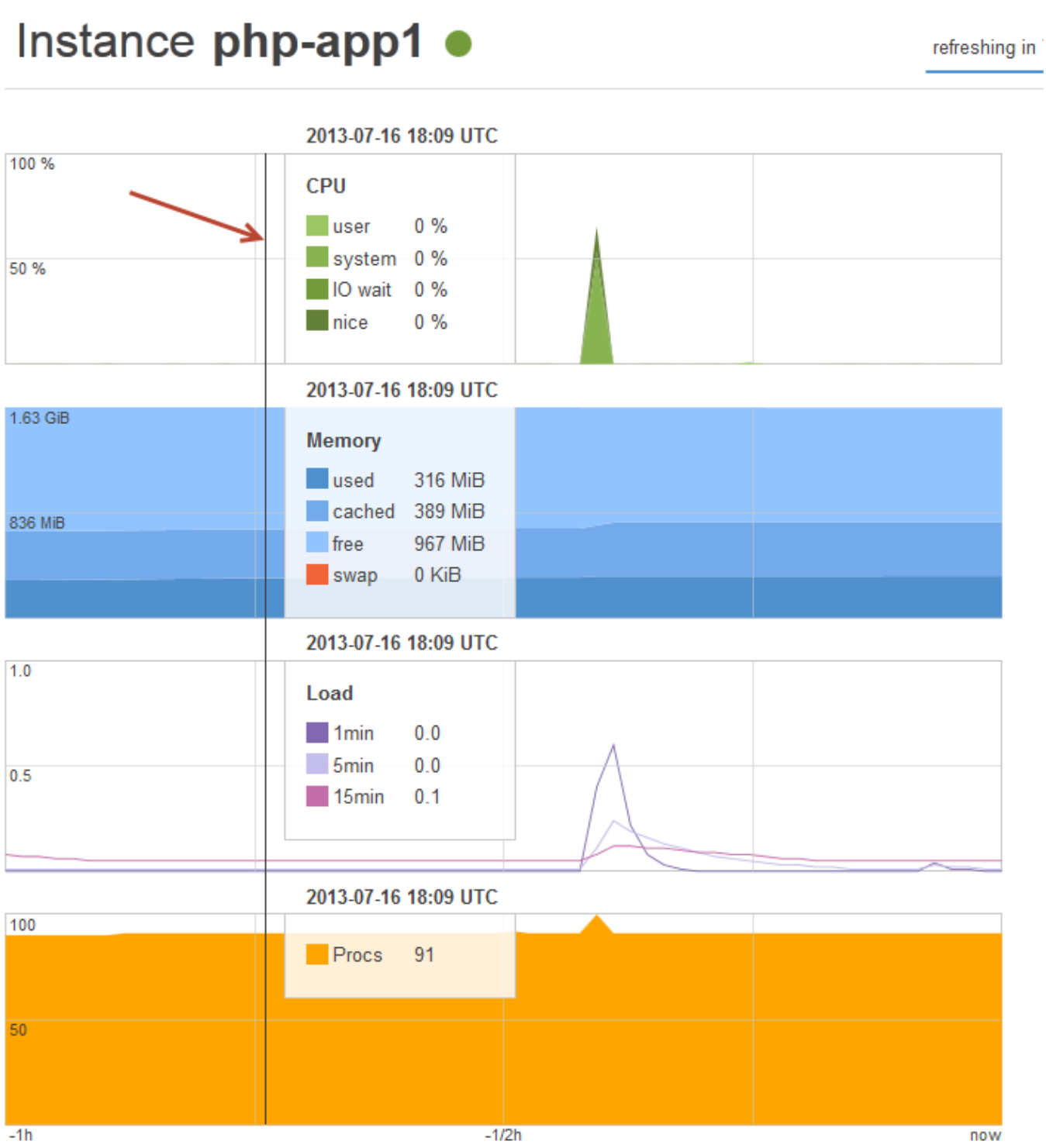
Para monitorar MyStack

1. No painel de navegação, clique em Monitoring, que mostra um conjunto de gráficos com métricas médias para cada camada. Use os menus de CPU System, Memory Used e Load para exibir as diferentes métricas relacionadas.

2. Clique em PHP App Server para ver as métricas para cada uma das instâncias da camada.



3. Clique em php-app1 para ver as métricas para a instância. Veja as métricas para qualquer ponto específico no tempo movendo a barra deslizante.


Instance php-app1
refreshing in
2013-07-16 18:09 UTC
100 %
50 %
CPU
user 0 %
system 0 %
IO wait 0 %
nice 0 %
2013-07-16 18:09 UTC
1.63 GiB
836 MiB
Memory
used 316 MiB
cached 389 MiB
free 967 MiB
swap 0 KiB
2013-07-16 18:09 UTC
1.0
0.5
Load
1min 0.0
5min 0.0
15min 0.1
2013-07-16 18:09 UTC
100
50
Procs 91
-1h
-1/2h
now

> **Note**
>
> AWS OpsWorks O Stacks também oferece suporte ao servidor de monitoramento Ganglia, o que pode ter vantagens para alguns aplicativos. Para obter mais informações, consulte Camada Ganglia.

Etapa 5: Excluir MyStack

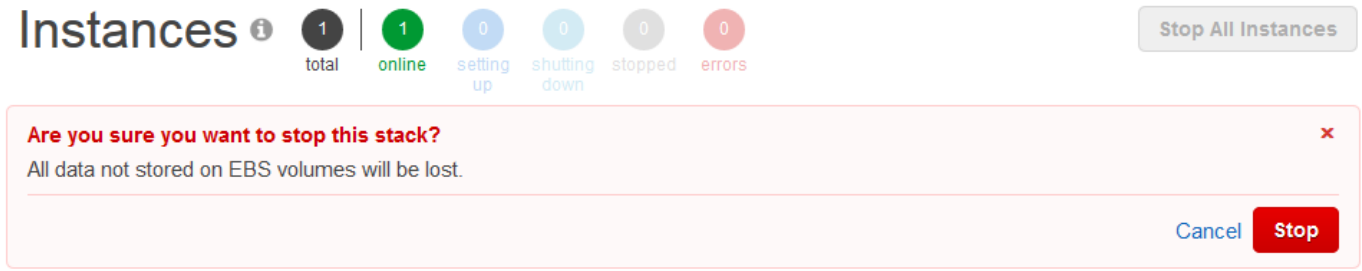
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Assim que você começar a usar os recursos da AWS, como EC2 instâncias da Amazon, você será cobrado com base no seu uso. Se você tiver terminado por enquanto, deverá interromper as instâncias para que não tenha cobranças indesejadas. Se você não precisar mais da pilha, poderá excluí-la.

Para excluir MyStack

1. Parar todas as instâncias

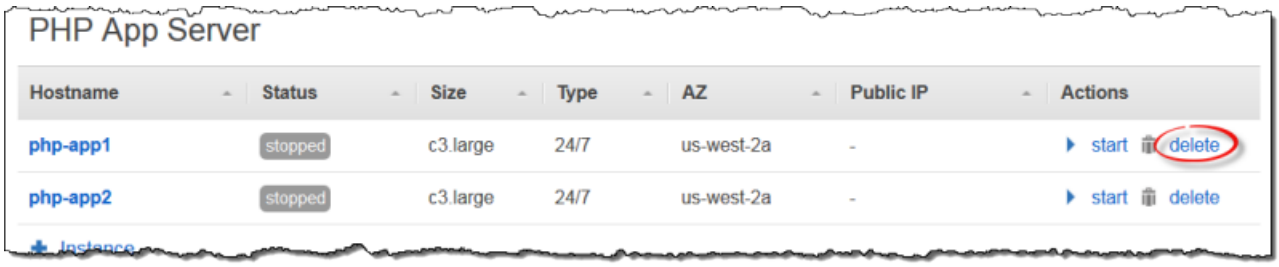
   Na página Instances, clique em Stop All Instances e em Stop quando precisar confirmar a operação.



Depois de clicar em Parar, o AWS OpsWorks Stacks encerra as EC2 instâncias associadas da Amazon, mas não quaisquer recursos associados, como endereços IP elásticos ou volumes do Amazon EBS.

2. Excluir todas as instâncias

   A interrupção da instância apenas encerra as EC2 instâncias associadas da Amazon. Depois que as instâncias estão com o status de interrompidas, você precisa excluir cada instância. Na camada PHP App Server, clique em delete na coluna Actions da instância php-app1.

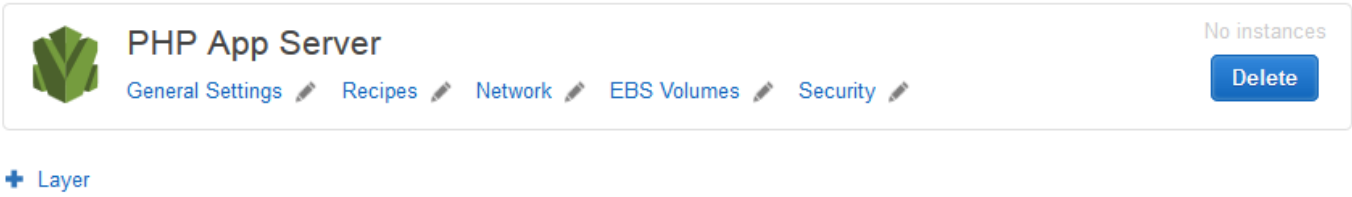


   AWS OpsWorks Em seguida, o Stacks solicita que você confirme a exclusão e mostra todos os recursos dependentes. Você pode optar por manter qualquer um ou todos esses recursos. Este exemplo não tem recursos dependentes, portanto, basta clicar em Delete.

   Repita o processo para php-app2 e a instância do MySQL, db-master1. Observe que db-master1 tem um volume Amazon Elastic Block Store associado, que é selecionado por padrão. Deixe-o selecionado para excluir o volume junto com a instância.

3. Exclua as camadas.

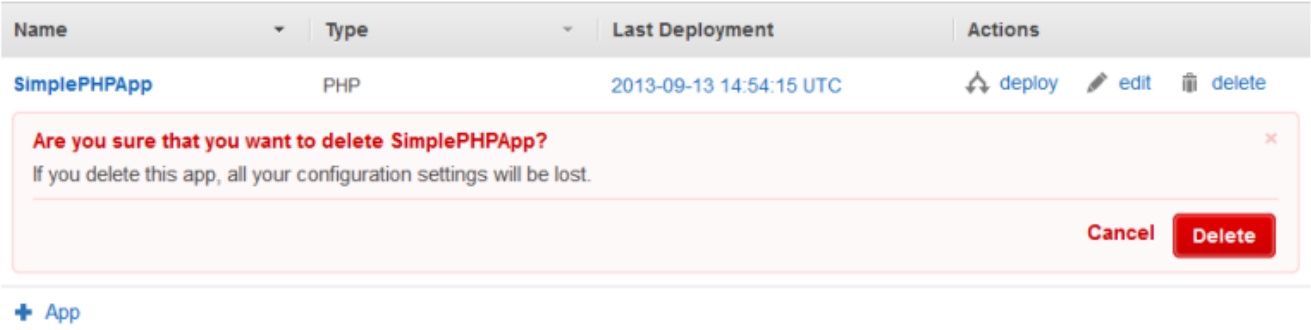
   Na página Layers, clique em Delete e, em seguida, clique em Delete para confirmar.



   Repita o processo para a layer MySQL.

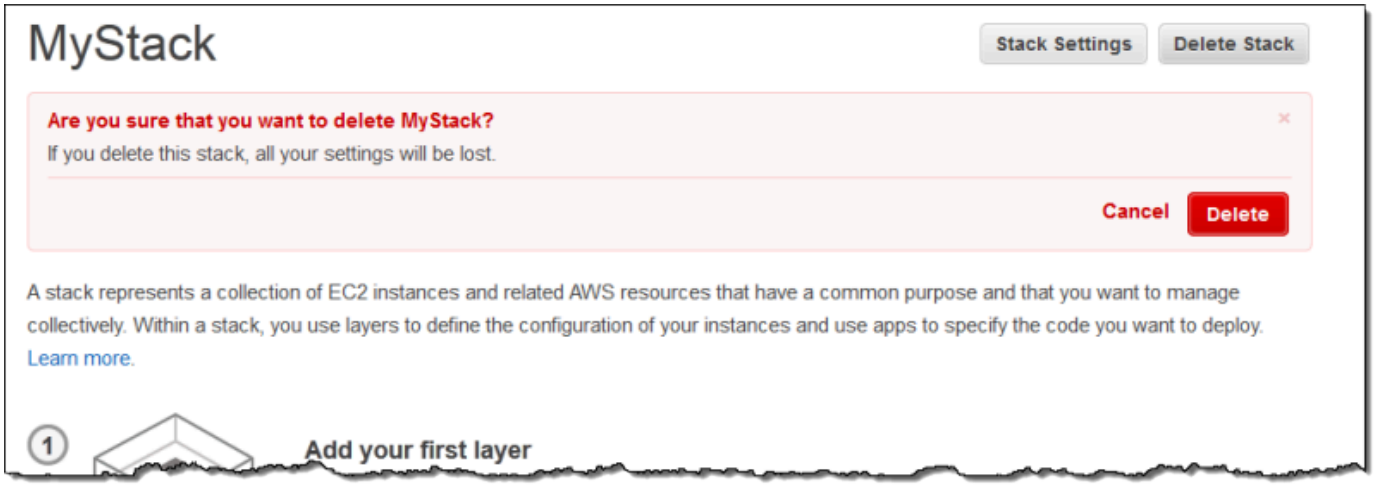
4. Excluir o aplicativo

   Na página Aplicativos, clique em excluir na coluna Ações do PHPApp aplicativo simples e, em seguida, clique em Excluir para confirmar.



5. Excluir MyStack

   Na página Stack, clique em Delete Stack e, em seguida, clique em Delete para confirmar.



Você chegou ao final deste passo a passo.

Criação da sua primeira stack do Node.js

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este exemplo descreve como criar uma stack do Linux que fornece suporte a um servidor de aplicativos do Node.js, e como implantar um aplicativo simples. A stack consiste nos seguintes componentes:

- Uma camada Node.js App Server com duas instâncias
- Um balanceador de carga do Elastic Load Balancing para distribuir o tráfego nas instâncias do servidor de aplicativos
- Uma camada de serviço do Amazon Relational Database Service (Amazon RDS) que fornece um banco de dados de back-end

Tópicos

- Pré-requisitos
- Implementação do aplicativo
- Criação do servidor de banco de dados e load balancer
- Criação da stack
- Implantação do aplicativo
- O que fazer em seguida?

Pré-requisitos

Esta orientação detalhada assume o seguinte:

- Você tem uma conta da AWS e uma compreensão básica de como usar o AWS OpsWorks Stacks.

  Se você é novato no AWS OpsWorks Stacks ou na AWS, aprenda o básico concluindo o tutorial introdutório em. Conceitos básicos das pilhas Linux do Chef 11
- Você tem um conhecimento básico de como implementar um aplicativo Node.js.

  Se você é um iniciante no uso do Node.js, aprenda os conceitos básicos com um tutorial introdutório como, por exemplo, o Node: Up and Running.
- Você já criou pelo menos uma stack na região da AWS que planeja usar para este exemplo.

  Quando você cria a primeira pilha em uma região, o AWS OpsWorks Stacks cria um grupo de segurança do Amazon Elastic Compute Cloud (Amazon EC2) para cada tipo de camada. Você precisa desses grupos de segurança para criar a instância de banco de dados (DB –

database) do Amazon RDS. Se você é iniciante no AWS OpsWorks Stacks, recomendamos que use neste exemplo a mesma região que usou ao seguir o tutorial emConceitos básicos das pilhas Linux do Chef 11. Se quiser usar uma região nova, crie uma stack nova na região; a stack não precisa ter layers ou instâncias. Assim que você cria a pilha, o AWS OpsWorks Stacks adiciona automaticamente um conjunto de grupos de segurança à região.

- Você criará sua pilha em uma VPC padrão.

  Você pode usar EC2 -Classic para este passo a passo, mas alguns detalhes serão um pouco diferentes. Por exemplo, com EC2 -Classic, você especifica a zona de disponibilidade (AZ) de uma instância em vez de sua sub-rede.

- Seu usuário do IAM tem permissões de acesso total ao AWS OpsWorks Stacks.

  Por motivos de segurança, recomendamos que você não use as credenciais raiz da sua conta para esta orientação detalhada. Em vez disso, crie um usuário com permissões de acesso total ao AWS OpsWorks Stacks e use essas credenciais com o Stacks. AWS OpsWorks Para obter mais informações, consulte Como criar um usuário administrativo do .

Implementação do aplicativo

Esta orientação detalhada usa um aplicativo simples do Express que se conecta à instância de banco de dados do Amazon RDS e lista os bancos de dados da instância.

Para implementar o aplicativo, crie um diretório chamado `nodedb` em um local conveniente na sua estação de trabalho e adicione os três arquivos a seguir nesse diretório.

Tópicos

- O descritor de pacote
- O arquivo de layout
- O arquivo de código

O descritor de pacote

Para criar o descritor do pacote do aplicativo, adicione um arquivo chamado `package.json`no diretório `nodedb`, com o conteúdo descrito a seguir. O `package.json` é obrigatório para aplicativos do Express e deve estar localizado no diretório raiz do aplicativo.

```
{
  "name": "Nodejs-DB",
  "description": "Node.js example application",
  "version": "0.0.1",
  "dependencies": {
    "express": "*",
    "ejs": "*",
    "mysql": "*"
  }
}
```

Este exemplo do `package.json` é bastante limitado. Ele define os atributos `name` e `version` necessários e lista os pacotes dependentes:

- `express` faz referência ao pacote Express.
- `ejs` faz referência ao pacote EJS, que o aplicativo usa para inserir texto em um arquivo de layout em HTML.
- `mysql` faz referência ao pacote node-mysql, que o aplicativo usa para se conectar à instância do RDS.

Para obter mais informações sobre arquivos de descritores de pacotes, consulte package.json.

O arquivo de layout

Para criar o arquivo de layout do aplicativo, adicione um diretório `views` ao diretório `nodedb` e, em seguida, adicione um arquivo ao `views` chamado `index.html` com o seguinte conteúdo:

```
<!DOCTYPE html>
<html>
<head>
  <title>AWS Opsworks Node.js Example</title>
</head>
<body>
  <h1>AWS OpsWorks Node.js Example</h1>
    <p>Amazon RDS Endpoint: <i><%= hostname %></i></p>
    <p>User: <i><%= username %></i></p>
    <p>Password: <i><%= password %></i></p>
    <p>Port: <i><%= port %></i></p>
    <p>Database: <i><%= database %></i></p>
```

```
    <p>Connection: <%= connectionerror %></p>
    <p>Databases: <%= databases %></p>
</body>
</html>
```

Neste exemplo, o arquivo de layout é um documento HTML simples que exibe alguns dados do Amazon RDS. Cada elemento `<%= ... =>` representa o valor de uma variável que é definida no arquivo de código do aplicativo, que criamos a seguir.

O arquivo de código

Para criar o arquivo de código do aplicativo, adicione um arquivo `server.js` ao diretório nodedb com o seguinte conteúdo.

> **Important**
>
> Com o AWS OpsWorks Stacks, o arquivo de código principal de um aplicativo Node.js deve ser nomeado `server.js` e estar localizado na pasta raiz do aplicativo.

```
var express = require('express');
var mysql = require('mysql');
var dbconfig = require('opsworks'); //[1] Include database connection data
var app = express();
var outputString = "";

app.engine('html', require('ejs').renderFile);

//[2] Get database connection data
app.locals.hostname = dbconfig.db['host'];
app.locals.username = dbconfig.db['username'];
app.locals.password = dbconfig.db['password'];
app.locals.port = dbconfig.db['port'];
app.locals.database = dbconfig.db['database'];
app.locals.connectionerror = 'successful';
app.locals.databases = '';

//[3] Connect to the Amazon RDS instance
var connection = mysql.createConnection({
    host: dbconfig.db['host'],
```

```
    user: dbconfig.db['username'],
    password: dbconfig.db['password'],
    port: dbconfig.db['port'],
    database: dbconfig.db['database']
});

connection.connect(function(err)
{
    if (err) {
        app.locals.connectionerror = err.stack;
        return;
    }
});

// [4] Query the database
connection.query('SHOW DATABASES', function (err, results) {
    if (err) {
        app.locals.databases = err.stack;
    }

    if (results) {
        for (var i in results) {
            outputString = outputString + results[i].Database + ', ';
        }
        app.locals.databases = outputString.slice(0, outputString.length-2);
    }
});

connection.end();

app.get('/', function(req, res) {
    res.render('./index.html');
});

app.use(express.static('public'));

//[5] Listen for incoming requests
app.listen(process.env.PORT);
```

Esse exemplo exibe as informações de conexão do banco de dados, executa consultas no servidor de banco de dados e exibe os bancos de dados do servidor. Você pode facilmente generalizá-lo para obter a interação com o banco de dados, conforme necessário. As observações a seguir se referem aos comentários numerados no código anterior.

[1] Incluir dados da conexão de banco de dados

Essa instrução `require` inclui os dados de conexão do banco de dados. Conforme descrito posteriormente, quando você anexa uma instância de banco de dados a um aplicativo, o AWS OpsWorks Stacks coloca os dados da conexão em um arquivo chamado`opsworks.js`, que é semelhante ao seguinte:

```
exports.db = {
  "host":"nodeexample.cdlqlk5uwd0k.us-west-2.rds.amazonaws.com",
  "database":"nodeexampledb",
  "port":3306,
  "username":"opsworksuser",
  "password":"your_pwd",
  "reconnect":true,
  "data_source_provider":"rds",
  "type":"mysql"}
```

`opsworks.js` está no diretório `shared/config` do aplicativo, `/srv/www/`*`app_shortname`*`/shared/config`. No entanto, o AWS OpsWorks Stacks coloca um link simbólico `opsworks.js` no diretório raiz do aplicativo, para que você possa incluir o objeto usando apenas. `require 'opsworks'`

[2] Obter os dados da conexão de banco de dados

Esse conjunto de instruções exibe os dados da conexão do `opsworks.js`atribuindo os valores do objeto db a um conjunto de propriedades `app.locals`, sendo que cada uma delas é mapeada para um dos elementos <%= ... %> no arquivo `index.html`. O documento renderizado substitui os elementos <%= ... %> pelos valores da propriedade correspondente.

[3] Conectar-se à instância do Amazon RDS

Este exemplo usa o `node-mysql` para acessar o banco de dados. Para se conectar ao banco de dados, o exemplo cria um objeto `connection` passando os dados da conexão ao `createConnection` e, em seguida, chamando o `connection.connect` para estabelecer a conexão.

[4] Consultar o banco de dados

Após estabelecer uma conexão, o exemplo chama o `connection.query` para consultar o banco de dados. Esse exemplo executa uma consulta simples para obter os nomes dos bancos de dados do servidor. A `query` retorna uma matriz de objetos `results`, um para cada banco

de dados, com o nome do banco de dados atribuído à propriedade `Database`. O exemplo concatena os nomes e os atribui ao `app.locals.databases`, que exibe a lista na página HTML renderizada.

Nesse exemplo há cinco bancos de dados, o banco de dados `nodeexampledb`, que você especificou quando criou a instância do RDS e quatro outros que são criados automaticamente pelo Amazon RDS.

[5] Escutar solicitações de entrada

A última instrução escuta as solicitações de entrada em uma porta especificada. Você não precisa especificar um valor explícito para a porta. Ao adicionar o aplicativo à sua pilha, você especifica se o aplicativo é compatível com solicitações HTTP ou HTTPS. AWS OpsWorks Em seguida, as pilhas definem a variável de `PORT` ambiente como 80 (HTTP) ou 443 (HTTPS), e você pode usar essa variável em seu aplicativo.

É possível escutar em outras portas, mas o grupo de segurança integrado da camada Node.js App Server, AWS- OpsWorks -NodeJS-App-Server, permite tráfego de entrada de usuários somente para as portas 80, 443 e 22 (SSH). Para permitir o tráfego de entrada de usuário em outras portas, crie um grupo de segurança com regras de entrada apropriadas e atribua-o à camada do Node.js App Server. Não modifique as regras de entrada editando o grupo de segurança integrado. Cada vez que você cria uma pilha, o AWS OpsWorks Stacks substitui os grupos de segurança integrados pelas configurações padrão, portanto, todas as alterações feitas serão perdidas.

> **Note**
>
> Você pode associar variáveis de ambiente personalizadas ao seu aplicativo quando cria ou atualiza o aplicativo associado. Você também pode passar os dados para o seu aplicativo usando um JSON personalizado e uma receita personalizada. Para obter mais informações, consulte Transmissão de dados para aplicativos.

Criação do servidor de banco de dados e load balancer

Este exemplo usa o servidor de banco de dados do Amazon RDS e as instâncias de balanceador de carga do Elastic Load Balancing. Você deve criar cada instância separadamente e, em seguida, incorporá-la a sua stack. Esta seção descreve como criar novas instâncias de banco de dados e de

load balancer. Você pode usar instâncias existentes, mas recomendamos que leia o procedimento até o fim para ter certeza que essas instâncias estão configuradas corretamente.

A seguir descrevemos como criar uma instância de banco de dados do RDS com o mínimo de configurações suficientes para este exemplo. Para obter mais informações, consulte o Guia do usuário do Amazon RDS.

Para criar a instância de banco de dados do RDS

1. Abra o console de .

   Abra o console do Amazon RDS e defina a região como Oeste dos EUA (Oregon). No painel de navegação, escolha RDS Dashboard e, em seguida, Launch DB Instance.
2. Especifique o mecanismo de banco de dados.

   Escolha MySQL Community Edition como o mecanismo de banco de dados.
3. Recuse a implantação multi-AZ.

   Escolha No, this instance... e, em seguida, Next. Você não precisa de implantação multi-AZ para este exemplo.
4. Defina as configurações básicas.

   Na página DB Instance Details, especifique as seguintes configurações:

   - DB Instance Class: db.t2.micro
   - Multi-AZ Deployment: No
   - Allocated Storage (Armazenamento alocado): **5** GB
   - DB Instance Identifier: **`nodeexample`**
   - Master Username (Nome do usuário mestre): **`opsworksuser`**
   - Master Password: uma senha de sua escolha

   Anote o identificador da instância, o nome de usuário e a senha para usar mais adiante, aceite as configurações padrão para as outras opções e, em seguida, escolha Next.
5. Defina as configurações avançadas.

   Na página Configure Advanced Settings, especifique as seguintes configurações:

   - Nome do banco de dados: **`nodeexampledb`**

- Grupo (s) de segurança de banco de dados: AWS- OpsWorks -DB-Master-Server

> **Note**
>
> O grupo de segurança AWS- OpsWorks -DB-Master-Server permite que somente as instâncias da sua pilha acessem o banco de dados. Se você deseja acessar o banco de dados diretamente, anexe um grupo de segurança adicional à instância de banco de dados do RDS com as regras de entrada apropriadas. Para obter mais informações, consulte Grupos de segurança do Amazon RDS. Você também pode controlar o acesso colocando a instância em uma VPC. Para obter mais informações, consulte Execução de uma Stack em uma VPC.

Anote o nome do banco de dados para usar mais adiante, aceite os valores padrão para as outras configurações e, em seguida, escolha Launch DB Instance.

O procedimento a seguir descreve como criar um balanceador de carga do Elastic Load Balancing para este exemplo. Para obter mais informações, consulte o Manual do usuário do Elastic Load Balancing.

Para criar o load balancer

1. Abra o EC2 console da Amazon.

   Abra o EC2 console da Amazon e verifique se a região está definida como Oeste dos EUA (Oregon). No painel de navegação, escolha Load Balancers, Create Load Balancer.

2. Defina o load balancer.

   Na página Define Load Balancer, especifique as seguintes configurações:

   - Nome: **`Node-LB`**
   - Criar LB dentro de: Minha VPC padrão

   Aceite as configurações padrão para as outras opções e, em seguida, escolha Next.

3. Atribua os grupos de segurança.

   Na página Assign Security Groups, especifique os seguintes grupos:

- default VPC security group
- AWS- OpsWorks -NodeJS-App-Server

Escolha Próximo. Na página Configure Security Settings, selecione Next. Você não precisa de um ouvinte protegido para este exemplo.

4. Configure a verificação de integridade.

   Na página Configure Health Check (Configurar verificação de integridade), defina Ping Path (Caminho de ping) como **/** e aceite os valores padrão para as outras configurações. Escolha Próximo. Na página Adicionar EC2 instâncias, escolha Avançar. Na página Adicionar tags, escolha Revisar e criar. AWS OpsWorks O Stacks lida com a tarefa de adicionar EC2 instâncias ao balanceador de carga, e você não precisará de tags neste exemplo.

5. Crie o load balancer.

   Na página Review, escolha Create para criar o load balancer.

Criação da stack

Você agora tem todos os componentes necessários para criar a stack.

Para criar a pilha

1. Faça login no console AWS OpsWorks Stacks.

   Faça login no console do AWS OpsWorks Stacks e selecione Add Stack (Adicionar pilha).

2. Crie a pilha.

   Para criar uma pilha nova, escolha Chef 11 pilha e, em seguida, especifique as seguintes configurações.

   - – **`NodeStack`**
   - Região: Oeste dos EUA (Oregon)

     Você pode criar uma pilha em qualquer região da AWS, mas recomendamos o Oeste dos EUA (Oregon) para os tutoriais.

Escolha Add Stack. Para obter mais informações sobre as definições de configuração de uma stack, consulte Criar uma nova pilha.

3. Adicione uma camada do &node; com um balanceador de carga anexado.

   Na NodeStackpágina, escolha Adicionar uma camada e, em seguida, especifique as seguintes configurações:

   - Tipo de camada: Node.js App Server
   - Balanceador de carga elástico: Nó-LB

   Aceite os valores padrão para as outras configurações e, em seguida, escolha Add Layer.

4. Adicione as instâncias à layer e as inicie.

   No painel de navegação, escolha Instances e, em seguida, adicione duas instâncias à layer do Rails App Server, da seguinte forma.

   1. Em Node.js App Server, escolha Adicionar instância.

      Defina Size como t2.micro, aceite os valores padrão para as outras configurações e, em seguida, escolha Add Instance.

   2. Escolha +Instance e, em seguida, adicione uma segunda instância t2.micro à layer em uma sub-rede diferente.

      Isso coloca a instância em uma zona de disponibilidade (AZ) diferente.

   3. Escolha Add instance.
   4. Para iniciar ambas as instâncias, escolha Start All Instances.

   Você atribuiu um balanceador de carga do Elastic Load Balancing a essa camada. Quando uma instância entra ou sai do estado on-line, o AWS OpsWorks Stacks registra ou cancela automaticamente o registro da instância com o balanceador de carga.

   > **Note**
   >
   > Para uma pilha de produção, recomendamos que você distribua suas instâncias do servidor de aplicativos em várias AZs. Se os usuários não conseguem se conectar

a uma zona de disponibilidade, o load balancer direciona o tráfego de entrada para instâncias nas zonas restantes, e seu site continua a funcionar.

5. Registre a instância de banco de dados do RDS na stack.

   No painel de navegação, escolha Resources e registre a instância de banco de dados do RDS na pilha, da seguinte forma.

   1. Escolha a guia RDS e, em seguida, as instâncias de Show Unregistered RDS DB.
   2. Escolha a instância nodeexampledb e, em seguida, especifique as seguintes configurações:
      - Usuário: o nome de usuário mestre que você especificou quando criou a instância; para este exemplo. **`opsworksuser`**.
      - Senha: a senha mestre que você especificou quando criou a instância.
   3. Escolha Registrar-se com Stack para adicionar a instância de banco de dados do RDS à pilha como uma camada de serviço do Amazon RDS.

   > **Warning**
   > AWS OpsWorks O Stacks não valida os valores do usuário ou da senha, ele simplesmente os passa para o aplicativo. Se você informá-los incorretamente, o aplicativo não conseguirá se conectar ao banco de dados.

   Para adicionar a instância de banco de dados do RDS à pilha como uma camada de serviço do Amazon RDS, escolha Registrar-se com Stack.

Implantação do aplicativo

Você deve armazenar o aplicativo em um repositório remoto. Quando você o implanta, o AWS OpsWorks Stacks implanta o código e os arquivos relacionados do repositório nas instâncias do servidor de aplicativos. Para sua conveniência, este exemplo usa um arquivamento público do Amazon Simple Storage Service (Amazon S3) como repositório, mas você também pode usar vários outros tipos de repositórios, incluindo o Git e o Subversion. Para obter mais informações, consulte Origem do aplicativo.

Como implantar o aplicativo

1. Empacote o aplicativo em um arquivo compactado.

   Crie um arquivo compactado `.zip` chamado nodedb.zip contendo o diretório `nodedb` e seus subdiretórios. Você também pode usar outros tipos de arquivos compactados, incluindo gzip, bzip2 ou tarball. Observe que o AWS OpsWorks Stacks não suporta arquivos tar não compactados. Para obter mais informações, consulte Origem do aplicativo.

2. Faça o upload do arquivo compactado para o Amazon S3.

   Faça upload do `nodedb.zip` para um bucket do Amazon S3, defina o arquivo como público e copie o URL do arquivo para usar mais diante. Para obter mais informações sobre como criar buckets e fazer upload de arquivos, consulte Comece a usar o Amazon Simple Storage Service.

   > **Note**
   >
   > AWS OpsWorks As pilhas também podem implantar arquivos privados de um bucket do Amazon S3, mas, para simplificar, este exemplo usa um arquivo público. Para obter mais informações, consulte Origem do aplicativo.

3. Crie um aplicativo AWS OpsWorks Stacks.

   Volte ao console do AWS OpsWorks Stacks, no painel de navegação, escolha Aplicativos e, em seguida, escolha Adicionar um aplicativo. Especifique as seguintes configurações:

   - Name: `NodeDB`.

     Essa string é o nome de exibição do aplicativo. Para a maioria dos propósitos, você precisa do nome curto do aplicativo, que o AWS OpsWorks Stacks gera a partir do nome de exibição, transformando todos os caracteres em minúsculas e removendo a pontuação. Neste exemplo, o nome abreviado é `nodedb`. Para verificar o nome abreviado de um aplicativo, depois de criar o aplicativo, escolha o aplicativo na página Apps para exibir sua página de detalhes.

   - Digite – `Node.js`.
   - Tipo de fonte de dados – `RDS`.
   - Database instance: escolha a instância de banco de dados do Amazon RDS que você registrou anteriormente.
   - Database name – Especifique o nome do banco de dados que você criou anteriormente, `nodeexampledb` neste exemplo.

- Tipo de repositório – `Http Archive`.

  Você deve usar esse tipo de repositório para arquivos públicos do Amazon S3. O tipo `S3 Archive` é usado apenas para arquivos privados.
- URL do repositório: o URL do Amazon S3 do arquivo compactado.

Use os valores padrão para as configurações restantes e clique em Add App para criar o aplicativo.

4. Implante o aplicativo.

   Acesse a página Apps e, na coluna Actions do aplicativo do NodeDB, escolha deploy. Em seguida, escolha Implantar para implantar o aplicativo nas instâncias do servidor. AWS OpsWorks O Stacks executa as receitas do Deploy em cada instância, que baixa o aplicativo do repositório e reinicia o servidor. Quando cada instância tiver uma marca de verificação verde e o Status for successful, a implantação estará concluída e o aplicativo estará pronto para começar a lidar com as solicitações.

   > **Note**
   >
   > Se a implantação falhar, escolha show na coluna Log para exibir o log do Chef para a implantação. As informações de erro ficam perto da parte inferior.

5. Abra o aplicativo .

   Para abrir o aplicativo, escolha Layers, escolha o load balancer e, em seguida, escolha o nome DNS do load balancer, que envia uma solicitação HTTP para o load balancer. Você deve ver algo parecido com o exemplo a seguir.

AWS OpsWorks Node.js Example

Amazon RDS Endpoint: *nodeexample.cdlqlk5uwd0k.us-west-2.rds.amazonaws.com*

User: *opsworksuser*

Password: *Your-Pwd*

Port: *3306*

Database: *nodeexampledb*

Connection: successful

Databases: information_schema, innodb, mysql, nodeexampledb, performance_schema

> **Note**
>
> AWS OpsWorks O Stacks implanta aplicativos automaticamente em novas instâncias durante a configuração. A implantação manual é necessária apenas para instâncias online. Para obter mais informações, consulte Implementação de aplicativos. Para ver uma discussão geral sobre a implantação, incluindo algumas estratégias mais sofisticadas de implantação, consulte Gerenciamento e implementação de aplicativos e guias de procedimentos.

O que fazer em seguida?

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta orientação detalhada apresentou as noções básicas da configuração de uma stack simples de servidor de aplicativos do Node.js. Aqui estão algumas sugestões sobre o que fazer a seguir.

Examine os livros de receitas integrados do Node.js

Se você quiser saber como as instâncias são configuradas em detalhes, consulte o livro de receitas integrado da camada, opsworks_nodejs, que contém as receitas e os arquivos relacionados que o AWS OpsWorks Stacks usa para instalar e configurar o software, e o livro de receitas integrado de implantação, que contém as receitas que o Stacks usa para implantar os aplicativos. AWS OpsWorks

Personalize a configuração do servidor

A stack do exemplo é bastante básica. Para usar na produção, você provavelmente vai precisar personalizar a stack. Para obter mais informações, consulte Personalizando pilhas AWS OpsWorks.

Adicione o suporte a SSL

Você pode ativar o suporte SSL para seu aplicativo e fornecer aos AWS OpsWorks Stacks os certificados apropriados ao criar o aplicativo. AWS OpsWorks Em seguida, o Stacks instala os certificados no diretório apropriado. Para obter mais informações, consulte Uso de SSL.

Adicione dados ao cache de memória

Os sites usados na produção muitas vezes melhoram seu desempenho armazenando dados em cache em um armazenamento de chave-valor na memória, como o Redis ou o Memcache. Você pode usar qualquer um com uma AWS OpsWorks pilha de pilhas. Para obter mais informações, consulte ElastiCache Redis e Memcached.

Use uma estratégia de implantação mais sofisticada

O exemplo usou uma estratégia simples de implantação de aplicativos, que implanta a atualização em cada instância simultaneamente. Essa abordagem é simples e rápida, mas não há margem para erro. Se a implantação falhar ou a atualização tiver algum problema, cada instância em sua stack de produção pode ser afetada, interrompendo ou desativando potencialmente seu site até que você possa corrigir o problema. Para obter mais informações sobre as estratégias de implantação, consulte Gerenciamento e implementação de aplicativos e guias de procedimentos.

Estenda a camada do Node.js App Server

Você pode estender a layer de várias maneiras. Por exemplo, você pode implementar receitas para executar scripts em instâncias ou implementar hooks de implantação do Chef para personalizar a implantação de aplicativos. Para obter mais informações, consulte Extensão de uma camada.

Defina variáveis de ambiente

Você pode passar dados para o seu aplicativo definindo variáveis de ambiente para o aplicativo associado. Quando você implanta o aplicativo, o AWS OpsWorks Stacks exporta essas variáveis para que você possa acessá-las do seu aplicativo. Para obter mais informações, consulte Usar variáveis de ambiente do .

Personalizando pilhas AWS OpsWorks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks As camadas integradas das pilhas fornecem uma funcionalidade padrão que é suficiente para muitas finalidades. No entanto, você pode encontrar um ou mais dos seguintes:

- A configuração padrão de uma camada integrada é adequada mas não é ideal; é recomendável otimizá-la para seus requisitos específicos.

  Por exemplo, você pode querer ajustar uma configuração de servidor Nginx de uma camada Static Web Server ao especificar seus próprios valores para as configurações, como o número máximo de processos de operador ou o valor `keepalivetimeout`.
- A funcionalidade da camada integrada é uma boa ideia, mas talvez seja necessário estendê-la ao instalar pacotes adicionais ou executar alguns scripts de instalação personalizados.

  Por exemplo, talvez seja necessário estender uma camada PHP App Server ao instalar também um servidor Redis.
- Você tem requisitos que não são processados por qualquer uma das camadas integradas.

  Por exemplo, o AWS OpsWorks Stacks não inclui camadas integradas para alguns servidores de banco de dados populares. Você pode criar uma camada personalizada que instale esses servidores nas instâncias da camada.

- Você está executando uma pilha do Windows, que é compatível apenas com camadas personalizadas.

AWS OpsWorks O Stacks fornece várias maneiras de personalizar camadas para atender às suas necessidades específicas. Os exemplos a seguir são listados em ordem crescente de complexidade e potência:

> **Note**
>
> Algumas destas abordagens funcionam apenas para pilhas do Linux. Consulte os tópicos a seguir para obter detalhes.

- Use JSON personalizado para substituir as configurações padrão de AWS OpsWorks pilhas.
- Implemente um livro de receitas personalizado do Chef com um arquivo de atributos que substitui as configurações padrão AWS OpsWorks do Stacks.
- Implemente um livro de receitas personalizado do Chef com um modelo que substitui ou estende um modelo padrão AWS OpsWorks do Stacks.
- Implemente um livro de receitas personalizado do Chef com uma receita simples que execute um script de shell.
- Implemente um livro de receitas personalizado do Chef com receitas personalizadas que executem tarefas como a criação e a configuração de diretórios, a instalação de pacotes, a criação de arquivos de configuração, a implementação de aplicativos e assim por diante.

Você também pode substituir receitas, dependendo da versão e do sistema operacional do Chef da pilha.

- Com as pilhas do Chef 0.9 e 11.4, você não pode substituir uma receita integrada ao implementar uma receita personalizada com o mesmo nome de livro de receitas e de receita.

  Para cada evento do ciclo de vida, o AWS OpsWorks Stacks sempre executa primeiro as receitas integradas, seguidas por qualquer receita personalizada. Como essas versões do Chef não executam uma receita com o mesmo nome de livro de receitas e de receita duas vezes, a receita integrada tem precedência e a receita personalizada não é executada.
- Você pode substituir receitas integradas em pilhas do Chef 11.10.

Para obter mais informações, consulte Instalação e precedência em livros de receitas.

- Você não pode substituir receitas integradas em pilhas do Windows.

  A forma como o AWS OpsWorks Stacks gerencia as execuções do Chef para pilhas do Windows não permite que as receitas integradas sejam substituídas.

> **Note**
>
> Como muitas das técnicas usam livros de receitas personalizados, você deve primeiro ler Livros de receitas e receitas se ainda não estiver familiarizado com a implementação do livro de receitas. Conceitos básicos do livro de receitasfornece um tutorial detalhado de introdução à implementação de livros de receitas personalizados e Implementando livros de receitas para pilhas AWS OpsWorks aborda alguns detalhes sobre como implementar livros de receitas para instâncias do AWS OpsWorks Stacks.

Tópicos

- Personalizando a configuração AWS OpsWorks de pilhas substituindo atributos
- Estendendo arquivos de configuração de AWS OpsWorks pilhas usando modelos personalizados
- Extensão de uma camada
- Criação de uma camada de servidor Tomcat personalizada
- Configuração de pilha e atributos de implantação

Personalizando a configuração AWS OpsWorks de pilhas substituindo atributos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Para pilhas do Windows e pilhas do Chef 12 Linux, o AWS OpsWorks Stacks usa execuções separadas do Chef para receitas integradas e receitas personalizadas. Isso significa que você não pode usar as técnicas discutidas nesta seção para sobrepor atributos integrados para stacks do Windows e Chef 12 Linux.

Receitas e modelos dependem de uma variedade de atributos Chef para informações específicas de instância ou stack, como configurações de layer ou de servidor de aplicativo. Esses atributos têm várias fontes:

- JSON personalizado: opcionalmente, você pode especificar atributos JSON personalizados ao criar, atualizar ou clonar uma pilha ou ao implantar um aplicativo.
- Atributos de configuração da pilha —AWS OpsWorks As pilhas definem esses atributos para armazenar as informações de configuração da pilha, incluindo as informações que você especifica nas configurações do console.
- Atributos de implantação — A AWS OpsWorks define atributos relacionados à implantação para eventos de implantação.
- Atributos de livros de receitas: em geral, livros de receitas integrados e personalizados incluem um ou mais arquivos de atributos, que contêm atributos representando valores específicos do livro de receitas, como definições de configuração do servidor de aplicativos.
- Chef: a ferramenta Ohai do Chef define atributos que representam uma ampla variedade de configurações do sistema, como tipo de CPU e memória instalada.

Para uma lista completa de configuração de stacks e atributos de implantação e atributos de livro de receitas integrado, consulte Atributos de implantação e configuração de pilha: Linux e Atributos de livros de receitas integrados. Para obter mais informações sobre atributos Ohai, consulte Ohai.

Quando ocorre um evento de ciclo de vida como Deploy ou Configure, ou quando você executa um comando de stack, como `execute_recipes` ou `update_packages`, o AWS OpsWorks Stacks faz o seguinte:

- Envia um comando correspondente para o agente em cada instância afetada.

  O agente executa as receitas apropriadas. Por exemplo, para um Implantar evento, o agente executa o Implantar receitas, seguido de quaisquer Implantar receitas personalizado.

- Une qualquer JSON personalizado e implementa atributos com os atributos de configuração do stack e instala-o nas instâncias.

Os atributos JSON personalizado, atributos de implantação e configuração de stack, atributos de livro de receitas e atributos Ohai são unidos em um objeto de nó, que fornece valores de atributo às receitas. Uma instância é essencialmente stateless contanto que os atributos de configuração de stack estejam envolvidos, incluindo qualquer JSON personalizado. Quando você executa um comando stack de implantação, as receitas associadas usam os atributos de configuração do stack baixados com o comando.

Tópicos

- Precedência de atributo
- Sobrepor atributos com JSON personalizado
- Substituindo atributos AWS OpsWorks de pilhas usando atributos personalizados do livro de receitas

Precedência de atributo

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Se um atributo for unicamente definido, Chef simplesmente o incorpora no objeto de nó. No entanto, qualquer fonte de atributo pode definir qualquer atributo, então é possível para o mesmo atributo ter múltiplas definições com valores diferentes. Por exemplo, livro de receitas `apache2` integrado define `node[:apache][:keepalive]`, mas você também pode definir esse atributo em JSON ou em um livro de receitas personalizados. Se um atributo tiver múltiplas definições, eles são avaliados em uma ordem descrita posteriormente e o objeto de nó recebe a definição com a precedência mais alta.

Um atributo é definido como seguinte:

```
node.type[:attribute][:sub_attribute][:...]=value
```

Se um atributo tiver várias definições, o tipo determina qual definição tem precedência e essa definição é incorporada ao objeto do nó. AWS OpsWorks O Stacks usa os seguintes tipos de atributos:

- padrão: esse é o tipo mais comum e significa essencialmente “usar este valor se o atributo ainda não tiver sido definido”. Se todas as definições de um atributo forem do tipo `default`, a primeira definição na ordem de avaliação tem precedência e os valores subsequentes são ignorados. Observe que o AWS OpsWorks Stacks define todas as definições de configuração e atributos de implantação da pilha como `default` tipo.
- normal: atributos com esse tipo sobrepõem quaisquer atributos `default` ou `normal` que foram definidos anteriormente na ordem da avaliação. Por exemplo, se o primeiro atributo for de um livro de receitas integrado e tiver um tipo `default` e o segundo for um atributo definido por usuário com um tipo `normal`, a segunda definição tem precedência.
- set: este é um tipo obsoleto que você pode ver em livros de receitas mais antigos. Foi anulado por `normal`, que tem a mesma precedência.

Chef oferece suporte a vários tipos de atributo adicionais, incluindo um tipo `automatic` que leva precedência sobre todas outras definições de atributo. As definições de atributo geradas pela ferramenta Ohai do Chef são todas do tipo `automatic`, então elas são, efetivamente, apenas leitura. Isso geralmente não é um problema, porque não há razão para substituí-los e eles são diferentes dos atributos dos AWS OpsWorks Stacks. No entanto, você deve ter cuidado ao nomear seus atributos de livro de receitas personalizados para que eles sejam diferentes dos atributos Ohai. Para obter mais informações, consulte Sobre atributos.

> **Note**
>
> A ferramenta Ohai é um executável que você pode rodar a partir da linha de comando. Para listar atributos Ohai de uma instância, entre na instância e execute `ohai` em uma janela de terminal. Esteja ciente de que isso produz uma saída muito longa.

Veja as etapas que incorporam as várias definições de atributo no objeto de nó:

1. Une quaisquer atributos de configuração de stack personalizados nos atributos de configuração de stack e implantação.

Atributos JSON personalizados podem ser definidos para o stack, ou para uma implantação em particular. Eles são os primeiros na ordem de avaliação e são tipos `normal` eficazes. Se um ou mais atributos de configuração de stack também estiverem definidos em JSON personalizado, os valores de JSON personalizado tomam precedência. Caso contrário, o AWS OpsWorks Stacks incorpora apenas os atributos JSON personalizados na configuração de stack.

2. Une quaisquer atributos JSON personalizados de implantação nos atributos de configuração de stack e implantação.

   Implantação de atributos JSON personalizados também é eficaz nos tipos `normal`, então eles tomam precedência sobre JSON de configuração de stack integrada e personalizada e JSON de implantação integrada.

3. Une os atributos de implementação e configuração do stack no objeto de nó da instância.
4. Une os atributos de livro de receitas integrados da instância no objeto de nó.

   Os atributos de livro de receitas integrados são todos do tipo `default`. Se um ou mais atributos de livro de receitas integrados também estiverem definidos nos atributos de configuração e implantação da pilha, normalmente porque você os definiu com um JSON personalizado, as definições de configuração da pilha terão precedência sobre as definições do livro de receitas incorporado. Todos os outros atributos de livro de receitas integrados são facilmente incorporados no objeto de nó.

5. Une os atributos de livro de receitas personalizados da instância no objeto de nó.

   Atributos de livro de receitas personalizados são, geralmente, do tipo `normal` ou `default`. Atributos únicos são incorporados no objeto de nó. Se quaisquer atributos de livro de receitas personalizado também foram definidos nas Etapas 1 a 3 (geralmente porque você os definiu com JSON personalizado), a precedência depende do tipo do atributo do livro de receitas personalizado:

   - Os atributos definidos nas Etapas 1 a 3 têm precedência sobre atributos de livros de receitas personalizados. `default`
   - Atributos `normal` de livros de receitas personalizados têm precedência sobre as definições das Etapas 1 a 3.

> **Important**
>
> Não use atributos `default` de livro de receitas personalizados para sobrepor configuração de stack ou atributos de livro de receitas integrados. Como atributos de livro de receitas personalizados são avaliados por último, os atributos `default` tem a menor precedência, e não podem sobrepor nada.

Sobrepor atributos com JSON personalizado

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Como o AWS OpsWorks Stacks gerencia a execução do Chef de forma diferente nas pilhas do Windows e nas pilhas do Linux, você não pode usar as técnicas discutidas nesta seção para pilhas do Windows.

A maneira mais simples de substituir um atributo AWS OpsWorks Stacks é defini-lo em JSON personalizado, que tem precedência sobre os atributos de configuração e implantação da pilha, bem como sobre os atributos integrados e personalizados do livro de receitas. `default` Para obter mais informações, consulte Precedência de atributo.

> **Important**
>
> Você deve sobrepor atributos de configuração de stack e implantação com cuidado. Por exemplo, substituir atributos no namespace `opsworks` pode interferir as receitas integradas. Para obter mais informações, consulte Configuração de pilha e atributos de implantação.

Você também pode usar JSON personalizado para definir atributos únicos, geralmente para transferir dados a suas receitas personalizadas. Os atributos são facilmente incorporados no objeto de nó, e receitas podem fazer referência a eles usando a sintaxe de nó Chef padrão.

Como especificar JSON personalizado

Para usar JSON para sobrepor um valor de atributo, você deve primeiro determinar o nome do atributo qualificado completamente por ele. Você então cria um objeto JSON que contém os atributos que deseja sobrepor, definido por seus valores preferidos. Por uma questão de comodidade, os documentos Atributos de implantação e configuração de pilha: Linux e Atributos de livros de receitas integrados comumente usou atributos de configuração de stack, implantação e livro de receitas integrado, incluindo seus nomes completamente qualificados.

As relações de filho-parente do objeto devem corresponder aos nós de Chef qualificados completamente apropriados. Por exemplo, suponha que você quer mudar os atributos Apache a seguir:

- O atributo `keepalivetimeout`, que tem o nó `node[:apache][:keepalivetimeout]` e um valor padrão de `3`.
- O atributo `logrotate` `schedule`, que tem um nó `node[:apache][:logrotate][:schedule]`, e um valor padrão de `"daily"`.

Para sobrepor os atributos e definir os valores para `5` e `"weekly"`, respectivamente, você usaria o JSON personalizado a seguir:

```
{
  "apache" : {
    "keepalivetimeout" : 5,
    "logrotate" : {
       "schedule" : "weekly"
    }
  }
}
```

Quando especificar JSON personalizado

É possível especificar uma estrutura JSON personalizada para as tarefas a seguir:

- Criar um novo stack

- Atualizar um stack
- Executar um comando stack
- Clonar um stack
- Implantar um aplicativo

Para cada tarefa, o AWS OpsWorks Stacks mescla os atributos JSON personalizados com os atributos de configuração e implantação da pilha e os envia às instâncias para serem mesclados ao objeto do nó. Entretanto, observe o seguinte:

- Se você especificar JSON personalizado quando criar, clonar ou atualizar um stack, os atributos são unidos em um nos atributos de configuração de stack e implantação para todos eventos de ciclo de vida subsequentes e comandos de stack.
- Se você especificar JSON personalizado para uma implantação, os atributos são unidos nos atributos de configuração de stack e implantação apenas para o evento correspondente.

  Se quiser usar esses atributos personalizados para implantações subsequentes, você deve especificar de forma explícita o JSON personalizado novamente.

É importante lembrar que os atributos afetam somente a instância quando são usados como receitas. Se você sobrepor um valor de atributo, mas nenhuma receita subsequente fizer referência ao atributo, a alteração não tem efeito. Você deve se certificar de que o JSON personalizado é enviado antes das receitas associadas serem executadas, ou certificar-se de que as receitas apropriadas são executadas novamente.

Melhores práticas do JSON personalizado

Você pode usar o JSON personalizado para substituir qualquer atributo do AWS OpsWorks Stacks, mas inserir as informações manualmente é um pouco complicado e não está sob nenhum tipo de controle de origem. JSON personalizado é mais eficaz para os seguintes fins:

- Quando você quiser sobrepor apenas um número pequeno de atributos, e não precisa usar livros de receita personalizados.

  Com o JSON personalizado, você pode evitar os custos indiretos de definir e manter um repositório de livro de receitas apenas para sobrepor um par de atributos.
- Valores confidenciais, como senhas ou chaves de autenticação.

Os atributos de livro de receitas são armazenados em um repositório, então, qualquer informação sensível tem o risco de ser comprometida. Em vez disso, defina atributos com valores fictícios e use JSON personalizado para definir os valores reais.

- Espera-se que os valores variem.

  Por exemplo, uma prática recomendada é ter seu stack de produção suportado por stacks de desenvolvimento separado e em estágio. Suponha que esses stacks suportam um aplicativo que aceita pagamentos. Se você usar JSON personalizado para especificar o endpoint do pagamento, pode especificar uma URL de teste para posicionar stack. Quando você estiver pronto para migrar um stack atualizado ao seu stack de produção, você pode usar os mesmos livros de receitas e usar JSON personalizado para definir o endpoint do pagamento à URL de produção.
- Valores que são específicos a um stack particular ou comando de implantação.

Substituindo atributos AWS OpsWorks de pilhas usando atributos personalizados do livro de receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Para pilhas do Windows, o AWS OpsWorks Stacks usa execuções separadas do Chef para receitas integradas e receitas personalizadas. Isso significa que você não pode usar as técnicas discutidas nesta seção para sobrepor atributos integrados para stacks do Windows.

O JSON personalizado é uma maneira conveniente de substituir a configuração da pilha do AWS OpsWorks Stacks e os atributos integrados do livro de receitas, mas tem algumas limitações. Em particular, você deve inserir manualmente o JSON para cada uso, portanto, não há nenhuma maneira eficiente para gerenciar as definições. Uma abordagem melhor é frequentemente usar

arquivos de atributo de livro de receitas para sobrepor atributos integrados. Fazer isso permite que você posicione as definições no controle de fonte.

O procedimento para usar arquivos de atributos personalizados para substituir as definições do AWS OpsWorks Stacks é simples.

Para substituir as definições de atributos AWS OpsWorks do Stacks

1. Defina um repositório de livro de receitas, conforme descrito em Livros de receitas e receitas.
2. Crie um livro de receitas com o mesmo nome que o livro de receitas integrado que contém os atributos que deseja sobrepor. Por exemplo, para sobrepor os atributos Apache, o livro de receitas deve ser nomeado apache2.
3. Adicione uma pasta `attributes` ao seu livro de receitas e adicione um arquivo à pasta nomeada `customize.rb`.
4. Adicione uma definição de atributo para o arquivo para cada um dos atributos de livro de receitas integrado que deseja sobrepor, defina para seu valor preferido. O atributo deve ser `normal` de um tipo ou superior e ter exatamente o mesmo nome de nó do atributo AWS OpsWorks Stacks correspondente. Para obter uma lista detalhada dos atributos do AWS OpsWorks Stacks, incluindo nomes de nós, consulte Atributos de implantação e configuração de pilha: Linux e. Atributos de livros de receitas integrados Para obter mais informações sobre atributos e arquivos de atributos, consulte Sobre arquivos de atributo.

   > **Important**
   >
   > Seus atributos devem ser do `normal` tipo para substituir os atributos do AWS OpsWorks Stacks; os `default` tipos não têm precedência. Por exemplo, se seu arquivo `customize.rb` contiver uma definição de atributo `default[:apache][:keepalivetimeout] = 5`, o atributo correspondente no arquivo de atributos `apache.rb` integrado é avaliado primeiro, e toma precedência. Para obter mais informações, consulte Sobrepor atributos.

5. Repita as etapas 2 a 4 para cada livro de receitas integrado com atributos que deseja sobrepor.
6. Ative livros de receitas personalizados para sua pilha e forneça as informações necessárias para que as AWS OpsWorks pilhas baixem seus livros de receitas para as instâncias da pilha. Para obter mais informações, consulte Instalação de livros de receitas personalizados.

> **Note**
>
> Para obter uma orientação mais detalhada sobre esse procedimento, consulte Sobrepor atributos integrados.

O objeto de nó usado por eventos de ciclo de vida subsequentes, comandos de implantação e comandos de pilha agora conterá suas definições de atributos em vez dos AWS OpsWorks valores de pilhas.

Por exemplo, para sobrepor as configurações integradas `keepalivetimeout` e `logrotate schedule` do Apache, discutidas em Como especificar JSON personalizado, adicione um livro de receitas `apache2` ao seu repositório e um arquivo `customize.rb` à pasta `attributes` do livro de receitas com os seguintes conteúdos.

```
normal[:apache][:keepalivetimeout] = 5
normal[:apache][:logrotate][:schedule] = 'weekly'
```

> **Important**
>
> Você não deve substituir os atributos do AWS OpsWorks Stacks modificando uma cópia do arquivo de atributos incorporado associado. Se, por exemplo, sua cópia `apache.rb` para sua pasta `apache2/attributes` e modificar algumas de suas configurações, você basicamente sobrepõem todos os atributos no arquivo integrado. Receitas usarão as definições de atributo de sua cópia e ignorarão o arquivo integrado. Se o AWS OpsWorks Stacks modificar posteriormente o arquivo de atributos integrado, as receitas não terão acesso às alterações, a menos que você atualize manualmente sua cópia.
>
> Para evitar essa situação, todos livros de receita integrados contêm um arquivo de atributos `customize.rb` vazio, que é necessário em todos módulos ao longo de um diretório `include_attribute`. Ao sobrepor atributos em sua cópia do `customize.rb`, você afeta apenas aqueles atributos específicos. As receitas obterão quaisquer outros valores de atributo dos arquivos de atributos integrados, e obterão automaticamente os valores atuais de qualquer atributo que você não sobrepôs.
>
> Essa abordagem ajuda você a manter o número de atributos em seu repositório de livro de receitas pequeno, que reduz seus custos indiretos de manutenção e torna as atualizações futuras mais fáceis de serem gerenciadas.

Estendendo arquivos de configuração de AWS OpsWorks pilhas usando modelos personalizados

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Como o AWS OpsWorks Stacks gerencia a execução do Chef de forma diferente nas pilhas do Windows e nas pilhas do Linux, você não pode usar as técnicas discutidas nesta seção para pilhas do Windows.

AWS OpsWorks O Stacks usa modelos para criar arquivos como arquivos de configuração, que normalmente dependem dos atributos de muitas das configurações. Se você usar atributos personalizados do JSON ou do livro de receitas personalizados para substituir as definições do AWS OpsWorks Stacks, suas configurações preferidas serão incorporadas aos arquivos de configuração no lugar das configurações do Stacks. AWS OpsWorks No entanto, o AWS OpsWorks Stacks não especifica necessariamente um atributo para cada configuração possível; ele aceita os padrões para algumas configurações e codifica outras diretamente no modelo. Você não pode usar atributos personalizados do JSON ou do livro de receitas personalizados para especificar as configurações preferidas se não houver um atributo AWS OpsWorks Stacks correspondente.

Você pode expandir o arquivo de configuração para incluir definições de configuração adicionais criando um modelo personalizado. Em seguida, você pode adicionar qualquer definição de configuração ou outro conteúdo de que precisa ao arquivo, e substituir as configurações codificadas. Para obter mais informações sobre modelos, consulte Modelos.

> **Note**
>
> Você pode substituir qualquer modelo integrado, exceto o opsworks-agent.monitrc.erb.

Para criar um modelo personalizado

1. Crie um livro de receitas com a mesma estrutura e os nomes de diretórios do livro de receitas integrado. Em seguida, crie um arquivo de modelo no diretório apropriado com o mesmo nome do modelo integrado que você deseja personalizar. Por exemplo, para usar um modelo personalizado para expandir o arquivo de configuração do Apache `httpd.conf`, você deve implementar um livro de receitas `apache2` em seu repositório e seu arquivo de modelo deve ser o `apache2/templates/default/apache.conf.erb`. Usar exatamente os mesmos nomes permite que o AWS OpsWorks Stacks reconheça o modelo personalizado e o use em vez do modelo incorporado.

   A abordagem mais simples é simplesmente copiar o arquivo de modelo integrado do GitHubrepositório do livro de receitas integrado para o seu livro de receitas e modificá-lo conforme necessário.

   > **Important**
   >
   > Não copie os arquivos do livro de receitas integrado, com exceção dos arquivos de modelo que você deseja personalizar. As cópias de outros tipos de arquivos de livros de receitas como, por exemplo, receitas criam recursos duplicados do Chef e podem causar erros.

   O livro de receitas também pode incluir atributos personalizados, receitas e arquivos relacionados, mas seus nomes de arquivos não devem ser duplicações dos nomes de arquivos integrados.

2. Personalize o arquivo de modelo para produzir um arquivo de configuração que atenda aos seus requisitos. Você pode adicionar mais configurações, excluir configurações existentes, substituir atributos codificados, e assim por diante.

3. Se você ainda não tiver feito isso, edite as configurações da pilha para habilitar os livros de receitas personalizados e especificar seu repositório de livros de receitas. Para obter mais informações, consulte Instalação de livros de receitas personalizados.

> **Note**
>
> Para obter uma orientação mais detalhada sobre esse procedimento, consulte Substituir modelos integrados.

Você não precisa implementar nenhuma receita ou adicionar receitas à configuração da camada para substituir um modelo. AWS OpsWorks O Stacks sempre executa as receitas integradas. Quando ele executa a receita que cria o arquivo de configuração, usa automaticamente o seu modelo personalizado em vez do modelo integrado.

> **Note**
>
> Se o AWS OpsWorks Stacks fizer alguma alteração no modelo integrado, seu modelo personalizado poderá ficar fora de sincronia e não funcionar mais corretamente. Por exemplo, suponha que seu modelo se refira a um arquivo dependente e o nome do arquivo seja alterado. AWS OpsWorks O Stacks não faz essas alterações com frequência e, quando um modelo muda, ele lista as alterações e oferece a opção de atualizar para uma nova versão. Você deve monitorar o repositório AWS OpsWorks Stacks em busca de alterações e atualizar manualmente seu modelo conforme necessário.

Extensão de uma camada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Algumas vezes, você precisa personalizar uma camada integrada além do que é possível lidar com a modificação de atributos do AWS OpsWorks Stacks ou com a personalização de modelos. Por exemplo, suponha que você precisa criar symlinks, definir modos de arquivos ou de pastas, instalar pacotes adicionais,e assim por diante. Você deve estender as camadas personalizadas para

fornecer mais do que a funcionalidade mínima. Neste caso, será necessário implementar um ou mais livros de receitas personalizados com receitas para lidar com as tarefas de personalização. Este tópico fornece alguns exemplos de como usar as receitas para estender uma camada.

Se você ainda não conhece o Chef, primeiro deverá ler Introdução a livros de receitas, que é um tutorial que apresenta as noções básicas de como implementar os livros de receitas para realizar uma variedade de tarefas comuns. Para obter um exemplo detalhado de como implementar uma camada personalizada, consulte Criação de uma camada de servidor Tomcat personalizada.

Tópicos

- Uso de receitas para executar scripts
- Uso de hooks de implantação do Chef
- Execução de trabalhos cron em instâncias do Linux
- Instalação e configuração de pacotes em instâncias do Linux

Uso de receitas para executar scripts

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Se você já tiver um script que realize as tarefas de personalização necessárias, a abordagem mais simples para estender uma camada quase sempre será implementar uma receita simples para executar o script. Assim, você poderá atribuir a receita aos eventos de ciclo de vida adequados, normalmente a configuração ou a implantação, ou usar o comando de pilha`execute_recipes` para executar a receita manualmente.

O exemplo a seguir executa um script de shell em instâncias Linux, mas você pode usar a mesma abordagem para outros tipos de script, incluindo PowerShell scripts do Windows.

```
cookbook_file "/tmp/lib-installer.sh" do
```

```
  source "lib-installer.sh"
  mode 0755
end

execute "install my lib" do
  command "sh /tmp/lib-installer.sh"
end
```

O recurso `cookbook_file` representa um arquivo que é armazenado em um subdiretório do diretório `files` de um livro de receitas e transfere o arquivo para um local especificado na instância. Este exemplo transfere um script de shell, `lib-installer.sh`, para o diretório `/tmp` da instância e define o modo de arquivo como `0755`. Para obter mais informações, consulte cookbook_file.

O recurso `execute` representa um comando, como um comando de shell. Este exemplo executa `lib-installer.sh`. Para obter mais informações, consulte execute.

Você também pode executar um script incorporando-o a uma receita. O exemplo a seguir executa um script bash, mas o Chef também oferece suporte a Csh, Perl, Python e Ruby.

```
script "install_something" do
  interpreter "bash"
  user "root"
  cwd "/tmp"
  code <<-EOH
    #insert bash script
  EOH
end
```

O recurso `script` representa um script. O exemplo especifica um intérprete bash, define o usuário como `"root"` e define o diretório de trabalho como `/tmp`. Em seguida, ele executa o script bash no bloco `code`, o que pode incluir quantas linhas forem necessárias. Para obter mais informações, consulte script.

Para obter mais informações sobre como usar receitas para executar scripts, consulte Exemplo 7: Execução de comandos e scripts. Para ver um exemplo de como executar um PowerShell script em uma instância do Windows, consulteExecutando um PowerShell script do Windows.

Uso de hooks de implantação do Chef

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode personalizar a implantação ao implementar uma receita personalizada para realizar as tarefas necessárias e atribuí-la ao evento Implantar da camada apropriada. Uma abordagem alternativa e algumas vezes mais simples, especialmente se não for necessário implementar um livro de receitas para outros fins, é usar hooks de implantação do Chef para executar o seu código de personalização. Além disso, o Implantar receitas personalizado é executado depois que a implantação tiver sido realizada pelas receitas integradas. Os hooks de implantação permitem que você interaja durante uma implantação, por exemplo, após o check-out do código do aplicativo do repositório, mas antes do reinício do Apache.

O Chef implanta aplicativos em quatro estágios:

- Check-out: faz download dos arquivos do repositório
- Migrar: executa uma migração conforme necessário
- Symlink: cria symlinks
- Reiniciar: reinicia o aplicativo

Os hooks de implantação do Chef oferecem uma maneira simples de personalizar uma implantação ao executarem opcionalmente um aplicativo Ruby fornecido pelo usuário após a conclusão de cada estágio. Para usar os hooks de implantação, implemente um ou mais aplicativos Ruby e coloque-os no diretório `/deploy` do seu aplicativo. (Se seu aplicativo não tiver um diretório `/deploy`, crie um no nível `APP_ROOT`). O aplicativo deve ter um dos seguintes nomes, que determina quando ele será executado.

- `before_migrate.rb` é executado após a conclusão do estágio Check-out, mas antes de Migração.
- `before_symlink.rb` é executado após a conclusão do estágio Migração, mas antes de Symlink.

- `before_restart.rb` é executado após a conclusão do estágio Symlink, mas antes de Reinício.
- `after_restart.rb` é executado após a conclusão do estágio Reinício.

Os hooks de implantação do Chef podem acessar o objeto de nó usando a sintaxe padrão do nó, assim como acontece com as receitas. Os hooks de implantação também podem acessar os valores de qualquer variável de ambiente de aplicativo especificada. No entanto, você deve usar `new_resource.environment["VARIABLE_NAME"]` para acessar o valor da variável em vez de `ENV["VARIABLE_NAME"]`.

Execução de trabalhos cron em instâncias do Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Um trabalho cron do Linux direciona o daemon cron para executar um ou mais comandos em uma programação especificada. Por exemplo, suponha que sua pilha seja compatível com um aplicativo de comércio eletrônico PHP. Você pode configurar um trabalho cron para que o servidor envie um relatório de vendas em um horário especificado a cada semana. Para obter mais informações sobre cron, consulte cron na Wikipédia. Para obter mais informações sobre como executar um trabalho cron diretamente em uma instância ou computador baseado em Linux, consulte O que são cron e crontab, e como usá-los? (em inglês), no site da base de dados de conhecimento da Universidade de Indiana.

Embora você possa configurar manualmente os trabalhos `cron` em instâncias baseadas no Linux individuais ao conectar-se a eles com SSH e ao editar suas entradas `crontab`, uma das principais vantagens do AWS OpsWorks Stacks é que você pode direcioná-lo para executar a tarefa em uma camada de instâncias inteira. O procedimento a seguir descreve como configurar um trabalho `cron` em instâncias de uma camada PHP App Server, mas você pode usar a mesma abordagem com qualquer camada.

Para configurar um trabalho **`cron`** nas instâncias de uma camada

1. Implemente um livro de receitas com uma receita com um recurso `cron` que configure o trabalho. O exemplo supõe que a receita se chama `cronjob.rb`; os detalhes da implementação serão descritos mais tarde. Para obter mais informações sobre livros de receitas e receitas, consulte Livros de receitas e receitas.
2. Instale o livro de receitas em sua pilha. Para obter mais informações, consulte Instalação de livros de receitas personalizados.
3. Faça com que o AWS OpsWorks Stacks execute a receita automaticamente nas instâncias da camada, atribuindo-a aos seguintes eventos do ciclo de vida. Para obter mais informações, consulte Execução automática de receitas.

   - Configuração — A atribuição `cronjob.rb` a esse evento faz com que o AWS OpsWorks Stacks execute a receita em todas as novas instâncias.
   - Implantar — A atribuição `cronjob.rb` a esse evento faz com que o AWS OpsWorks Stacks execute a receita em todas as instâncias on-line quando você implanta ou reimplanta um aplicativo na camada.

   Você também pode executar manualmente a receita em instâncias online usando o comando `Execute Recipes` da pilha. Para obter mais informações, consulte Executa comandos de pilha.

A seguir, o exemplo de `cronjob.rb` configura um trabalho cron para executar um aplicativo PHP implementado pelo usuário uma vez por semana que coleta os dados de vendas do servidor e envia um relatório por e-mail. Para obter mais exemplos de como usar um recurso cron, consulte cron.

```
cron "job_name" do
  hour "1"
  minute "10"
  weekday "6"
  command "cd /srv/www/myapp/current && php .lib/mailing.php"
end
```

O `cron` é um recurso do Chef que representa um trabalho `cron`. Quando o AWS OpsWorks Stacks executa a receita em uma instância, o provedor associado trata dos detalhes da configuração do trabalho.

- *job_name* é um nome definido pelo usuário para o trabalho `cron`, como `weekly report`.
- `hour`/`minute`/`weekday` especificam quando os comandos devem ser executados. Este exemplo executa os comandos todos os sábados à 1:10h.
- `command` especifica os comandos a serem executados.

  Este exemplo executa dois comandos. O primeiro navega até o diretório `/srv/www/myapp/current`. O segundo executa o aplicativo `mailing.php` implementado pelo usuário, que coleta os dados de vendas e envia o relatório.

> **Note**
>
> Por padrão, o comando `bundle` não funciona com trabalhos `cron`. O motivo é que o AWS OpsWorks Stacks instala o bundler no diretório. `/usr/local/bin` Para usar `bundle` com um trabalho `cron`, você deve adicionar explicitamente o caminho `/usr/local/bin` ao trabalho cron. Além disso, como a variável de ambiente $PATH não pode se expandir no trabalho `cron`, uma melhor prática é adicionar explicitamente qualquer informação sobre caminho necessária ao trabalho sem depender da expansão da variável $PATH. Os exemplos a seguir mostram duas maneiras de usar `bundle` em um trabalho `cron`.
>
> ```
> cron "my first task" do
>  path "/usr/local/bin"
>  minute "*/10"
>  command "cd /srv/www/myapp/current && bundle exec my_command"
> end
> ```
>
> ```
> cron_env = {"PATH" => "/usr/local/bin"}
> cron "my second task" do
>  environment cron_env
>  minute "*/10"
>  command "cd /srv/www/myapp/current && /usr/local/bin/bundle exec my_command"
> end
> ```

Se a sua pilha tiver vários servidores de aplicativos, a atribuição de `cronjob.rb` aos eventos de ciclo de vida da camada PHP App Server pode não ser uma abordagem ideal. Por exemplo, a receita

é executada em todas as instâncias da camada e, portanto, você receberá vários relatórios. Uma abordagem melhor é usar uma camada personalizada para garantir que apenas um servidor envie um relatório.

Para executar uma receita em apenas uma das instâncias de uma camada

1. Crie uma camada personalizada chamada, por exemplo, PHPAdmin e `cronjob.rb` atribua aos seus eventos de configuração e implantação. As camadas personalizadas não precisam necessariamente fazer muito. Nesse caso, PHPAdmin basta executar uma receita personalizada em suas instâncias.
2. Atribua uma das instâncias do PHP App Server AdminLayer a. Se uma instância pertencer a mais de uma camada, o AWS OpsWorks Stacks executa as receitas integradas e personalizadas de cada camada.

Como apenas uma instância pertence ao PHP App Server e às PHPAdmin camadas, `cronjob.rb` é executada somente nessa instância e você recebe apenas um relatório.

Instalação e configuração de pacotes em instâncias do Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As camadas integradas dão suporte apenas a determinados pacotes. Para obter mais informações, consulte Camadas. Você pode instalar outros pacotes, como um servidor Redis, ao implementar as receitas personalizadas para lidar com as tarefas associadas de instalação, de configuração e de implantação. Em alguns casos, a melhor abordagem é estender uma camada integrada para que ela instale o pacote em suas instâncias junto com os pacotes padrão da camada. Por exemplo, se você tiver uma pilha que ofereça suporte a um aplicativo PHP e se quiser incluir um servidor Redis, poderá estender a camada PHP App Server para instalar e configurar um servidor Redis nas instâncias da camada além de um servidor de aplicativos PHP.

Uma receita de instalação de pacote normalmente precisa executar tarefas como estas:

- Crie um ou mais diretórios e defina seus modos.
- Crie um arquivo de configuração desde um modelo.
- Execute o instalador para instalar o pacote na instância.
- Inicie um ou mais serviços.

Para obter um exemplo de como instalar um servidor Tomcat, consulte Criação de uma camada de servidor Tomcat personalizada. O tópico descreve como configurar uma camada personalizada do Redis, mas você pode usar o mesmo código para instalar e configurar o camada em uma camada integrada. Para ver exemplos de como instalar outros pacotes, consulte os livros de receitas integrados, em https://github.com/aws/opsworks-cookbooks.

Criação de uma camada de servidor Tomcat personalizada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este tópico descreve como implementar uma camada personalizada para uma pilha do Linux. No entanto, os princípios básicos e parte do código também podem ser adaptados para implementar camadas personalizadas para pilhas do Windows, especialmente aquelas na seção sobre implantação de aplicações.

A maneira mais simples de usar pacotes não padrão em instâncias do AWS OpsWorks Stacks é estender uma camada existente. No entanto, essa abordagem instala e executa os pacotes padrão e não padrão nas instâncias da camada, o que nem sempre é desejável. Uma abordagem um pouco mais exigente, mas mais eficiente, é implementar uma camada personalizada, que oferece a você um controle quase completo sobre as instâncias da camada, inclusive o seguinte:

- Quais pacotes são instalados

- Como cada pacote está configurado
- Como implantar aplicações de um repositório na instância

Independentemente de usar o console ou a API, você cria e gerencia uma camada personalizada da mesma maneira que qualquer outra camada, conforme descrito em Camadas personalizadas. No entanto, as receitas integradas de uma camada personalizada realizam apenas algumas tarefas bem básicas, como instalar um cliente do Ganglia para relatar métricas a um mestre do Ganglia. Para deixar as instâncias de uma camada personalizada mais do que minimamente funcionais, você deve implementar um ou mais livros de receitas personalizados com receitas do Chef e arquivos relacionados para processar as tarefas de instalação e configuração de pacotes, implantação de aplicações etc. Porém, você não precisa necessariamente implementar tudo do zero. Por exemplo, se armazenar aplicações em um dos repositórios padrão, você poderá usar as receitas de implantação integradas para processar boa parte do trabalho de instalação das aplicações nas instâncias da camada.

> **Note**
>
> Se você ainda não conhece o Chef, primeiro deverá ler Introdução a livros de receitas, que é um tutorial que apresenta as noções básicas de como implementar os livros de receitas para realizar uma variedade de tarefas comuns.

O passo a passo a seguir descreve como implementar uma camada personalizada que dê suporte a um servidor de aplicações Tomcat. A camada se baseia em um livro de receitas personalizado chamado Tomcat, que inclui receitas para processar a instalação de pacotes, a implantação etc. O passo a passo inclui trechos do livro de receitas Tomcat. Você pode baixar o livro de receitas completo em seu GitHub repositório. Caso não esteja familiarizado com Opscode Chef, você deve ler primeiramente Livros de receitas e receitas.

> **Note**
>
> AWS OpsWorks O Stacks inclui uma camada completa do Java App Server para uso em produção. A finalidade do livro de receitas do Tomcat é mostrar como implementar camadas personalizadas, de maneira que ele só dê suporte a uma versão limitada do Tomcat que não inclui recursos como SSL. Para obter um exemplo de uma implementação com todos os recursos, consulte o livro de receitas opsworks_java integrado.

O livro de receitas do Tomcat dá suporte a uma camada personalizada cujas instâncias têm as seguintes características:

- Elas dão suporte a um servidor de aplicações Java Tomcat com um front-end do Apache.
- O Tomcat é configurado para permitir que aplicações usem um objeto `DataSource` JDBC para se conectar a uma instância MySQL à parte, que funciona como um armazenamento de dados back-end.

O livro de receitas deste projeto envolve vários componentes principais:

- Arquivo de atributos contém definições de configuração usadas por várias receitas.
- Receitas de instalação são atribuídas ao evento do ciclo de vida de configuração da camada. Eles serão executados depois que uma instância tiver sido inicializada e realizarão tarefas como a instalação de pacotes e a criação de arquivos de configuração.
- Receitas de configuração são atribuídas ao evento do ciclo de vida de configuração da camada. Elas serão executadas depois das alterações de configuração da pilha, principalmente quando as instâncias ficam on-line ou off-line, e processarão todas as alterações obrigatórias feitas na configuração.
- Receitas de implantação são atribuídas ao evento do ciclo de vida de implantação da camada. Elas serão executadas depois das receitas de instalação e quando você implantar manualmente uma aplicação para instalar o código e os arquivos relacionados nas instâncias de camada e processar tarefas relacionadas, como reiniciar serviços.

A seção final, Criar uma pilha e executar uma aplicação, descreve como criar uma pilha que inclua uma camada personalizada com base no livro de receitas do Tomcat e como implantar e executar uma aplicação JSP simples que exiba dados de um banco de dados do MySQL em execução em uma instância que pertença a uma camada MySQL separada.

> **Note**
>
> As receitas do livro de receitas do Tomcat dependem de algumas receitas integradas do AWS OpsWorks Stacks. Para esclarecer a origem de cada receita, este tópico identifica receitas que usam a convenção nome_do_livro_de_receitas::nome_da_receita do Chef.

Tópicos

- Arquivo de atributos
- Receitas de instalação
- Receitas de configuração
- Receitas de implantação
- Criar uma pilha e executar uma aplicação

Arquivo de atributos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Antes de observar as receitas, será útil examinar primeiramente o arquivo de atributos do livro de receitas do Tomcat, que contém várias definições de configuração usadas pelas receitas. Os atributos não são obrigatórios. Você pode simplesmente codificar esses valores nas receitas ou nos modelos. No entanto, se você definir as configurações usando atributos, poderá usar o console ou a API do AWS OpsWorks Stacks para modificar os valores definindo atributos JSON personalizados, o que é mais simples e flexível do que reescrever a receita ou o código do modelo toda vez que quiser alterar uma configuração. Essa abordagem permite, por exemplo, usar o mesmo livro de receitas para várias pilhas, mas configurar o servidor Tomcat de maneira diferente para cada pilha. Para obter mais informações sobre atributos e como substitui-los, consulte Sobrepor atributos.

O exemplo a seguir mostra o arquivo de atributos completo, `default.rb`, localizado no diretório `attributes` do livro de receitas do Tomcat.

```
default['tomcat']['base_version'] = 6
default['tomcat']['port'] = 8080
default['tomcat']['secure_port'] = 8443
default['tomcat']['ajp_port'] = 8009
default['tomcat']['shutdown_port'] = 8005
default['tomcat']['uri_encoding'] = 'UTF-8'
```

```
default['tomcat']['unpack_wars'] = true
default['tomcat']['auto_deploy'] = true
case node[:platform]
when 'centos', 'redhat', 'fedora', 'amazon'
  default['tomcat']['java_opts'] = ''
when 'debian', 'ubuntu'
  default['tomcat']['java_opts'] = '-Djava.awt.headless=true -Xmx128m -XX:
+UseConcMarkSweepGC'
end
default['tomcat']['catalina_base_dir'] = "/etc/tomcat#{node['tomcat']['base_version']}"
default['tomcat']['webapps_base_dir'] = "/var/lib/tomcat#{node['tomcat']
['base_version']}/webapps"
default['tomcat']['lib_dir'] = "/usr/share/tomcat#{node['tomcat']['base_version']}/lib"
default['tomcat']['java_dir'] = '/usr/share/java'
default['tomcat']['mysql_connector_jar'] = 'mysql-connector-java.jar'
default['tomcat']['apache_tomcat_bind_mod'] = 'proxy_http' # or: 'proxy_ajp'
default['tomcat']['apache_tomcat_bind_config'] = 'tomcat_bind.conf'
default['tomcat']['apache_tomcat_bind_path'] = '/tc/'
default['tomcat']['webapps_dir_entries_to_delete'] = %w(config log public tmp)
case node[:platform]
when 'centos', 'redhat', 'fedora', 'amazon'
  default['tomcat']['user'] = 'tomcat'
  default['tomcat']['group'] = 'tomcat'
  default['tomcat']['system_env_dir'] = '/etc/sysconfig'
when 'debian', 'ubuntu'
  default['tomcat']['user'] = "tomcat#{node['tomcat']['base_version']}"
  default['tomcat']['group'] = "tomcat#{node['tomcat']['base_version']}"
  default['tomcat']['system_env_dir'] = '/etc/default'
end
```

As configurações propriamente ditas serão abordadas depois na seção relacionada. As seguintes observações normalmente se aplicam:

- Como todas as definições de nó são do tipo `default`, você pode substituí-las por atributos JSON personalizados.
- O arquivo usa uma instrução `case` para definir condicionalmente alguns valores de atributo com base no sistema operacional da instância.

  O nó `platform` é gerado pela ferramenta Ohai do Chef e representa o sistema operacional da instância.

Receitas de instalação

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As receitas de instalação são atribuídas ao evento do ciclo de vida de configuração da camada e executadas após a inicialização de uma instância. Elas realizam tarefas como instalar pacotes, criar arquivos de configuração e iniciar serviços. Depois que as receitas de configuração terminarem de ser AWS OpsWorks executadas, o Stacks executa as receitas de implantação para implantar qualquer aplicativo na nova instância.

Tópicos

- tomcat::setup
- tomcat::install
- tomcat::service
- tomcat::container_config
- tomcat::apache_tomcat_bind

tomcat::setup

A receita `tomcat::setup` deve ser atribuída a um evento do ciclo de vida de configuração de uma camada.

```
include_recipe 'tomcat::install'
include_recipe 'tomcat::service'

service 'tomcat' do
  action :enable
end
```

```
# for EBS-backed instances we rely on autofs
bash '(re-)start autofs earlier' do
  user 'root'
  code <<-EOC
    service autofs restart
  EOC
  notifies :restart, resources(:service => 'tomcat')
end

include_recipe 'tomcat::container_config'
include_recipe 'apache2'
include_recipe 'tomcat::apache_tomcat_bind'
```

A receita `tomcat::setup` é mais uma metarreceita. Ela inclui um conjunto de receitas dependentes que processam a maioria dos detalhes de como instalar e configurar o Tomcat e os pacotes relacionados. A primeira parte do `tomcat::setup` executa as seguintes receitas, que serão abordadas posteriormente:

- A receita tomcat::install instala o pacote do servidor Tomcat.
- A receita tomcat::service configura o serviço Tomcat.

A parte intermediária de `tomcat::setup` habilita e inicia o serviço Tomcat:

- O service resource do Chef habilita o serviço Tomcat na inicialização.
- O bash resource do Chef executa um script Bash para iniciar o daemon autofs, necessário para instâncias com suporte do Amazon EBS. O recurso acaba notificando o recurso `service` para reiniciar o serviço Tomcat.

  Para obter mais informações, consulte: autofs (para Amazon Linux) ou Autofs (para Ubuntu).

A parte final de `tomcat::setup` cria arquivos de configuração, além de instalar e configurar o servidor Apache front-end:

- A receita tomcat::container_config cria arquivos de configuração.
- A `apache2` receita (que é uma abreviatura de `apache2::default`) é uma receita integrada do AWS OpsWorks Stacks que instala e configura um servidor Apache.
- A receita tomcat::apache_tomcat_bind configura o servidor Apache para funcionar como um front-end para o servidor Tomcat.

> **Note**
>
> Muitas vezes, você pode economizar tempo e esforço usando receitas integradas para realizar algumas das tarefas obrigatórias. Esta receita usa a receita `apache2::default` integrada para instalar o Apache, em vez de implementá-lo do zero. Para obter outro exemplo de como usar receitas integradas, consulte Receitas de implantação.

As seções a seguir descrevem as receitas de instalação do livro de receitas Tomcat em mais detalhes. Para obter mais informações sobre as receitas `apache2`, consulte opsworks-cookbooks/apache2.

tomcat::install

A receita `tomcat::install` instala o servidor Tomcat, o OpenJDK e uma biblioteca de conectores Java que processa a conexão com o servidor MySQL.

```
tomcat_pkgs = value_for_platform(
  ['debian', 'ubuntu'] => {
    'default' => ["tomcat#{node['tomcat']['base_version']}", 'libtcnative-1',
 'libmysql-java']
  },
  ['centos', 'redhat', 'fedora', 'amazon'] => {
    'default' => ["tomcat#{node['tomcat']['base_version']}", 'tomcat-native', 'mysql-
connector-java']
  },
  'default' => ["tomcat#{node['tomcat']['base_version']}"]
)

tomcat_pkgs.each do |pkg|
  package pkg do
    action :install
  end
end

link ::File.join(node['tomcat']['lib_dir'], node['tomcat']['mysql_connector_jar']) do
  to ::File.join(node['tomcat']['java_dir'], node['tomcat']['mysql_connector_jar'])
  action :create
end

# remove the ROOT webapp, if it got installed by default
```

```
include_recipe 'tomcat::remove_root_webapp'
```

A receita executa as seguintes tarefas:

1. Cria uma lista de pacotes a serem instalados, dependendo do sistema operacional da instância.
2. Instala cada pacote na lista.

   O package resource do Chef usa o provedor apropriado (`yum` para Amazon Linux e `apt-get` para Ubuntu) a fim de processar a instalação. Os provedores de pacotes instalam o OpenJDK como uma dependência do Tomcat, mas a biblioteca de conectores MySQL deve ser instalada explicitamente.
3. Usa um recurso link resource do Chef para criar um symlink no diretório lib do servidor do Tomcat para o a biblioteca de conectores do MySQL no JDK.

   Usando os valores de atributo padrão, o diretório lib do Tomcat é `/usr/share/tomcat6/lib` e a biblioteca de conectores do MySQL (`mysql-connector-java.jar`) está em `/usr/share/java/`.

A receita `tomcat::remove_root_webapp` remove o aplicativo web ROOT (`/var/lib/tomcat6/webapps/ROOT` por padrão) para evitar alguns problemas de segurança.

```
ruby_block 'remove the ROOT webapp' do
  block do
    ::FileUtils.rm_rf(::File.join(node['tomcat']['webapps_base_dir'], 'ROOT'), :secure
 => true)
  end
  only_if { ::File.exists?(::File.join(node['tomcat']['webapps_base_dir'], 'ROOT'))
 && !::File.symlink?(::File.join(node['tomcat']['webapps_base_dir'], 'ROOT')) }
end
```

A instrução `only_if` garante que a receita só remova o arquivo caso ele exista.

> **Note**
>
> A versão do Tomcat é especificada pelo atributo `['tomcat']['base_version']`, definido como 6 no arquivo de atributos. Para instalar o Tomcat 7, você pode usar atributos JSON personalizados para substituir o atributo. Basta editar as configurações da pilha e inserir o seguinte JSON na caixa Custom Chef JSON ou adicioná-la a qualquer JSON personalizado existente:

```
{
  'tomcat' : {
    'base_version' : 7
  }
}
```

O atributo JSON personalizado substitui o atributo padrão e define a versão do Tomcat como 7. Para obter mais informações sobre como substituir esses atributos, consulte Sobrepor atributos.

tomcat::service

A receita `tomcat::service` cria a definição de serviço do Tomcat.

```
service 'tomcat' do
  service_name "tomcat#{node['tomcat']['base_version']}"

  case node[:platform]
  when 'centos', 'redhat', 'fedora', 'amazon'
    supports :restart => true, :reload => true, :status => true
  when 'debian', 'ubuntu'
    supports :restart => true, :reload => false, :status => true
  end

  action :nothing
end
```

A receita usa o service resource do Chef para especificar o nome do serviço do Tomcat (tomcat6, por padrão) e define o atributo `supports` para determinar como o Chef gerencia os comandos de reinicialização, recarregamento e status do serviço nos sistemas operacionais diferentes.

- `true` indica que o Chef pode usar o script init ou outro provedor de serviços para executar o comando
- `false` indica que o Chef deve tentar executar o comando por outros meios.

O `action` é definido como `:nothing`. Para cada evento do ciclo de vida, o AWS OpsWorks Stacks inicia uma execução do Chef para executar o conjunto apropriado de receitas. O livro de receitas do Tomcat segue um padrão comum de fazer uma receita criar a definição de serviço, mas não reiniciá-lo. Outras receitas na execução do Chef processam a reinicialização, normalmente incluindo um comando `notifies` nos recursos `template` usados para criar arquivos de configuração. Notificações são uma maneira prática de reiniciar um serviço porque elas só fazem isso caso a configuração tenha sido alterada. Além disso, caso uma execução do Chef tenha várias notificações de reinicialização para um serviço, o Chef reinicia o serviço uma vez, no máximo. Essa prática evita problemas que possam ocorrer durante a tentativa de reiniciar um serviço que não esteja totalmente operacional, que é uma fonte comum de erros do Tomcat.

O serviço Tomcat deve ser definidos para qualquer execução do Chef que use notificações de reinicialização. Portanto, `tomcat::service` é incluído em vários receitas, para garantir que o serviço seja definido para cada execução do Chef. Não haverá penalidade se uma execução do Chef incluir várias instâncias de `tomcat::service` porque o Chef garante que uma receita seja executada somente uma vez por execução, independentemente de quantas vezes ele seja incluído.

tomcat::container_config

A receita `tomcat::container_config` cria arquivos de configuração com base em arquivos de modelo do livro de receitas.

```
include_recipe 'tomcat::service'

template 'tomcat environment configuration' do
  path ::File.join(node['tomcat']['system_env_dir'], "tomcat#{node['tomcat']
['base_version']}")
  source 'tomcat_env_config.erb'
  owner 'root'
  group 'root'
  mode 0644
  backup false
  notifies :restart, resources(:service => 'tomcat')
end

template 'tomcat server configuration' do
  path ::File.join(node['tomcat']['catalina_base_dir'], 'server.xml')
  source 'server.xml.erb'
  owner 'root'
  group 'root'
```

```
  mode 0644
  backup false
  notifies :restart, resources(:service => 'tomcat')
end
```

A receita chama primeiro `tomcat::service`, que define o serviço, se necessário. A maior parte da receita consiste em dois template resources, cada um criando um arquivo de configuração com base em um dos arquivos de modelo do livro de receitas, define as propriedades do arquivo e notifica o Chef para reiniciar o serviço.

Arquivo de configuração do ambiente do Tomcat

O primeiro recurso `template` usa o arquivo de modelo `tomcat_env_config.erb` para criar um arquivo de configuração do ambiente do Tomcat, usado para definir variáveis do ambiente, como `JAVA_HOME`. O nome do arquivo padrão é o argumento do recurso `template`. `tomcat::container_config` usa um atributo `path` para substituir o valor padrão e o nome do arquivo de configuração `/etc/sysconfig/tomcat6` (Amazon Linux) ou `/etc/default/tomcat6` (Ubuntu). O recurso `template` também especifica o proprietário, o grupo e as configurações de modo do arquivo e leva o Chef a não criar arquivos de backup.

Caso você observe o código-fonte, na verdade, há três versões de `tomcat_env_config.erb`, cada uma em um subdiretório diferente do diretório `templates`. Os diretórios `ubuntu` e `amazon` contêm os modelos para os respectivos sistemas operacionais. A pasta `default` contém um modelo fictício com uma única linha de comentário, usada apenas caso você tente executar essa receita em uma instância com um sistema operacional não compatível. A receita `tomcat::container_config` não precisa especificar qual `tomcat_env_config.erb` usar. O Chef escolhe automaticamente o diretório apropriado para o sistema operacional da instância com base em regras descritas em File Specificity.

Os arquivos `tomcat_env_config.erb` deste exemplo consistem em grande parte de comentários. Para definir variáveis de ambiente adicionais, basta excluir o comentário das linhas apropriadas e fornecer os valores preferidos.

> **Note**
>
> Qualquer definição de configuração que possa mudar deve ser determinada como um atributo, em vez de codificada no modelo. Assim, você não precisa reescrever o modelo para alterar uma configuração, bastando substituir o atributo.

O modelo do Amazon Linux define apenas uma variável de ambiente, conforme mostrado no trecho a seguir.

```
...
# Use JAVA_OPTS to set java.library.path for libtcnative.so
#JAVA_OPTS="-Djava.library.path=/usr/lib"

JAVA_OPTS="${JAVA_OPTS} <%= node['tomcat']['java_opts'] %>"

# What user should run tomcat
#TOMCAT_USER="tomcat"
...
```

JAVA_OPTS pode ser usado para especificar opções de Java, como o caminho da biblioteca. Usando os valores de atributo padrão, o modelo não define opções de Java para o Amazon Linux. Você pode definir as próprias opções de Java substituindo o atributo `['tomcat']['java_opts']`, por exemplo, usando atributos JSON personalizados. Para obter um exemplo, consulte Criar um stack.

O modelo do Ubuntu define várias variáveis de ambiente, conforme mostrado no trecho do modelo a seguir.

```
# Run Tomcat as this user ID. Not setting this or leaving it blank will use the
# default of tomcat<%= node['tomcat']['base_version'] %>.
TOMCAT<%= node['tomcat']['base_version'] %>_USER=tomcat<%= node['tomcat']
['base_version'] %>
...
# Run Tomcat as this group ID. Not setting this or leaving it blank will use
# the default of tomcat<%= node['tomcat']['base_version'] %>.
TOMCAT<%= node['tomcat']['base_version'] %>_GROUP=tomcat<%= node['tomcat']
['base_version'] %>
...
JAVA_OPTS="<%= node['tomcat']['java_opts'] %>"

<% if node['tomcat']['base_version'].to_i < 7 -%>
# Unset LC_ALL to prevent user environment executing the init script from
# influencing servlet behavior.  See Debian bug #645221
unset LC_ALL
<% end -%>
```

Usando valores de atributo padrão, o modelo define as variáveis de ambiente do Ubuntu da seguinte forma:

- `TOMCAT6_USER` e `TOMCAT6_GROUP`, que representam o usuário e o grupo do Tomcat, são definidos como `tomcat6`.

  Caso você defina ['tomcat'] ['base_version'] como `tomcat7`, os nomes de variável são resolvidos como `TOMCAT7_USER` e `TOMCAT7_GROUP`, e ambos são definidos como `tomcat7`.
- `JAVA_OPTS` é definido como `-Djava.awt.headless=true -Xmx128m -XX:+UseConcMarkSweepGC`:
  - A definição de `-Djava.awt.headless` como `true` informa o mecanismo de gráficos que a instância é descentralizada e não tem um console, o que resolve o comportamento de determinadas aplicações gráficas.
  - `-Xmx128m` garante que o JVM tenha recursos de memória adequados, 128 MB para este exemplo.
  - `-XX:+UseConcMarkSweepGC` especifica a coleta de lixo de varredura de marca simultânea, o que ajuda a limitar pausas induzidas pela coleta de lixo.

    Para obter mais informações, consulte: Concurrent Mark Sweep Collector Enhancements.
- Caso a versão do Tomcat seja inferior à versão 7, o modelo cancela a definição de `LC_ALL`, que resolve um bug do Ubuntu.

> **Note**
>
> Com os atributos padrão, algumas dessas variáveis de ambiente são simplesmente definidas como os valores padrão. No entanto, a definição explícita das variáveis de ambiente como atributos significa que você pode definir atributos JSON personalizados para substituir os atributos padrão e fornecer valores personalizados. Para obter mais informações sobre como substituir esses atributos, consulte Sobrepor atributos.

Para obter os arquivos de modelo completos, consulte o source code.

Arquivo de configuração Server.xml

O segundo `template` recurso é usado `server.xml.erb` para criar o arquivo `system.xml` de configuração, que configura o servlet/JSP contêiner. `server.xml.erb`não contém configurações específicas do sistema operacional, portanto, está no `template` subdiretório do `default` diretório.

O modelo usa configurações padrão, mas pode criar um arquivo `system.xml` para o Tomcat 6 ou o Tomcat 7. Por exemplo, o código a seguir na seção do servidor do modelo configura as escutas corretamente para a versão especificada.

```
<% if node['tomcat']['base_version'].to_i > 6 -%>
  <!-- Security listener. Documentation at /docs/config/listeners.html
  <Listener className="org.apache.catalina.security.SecurityListener" />
  -->
<% end -%>
  <!--APR library loader. Documentation at /docs/apr.html -->
  <Listener className="org.apache.catalina.core.AprLifecycleListener" SSLEngine="on" />
  <!--Initialize Jasper prior to webapps are loaded. Documentation at /docs/jasper-
howto.html -->
  <Listener className="org.apache.catalina.core.JasperListener" />
  <!-- Prevent memory leaks due to use of particular java/javax APIs-->
  <Listener className="org.apache.catalina.core.JreMemoryLeakPreventionListener" />
<% if node['tomcat']['base_version'].to_i < 7 -%>
  <!-- JMX Support for the Tomcat server. Documentation at /docs/non-existent.html -->
  <Listener className="org.apache.catalina.mbeans.ServerLifecycleListener" />
<% end -%>
  <Listener className="org.apache.catalina.mbeans.GlobalResourcesLifecycleListener" />
<% if node['tomcat']['base_version'].to_i > 6 -%>
  <Listener className="org.apache.catalina.core.ThreadLocalLeakPreventionListener" />
<% end -%>
```

O modelo usa atributos em vez de configurações codificadas, de maneira que você possa alterar facilmente as configurações definindo atributos JSON personalizados. Por exemplo:

```
<Connector port="<%= node['tomcat']['port'] %>" protocol="HTTP/1.1"
           connectionTimeout="20000"
           URIEncoding="<%= node['tomcat']['uri_encoding'] %>"
           redirectPort="<%= node['tomcat']['secure_port'] %>" />
```

Para obter mais informações, consulte o source code.

tomcat::apache_tomcat_bind

A receita `tomcat::apache_tomcat_bind` permite que o servidor Apache funcione como o front-end do Tomcat, recebendo solicitações de entrada e as encaminhando para o Tomcat e retornando as respostas ao cliente. Este exemplo usa mod_proxy como o proxy/gateway do Apache.

```
execute 'enable mod_proxy for apache-tomcat binding' do
  command '/usr/sbin/a2enmod proxy'
  not_if do
    ::File.symlink?(::File.join(node['apache']['dir'], 'mods-enabled', 'proxy.load'))
 || node['tomcat']['apache_tomcat_bind_mod'] !~ /\Aproxy/
  end
end

execute 'enable module for apache-tomcat binding' do
  command "/usr/sbin/a2enmod #{node['tomcat']['apache_tomcat_bind_mod']}"
  not_if {::File.symlink?(::File.join(node['apache']['dir'], 'mods-enabled',
 "#{node['tomcat']['apache_tomcat_bind_mod']}.load"))}
end

include_recipe 'apache2::service'

template 'tomcat thru apache binding' do
  path ::File.join(node['apache']['dir'], 'conf.d', node['tomcat']
['apache_tomcat_bind_config'])
  source 'apache_tomcat_bind.conf.erb'
  owner 'root'
  group 'root'
  mode 0644
  backup false
  notifies :restart, resources(:service => 'apache2')
end
```

Para habilitar `mod_proxy`, você deve habilitar o módulo `proxy` e um módulo com base em protocolo. Você tem duas opções para o módulo de protocolo:

- HTTP: `proxy_http`
- JServ Protocolo Apache (AJP): `proxy_ajp`

  AJP é um protocolo do Tomcat interno.

Os execute resources da receita executam o comando `a2enmod`, que habilita o módulo especificado criando os symlinks obrigatórios:

- O primeiro recurso `execute` habilita o módulo `proxy`.
- O segundo recurso `execute` habilita o módulo de protocolo, definido como `proxy_http`, por padrão.

  Caso prefira usar AJP, você pode definir JSON personalizada para substituir o atributo `apache_tomcat_bind_mod` e defini-lo como `proxy_ajp`.

A `apache2::service` receita é uma receita integrada do AWS OpsWorks Stacks que define o serviço Apache. Para obter mais informações, consulte a receita no GitHub repositório AWS OpsWorks Stacks.

O recurso `template` usa `apache_tomcat_bind.conf.erb` para criar um arquivo de configuração que, por padrão, é chamado `tomcat_bind.conf`. Ele salva o arquivo no diretório `['apache']['dir']/.conf.d`. O atributo `['apache']['dir']` é definido no arquivo de atributos `apache2` integrados, e é definido por padrão como `/etc/httpd` (Amazon Linux) ou `/etc/apache2` (Ubuntu). Caso o recurso `template` crie ou altere o arquivo de configuração, o comando `notifies` programa uma reinicialização do serviço do Apache.

```
<% if node['tomcat']['apache_tomcat_bind_mod'] == 'proxy_ajp' -%>
ProxyPass <%= node['tomcat']['apache_tomcat_bind_path'] %> ajp://localhost:<%=
 node['tomcat']['ajp_port'] %>/
ProxyPassReverse <%= node['tomcat']['apache_tomcat_bind_path'] %> ajp://localhost:<%=
 node['tomcat']['ajp_port'] %>/
<% else %>
ProxyPass <%= node['tomcat']['apache_tomcat_bind_path'] %> http://localhost:<%=
 node['tomcat']['port'] %>/
ProxyPassReverse <%= node['tomcat']['apache_tomcat_bind_path'] %> http://localhost:<%=
 node['tomcat']['port'] %>/
<% end -%>
```

O modelo usa as ProxyPassReversediretivas ProxyPasse para configurar a porta usada para transmitir tráfego entre o Apache e o Tomcat. Como ambos os servidores estão na mesma instância, eles podem usar um URL localhost e são definidos por padrão como `http://localhost:8080`.

Receitas de configuração

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As receitas de configuração são atribuídas ao evento do ciclo de vida de configuração da camada, que ocorre em todas as instâncias da pilha sempre que uma instância entra ou sai do estado online. Você usa receitas de configuração a fim de ajustar a configuração de uma instância para responder à alteração, conforme apropriado. Ao implementar uma receita de configuração, lembre-se de que uma alteração feita na configuração da pilha pode envolver instâncias que não têm nada a ver com essa camada. A receita deve ser capaz de responder devidamente, o que pode significar não fazer nada em alguns casos.

tomcat::configure

A receita `tomcat::configure` se destina a um evento do ciclo de vida de configuração da camada.

```
include_recipe 'tomcat::context'
# Optional: Trigger a Tomcat restart in case of a configure event, if relevant
# settings in custom JSON have changed (e.g. java_opts/JAVA_OPTS):
#include_recipe 'tomcat::container_config'
```

A receita `tomcat::configure` é basicamente uma metarreceita que executa duas receitas dependentes.

1. A receita `tomcat::context` cria um arquivo de configuração de contexto da aplicação da web.

   Este arquivo configura os recursos JDBC usados pelas aplicações para se comunicar com a instância do MySQL, conforme abordado na próxima seção. A execução dessa receita em resposta a um evento de configuração permite que a camada atualize o arquivo de configuração de contexto da aplicação da web caso a camada do banco de dados tenha sido alterada.

2. A receita de configuração `tomcat::container_config` é reexecutada para capturar eventuais alterações feitas na configuração do contêiner.

O `include` de `tomcat::container_config` é comentado neste exemplo. Caso queira usar o JSON personalizado para modificar as configurações do Tomcat, você pode remover o comentário. Um evento do ciclo de vida de configuração acaba executando `tomcat::container_config`, que atualiza os arquivos de configuração relacionados ao Tomcat, conforme descrito em tomcat::container_config e reinicia o serviço Tomcat.

tomcat::context

O livro de receitas do Tomcat permite que os aplicativos acessem um servidor de banco de dados MySQL, que pode ser executado em uma instância separada, usando um objeto J2EE. DataSource Com o Tomcat, você pode habilitar a conexão criando e instalando um arquivo de configuração de contexto da aplicação da web para cada aplicação. Este arquivo define a relação entre a aplicação e o recurso JDBC que a aplicação usará para se comunicar com o banco de dados. Para obter mais informações, consulte The Context Container.

A finalidade principal da receita `tomcat::context` é criar esse arquivo de configuração.

```
include_recipe 'tomcat::service'

node[:deploy].each do |application, deploy|
  context_name = deploy[:document_root].blank? ? application : deploy[:document_root]

  template "context file for #{application} (context name: #{context_name})" do
    path ::File.join(node['tomcat']['catalina_base_dir'], 'Catalina', 'localhost',
 "#{context_name}.xml")
    source 'webapp_context.xml.erb'
    owner node['tomcat']['user']
    group node['tomcat']['group']
    mode 0640
    backup false
    only_if { node['datasources'][context_name] }
    variables(:resource_name => node['datasources'][context_name], :webapp_name =>
 application)
    notifies :restart, resources(:service => 'tomcat')
  end
end
```

Além dos atributos do livro de receitas do Tomcat, essa receita usa os atributos de configuração e implantação da pilha que o AWS OpsWorks Stacks instala com o evento Configure. O serviço AWS OpsWorks Stacks adiciona atributos ao objeto de nó de cada instância que contêm as informações que as receitas normalmente obteriam usando pacotes de dados ou pesquisas e instala os atributos em cada instância. Os atributos contêm informações detalhadas sobre a configuração da pilha, as aplicações implantadas e eventuais dados personalizados que um usuário deseja incluir. As receitas podem obter dados dos atributos de configuração e implantação da pilha usando a sintaxe de nó do Chef padrão. Para obter mais informações, consulte Configuração de pilha e atributos de implantação. Com as pilhas do Chef 11.10, você também pode usar o Chef para obter os dados configuração e implantação da pilha. Para obter mais informações, consulte Utilização da pesquisa do Chef.

`deploy`attributes se refere ao `[:deploy]` namespace, que contém atributos relacionados à implantação que são definidos por meio do console ou da API ou gerados pelo serviço Stacks. AWS OpsWorks O atributo `deploy` inclui um atributo para cada aplicação implantada, com o nome abreviado da aplicação. Cada atributo da aplicação contém um conjunto de atributos que caracterizam a aplicação, como a raiz do documento (`[:deploy][:`*`appname`*`][:document_root]`).

A receita `context` primeiro garante que o serviço seja definido para esta execução do Chef chamando tomcat::service. Em seguida, ele define uma variável `context_name` que representa o nome do arquivo de configuração, excluindo a extensão `.xml`. Caso você use a raiz do documento padrão, `context_name` é definido como o nome abreviado da aplicação. Do contrário, ele é definido como a raiz do documento especificada. O exemplo abordado em Criar uma pilha e executar uma aplicação define a raiz do documento como `"ROOT"`, de maneira que o contexto seja ROOT e o arquivo de configuração se chame `ROOT.xml`.

A maior parte da receita passa pela lista de aplicações implantadas e, para cada aplicação, usa o modelo `webapp_context.xml.erb` para criar um arquivo de configuração. O exemplo implanta apenas uma aplicação, mas a definição do atributo `deploy` exige que você o trate como uma lista de aplicações independentemente disso.

O modelo `webapp_context.xml.erb` não é específico para sistema operacional, por isso está localizado no subdiretório `templates`, no diretório `default`.

A receita cria o arquivo de configuração da seguinte forma:

- Usando-se valores de atributo padrão, o nome do arquivo de configuração é definido como *`context_name`*`.xml` e instalado no diretório `/etc/tomcat6/Catalina/localhost/`.

O nó `['datasources']` dos atributos de configuração da pilha contém um ou mais atributos, cada um mapeando um nome de contexto para o recurso de dados JDBC que a aplicação associada usará para se comunicar com o banco de dados. O nó e o conteúdo são definidos com o JSON personalizado quando você cria a pilha, conforme descrito posteriormente em Criar uma pilha e executar uma aplicação. O exemplo tem um único atributo que associa o nome de contexto ROOT a um recurso JDBC chamado jdbc/mydb.

- Usando valores de atributo padrão, o usuário do arquivo e o grupo são definidos como os valores definidos pelo pacote do Tomcat: `tomcat` (Amazon Linux) ou `tomcat6` (Ubuntu).
- O recurso `template` só cria o arquivo de configuração caso o nó `['datasources']` exista e inclui um atributo `context_name`.
- O recurso `template` define duas variáveis, `resource_name` e `webapp_name`.

  `resource_name` é definido como o nome do recurso associado a `context_name` e `webapp_name` é definido como o nome abreviado da aplicação.
- O recurso do modelo reinicia o serviço Tomcat para carregar e ativar as alterações.

O modelo `webapp_context.xml.erb` consiste em um elemento `Context` que contém um elemento `Resource` com o próprio conjunto de atributos.

Os atributos `Resource` caracterizam a configuração do contexto:

- name: o nome do recurso JDBC, definido como o valor `resource_name` definido em `tomcat::context`.

  Por exemplo, o nome do recurso é definido como jdbc/mydb.
- auth e type: estas são configurações padrão para conexões JDBC `DataSource`.
- maxActive, maxIdle e maxWait: o número máximo de conexões ativas e inativas, e o tempo de espera máximo para uma conexão ser retornada.
- username e password: o nome de usuário e a senha raiz do banco de dados, obtidos junto aos atributos `deploy`.
- driverClassName— O nome da classe do driver JDBC, que é definido como o driver MySQL.
- url: a conexão URL.

  O prefixo depende do banco de dados. Ele deve ser definido como `jdbc:mysql` para MySQL, `jdbc:postgresql` para Postgres e `jdbc:sqlserver` para SQL Server. O exemplo define o

URL como `jdbc:mysql://host_IP_Address:3306:simplejsp`, em que *`simplejsp`* é o nome abreviado da aplicação.

- factory: a fábrica `DataSource`, obrigatória para bancos de dados MySQL.

Para obter mais informações sobre esse arquivo de configuração, consulte o DataSources tópico Usando no wiki do Tomcat.

Receitas de implantação

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As receitas de implantação são atribuídas ao evento do ciclo de vida de implantação da camada. Normalmente, ocorre em todas as instâncias da pilha sempre que você implanta um aplicativo, embora você possa, opcionalmente, restringir o evento somente a instâncias especificadas. AWS OpsWorks O Stacks também executa as receitas de implantação em novas instâncias, após a conclusão das receitas de configuração. A finalidade principal das receitas de implantação é implantar código e arquivos relacionados de um repositório nas instâncias da camada do servidor de aplicações. No entanto, você normalmente executa receitas de implantação em outras camadas também. Isso permite que as instâncias dessas camadas, por exemplo, atualizem a configuração para acomodar a aplicação recém-implantada. Quando você implementar uma receita de implantação, lembre-se de que um evento de implantação não necessariamente significa que aplicações estejam sendo implantadas na instância. Pode ser simplesmente uma notificação de que aplicações estão sendo implantadas em outras instâncias na pilha para permitir que a instância faça eventuais atualizações necessárias. A receita deve ser capaz de responder adequadamente, o que pode significar não fazer nada.

AWS OpsWorks O Stacks implanta automaticamente aplicativos dos tipos de aplicativos padrão nas camadas correspondentes do servidor de aplicativos incorporado. Para implantar aplicações em uma camada personalizada, você deve implementar receitas de implantação personalizadas que fazem download dos arquivos da aplicação de um repositório no local apropriado na instância. No entanto,

você normalmente pode limitar a quantidade de código que deve gravar usando deploy cookbook integrada para lidar com alguns aspectos da implantação. Por exemplo, caso você armazene os arquivos em um dos repositórios compatíveis, o livro de receitas integrado pode processar os detalhes do download dos arquivos do repositório para as instâncias da camada.

A receita `tomcat::deploy` deve ser atribuída ao evento do ciclo de vida de implementação.

```
include_recipe 'deploy'

node[:deploy].each do |application, deploy|
  opsworks_deploy_dir do
    user deploy[:user]
    group deploy[:group]
    path deploy[:deploy_to]
  end

  opsworks_deploy do
    deploy_data deploy
    app application
  end
...
```

A receita `tomcat::deploy` usa o livro de receitas de implantação integrado em aspectos da implantação que não sejam específicos da aplicação. A receita `deploy` (abreviada para a receita `deploy::default` integrada) é uma receita integrada que processa os detalhes da configuração de usuários, grupos etc., com base em dados dos atributos `deploy`.

A receita usa duas definições do Chef integradas `opsworks_deploy_dir` e `opworks_deploy` para instalar a aplicação.

A definição `opsworks_deploy_dir` configura a estrutura do diretório, com base em dados do JSON de implantação da aplicação. As definições são basicamente uma maneira prática de empacotar definições de recurso e estão localizadas no diretório `definitions` de um livro de receitas. As receitas podem usar definições como os recursos, mas a definição propriamente dita não tem um provedor associado, apenas os recursos incluídos na definição. Você pode definir variáveis na receita, que são passadas para as definições de recurso subjacentes. A receita `tomcat::deploy` define variáveis `user`, `group` e `path` com base em dados do JSON de implantação. Eles são passados para o directory resource da definição, que gerencia os diretórios.

> **Note**
>
> O usuário e o grupo da aplicação implantado são determinados pelos atributos `[:opsworks][:deploy_user][:user]` e `[:opsworks][:deploy_user][:group]`, definidos no arquivo de atributos `deploy.rb` do livro de receitas de implantação integrado. O valor padrão de `[:opsworks][:deploy_user][:user]` é `deploy`. O valor padrão de `[:opsworks][:deploy_user][:group]` depende do sistema operacional da instância:
>
> - Para instâncias do Ubuntu, o grupo padrão é `www-data`.
> - Para instâncias do Amazon Linux membros de uma camada do Rails App Server que use Nginx e Unicorn, o grupo padrão é `nginx`.
> - Para todas as outras instâncias do Amazon Linux, o grupo padrão é `apache`.
>
> Você pode alterar a definição usando um JSON personalizado ou um arquivo de atributos personalizado para substituir o atributo apropriado. Para obter mais informações, consulte Sobrepor atributos.

A outra definição, `opsworks_deploy`, processa os detalhes de verificação do código da aplicação e dos arquivos relacionados do repositório e os implanta na instância, com base em dados dos atributos `deploy`. Você pode usar essa definição para qualquer tipo de aplicação; detalhes da implantação, como os nomes de diretório especificados no console ou por meio da API e colocar os atributos `deploy`. No entanto, `opsworks_deploy` só funciona para os quatro tipos de repositório compatíveis: Git, Subversion, S3 e HTTP. Você deve implementar esse código sozinho caso queira usar um tipo de repositório diferente.

Você instala os arquivos de uma aplicação no diretório `webapps` do Tomcat. Uma prática comum é copiar os arquivos diretamente para `webapps`. No entanto, a implantação do AWS OpsWorks Stacks foi projetada para reter até cinco versões de um aplicativo em uma instância, para que você possa reverter para uma versão anterior, se necessário. AWS OpsWorks Portanto, o Stacks faz o seguinte:

1. Implanta aplicativos em um diretório distinto cujo nome contém um time stamp, como `/srv/www/my_1st_jsp/releases/20130731141527`.
2. Cria um symlink chamado `current`, como `/srv/www/my_1st_jsp/current`, para esse diretório exclusivo.

3. Caso ele ainda não exista, cria um symlink com base no diretório `webapps` para o symlink `current` criado na Etapa 2.

Caso você precise reverter para uma versão anterior, modifique o symlink `current` a fim de apontar para um diretório distinto que contenha o carimbo de data e hora apropriado, por exemplo, alterando o link de destino de `/srv/www/my_1st_jsp/current`.

A seção intermediária de `tomcat::deploy` configura o symlink.

```
  ...
  current_dir = ::File.join(deploy[:deploy_to], 'current')
  webapp_dir = ::File.join(node['tomcat']['webapps_base_dir'],
 deploy[:document_root].blank? ? application : deploy[:document_root])

  # opsworks_deploy creates some stub dirs, which are not needed for typical webapps
  ruby_block "remove unnecessary directory entries in #{current_dir}" do
    block do
      node['tomcat']['webapps_dir_entries_to_delete'].each do |dir_entry|
        ::FileUtils.rm_rf(::File.join(current_dir, dir_entry), :secure => true)
      end
    end
  end

  link webapp_dir do
    to current_dir
    action :create
  end
  ...
```

A receita cria primeiramente duas variáveis, `current_dir` e `webapp_dir`, para representar os diretórios `current` e `webapp`, respectivamente. Em seguida, ele usa um recurso `link` para vincular `webapp_dir` a `current_dir`. A `deploy::default` receita do AWS OpsWorks Stacks cria alguns diretórios stub que não são necessários para este exemplo, então a parte central do trecho os remove.

A parte final de `tomcat::deploy` reinicia o serviço do Tomcat, se necessário.

```
  ...
  include_recipe 'tomcat::service'
```

```
  execute 'trigger tomcat service restart' do
    command '/bin/true'
    not_if { node['tomcat']['auto_deploy'].to_s == 'true' }
    notifies :restart, resources(:service => 'tomcat')
  end
end

include_recipe 'tomcat::context'
```

A receita executa primeiramente `tomcat::service` a fim de garantir que o serviço seja definido para essa execução do Chef. Ela acaba usando um execute resource a fim de notificar o serviço para reiniciar, mas somente se `['tomcat']['auto_deploy']` for definido como `'true'`. Do contrário, o Tomcat escuta alterações no diretório `webapps`, que torna uma reinicialização do serviço do Tomcat explícito desnecessário.

> **Note**
>
> O recurso `execute` não executa realmente nada substantivo; `/bin/true` é um script do shell fictício que simplesmente retorna um código de êxito. Ele é usado aqui apenas como uma maneira cômoda para gerar uma notificação de reinicialização. Conforme mencionado anteriormente, o uso de notificações garante que os serviços não sejam reiniciados com muita frequência.

Por fim, `tomcat::deploy` executa `tomcat::context`, que atualiza o arquivo de configuração do contexto da aplicação da web caso você tenha alterado o banco de dados back-end.

Criar uma pilha e executar uma aplicação

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção mostra como usar o livro de receitas do Tomcat para implementar uma configuração de pilha básica que execute uma aplicação JSP simples chamada SimpleJSP. A pilha consiste em uma camada personalizada baseada em Tomcat chamada e TomCustom uma camada MySQL. O SimpleJSP é implantado TomCustom e exibe algumas informações do banco de dados MySQL. Se você ainda não está familiarizado com o básico de como usar o AWS OpsWorks Stacks, leia primeiro. Conceitos básicos das pilhas Linux do Chef 11

A aplicação SimpleJSP

A aplicação SimpleJSP demonstra os conceitos básicos de como configurar uma conexão de banco de dados e recuperar dados do banco de dados MySQL da pilha.

```
<html>
  <head>
    <title>DB Access</title>
  </head>
  <body>
    <%@ page language="java" import="java.sql.*,javax.naming.*,javax.sql.*" %>
    <%
      StringBuffer output = new StringBuffer();
      DataSource ds = null;
      Connection con = null;
      Statement stmt = null;
      ResultSet rs = null;
      try {
        Context initCtx = new InitialContext();
        ds = (DataSource) initCtx.lookup("java:comp/env/jdbc/mydb");
        con = ds.getConnection();
        output.append("Databases found:<br>");
        stmt = con.createStatement();
        rs = stmt.executeQuery("show databases");
        while (rs.next()) {
          output.append(rs.getString(1));
          output.append("<br>");
        }
      }
      catch (Exception e) {
        output.append("Exception: ");
        output.append(e.getMessage());
        output.append("<br>");
      }
      finally {
```

```
        try {
          if (rs != null) {
            rs.close();
          }
          if (stmt != null) {
            stmt.close();
          }
          if (con != null) {
            con.close();
          }
        }
        catch (Exception e) {
          output.append("Exception (during close of connection): ");
          output.append(e.getMessage());
          output.append("<br>");
        }
      }
    %>
    <%= output.toString() %>
  </body>
</html>
```

SimpleJSP usa um objeto `DataSource` para se comunicar com o banco de dados MySQL. O Tomcat usa os dados no web app context configuration file para criar e inicializar um objeto `DataSource` e o associar a um nome lógico. Em seguida, ele registra o nome lógico usando um serviço de nomenclatura Java Naming and Directory Interface (JNDI). Para obter uma instância do objeto `DataSource` apropriado, você cria um objeto `InitialContext` e passa o nome lógico do recurso para o método `lookup` do objeto, que recupera o objeto apropriado. O nome lógico do exemplo SimpleJSP, `java:comp/env/jdbc/mydb`, tem os seguintes componentes:

- O namespace raiz, `java`, separado do resto do nome por dois-pontos (:).
- Eventuais namespaces adicionais, separados por barras (/).

  O Tomcat adiciona automaticamente os recursos ao namespace `comp/env`.
- O nome do recurso, definido no arquivo de configuração do contexto da aplicação da web e separado dos namespaces por uma barra.

  O nome do recurso deste exemplo é `jdbc/mydb`.

Para estabelecer uma conexão com o banco de dados, SimpleJSP faz o seguinte:

1. Chama o método `DataSource` do objeto `getConnection`, que retorna um objeto `Connection`.
2. Chama o método `Connection` do objeto `createStatement` para criar um objeto `Statement`, que você usa para se comunicar com o banco de dados.
3. Comunica-se com o banco de dados chamando o método `Statement` apropriado.

   SimpleJSP chama `executeQuery` para executar uma consulta SHOW DATABASES, que lista os bancos de dados do servidor.

O método `executeQuery` retorna um objeto `ResultSet`, que contém os resultados da consulta. SimpleJSP obtém os nomes do banco de dados do objeto `ResultSet` retornado e os concatena para criar uma string de saída. Por fim, o exemplo fecha os objetos `ResultSet`, `Statement` e `Connection`. Para obter mais informações sobre JSP e JDBC, consulte JavaServer Pages Technology e JDBC Basics, respectivamente.

Para usar SimpleJSP com uma pilha, você deve colocá-lo em um repositório. Você pode usar qualquer um dos repositórios compatíveis, mas para usar SimpleJSP com a pilha de exemplo abordada na seção a seguir, você deve colocá-lo em um arquivo S3 público. Para obter informações sobre como usar os outros repositórios padrão, consulte Repositórios de livro de receitas.

Para colocar SimpleJSP em um repositório de arquivo S3

1. Copie o código de exemplo para um arquivo chamado `simplejsp.jsp` e coloque o arquivo em um diretório chamado `simplejsp`.
2. Crie um arquivo `.zip` do diretório `simplejsp`.
3. Crie um bucket do Amazon S3 público, faça upload de `simplejsp.zip` no bucket e torne o arquivo público.

   Para obter uma descrição de como realizar esta tarefa, consulte Começar a usar o Amazon Simple Storage Service.

Criar um stack

Para executar SimpleJSP você precisa de uma pilha com as camadas a seguir.

- A camada do MySQL, que dá suporte ao servidor MySQL back-end.

- Uma camada personalizada que usa o livro de receitas do Tomcat para dar suporte a instâncias do servidor Tomcat.

Para criar a pilha

1. No painel de AWS OpsWorks pilhas, clique em Adicionar pilha para criar uma nova pilha e clique em Avançado **>>** para exibir todas as opções. Configure a pilha da maneira a seguir.

   - Nome — Um nome de pilha definido pelo usuário; este exemplo usa. TomStack
   - Usar livros de receitas do Chef personalizados: defina o botão seletor como Sim para exibir opções adicionais.
   - Tipo de repositório: Git.
   - URL do repositório: `git://github.com/amazonwebservices/opsworks-example-cookbooks.git`.
   - JSON do Chef personalizado: adicione o seguinte JSON:

   ```
   {
     "tomcat": {
       "base_version": 7,
       "java_opts": "-Djava.awt.headless=true -Xmx256m"
     },
     "datasources": {
       "ROOT": "jdbc/mydb"
     }
   }
   ```

   Para as opções restantes, você pode aceitar os valores padrão.

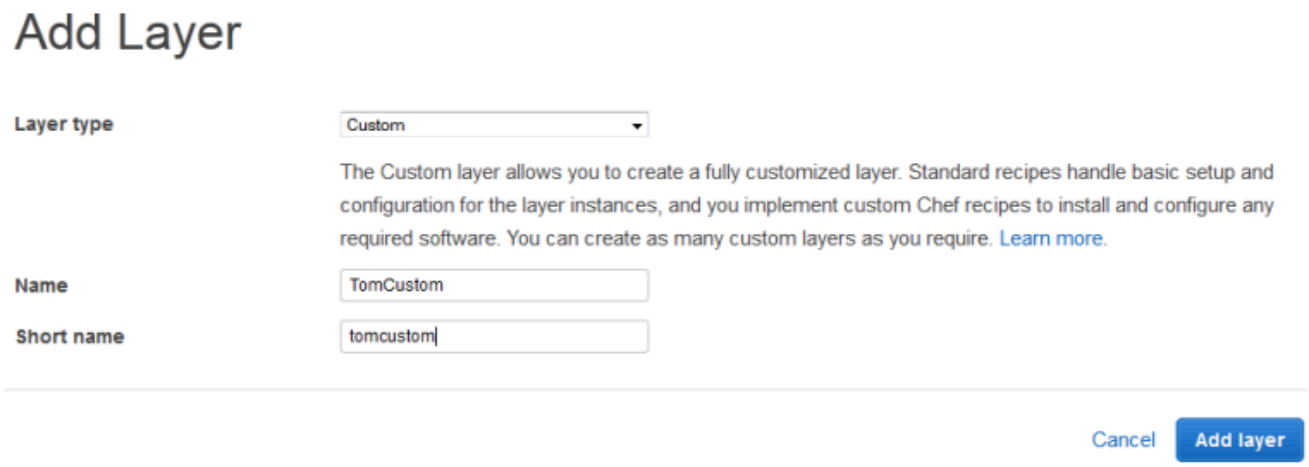
   O JSON personalizado faz o seguinte:

   - Substitui o atributo `['base_version']` do livro de receitas do Tomcat para definir o Tomcat versão como 7; o valor padrão é 6.
   - Substitui o atributo `['java_opts']` do livro de receitas do Tomcat para especificar que a instância é descentralizada e definir o tamanho máximo do heap do JVM como 256 MB; o valor padrão não define opções para instâncias que executem o Amazon Linux.

- Especifica o valor do atributo `['datasources]`, que atribui um nome de recurso JDBC (jdbc/mydb) ao nome de contexto da aplicação da web (ROOT), conforme abordado em tomcat::context.

  Este último atributo não tem valor padrão. Você deve defini-lo com JSON personalizado.

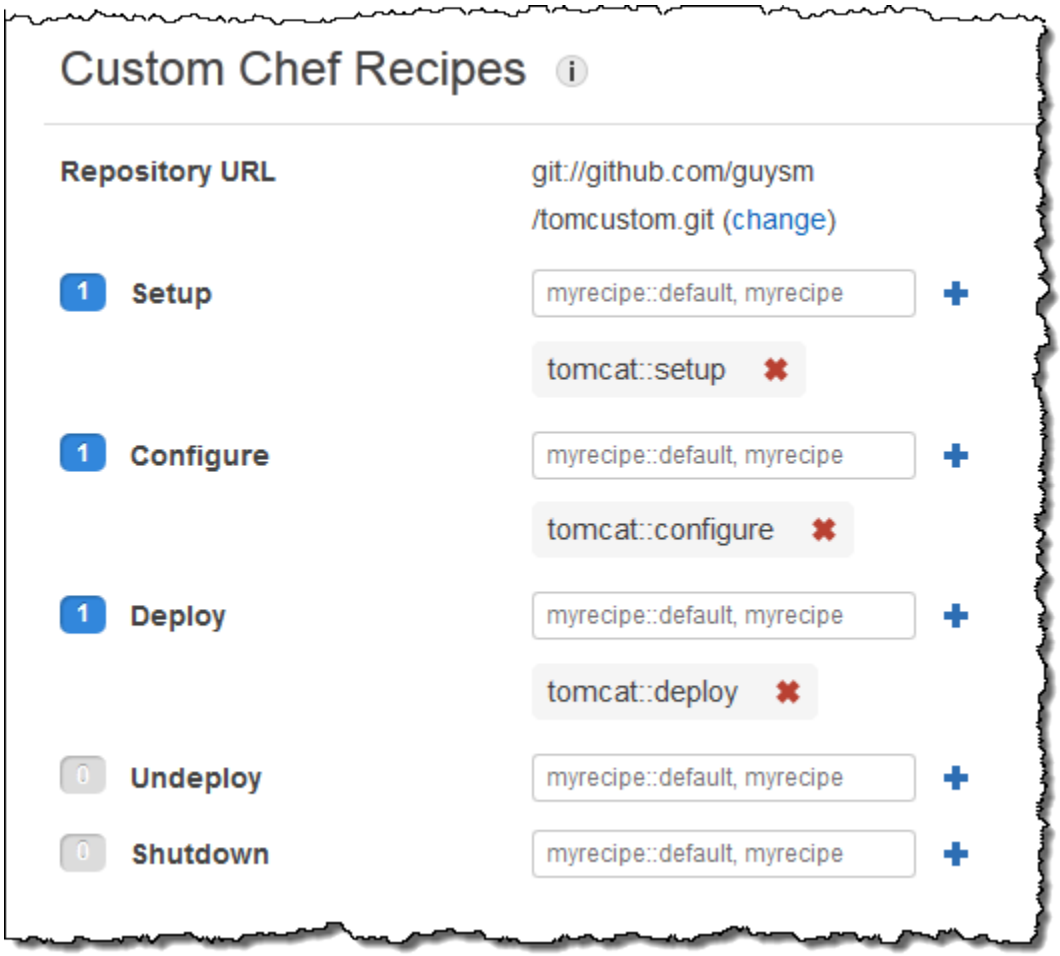
2. Clique em Add a layer. Em Layer type, selecione MySQL. Em seguida, clique em Add Layer.
3. No painel de navegação, clique em Instances e em Add an instance. Clique em Add Instance para aceitar os padrões. Na linha da instância, clique em start.
4. Retorne à página Layers e clique em + Layer para adicionar uma camada. Para Layer type (Tipo de camada), clique em Custom (Personalizado). O exemplo usa **`TomCustom`** e **`tomcustom`** como o nome da camada e o nome abreviado, respectivamente.



5. Na página Layers, para a camada personalizada, clique em Recipes e, em seguida, em Edit. Em Custom Chef Recipes, atribua as receitas do livro do Tomcat aos eventos do ciclo de vida da camada, da seguinte forma:

   - Para Setup (Instalar), digite **`tomcat::setup`** e clique em +.
   - Para Configure (Configurar), digite **`tomcat::configure`** e clique em +.
   - Para Deploy (Implantar), digite **`tomcat::deploy`** e clique em +. Em seguida, clique em Save.

   .



6. No painel de navegação, clique em Apps e em Add an app. Especifique as opções a seguir e clique em Add App:

   - Nome — O nome do aplicativo; o exemplo usa SimpleJSP e o nome curto gerado pelo AWS OpsWorks Stacks será simplejsp.
   - Tipo de aplicativo: defina essa opção como Outro.

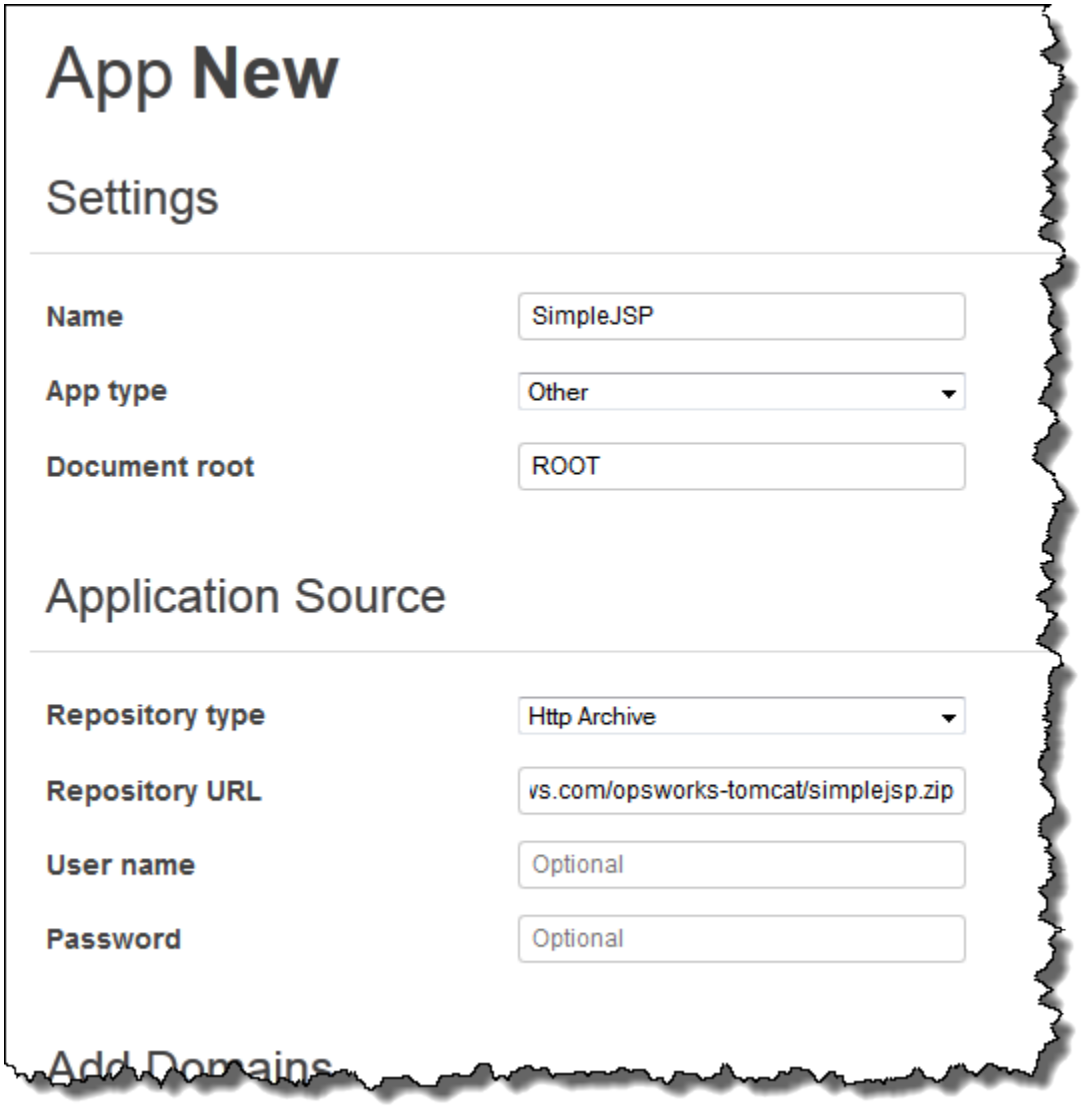
AWS OpsWorks O Stacks implanta automaticamente os tipos de aplicativos padrão nas instâncias do servidor associadas. Caso você defina App type (Tipo de aplicativo) como Other (Outro), o AWS OpsWorks Stacks simplesmente executará as receitas de implantação e permitirá que elas lidem com a implantação.

- Raiz do documento: defina essa opção como **ROOT**.

  O valor Document root especifica o nome do contexto.

- Tipo de repositório: defina essa opção como Arquivamento do S3.
- URL do repositório: defina como o URL do Amazon S3 do aplicativo que você criou anteriormente.

Use as configurações padrão para as outras opções.



7. Use a página Instâncias para adicionar uma instância à TomCustom camada e iniciá-la. AWS OpsWorks O Stacks executa automaticamente as receitas de implantação em uma nova instância após a conclusão das receitas de configuração. Portanto, iniciar a instância também implanta o SimpleJSP.
8. Quando a TomCustom instância estiver on-line, clique no nome da instância na página Instâncias para ver seus detalhes. Copie o endereço IP público. Em seguida, construa uma URL da seguinte forma: http://*`publicIPappname.jsp`*/tc/. Para o exemplo, esse URL será semelhante a **`http://50.218.191.172/tc/simplejsp.jsp`**.

> **Note**
>
> O URL do Apache que encaminha solicitações para o Tomcat é definido como o atributo `['tomcat']['apache_tomcat_bind_path']` padrão, `/tc/`. A raiz do documento

SimpleJSP é definida como `ROOT`, que é um valor especial resolvido como `/`. Por isso, o URL é ".../tc/simplejsp.jsp".

9. Cole o URL da etapa anterior no navegador. Você deve ver o seguinte:

```
Databases found:
information_schema
simplejsp
test
```

> **Note**
>
> Se sua pilha tiver uma instância do MySQL AWS OpsWorks , o Stacks cria automaticamente um banco de dados para cada aplicativo, nomeado com o nome abreviado do aplicativo.

Configuração de pilha e atributos de implantação

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Quando o AWS OpsWorks Stacks executa um comando em uma instância, por exemplo, um comando de implantação em resposta a um evento do ciclo de vida do Deploy, ele adiciona um conjunto de atributos ao objeto node da instância que descreve a configuração atual da pilha. Para os comandos Deploy events e Execute Recipes stack, o AWS OpsWorks Stacks instala atributos de implantação, que fornecem algumas informações adicionais de implantação. Para obter mais informações sobre o objeto de nó, consulte Sobrepor atributos. Para uma lista de configuração de pilhas e atributos de implantação mais comumente usados, incluindo nomes de nós completamente qualificados, consulte Atributos de implantação e configuração de pilha: Linux e Atributos de livros de receitas integrados.

> **Note**
>
> Em pilhas no Linux, a lista completa desses atributos, formatada como um objeto JSON, pode ser acessada usando o agente comando get_json da CLI.

As seções a seguir mostram os atributos associados a um evento Configurar e um evento Implantar para uma pilha simples, que contém o seguinte:

- Uma camada PHP App Server com duas instâncias
- Uma HAProxy camada com uma instância

Os exemplos são de uma das instâncias PHP App Server, php-app1. Por conveniência, os atributos são formatados como um objeto JSON. A estrutura do objeto é mapeada para os nomes totalmente qualificados dos atributos. Por exemplo, o atributo `node[:opsworks][:ruby_version]` aparece como se segue em uma representação JSON.

```
{
  "opsworks": {
    ...
    "ruby_version": "1.8.7",
    ...
  }
}
```

Tópicos

- Configurar Atributos
- Atributos de implantação

Configurar Atributos

O objeto JSON a seguir mostra os atributos para um evento Configurar, que ocorre em cada instância na pilha quando uma instância fica online ou off-line. Os atributos incluem os atributos de configuração da stack integrada e qualquer atributo personalizado de JSON que foram definidos para a pilha anterior ao evento (nenhuma neste exemplo). Foi editado por questão de tamanho. Para uma

descrição mais detalhada dos vários atributos, consulte Atributos de implantação e configuração de pilha: Linux e Atributos de livros de receitas integrados.

```
{
  "opsworks": {
    "layers": {
      "php-app": {
        "id": "4a2a56c8-f909-4b39-81f8-556536d20648",
        "instances": {
          "php-app2": {
            "elastic_ip": null,
            "region": "us-west-2",
            "booted_at": "2013-02-26T20:41:10+00:00",
            "ip": "192.0.2.0",
            "aws_instance_id": "i-34037f06",
            "availability_zone": "us-west-2a",
            "instance_type": "c1.medium",
            "private_dns_name": "ip-10-252-0-203.us-west-2.compute.internal",
            "private_ip": "10.252.0.203",
            "created_at": "2013-02-26T20:39:39+00:00",
            "status": "online",
            "backends": 8,
            "public_dns_name": "ec2-192-0-2-0.us-west-2.compute.amazonaws.com"
          },
          "php-app1": {
            ...
          }
        },
        "name": "PHP Application Server"
      },
      "lb": {
        "id": "15c86142-d836-4191-860f-f4d310440f14",
        "instances": {
          "lb1": {
           ...
          }
        },
        "name": "Load Balancer"
      }
    },
    "agent_version": "104",
    "applications": [
```

```
    ],
    "stack": {
      "name": "MyStack"
    },
    "ruby_version": "1.8.7",
    "sent_at": 1361911623,
    "ruby_stack": "ruby_enterprise",
    "instance": {
      "layers": [
        "php-app"
      ],
      "region": "us-west-2",
      "ip": "192.0.2.0",
      "id": "45ef378d-b87c-42be-a1b9-b67c48edafd4",
      "aws_instance_id": "i-32037f00",
      "availability_zone": "us-west-2a",
      "private_dns_name": "ip-10-252-84-253.us-west-2.compute.internal",
      "instance_type": "c1.medium",
      "hostname": "php-app1",
      "private_ip": "10.252.84.253",
      "backends": 8,
      "architecture": "i386",
      "public_dns_name": "ec2-192-0-2-0.us-west-2.compute.amazonaws.com"
    },
    "activity": "configure",
    "rails_stack": {
      "name": null
    },
    "deployment": null,
    "valid_client_activities": [
      "reboot",
      "stop",
      "setup",
      "configure",
      "update_dependencies",
      "install_dependencies",
      "update_custom_cookbooks",
      "execute_recipes"
    ]
  },
  "opsworks_custom_cookbooks": {
    "recipes": [
```

```
    ],
    "enabled": false
  },
  "recipes": [
    "opsworks_custom_cookbooks::load",
    "opsworks_ganglia::configure-client",
    "ssh_users",
    "agent_version",
    "mod_php5_apache2::php",
    "php::configure",
    "opsworks_stack_state_sync",
    "opsworks_custom_cookbooks::execute",
    "test_suite",
    "opsworks_cleanup"
  ],
  "opsworks_rubygems": {
    "version": "1.8.24"
  },
  "ssh_users": {
  },
  "opsworks_bundler": {
    "manage_package": null,
    "version": "1.0.10"
  },
  "deploy": {
  }
}
```

A maioria das informações está sob o atributo `opsworks`, que normalmente é chamado de um namespace. A lista a seguir descreve os principais atributos:

- `layers` atributos: um conjunto de atributos, cada descrevendo a configuração de uma das camadas da pilha.

  As camadas são identificados pelas abreviações, `php-app` e `lb` para este exemplo. Para obter mais informações sobre abreviações para outras camadas , consulte AWS OpsWorks Referência de camadas de pilhas.
- `instances` atributos: cada camada tem um elemento `instances`, que inclui um atributo para cada uma das instâncias on-line das camadas, com o nome do nome abreviado da instância.

  A camada PHP App Server tem duas instâncias, `php-app1` e `php-app2`. A HAProxy camada tem uma instância,`lb1`.

> **Note**
>
> O elemento `instances` contém somente as instâncias que estão no estado online quando os atributos de pilha e de implantação são criados.

- Atributos da instância: cada atributo da instância contém um conjunto de atributos que caracterizam a instância, como o endereço IP privado da instância e nome de DNS privado. Em resumo, o exemplo mostra apenas o atributo `php-app2` em detalhes, os outros contêm informações semelhantes.
- `applications`: uma lista de aplicativos implantados, não usados neste exemplo.
- `stack`: o nome da pilha. `MyStack` neste exemplo.
- `instance`: a instância em que esses atributos são instalados. `php-app1` neste exemplo. As receitas podem usar esse atributo para conseguir informações sobre a instância em que está sendo executada, como o endereço IP público da instância.
- `activity`: a atividade geradora dos atributos. Um evento Configure, neste exemplo.
- `rails_stack`: a pilha Rails para pilhas que incluem uma camada Rails App Server.
- `deployment`: se esses atributos são associados a uma implantação. Ele é definido como `null` para este exemplo, pois eles são associados a um evento Configurar.
- `valid_client_activities`: uma lista de atividades de cliente válido.

O atributo `opsworks` é seguido por vários outros atributos de nível superior, incluindo os seguintes:

- `opsworks_custom_cookbooks`: se os livros de receitas personalizados estão habilitados. Se esse for o caso, o atributo inclui uma lista de receitas personalizadas.
- `recipes`: as receitas que foram executadas por essa atividade.
- `opsworks_rubygems`— A RubyGems versão da instância.
- `ssh_users`: uma lista de usuários SSH. Nenhum neste exemplo.
- `opsworks_bundler`: a versão do bundler e se ela está habilitada.
- `deploy`: informações sobre atividades de implantação. Nenhuma neste exemplo.

Atributos de implantação

Os atributos para um evento Implantar ou comando de pilha para executar receitas consistem na configuração da pilha integrada e nos atributos de implantação, além de quaisquer outros atributos

personalizados de pilha ou implantação (nenhum para este exemplo). O objeto JSON a seguir mostra os atributos de php-app1 que estão associados a um evento Implantar que implantou o aplicativo SimplePHP às instâncias PHP da pilha. Grande parte do objeto consiste nos atributos de configuração da pilha que são semelhantes aos atributos para o evento Configurar descrito na seção anterior, assim, o exemplo tem o foco principal nos atributos específicos da implantação. Para uma descrição mais detalhada dos vários atributos, consulte Atributos de implantação e configuração de pilha: Linux e Atributos de livros de receitas integrados.

```
{
   ...
  "opsworks": {
    ...
    "activity": "deploy",
    "applications": [
      {
        "slug_name": "simplephp",
        "name": "SimplePHP",
        "application_type": "php"
      }
    ],
    "deployment": "5e6242d7-8111-40ee-bddb-00de064ab18f",
    ...
  },
  ...
{
  "ssh_users": {
  },
  "deploy": {
    "simplephpapp": {
      "application": "simplephpapp",
      "application_type": "php",
      "environment_variables": {
        "USER_ID": "168424",
        "USER_KEY": "somepassword"
      },
      "auto_bundle_on_deploy": true,
      "deploy_to": "/srv/www/simplephpapp",
      "deploying_user": "arn:aws:iam::123456789012:user/guysm",
      "document_root": null,
      "domains": [
        "simplephpapp"
```

```
      ],
      "migrate": false,
      "mounted_at": null,
      "rails_env": null,
      "restart_command": "echo 'restarting app'",
      "sleep_before_restart": 0,
      "ssl_support": false,
      "ssl_certificate": null,
      "ssl_certificate_key": null,
      "ssl_certificate_ca": null,
      "scm": {
        "scm_type": "git",
        "repository": "git://github.com/amazonwebservices/opsworks-demo-php-simple-
app.git",
        "revision": "version1",
        "ssh_key": null,
        "user": null,
        "password": null
      },
      "symlink_before_migrate": {
        "config/opsworks.php": "opsworks.php"
      },
      "symlinks": {
      },
      "database": {
      },
      "memcached": {
        "host": null,
        "port": 11211
      },
      "stack": {
        "needs_reload": false
      }
    }
  },
}
```

O atributo `opsworks` é idêntico ao exemplo na seção anterior. As seções a seguir são mais relevantes para a implantação:

- `activity`: o evento associado a esses atributos. Neste exemplo, um evento Deploy.

- `applications`: contém um conjunto de atributos para cada aplicativo que fornecem os nomes dos aplicativos, nomes slug e tipos.

  O nome do slug é o nome curto do aplicativo, que o AWS OpsWorks Stacks gera a partir do nome do aplicativo. O nome slug para SimplePHP é simplephp.
- `deployment`: o ID de implantação, que identifica unicamente uma implantação.

O atributo `deploy` inclui informações sobre os aplicativos que estão sendo implantados. Por exemplo, as receitas integradas de implantação usam os dados no atributo `deploy` para instalar arquivos nos diretórios adequados e criar arquivos com conexão nos bancos de dados. O atributo `deploy` inclui um atributo para cada aplicativo implantado, com o nome abreviado do aplicativo. Cada atributo do aplicativo inclui os seguintes atributos:

- `environment_variables`: contém qualquer variável ambiental definida para o aplicativo. Para obter mais informações, consulte Variáveis de ambiente.
- `domains`: por padrão, o domínio é o nome abreviado do aplicativo, que é simplephpapp para este exemplo. Se você tiver atribuído domínios personalizados, eles também aparecerão aqui. Para obter mais informações, consulte Usando domínios predefinidos.
- `application`: o nome abreviado do aplicativo.
- `scm`: esse elemento contém as informações necessárias para fazer download dos arquivos do aplicativo do repositório. Neste exemplo, um repositório Git.
- `database`: informações do banco de dados, caso a pilha inclua uma camada de banco de dados.
- `document_root`: a raiz do documento, que é definida como `null` neste exemplo, indicando que a raiz é pública.
- `ssl_certificate_ca`, `ssl_support`, `ssl_certificate_key`: indica se o aplicativo tem suporte para SSL. Se esse for o caso, os atributos `ssl_certificate_key` e `ssl_certificate_ca` são definidos para certificados correspondentes.
- `deploy_to`: o diretório raiz do aplicativo.

Introdução a livros de receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Uma pilha AWS OpsWorks Stacks em nível de produção normalmente requer alguma personalização, o que geralmente significa implementar um livro de receitas personalizado do Chef com uma ou mais receitas, arquivos de atributos ou arquivos de modelo. Este tópico é um tutorial de introdução à implementação de livros de receitas para AWS OpsWorks Stacks.

Para obter mais informações sobre como o AWS OpsWorks Stacks usa livros de receitas, que incluem uma breve introdução geral aos livros de receitas, consulte. Livros de receitas e receitas Para obter mais informações sobre como implementar e testar receitas do Chef, consulte Test-Driven Infrastructure with Chef, 2nd Edition.

Os exemplos do tutorial são divididos em duas seções:

- Conceitos básicos do livro de receitas é um conjunto de demonstrações de exemplo projetadas para os usuários que não estão familiarizados com o Chef; usuários experientes no Chef podem ignorar essa seção.

  Os exemplos mostrarão as noções básicas de como implementar livros de receitas para executar tarefas comuns, como instalação de pacotes ou criação de diretórios. Para simplificar o processo, você usará duas ferramentas úteis, Vagrant e Test Kitchen, para executar a maioria dos exemplos localmente em uma máquina virtual. Antes de começar Conceitos básicos do livro de receitas, você deve ler Vagrant e Test Kitchen para saber como instalar e usar essas ferramentas. Como o Test Kitchen ainda não é compatível com o Windows, os exemplos são todos para o Linux, com observações indicando como adaptá-los para o Windows.
- Implementando livros de receitas para pilhas AWS OpsWorksdescreve como implementar receitas para AWS OpsWorks pilhas, inclusive para pilhas do Windows.

  Ele também inclui alguns tópicos mais avançados como, por exemplo, como usar Berkshelf para gerenciar livros de receitas externos. Os exemplos são escritos para novos usuários do Chef, da mesma forma que os exemplos em Conceitos básicos do livro de receitas. No entanto, o AWS OpsWorks Stacks funciona de forma um pouco diferente do Chef Server, por isso recomendamos que usuários experientes do Chef pelo menos leiam esta seção.

Vagrant e Test Kitchen

Se você trabalha com receitas para instâncias do Linux, o Vagrant e o Test Kitchen são ferramentas muito úteis para aprendizado e desenvolvimento e testes iniciais. Este oferece descrições breves do Vagrant e do Test Kitchen e indica as instruções de instalação e demonstrações para que você conheça e se familiarize com os conceitos básicos de como usar as ferramentas. Embora o Vagrant seja compatível com o Windows, o Test Kitchen não é; portanto, somente exemplos para Linux são fornecidos para essas ferramentas.

Vagrant

O Vagrant fornece um ambiente consistente para execução e teste de código em uma máquina virtual. Ele é compatível com uma grande variedade de ambientes, chamados de caixas do Vagrant, cada um representando um sistema operacional configurado. Para AWS OpsWorks Stacks, os ambientes de interesse são baseados nas distribuições Ubuntu, Amazon ou Red Hat Enterprise Linux (RHEL), então os exemplos usam principalmente uma caixa Vagrant chamada. `opscode-ubuntu-12.04`

O Vagrant está disponível para sistemas Linux, Windows e Macintosh; portanto, você pode usar a estação de trabalho de sua preferência para implementar e testar receitas em qualquer sistema operacional compatível. Os exemplos deste capítulo foram criados em um sistema Linux Ubuntu, mas converter os procedimentos para os sistemas Windows ou Macintosh é simples.

O Vagrant é basicamente um wrapper para um provedor de virtualização. A maioria dos exemplos usa o VirtualBoxprovedor. VirtualBox é gratuito e está disponível para sistemas Linux, Windows e Macintosh. O passo a passo do Vagrant fornece instruções de instalação, caso você ainda não tenha VirtualBox em seu sistema. Observe que você pode executar ambientes baseados no Ubuntu, VirtualBox mas o Amazon Linux está disponível somente para instâncias da Amazon. EC2 No entanto, você pode executar um sistema operacional semelhante, como o CentOS VirtualBox, o que é útil para desenvolvimento e teste iniciais.

Para obter informações sobre outros provedores, consulte a documentação do Vagrant. Em particular, o provedor de `vagrant-aws` plug-in permite que você use o Vagrant com instâncias da Amazon. EC2 Esse provedor é particularmente útil para testar receitas no Amazon Linux, que está disponível somente em EC2 instâncias da Amazon. O provedor `vagrant-aws` é gratuito, mas você precisa ter uma conta da AWS e pagar por todos os recursos da AWS que usar.

Nesse ponto, você deve passar pela demonstração Conceitos básicos do Vagrant, que descreve como instalá-lo na estação de trabalho e ensina os conceitos básicos de como usá-lo. Observe que

os exemplos deste capítulo não usam um repositório Git. Portanto, você pode omitir essa parte da demonstração, se preferir.

Test Kitchen

O Test Kitchen simplifica o processo de execução e teste dos livros de receitas no Vagrant. Como uma questão prática, você raramente precisará usar o Vagrant diretamente, isso se alguma vez precisar. O Test Kitchen executa tarefas mais comuns, incluindo:

- Como executar uma instância no Vagrant.
- Como transferir livros de receitas para a instância.
- Como executar as receitas do livro de receitas na instância.
- Como testar as receitas de um livro de receitas na instância.
- Como usar o SSH para fazer login na instância.

Em vez de instalar o gem do Test Kitchen diretamente, recomendamos instalar o Chef DK. Além do próprio Chef, este pacote inclui Test Kitchen, Berkshelf e várias outras ferramentas úteis. ChefSpec

Nesse ponto, você deve passar pela demonstração Conceitos básicos do Test Kitchen, que apresenta os conceitos básicos de como usá-lo para executar e testar receitas.

> **Note**
>
> Os exemplos neste capítulo usam o Test Kitchen como uma forma conveniente para executar receitas. Se você preferir, pode interromper a demonstração Conceitos básicos após concluir a seção Como verificar manualmente, que abrange tudo que é preciso saber para os exemplos. No entanto, o Test Kitchen é principalmente uma plataforma de teste compatível com estruturas de teste como o sistema automatizado de testes bash (BATS). Você deve concluir o restante da demonstração em algum momento para aprender como usar o Test Kitchen para testar as receitas.

Conceitos básicos do livro de receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode usar livros de receitas para realizar uma grande variedade de tarefas. Os tópicos a seguir pressupõem que você seja novo no Chef e descrevem como usar livros de receitas para realizar algumas tarefas comuns. Como o Test Kitchen ainda não é compatível com o Windows, os exemplos são todos para o Linux, com observações indicando como adaptá-los para o Windows. Caso você seja novo no Chef, recomendamos passar por esses exemplos, mesmo que vá trabalhar com o Windows. A maioria dos exemplos neste tópico pode ser usada em instâncias do Windows com algumas alterações modestas, observadas nos exemplos. Como todos os exemplos são executados em uma máquina virtual, você sequer precisa ter um computador com Linux. Basta instalar o Vagrant e o Test Kitchen na estação de trabalho regular.

> **Note**
>
> Caso você queira executar essas receitas em uma instância do Windows, a abordagem mais simples é criar uma pilha do Windows e executá-las em uma das instâncias da pilha. Para obter mais informações sobre como executar receitas em uma instância do AWS OpsWorks Stacks Windows, consulteExecutar uma receita na instância Windows.

Antes de continuar, certifique-se de que você tenha instalado o Vagrant e o Test Kitchen e passado pelo passo a passo de ambos os conceitos básicos. Para obter mais informações, consulte Vagrant e Test Kitchen.

Tópicos

- Estrutura da receita
- Exemplo 1: Instalação de pacotes
- Exemplo 2: Gerenciamento de usuários
- Exemplo 3: Criação de diretórios
- Exemplo 4: Adição do controle de fluxo
- Exemplo 5: Uso de atributos
- Exemplo 6: Criação de arquivos
- Exemplo 7: Execução de comandos e scripts

- Exemplo 8: Gerenciamento de serviços
- Exemplo 9: Usando EC2 instâncias da Amazon
- Próximas etapas

Estrutura da receita

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Livro de receitas é basicamente um conjunto de receitas, que podem executar uma grande variedade de tarefas em uma instância. Para esclarecer como implementar receitas, é útil ver um exemplo simples. Esta é a receita de configuração para a camada HAProxyintegrada. Basta se concentrar na estrutura geral a esta altura e não se preocupar muito com os detalhes; eles serão abordados nos exemplos subsequentes.

```
package 'haproxy' do
  action :install
end

if platform?('debian','ubuntu')
  template '/etc/default/haproxy' do
    source 'haproxy-default.erb'
    owner 'root'
    group 'root'
    mode 0644
  end
end

include_recipe 'haproxy::service'

service 'haproxy' do
  action [:enable, :start]
end
```

```
template '/etc/haproxy/haproxy.cfg' do
  source 'haproxy.cfg.erb'
  owner 'root'
  group 'root'
  mode 0644
  notifies :restart, "service[haproxy]"
end
```

> **Note**
>
> Para esse e outros exemplos de receitas e arquivos relacionados funcionais, consulte Receitas integradas do AWS OpsWorks Stacks.

O exemplo destaca os principais elementos de receita, descritos nas seções a seguir.

Tópicos

- Recursos
- Controle de fluxo
- Receitas incluídas

Recursos

As receitas consistem mais em um conjunto de recursos do Chef. Cada uma especifica um determinado aspecto do estado final da instância, como um pacote ou um serviço a ser iniciado. O exemplo tem quatro recursos:

- Um `package` recurso, que representa um pacote instalado, um HAProxy servidor para este exemplo.
- Um `service` recurso, que representa um serviço, o HAProxy serviço deste exemplo.
- Dois `template` recursos, que representam arquivos que devem ser criados a partir de um modelo especificado, dois arquivos de HAProxy configuração para este exemplo.

Os recursos apresentam uma maneira declarativa de especificar o estado da instância. Nos bastidores, cada recurso tem um provedor associado que executa as tarefas necessárias, como a instalação de pacotes, a criação e a configuração de diretórios, a inicialização de serviços etc. Caso

os detalhes da tarefa dependam do sistema operacional específico, o recurso tem vários provedores e usa o adequado para o sistema. Por exemplo, em um sistema Red Hat Linux, o provedor `package` usa `yum` para instalar os pacotes. Em um sistema Ubuntu Linux, o provedor `package` usa `apt-get`.

Você implementa um recurso como um bloco de código Ruby com o formato geral a seguir.

```
resource_type "resource_name" do
  attribute1 'value1'
  attribute2 'value2'
  ...
  action :action_name
  notifies : action 'resource'
end
```

Os elementos são:

Tipo de recurso

(Obrigatório) O exemplo inclui três tipos de recurso, `package`, `servicee` `template`.

Nome do recurso

(Obrigatório) O nome identifica o recurso específico e, às vezes, é usado como um valor padrão para um dos atributos. No exemplo, `package` representa um pacote de recursos chamado `haproxy`, e o primeiro recurso `template` representa um arquivo de configuração chamado `/etc/default/haproxy`.

Atributos

(Opcional) Os atributos especificam a configuração do recurso e variam de acordo com o tipo de recurso e como você deseja configurar o recurso.

- Os recursos `template` do exemplo definem explicitamente um conjunto de atributos que especificam a origem do arquivo criado, o proprietário, o grupo e o modo.
- Os recursos `package` e `service` do exemplo não definem explicitamente nenhum atributo.

  O nome do recurso costuma ser o valor padrão de um atributo obrigatório e, às vezes, é tudo o que basta. Por exemplo, o nome do recurso é o valor padrão do atributo do `package` do recurso `package_name`, o único atributo obrigatório.

Também existem alguns atributos especializadas chamados atributos de proteção, que especificam quando o provedor de recursos deve realizar uma ação. Por exemplo, o atributo

`only_if` leva o provedor de recursos a só executar a ação caso uma condição especificada seja atendida. A HAProxy receita não usa atributos de proteção, mas eles são usados por vários dos exemplos a seguir.

Ações e notificações

(Opcional) Actions e notificações especificam quais tarefas o provedor deve realizar.

- `action` leva o provedor a executar uma ação especificada, como instalar ou criar.

  Cada recurso tem um conjunto de ações que dependem do recurso específico, um dos quais é a ação padrão. No exemplo, a ação do recurso `package` é `install`, que leva o provedor a instalar o pacote. Como o primeiro recurso `template` não tem elemento `action`, o provedor utiliza a ação `create` padrão.

- `notifies` leva outro provedor do recurso a realizar uma ação, mas somente caso o estado do recurso tenha sido alterado.

  `notifies` costuma ser usado com recursos como `template` e `file` para realizar tarefas como reiniciar um serviço após a modificação de um arquivo de configuração. Os recursos não têm notificações padrão. Caso você queira uma notificação, o recurso deve ter um elemento `notifies` explícito. Na HAProxy receita, o segundo `template` recurso notifica o recurso haproxy `service` para reiniciar o HAProxy serviço se o arquivo de configuração associado tiver sido alterado.

Às vezes, os recursos dependem do sistema operacional.

- Alguns recursos só podem ser usados em sistemas Linux ou Windows.

  Por exemplo, package instala pacotes em sistemas Linux e windows_package instala pacotes em sistemas Windows.

- Alguns recursos podem ser usados com qualquer sistema operacional, mas têm atributos específicos de um determinado sistema.

  Por exemplo, o recurso file pode ser usado em sistemas Linux ou Windows, mas tem conjuntos de atributos separados para configurar permissões.

Para obter descrições dos recursos padrão, inclusive os atributos, as ações e as notificações disponíveis para cada recurso, consulte About Resources and Providers.

Controle de fluxo

Como receitas são aplicações do Ruby, você pode usar as estruturas de controle do Ruby para incorporar o controle de fluxo a uma receita. Por exemplo, você pode usar a lógica condicional do Ruby para que a receita se comporte de maneira diferente em sistemas diferentes. A HAProxy receita inclui um `if` bloco que usa um `template` recurso para criar um arquivo de configuração, mas somente se a receita estiver sendo executada em um sistema Debian ou Ubuntu.

Outro cenário comum é usar um loop para executar um recurso várias vezes com configurações de atributo diferentes. Por exemplo, você pode criar um conjunto de diretórios usando um loop para executar um recurso `directory` várias vezes com nomes de diretórios diferentes.

> **Note**
>
> Caso você não esteja familiarizado com o Ruby, consulte Just Enough Ruby for Chef, que aborda o que você precisa saber para a maioria das receitas.

Receitas incluídas

`include_recipe` inclui outras receitas no código, o que permite modularizar as receitas e reutilizar o mesmo código em várias. Quando você executa a receita host, o Chef substitui cada elemento `include_recipe` pelo código da receita especificado antes de executar a receita host. Você identifica uma receita incluída usando a sintaxe `cookbook_name::recipe_name` do Chef, em que `recipe_name` omite a extensão `.rb`. O exemplo inclui uma receita,`haproxy::service`, que representa o HAProxy serviço.

> **Note**
>
> Caso use `include_recipe` em receitas em execução no Chef 11.10 e posterior para incluir uma receita de outro livro de receitas, você deve usar uma instrução `depends` para declarar a dependência no arquivo `metadata.rb` do livro de receitas. Para obter mais informações, consulte Implementação de receitas: Chef 11.10.

Exemplo 1: Instalação de pacotes

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A instalação do pacote é um dos usos mais comuns de receitas e pode ser bem simples, dependendo do pacote. Por exemplo, a receita a seguir instala o Git em um sistema Linux.

```
package 'git' do
  action :install
end
```

O recurso `package` processa a instalação do pacote. Neste exemplo, você não precisa especificar nenhum atributo. O nome do recurso é o valor padrão do atributo `package_name`, que identifica o pacote. A ação `install` leva o provedor a instalar o pacote. Você pode simplificar ainda mais o código ignorando `install`; trata-se da ação padrão do recurso `package`. Quando você executa a receita, o Chef usa o provedor apropriado para instalar o pacote. No sistema Ubuntu que você usará para o exemplo, o provedor instala o Git chamando `apt-get`.

> **Note**
>
> A instalação do software em um sistema Windows exige um procedimento um pouco diferente. Para obter mais informações, consulte Instalar o softwares do Windows.

Para usar o Test Kitchen a fim de executar essa receita no Vagrant, você primeiro precisa configurar um livro de receitas, além de inicializar e configurar o Test Kitchen. Este procedimento se destina um sistema Linux, mas é essencialmente igual para sistemas Windows e Macintosh. Comece abrindo uma janela Terminal; todos os exemplos deste capítulo usam ferramentas de linha de comando.

Para preparar o livro de receitas

1. No diretório inicial, crie um subdiretório chamado `opsworks_cookbooks`, que conterá todos os livros de receitas deste capítulo. Em seguida, crie um subdiretório para este livro de receitas chamado `installpkg` e navegue até ele.
2. Em `installpkg`, crie um arquivo chamado `metadata.rb` que contém o código a seguir.

```
name "installpkg"
version "0.1.0"
```

   Para simplificar, os exemplos deste capítulo apenas especificam o nome e a versão do livro de receitas, mas `metadata.rb` pode conter uma grande variedade de metadados sobre o livro de receitas. Para obter mais informações, consulte Sobre metadados do livro de receitas.

   > **Note**
   >
   > Não se esqueça de criar `metadata.rb` antes de inicializar o Test Kitchen; ele usa os dados para criar o arquivo de configuração padrão.

3. Em `installpkg`, execute `kitchen init`, que inicializa o Test Kitchen e instala o driver do Vagrant padrão.
4. O comando `kitchen init` cria um arquivo de configuração YAML em `installpkg` chamado `.kitchen.yml`. Abra o arquivo no editor de texto favorito. O arquivo `.kitchen.yml` inclui uma seção `platforms` que especifica em quais sistemas as receitas devem ser executadas. O Test Kitchen cria uma instância e executa as receitas especificadas em cada plataforma.

   > **Note**
   >
   > Por padrão, o Test Kitchen executa receitas uma plataforma por vez. Se você adicionar um argumento `-p` a qualquer comando que cria uma instância, o Test Kitchen executará as receitas em todas as plataformas, em paralelo.

   Como uma única plataforma é suficiente para esse exemplo, edite `.kitchen.yml` para remover a plataforma `centos-6.4`. Seu arquivo `.kitchen.yml` deve se parecer com o seguinte:

```
---
driver:
  name: vagrant

provisioner:
  name: chef_solo

platforms:
  - name: ubuntu-12.04

suites:
  - name: default
    run_list:
      - recipe[installpkg::default]
    attributes:
```

O Test Kitchen só executa as receitas que estejam na lista de execuções `.kitchen.yml`. Você identifica receitas usando o `[`*`cookbook_name`*`::`*`recipe_name`*`]` formato, onde *`recipe_name`* omite a `.rb` extensão. Inicialmente, a lista de execuções `.kitchen.yml` contém a receita padrão do livro de receitas, `installpkg::default`. Como essa é a receita que implementará, você não precisa modificar a lista de execução.

5. Crie um subdiretório de `installpkg` chamado `recipes`.

   Caso um livro de receitas contenha receitas (a maioria deles contém), elas devem estar no subdiretório `recipes`.

Você já pode adicionar a receita ao livro de receitas e usar o Test Kitchen para executá-la em uma instância.

Para executar a receita

1. Crie um arquivo chamado `default.rb` que contenha o código de exemplo de instalação do Git no início da seção o salve no subdiretório `recipes`.
2. No diretório `installpkg`, execute `kitchen converge`. Este comando inicia uma nova instância do Ubuntu no Vagrant, copia os livros de receitas para a instância e inicia uma execução do Chef para realizar as receitas na lista de execução `.kitchen.yml`.

3. Para verificar se a receita foi bem-sucedida, execute `kitchen login`, que abre uma conexão SSH para a instância. Em seguida, execute `git --version` para verificar se o Git foi instalado com êxito. Para retornar à estação de trabalho, execute `exit`.
4. Quando terminar, execute `kitchen destroy` para desligar a instância. O exemplo a seguir usa um livro de receitas diferente.

Este exemplo foi uma boa maneira de começar, mas é especialmente simples. A instalação de outros pacotes pode ser mais complicada; talvez você possa precisar seguir uma ou todas as seguintes opções:

- Crie e configure um usuário.
- Crie um ou mais diretórios para dados, logs etc.
- Instale um ou mais arquivos de configuração.
- Especifique um nome de pacote ou valores de atributo diferentes para sistemas operacionais distintos.
- Inicie um serviço e o reinicie, conforme necessário.

Os exemplos a seguir descrevem como resolver esses problemas, além de algumas outras operações úteis.

Exemplo 2: Gerenciamento de usuários

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Outra tarefa simples é o gerenciamento de usuários em uma instância. O código a seguir adiciona um novo usuário a uma instância do Linux.

```
user "myuser" do
```

```
  home "/home/newuser"
  shell "/bin/bash"
end
```

Você usa um recurso user para gerenciar usuários nos sistemas Linux e Windows, embora alguns atributos se apliquem a apenas um sistema. O exemplo cria um usuário chamado `myuser` e especifica o diretório inicial e o shell. Como não há ação especificada, o recurso usa a ação `create` padrão. Você pode adicionar atributos a `user` para especificar várias outras configurações, como a senha ou a ID do grupo. Você também pode usar `user` em tarefas de gerenciamento de usuário relacionadas, como a modificação das configurações do usuário ou a exclusão de usuários. Para obter mais informações, consulte user.

Para executar a receita

1. Crie um diretório em `opsworks_cookbooks` chamado `newuser` e navegue até ele.
2. Crie um arquivo `metadata.rb` que contenha o código a seguir e o salve em `newuser`.

   ```
   name "newuser"
   version "0.1.0"
   ```

3. Inicialize e configure o Test Kitchen, conforme descrito em Exemplo 1: Instalação de pacotes e adicione um diretório `recipes` dentro do diretório `newuser`.
4. Adicione o arquivo `default.rb` com a receita de exemplo ao diretório `recipes` do livro de receitas.
5. Execute `kitchen converge` para executar a receita.
6. Use `kitchen login` para fazer logon na instância e verificar a existência do novo usuário executando `cat /etc/passwd`. O usuário `myuser` deve estar na parte inferior do arquivo.

Exemplo 3: Criação de diretórios

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Ao instalar um pacote em uma instância, você normalmente precisa criar alguns arquivos de configuração e colocá-los nos diretórios indicados. No entanto, esses diretórios talvez ainda não existam. Você talvez também precise criar diretórios para arquivos de dados, log etc. Por exemplo, você inicializa primeiramente o sistema Ubuntu usado para a maioria dos exemplos, o diretório `/srv` não tem subdiretórios. Se estiver instalando um servidor de aplicativos, você provavelmente desejará um diretório `/srv/www/` e talvez alguns subdiretórios para arquivos de dados, logs etc. A receita a seguir cria `/srv/www/` em uma instância.

```
directory "/srv/www/" do
  mode 0755
  owner 'root'
  group 'root'
  action :create
end
```

Você usa um `directory` resource para criar e configurar diretórios nos sistemas Linux e Windows, embora alguns atributos sejam usados de maneira diferente. Como o nome do recurso é o valor padrão do atributo `path` do recurso, o exemplo cria `/srv/www/` e especifica as propriedades `mode`, `owner` e `group`.

Para executar a receita

1. Crie um diretório dentro de `opsworks_cookbooks` chamado `createdir` e navegue até ele.
2. Inicialize e configure o Test Kitchen, conforme descrito em Exemplo 1: Instalação de pacotes e adicione um diretório `recipes` dentro de `createdir`.
3. Adicione um arquivo `default.rb` com o código da receita ao subdiretório `recipes` do livro de receitas.
4. Execute `kitchen converge` para executar a receita.
5. Execute `kitchen login`, navegue até `/srv` e verifique se ele tem um subdiretório `www`.
6. Execute `exit` para retornar à estação de trabalho, mas deixe a instância em execução.

> **Note**
>
> Para criar um diretório relativo ao diretório inicial na instância, use `#{ENV['HOME']}` para representar o diretório inicial. Por exemplo, a opção a seguir cria o diretório `~/shared`.

```
directory "#{ENV['HOME']}/shared" do
  ...
end
```

Suponhamos que você queira criar um diretório mais aninhado, como `/srv/www/shared`. Você pode modificar a receita anterior da maneira a seguir.

```
directory "/srv/www/shared" do
  mode 0755
  owner 'root'
  group 'root'
  action :create
end
```

Para executar a receita

1. Substitua o código em `default.rb` pela receita anterior.
2. Execute `kitchen converge` no diretório `createdir`.
3. Para verificar se o diretório foi mesmo criado, execute `kitchen login`, navegue até `/srv/www` e verifique se ele contém um subdiretório `shared`.
4. Execute `kitchen destroy` para desligar a instância.

Você perceberá que o comando `kitchen converge` foi executado muito mais rapidamente. Isso porque a instância já está em execução, logo, não há necessidade de inicializar a instância, instalar o Chef e assim por diante. Teste o Kitchen apenas a fim de copiar o livro de receitas atualizado para a instância e iniciar uma execução do Chef.

Agora reexecute `kitchen converge`, que executa a receita em uma nova instância. Você já verá o resultado a seguir.

```
Chef Client failed. 0 resources updated in 1.908125788 seconds
[2014-06-20T20:54:26+00:00] ERROR: directory[/srv/www/shared] (createdir::default line
 1) had an error: Chef::Exceptions::EnclosingDirectoryDoesNotExist: Parent directory /
srv/www does not exist, cannot create /srv/www/shared
```

```
[2014-06-20T20:54:26+00:00] FATAL: Chef::Exceptions::ChildConvergeError: Chef run
 process exited unsuccessfully (exit code 1)
>>>>>> Converge failed on instance <default-ubuntu-1204>.
>>>>>> Please see .kitchen/logs/default-ubuntu-1204.log for more details
>>>>>> ------Exception-------
>>>>>> Class: Kitchen::ActionFailed
>>>>>> Message: SSH exited (1) for command: [sudo -E chef-solo --config /tmp/kitchen/
solo.rb --json-attributes /tmp/kitchen/dna.json  --log_level info]
>>>>>> ----------------------
```

O que aconteceu? O problema é que, por padrão, um recurso `directory` só pode criar um diretório por vez; ele não pode criar uma cadeia de diretórios. O motivo pelo qual a receita funcionou antes é que a primeira receita que você executou na instância já tinha criado `/srv/www`, logo, a criação de `/srv/www/shared` gerou apenas um subdiretório.

> **Note**
>
> Quando você executar `kitchen converge`, certifique-se de que saiba se está executando as receitas em uma instância nova ou existente. Você pode obter resultados diferentes.

Para criar uma cadeia de subdiretórios, adicione um atributo `recursive` a `directory` e o defina como `true`. A receita a seguir cria `/srv/www/shared` diretamente em uma instância limpa.

```
directory "/srv/www/shared" do
  mode 0755
  owner 'root'
  group 'root'
  recursive true
  action :create
end
```

Exemplo 4: Adição do controle de fluxo

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Algumas receitas são apenas uma série de recursos do Chef. Neste caso, quando você executa a receita, ela simplesmente executa cada um dos provedores de recursos em sequência. No entanto, costuma ser útil ter um caminho de execução mais sofisticado. Estes são dois cenários comuns:

- Você deseja que uma receita execute o mesmo recurso várias vezes com configurações de atributo diferentes.
- Você deseja usar configurações de atributo diferentes em sistemas operacionais distintos.

Você pode abordar cenários como esses incorporando estruturas de controle do Ruby à receita. Esta seção mostra como modificar a receita de Exemplo 3: Criação de diretórios para resolver ambos os cenários.

Tópicos

- Iteração
- Lógica condicional

Iteração

Exemplo 3: Criação de diretórios mostrou como usar um recurso `directory` para criar um diretório ou uma cadeia de diretórios. No entanto, suponhamos que você queira criar dois diretórios separados, `/srv/www/config` e `/srv/www/shared`. Você pode implementar um recurso de diretório separado para cada diretório, mas essa abordagem pode atrapalhar caso você queira criar muitos diretórios. A receita a seguir mostra uma maneira mais simples de realizar a tarefa.

```
[ "/srv/www/config", "/srv/www/shared" ].each do |path|
  directory path do
    mode 0755
    owner 'root'
    group 'root'
    recursive true
    action :create
  end
end
```

Em vez de usar um recurso de diretório separado para cada subdiretório, a receita usa uma coleção de strings que contém os caminhos de subdiretório. O método do Ruby `each` executa o recurso uma vez para cada elemento de coleção, começando pelo primeiro. O valor do elemento é representado no recurso pela variável `path`, que, neste caso, representa o caminho do diretório. Você pode adaptar facilmente este exemplo para criar qualquer número de subdiretórios.

Para executar a receita

1. Permaneça no diretório `createdir`; você usará esse livro de receitas nos vários próximos exemplos.
2. Caso você ainda não tenha feito isso, execute `kitchen destroy` de maneira a começar com uma instância limpa.
3. Substitua o código em `default.rb` pelo exemplo e execute `kitchen converge`.
4. Faça logon na instância; você verá os diretórios recém-criados em `/srv`.

Você pode usar uma tabela de hash a fim de especificar dois valores para cada iteração. A receita a seguir cria `/srv/www/config` e `/srv/www/shared`, cada um com um modo diferente.

```
{ "/srv/www/config" => 0644, "/srv/www/shared" => 0755 }.each do |path, mode_value|
  directory path do
    mode mode_value
    owner 'root'
    group 'root'
    recursive true
    action :create
  end
end
```

Para executar a receita

1. Caso você ainda não tenha feito isso, execute `kitchen destroy` de maneira a começar com uma instância limpa.
2. Substitua o código em `default.rb` pelo exemplo e execute `kitchen converge`.
3. Faça logon na instância; você verá os diretórios recém-criados em `/srv` com os modos especificados.

> **Note**
>
> AWS OpsWorks As receitas de pilhas geralmente usam essa abordagem para extrair valores do JSON de configuração e implantação da pilha, que é basicamente uma grande tabela de hash, e inseri-los em um recurso. Para obter um exemplo, consulte Receitas de implantação.

Lógica condicional

Você também pode usar a lógica condicional do Ruby para criar várias ramificações de execução. A receita a seguir usa a lógica `if-elsif-else` para estender o exemplo anterior, de maneira que ela crie um subdiretório chamado `/srv/www/shared`, mas apenas em sistemas Debian e Ubuntu. Para todos os outros sistemas, ela registra uma mensagem de erro exibida na saída do Test Kitchen.

```
if platform?("debian", "ubuntu")
  directory "/srv/www/shared" do
    mode 0755
    owner 'root'
    group 'root'
    recursive true
    action :create
  end
else
  log "Unsupported system"
end
```

Para executar a receita de exemplo

1. Caso a instância ainda esteja em execução, execute `kitchen destroy` para desligá-la.
2. Substitua o código em `default.rb` pelo código de exemplo.
3. Edite `.kitchen.yml` para adicionar um sistema CentOS 6.4 à lista de plataformas. A seção `platforms` do arquivo deve ser semelhante.

```
...
platforms:
  - name: ubuntu-12.04
  - name: centos-6.4
```

```
...
```

4. Execute `kitchen converge`, que irá criar uma instância e executar as receitas de cada plataforma em `.kitchen.yml` na sequência.

> **Note**
>
> Caso você queira apenas convergir uma instância, adicione o nome da instância como um parâmetro. Por exemplo, para convergir a receita apenas na plataforma Ubuntu, execute `kitchen converge default-ubuntu-1204`. Caso você esqueça os nomes da plataforma, basta executar `kitchen list`.

Você deve ver a mensagem de log na parte CentOS da saída do Test Kitchen, que será semelhante ao seguinte:

```
...
Converging 1 resources
Recipe: createdir::default
* log[Unsupported system] action write[2014-06-23T19:10:30+00:00] INFO: Processing
 log[Unsupported system] action write (createdir::default line 12)
[2014-06-23T19:10:30+00:00] INFO: Unsupported system

[2014-06-23T19:10:30+00:00] INFO: Chef Run complete in 0.004972162 seconds
```

Você já pode fazer logon nas instâncias e verifique se os diretórios foram criados ou não. No entanto, não basta executar `kitchen login` agora. Você deve especificar qual instância anexando o nome da plataforma; por exemplo, `kitchen login default-ubuntu-1204`.

> **Note**
>
> Caso um comando do Test Kitchen utilize o nome de uma instância, você não precisa digitar o nome completo. O Test Kitchen trata o nome de uma instância como uma expressão regular do Ruby. Assim, você precisa apenas de caracteres suficientes para apresentar uma correspondência exclusiva. Por exemplo, você pode convergir apenas a instância do Ubuntu executando `kitchen converge ub` ou fazer logon na instância do CentOS executando `kitchen login 64`.

A pergunta que você provavelmente tem a esta altura é como a receita sabe em qual plataforma está sendo executada. O Chef executa uma ferramenta chamada Ohai em todas as execuções que coletam dados do sistema, inclusive a plataforma, e representa como um conjunto de atributos em uma estrutura chamada de objeto nó. O método `platform?` do Chef compara os sistemas entre parênteses com o valor da plataforma Ohai e retorna verdadeiro caso haja correspondência de um deles.

Você pode consultar o valor de um atributo nó diretamente no código usando `node['attribute_name']`. O valor da plataforma, por exemplo, é representado por `node['platform']`. Você pode, por exemplo, ter escrito o exemplo anterior da maneira a seguir.

```
if node[:platform] == 'debian' or node[:platform] == 'ubuntu'
  directory "/srv/www/shared" do
    mode 0755
    owner 'root'
    group 'root'
    recursive true
    action :create
  end
else
  log "Unsupported system"
end
```

Um motivo comum para incluir lógica condicional em uma receita é acomodar o fato de que, às vezes, famílias do Linux diferentes usam nomes diferentes para pacotes, diretórios etc. Por exemplo, o nome do pacote do Apache é `httpd` em sistemas CentOS e `apache2` em sistemas Ubuntu.

Caso você só precise de uma string diferentes para sistemas distintos, o método `value_for_platform` do Chef é uma solução mais simples do que `if-elsif-else`. A receita a seguir cria um diretório `/srv/www/shared` em sistemas CentOS, um diretório `/srv/www/data` em sistemas Ubuntu, e `/srv/www/config` em todos os outros.

```
data_dir = value_for_platform(
  "centos" => { "default" => "/srv/www/shared" },
  "ubuntu" => { "default" => "/srv/www/data" },
  "default" => "/srv/www/config"
)
directory data_dir do
  mode 0755
```

```
  owner 'root'
  group 'root'
  recursive true
  action :create
end
```

`value_for_platform` atribui o caminho apropriado a `data_dir` e o recurso `directory` usa esse valor para criar o diretório.

Para executar a receita de exemplo

1. Caso a instância ainda esteja em execução, execute `kitchen destroy` para desligá-la.
2. Substitua o código em `default.rb` pelo código de exemplo.
3. Execute `kitchen converge` e faça logon em cada instância para verificar se os diretórios apropriados estão presentes.

Exemplo 5: Uso de atributos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As receitas nas seções anteriores usaram valores codificados em tudo que não fosse a plataforma. Essa abordagem poderá ser incômoda se, por exemplo, você quiser usar o mesmo valor em mais de uma receita. Você pode definir valores separadamente de receitas incluindo um arquivo de atributo no livro de receitas.

Arquivo de atributo é uma aplicação do Ruby que atribui valores a um ou mais atributos. Ele deve estar na pasta `attributes` do livro de receitas. O Chef incorpora os atributos ao objeto nó, e qualquer receita pode usar os valores de atributo referenciando o atributo. Este tópico mostra como modificar a receita de Iteração para usar atributos. Aqui está a receita original para referência.

```
[ "/srv/www/config", "/srv/www/shared" ].each do |path|
  directory path do
    mode 0755
    owner 'root'
    group 'root'
    recursive true
    action :create
  end
end
```

A seguir, a definição de atributos para os valores de nome, modo, proprietário e grupo do subdiretório.

```
default['createdir']['shared_dir'] = 'shared'
default['createdir']['config_dir'] = 'config'
default['createdir']['mode'] = 0755
default['createdir']['owner'] = 'root'
default['createdir']['group'] = 'root'
```

Observe o seguinte:

- Toda definição começa com um tipo de atributo.

  Caso um atributo seja definido mais de uma vez, talvez em arquivos de atributo diferentes, o tipo de atributo especifica a precedência do atributo, que determina qual definição é incorporada ao objeto nó. Para obter mais informações, consulte Precedência de atributo. Todas as definições neste exemplo têm o tipo de atributo `default`, que é o tipo usual para essa finalidade.
- Os atributos têm nomes aninhados.

  O nó objeto é basicamente uma tabela de hash que pode ser aninhada profundamente de maneira arbitrária, de maneira que nomes de atributo possam ser e normalmente são aninhados. Este arquivo de atributo segue uma prática padrão de usar um nome aninhado com o nome do livro de receitas, `createdir`, como o primeiro elemento.

A razão para usar createdir como o primeiro elemento do atributo é que, quando você faz uma execução do Chef, o Chef incorpora os atributos de todos os livros de receitas ao objeto nó. Com o AWS OpsWorks Stacks, o objeto node inclui um grande número de atributos dos livros de receitas integrados, além de quaisquer atributos que você definir. A inclusão do nome do livro de receitas no nome do atributo reduz o risco de uma colisão de nomes com atributos de outro livro de receitas,

especialmente caso o atributo tenha um nome como `port` ou `user`. Não nomeie um atributo com algo como `[:apache2][:user]`, por exemplo, a menos que você queira substituir o valor desse atributo. Para obter mais informações, consulte Usando atributos de livro de receitas personalizado.

O exemplo a seguir mostra a receita original usando atributos, em vez de valores codificados.

```
[ "/srv/www/#{node['createdir']['shared_dir']}", "/srv/www/#{node['createdir']
['config_dir']}" ].each do |path|
  directory path do
    mode node['createdir']['mode']
    owner node['createdir']['owner']
    group node['createdir']['group']
    recursive true
    action :create
  end
end
```

> **Note**
>
> Caso você queira incorporar um valor de atributo a uma string, encapsule-o com `#{}`. No exemplo anterior, `#{node['createdir']['shared_dir']}` acrescenta "shared" a "/srv/www/".

Para executar a receita

1. Execute `kitchen destroy` para começar com uma instância limpa.
2. Substitua o código em `recipes/default.rb` pelo exemplo de receita anterior.
3. Crie um subdiretório de `createdir` chamado `attributes` e adicione um arquivo chamado `default.rb` que contenha as definições de atributo.
4. Edite `.kitchen.yml` para remover CentOS da lista de plataformas.
5. Execute `kitchen converge` e faça logon na instância para verificar se os diretórios `/srv/www/shared` e `/srv/www/config` estão lá.

> **Note**
>
> Com o AWS OpsWorks Stacks, definir valores como atributos fornece um benefício adicional; você pode usar o JSON personalizado para substituir esses valores por pilha ou até mesmo por implantação. Isso pode ser útil em várias finalidades, inclusive a seguinte:
>
> - Você pode personalizar o comportamento das receitas, como definições de configuração ou nomes de usuário, sem a necessidade de modificar o livro de receitas.
>
> Você pode, por exemplo, usar o mesmo livro de receitas em pilhas personalizado e usar JSON personalizado a fim de especificar definições de configuração principais para uma determinada pilha. Isso economiza o tempo e o esforço necessários para modificar o livro de receitas ou usar um livro de receitas diferente para cada pilha.
> - Você não precisa colocar informações potencialmente confidenciais, como senhas de banco de dados, no repositório do livro de receitas.
>
> Em vez disso, você pode usar um atributo para definir um valor padrão e usar o JSON personalizado para substituir esse valor pelo real.
>
> Para obter mais informações sobre como usar JSON personalizado para substituir atributos, consulte Sobrepor atributos.

O arquivo de atributo é chamado de `default.rb` porque é uma aplicação do Ruby, caso ainda seja simples. Isso significa que você pode, por exemplo, usar lógica condicional para especificar valores de atributo com base no sistema operacional. Em Lógica condicional, você especificou um nome de subdiretório diferente para famílias do Linux diferentes na receita. Com um arquivo de atributo, você pode colocar a lógica condicional no arquivo de atributo.

O arquivo de atributo a seguir usa `value_for_platform` para especificar um valor de atributo `['shared_dir']` diferente, dependendo do sistema operacional. Para outras condições, você pode usar a lógica `if-elsif-else` do Ruby ou uma instrução `case`.

```
data_dir = value_for_platform(
  "centos" => { "default" => "shared" },
  "ubuntu" => { "default" => "data" },
  "default" => "user_data"
)
```

```
default['createdir']['shared_dir'] = data_dir
default['createdir']['config_dir'] = "config"
default['createdir']['mode'] = 0755
default['createdir']['owner'] = 'root'
default['createdir']['group'] = 'root'
```

Para executar a receita

1. Execute `kitchen destroy` para começar com uma instância nova.
2. Substitua o código em `attributes/default.rb` pelo exemplo anterior.
3. Edite `.kitchen.yml` para adicionar uma plataforma do CentOS à seção de plataformas, conforme descrito em Lógica condicional.
4. Execute `kitchen converge` e faça logon nas instâncias para verificar se os diretórios estão lá.

Quando terminar, execute `kitchen destroy` para encerrar a instância. O exemplo a seguir usa um novo livro de receitas.

Exemplo 6: Criação de arquivos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Depois de ter criado diretórios, você normalmente precisa preenchê-los com arquivos de configuração, arquivos de dados etc. Este tópico mostra duas maneiras de instalar arquivos em uma instância.

Tópicos

- Instalação de um arquivo usando um livro de receitas
- Criação de um arquivo usando um modelo

Instalação de um arquivo usando um livro de receitas

A maneira mais simples de instalar um arquivo em uma instância é usando um recurso `cookbook_file`, que copia um arquivo do livro de receitas para um local especificado na instância para os sistemas Linux e Windows. Este exemplo estende a receita do Exemplo 3: Criação de diretórios para adicionar um arquivo de dados a `/srv/www/shared` após a criação do diretório. Para referência, aqui está a receita original.

```
directory "/srv/www/shared" do
  mode 0755
  owner 'root'
  group 'root'
  recursive true
  action :create
end
```

Para configurar o livro de receitas

1. No diretório `opsworks_cookbooks`, crie um diretório chamado `createfile` e navegue até ele.
2. Adicione um arquivo `metadata.rb` a `createfile`, com o seguinte conteúdo.

   ```
   name "createfile"
   version "0.1.0"
   ```

3. Inicialize e configure o Test Kitchen, conforme descrito em Exemplo 1: Instalação de pacotes e remova CentOS da lista `platforms`.
4. Adicione um subdiretório `recipes` a `createfile`.

O arquivo a ser instalado contém os dados JSON a seguir.

```
{
  "my_name" : "myname",
  "your_name" : "yourname",
  "a_number" : 42,
  "a_boolean" : true
```

```
}
```

Para configurar o arquivo de dados

1. Adicione um subdiretório `files` e `createfile` e um subdiretório `default` a `files`. Qualquer arquivo instalado por você com `cookbook_file` deve estar em um subdiretório de `files`, como `files/default` neste exemplo.

   > **Note**
   >
   > Caso queira especificar arquivos diferentes para sistemas distintos, você pode colocar o arquivo específico de cada sistema em uma subpasta chamada considerando o sistema, como `files/ubuntu`. O recurso `cookbook_file` copia o arquivo específico do sistema apropriado, caso ele exista, e usa o arquivo `default`. Para obter mais informações, consulte cookbook_file.

2. Crie um arquivo chamado `example_data.json` com o JSON do exemplo anterior e adicione-o a `files/default`.

A receita a seguir copia `example_data.json` para um local especificado.

```
directory "/srv/www/shared" do
  mode 0755
  owner 'root'
  group 'root'
  recursive true
  action :create
end

cookbook_file "/srv/www/shared/example_data.json" do
  source "example_data.json"
  mode 0644
  action :create_if_missing
end
```

Depois que cria `/srv/www/shared`, o recurso `cookbook_file` copia `example_data.json` para esse diretório e também define o usuário, o grupo e o modo do arquivo.

> **Note**
>
> O recurso `cookbook_file` apresenta uma nova ação: `create_if_missing`. Você também pode usar uma ação `create`, mas ela substitui um arquivo existente. Caso você não queira substituir nada, use `create_if_missing`, que só instala `example_data.json` caso ainda não exista.

Para executar a receita

1. Execute `kitchen destroy` para começar com uma instância nova.
2. Crie um arquivo `default.rb` que contenha a receita anterior e o salve em `recipes`.
3. Execute `kitchen converge` e faça logon na instância para verificar se o arquivo `/srv/www/shared` contém `example_data.json`.

Criação de um arquivo usando um modelo

O recurso `cookbook_file` é útil para algumas finalidades, mas só instala o arquivo que você tem no livro de receitas. Um recurso `template` oferece uma maneira mais flexível de instalar um arquivo em uma instância do Windows ou do Linux o criando dinamicamente com base em um modelo. Você pode acabar determinando os detalhes do conteúdo do arquivo no runtime e alterá-los conforme necessário. Por exemplo, você talvez queira que um arquivo de configuração tenha uma determinada configuração ao iniciar a instância e modificar a configuração depois, quando adicionar mais instâncias à pilha.

Este exemplo modifica o livro de receitas `createfile` a fim de usar um recurso `template` para instalar uma versão levemente modificada de `example_data.json`.

Aqui está como será a aparência do arquivo instalado.

```
{
  "my_name" : "myname",
  "your_name" : "yourname",
  "a_number" : 42,
  "a_boolean" : true,
  "a_string" : "some string",
  "platform" : "ubuntu"
```

```
}
```

Os recursos de modelo costumam ser usados com arquivos de atributo. Assim, o exemplo usa um para definir os valores a seguir.

```
default['createfile']['my_name'] = 'myname'
default['createfile']['your_name'] = 'yourname'
default['createfile']['install_file'] = true
```

Para configurar o livro de receitas

1. Exclua o diretório `createfile` e seu conteúdo do livro de receitas `files`.
2. Adicione um subdiretório `attributes` a `createfile` e um arquivo `default.rb` a `attributes` que contenha as definições de atributo anteriores.

Modelo é um arquivo `.erb` que é basicamente uma cópia do arquivo final, com alguns dos conteúdos representados por espaços reservados. Quando cria o arquivo, o recurso `template` copia o conteúdo do modelo para o arquivo especificado e substitui os espaços reservados pelos valores atribuídos. Veja o modelo de `example_data.json`.

```
{
  "my_name" : "<%= node['createfile']['my_name'] %>",
  "your_name" : "<%= node['createfile']['your_name'] %>",
  "a_number" : 42,
  "a_boolean" : <%= @a_boolean_var %>,
  "a_string" : "<%= @a_string_var %>",
  "platform" : "<%= node['platform'] %>"
}
```

Os valores `<%=...%>` são os espaços reservados.

- `<%=node[...]%>` representa um valor de atributo do nó.

  Para este exemplo, o valor "your_name" é um espaço reservado que representa um dos valores de atributo do arquivo de atributo do livro de receitas.
- `<%=@...%>` representa o valor de uma variável definido no recurso do modelo, conforme abordado resumidamente.

Como criar o arquivo de modelo do

1. Adicione um subdiretório `templates` ao livro de receitas `createfile` e um subdiretório `default` a `templates`.

   > **Note**
   >
   > O diretório `templates` funciona de maneira muito semelhante ao diretório `files`. Você pode colocar modelos específicos do sistema em um subdiretório, como `ubuntu`, nomeado por causa do sistema. O recurso `template` usa o modelo específico do sistema apropriado caso ele exista e acaba usando o modelo `default`.

2. Crie um arquivo chamado `example_data.json.erb` e coloque-o no diretório `templates/default`. O nome do modelo é arbitrário, mas você normalmente o cria acrescentando `.erb` ao nome do arquivo, inclusive extensões.

A receita a seguir usa um recurso `template` para criar `/srv/www/shared/example_data.json`.

```
directory "/srv/www/shared" do
  mode 0755
  owner 'root'
  group 'root'
  recursive true
  action :create
end

template "/srv/www/shared/example_data.json" do
  source "example_data.json.erb"
  mode 0644
  variables(
    :a_boolean_var => true,
    :a_string_var => "some string"
  )
  only_if {node['createfile']['install_file']}
end
```

O recurso `template` cria `example_data.json` usando um modelo e o instala em `/srv/www/shared`.

- O nome do modelo, `/srv/www/shared/example_data.json`, especifica o caminho e o nome do arquivo instalado.
- O atributo `source` especifica o modelo usado para criar o arquivo.
- O atributo `mode` especifica o modo do arquivo instalado.
- O recurso define duas variáveis, `a_boolean_var` e `a_string_var`.

  Quando cria `example_data.json`, o recurso substitui os espaços reservados da variável no modelo pelos valores correspondentes do recurso.
- O atributo `only_if` guard só leva o recurso a criar o arquivo caso `['createfile']` `['install_file']` esteja definido como `true`.

Para executar a receita

1. Execute `kitchen destroy` para começar com uma instância nova.
2. Substitua o código em `recipes/default.rb` pelo exemplo anterior.
3. Execute `kitchen converge` e faça logon na instância para verificar se o arquivo está em `/srv/www/shared` e tem o conteúdo correto.

Quando terminar, execute `kitchen destroy` para desligar a instância. A próxima seção usa um livro de receitas novo.

Exemplo 7: Execução de comandos e scripts

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Os recursos do Chef podem processar uma grande variedade de tarefas em uma instância, mas, às vezes, é preferível usar um comando do shell ou um script. Por exemplo, você talvez já tenha scripts que usa para realizar determinadas tarefas, e será mais fácil continuar usando eles do que

implementar um código novo. Esta seção mostra como executar comandos ou scripts em uma instância.

Tópicos

- Execução de comandos
- Execução de scripts

Execução de comandos

O recurso `script` executa um ou mais comandos. Como dá suporte a interpretadores de comando csh, bash, Perl, Python e Ruby, ele pode ser usado nos sistemas Linux ou Windows, desde que estes tenham os interpretadores apropriados instalados. Este tópico mostra como executar um comando bash simples em uma instância do Linux. O Chef também dá suporte aos recursos powershell_script e batch para executar scripts no Windows. Para obter mais informações, consulte Executando um PowerShell script do Windows.

Para começar

1. No diretório `opsworks_cookbooks`, crie um diretório chamado `script` e navegue até ele.
2. Adicione um arquivo `metadata.rb` a `script`, com o seguinte conteúdo.

```
name "script"
version "0.1.0"
```

3. Inicialize e configure o Test Kitchen, conforme descrito em Exemplo 1: Instalação de pacotes e remova CentOS da lista `platforms`.
4. Em `script`, crie um diretório chamado `recipes`.

Você pode executar comandos usando o recurso `script` propriamente dito, mas o Chef também dá suporte a um conjunto de versões específicas do interpretador de comandos do recurso, nomeadas segundo o interpretador. A receita a seguir usa um recurso `bash` para executar um script bash simples.

```
bash "install_something" do
  user "root"
  cwd "/tmp"
```

```
  code <<-EOH
    touch somefile
  EOH
  not_if do
    File.exists?("/tmp/somefile")
  end
end
```

O recurso `bash` é configurado da maneira a seguir.

- Ele usa a ação padrão, `run`, que executa os comandos no bloco `code`.

  Este exemplo tem um comando `touch somefile`, mas um `code` bloco pode conter vários comandos.
- O atributo `user` especifica o usuário que executa o comando.
- O atributo `cwd` especifica o diretório de trabalho.

  Para este exemplo, `touch` cria um arquivo no diretório `/tmp`.
- O atributo de proteção `not_if` leva o recurso a não tomar ação alguma caso o arquivo já exista.

Para executar a receita

1. Crie um arquivo `default.rb` que contenha o código de exemplo anterior e o salve em `recipes`.
2. Execute `kitchen converge` e faça logon na instância para verificar se o arquivo está em `/tmp`.

Execução de scripts

O recurso `script` é prático, especialmente caso você precise executar apenas um ou dois comandos, mas normalmente é preferível armazenar o script em um arquivo e executar o arquivo. O recurso `execute` executa um arquivo executável especificado, inclusive arquivos de script, no Linux ou no Windows. Este tópico modifica o livro de receitas `script` do exemplo anterior a fim de usar `execute` para executar um script de shell simples. Você pode estender facilmente o exemplo para scripts mais complexos ou outros tipos de arquivo executável.

Para configurar o arquivo de script

1. Adicione um subdiretório `files` e `script` e um subdiretório `default` a `files`.

2. Crie um arquivo chamado `touchfile` que contenha o seguinte e adicione-o a `files/default`. Uma linha do interpretador do Bash comum é usada nesse exemplo, mas substitua um interpretador que funcione para o ambiente do shell, caso necessário.

   ```
   #!/usr/bin/env bash
   touch somefile
   ```

   O arquivo de script pode conter qualquer número de comandos. Por uma questão de comodidade, esse script de exemplo tem apenas um único comando `touch`.

A receita a seguir executa o script.

```
cookbook_file "/tmp/touchfile" do
  source "touchfile"
  mode 0755
end

execute "touchfile" do
  user "root"
  cwd "/tmp"
  command "./touchfile"
end
```

O recurso `cookbook_file` copia o arquivo de script para `/tmp` e define o modo para tornar o arquivo executável. O recurso `execute` acaba executando o arquivo da seguinte forma:

- O atributo `user` especifica o usuário do comando (`root` neste exemplo).
- O atributo `cwd` especifica o diretório de trabalho (`/tmp` neste exemplo).
- O atributo `command` especifica o script a ser executado (`touchfile`, neste exemplo), localizado no diretório de trabalho.

Para executar a receita

1. Substitua o código em `recipes/default.rb` pelo exemplo anterior.
2. Execute `kitchen converge` e faça logon na instância para verificar se `/tmp` já contém o arquivo de script, com o modo definido como 0755 e `somefile`.

Quando terminar, execute `kitchen destroy` para desligar a instância. A próxima seção usa um livro de receitas novo.

Exemplo 8: Gerenciamento de serviços

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Pacotes como servidores de aplicação normalmente têm um serviço associado que deve ser iniciado, parado, reiniciado etc. Por exemplo, você precisará iniciar o serviço Tomcat depois de instalar o pacote ou a instância terminar a inicialização e reiniciar o serviço sempre que modificar o arquivo de configuração. Este tópico aborda os conceitos básicos de como gerenciar um serviço em uma instância do Linux usando um servidor de aplicações Tomcat como um exemplo. O recurso de serviço funciona da mesma maneira que instâncias do Windows, embora haja algumas diferenças em detalhes. Para obter mais informações, consulte `service`.

> **Note**
>
> O exemplo faz uma instalação do Tomcat muito mínima, apenas o suficiente para demonstrar os conceitos básicos de como usar um recurso `service`. Para obter um exemplo de como implementar receitas para um servidor Tomcat mais funcional, consulte Criação de uma camada de servidor Tomcat personalizada.

Tópicos

- Definição e inicialização de um serviço
- Uso de notifies para iniciar ou reiniciar um serviço

Definição e inicialização de um serviço

Esta seção mostra os conceitos básicos de como definir e iniciar um serviço.

Para começar

1. No diretório `opsworks_cookbooks`, crie um diretório com nome `tomcat` e navegue até ele.
2. Adicione um arquivo `metadata.rb` a `tomcat`, com o seguinte conteúdo.

```
name "tomcat"
version "0.1.0"
```

3. Inicialize e configure o Test Kitchen, conforme descrito em Exemplo 1: Instalação de pacotes e remova CentOS da lista `platforms`.
4. Adicione um subdiretório `recipes` a `tomcat`.

Você usa um recurso `service` para gerenciar um serviço. A receita padrão a seguir instala o Tomcat e inicia o serviço.

```
execute "install_updates" do
  command "apt-get update"
end

package "tomcat7" do
    action :install
end

include_recipe 'tomcat::service'

service 'tomcat' do
  action :start
end
```

A receita faz o seguinte:

- O recurso `execute` executa `apt-get update` para instalar as atualizações do sistema atual.

  Para a instância do Ubuntu usada neste exemplo, você deverá instalar as atualizações antes de instalar o Tomcat. Outros sistemas podem ter requisitos diferentes.
- O recurso `package` instala o Tomcat 7.
- A receita`tomcat::service` incluída define o serviço e será abordada depois.

- O recurso `service` inicia o serviço Tomcat.

  Você também pode usar esse recurso para emitir outros comandos, como parar e reiniciar o serviço.

O exemplo a seguir mostra a receita `tomcat::service`.

```
service 'tomcat' do
  service_name "tomcat7"
  supports :restart => true, :reload => false, :status => true
  action :nothing
end
```

Essa receita cria a definição do serviço Tomcat da seguinte forma:

- O nome do recurso, `tomcat`, é usado por outras receitas para referenciar o serviço.

  Por exemplo, `default.rb` referencia `tomcat` para iniciar o serviço.
- O recurso `service_name` especifica o nome do serviço.

  Quando você listar os serviços na instância, o serviço Tomcat será chamado de tomcat7.
- `supports` especifica como o Chef gerencia os comandos `restart`, `reloade` `status` do serviço.
  - `true` indica que o Chef pode usar o script init ou outro provedor de serviços para executar o comando.
  - `false` indica que o Chef deve tentar executar o comando por outros meios.

Observe que `action` é definido como `:nothing`, que leva o recurso a não realizar ação alguma. O recurso de serviço não dá suporte a ações como `start` e `restart`. No entanto, esse livro de receitas segue uma prática padrão de como usar uma definição de serviço que não realiza ação alguma e iniciar ou reiniciar o serviço em outro lugar. Cada receita que inicia ou reinicia um serviço deve primeiramente defini-la. Assim, a abordagem mais simples é colocar a definição de serviço em uma receita à parte e incluí-la em outras receitas, conforme necessário.

> **Note**
>
> Por uma questão de simplicidade, a receita padrão deste exemplo usa um recurso `service` para iniciar o serviço após a execução da definição do serviço. Uma implementação de

produção normalmente inicia ou reinicia um serviço usando `notifies`, conforme será debatido depois.

Para executar a receita

1. Crie um arquivo `default.rb` que contenha o exemplo de receita padrão e o salve em `recipes`.
2. Crie um arquivo `service.rb` que contenha o exemplo de definição do serviço e o salve em `recipes`.
3. Execute `kitchen converge`, faça logon na instância e execute o comando a seguir para verificar se o serviço está em execução.

```
sudo service tomcat7 status
```

> **Note**
>
> Se estivesse executando `service.rb` separadamente de `default.rb`, você teria que editar `.kitchen.yml` para adicionar `tomcat::service` à lista de execuções. No entanto, quando você incluir uma receita, o código será incorporado à receita pai antes da receita ser executada. Por isso, `service.rb` é basicamente uma parte de `default.rb` e não exige uma entrada da lista de execuções à parte.

Uso de notifies para iniciar ou reiniciar um serviço

As implementações de produção normalmente não usam `service` para iniciar ou reiniciar um serviço. Em vez disso, elas adicionam `notifies` a qualquer um dos vários recursos. Por exemplo, caso queira reiniciar o serviço depois de modificar um arquivo de configuração, você inclui `notifies` no recurso `template` associado. O uso de `notifies` tem as vantagens a seguir em relação ao uso de um recurso `service` para reiniciar explicitamente o serviço.

- O elemento `notifies` só reiniciará o serviço se o arquivo de configuração associado tiver sido alterado. Assim, não há risco de causar uma reinicialização desnecessária do serviço.
- O Chef reinicia o serviço no máximo uma vez ao final de cada execução, independentemente de quantos `notifies` a execução contenha.

Por exemplo, a execução do Chef pode incluir vários recursos do modelo, e cada um deles modifica um arquivo de configuração diferente e exige a reinicialização de um serviço caso o arquivo tenha sido alterado. No entanto, você normalmente só deseja reiniciar o serviço uma vez, ao final da execução do Chef. Do contrário, você pode tentar reiniciar um serviço que ainda não esteja totalmente operacional após uma reinicialização anterior, o que pode causar erros.

Este exemplo modifica `tomcat::default` para incluir um recurso `template` que usa `notifies` para reiniciar o serviço. Um exemplo realista usaria um recurso de modelo que cria uma versão personalizada de um dos arquivos de configuração do Tomcat, mas eles acabam sendo longos e complexos. Para simplificar, o exemplo usa apenas o recurso de modelo de Criação de um arquivo usando um modelo. Ele não tem a ver com o Tomcat, mas oferece uma maneira simples de mostrar como usar `notifies`. Para obter um exemplo sobre como usar modelos para criar arquivos de configuração do Tomcat, consulte Receitas de instalação.

Para configurar o livro de receitas

1. Adicione um subdiretório `templates` e `tomcat` e um subdiretório `default` a `templates`.
2. Copie o modelo `example_data.json.erb` do livro de receitas `createfile` para o diretório `templates/default`.
3. Adicione um subdiretório `attributes` a `tomcat`.
4. Copie o arquivo de atributo `default.rb` do livro de receitas `createfile` para o diretório `attributes`.

A receita a seguir usa `notifies` para reiniciar o serviço Tomcat.

```
execute "install_updates" do
  command "apt-get update"
end

package "tomcat7" do
    action :install
end

include_recipe 'tomcat::service'

service 'tomcat' do
```

```
  action :enable
end

directory "/srv/www/shared" do
  mode 0755
  owner 'root'
  group 'root'
  recursive true
  action :create
end

template "/srv/www/shared/example_data.json" do
  source "example_data.json.erb"
  mode 0644
  variables(
    :a_boolean_var => true,
    :a_string_var => "some string"
  )
  only_if {node['createfile']['install_file']}
  notifies :restart, resources(:service => 'tomcat')
end
```

O exemplo mescla a receita de Criação de um arquivo usando um modelo à receita da seção anterior, com duas alterações significativas:

- O recurso `service` ainda está lá, mas agora atende a uma finalidade um pouco diferente.

  A ação `:enable` habilita o serviço Tomcat na inicialização.
- O recurso de modelo já inclui `notifies`, que reinicia o serviço Tomcat caso `example_data.json` tenha sido alterado.

  Isso garante que o serviço seja iniciado quando Tomcat é instalado e reiniciado inicialmente após cada alteração feita na configuração.

Para executar a receita

1. Execute `kitchen destroy` para começar com uma instância limpa.
2. Substitua o código em `default.rb` pelo exemplo anterior.
3. Execute `kitchen converge`, faça logon na instância e verifique se o serviço está em execução.

> **Note**
>
> Caso queira reiniciar um serviço, mas a receita não inclua um recurso, como `template` que dê suporte a `notifies`, você pode usar um recurso `execute` fictício em seu lugar. Por exemplo
>
> ```
> execute 'trigger tomcat service restart' do
>  command 'bin/true'
>  notifies :restart, resources(:service => 'tomcat')
> end
> ```
>
> O recurso `execute` deve ter um atributo `command`, mesmo que você só esteja usando o recurso como uma maneira de executar `notifies`. Esse exemplo contorna esse requisito executando `/bin/true`, que é um comando do shell que simplesmente retorna um código de êxito.

Exemplo 9: Usando EC2 instâncias da Amazon

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Até agora, você está executando instâncias localmente em VirtualBox. Embora isso seja rápido e fácil, você eventualmente desejará testar suas receitas em uma EC2 instância da Amazon. Em particular, se você quiser executar receitas no Amazon Linux, elas estarão disponíveis somente na Amazon EC2. Você pode usar um sistema similar, como o CentOS, para implementação e testes preliminares, mas a única maneira de testar totalmente suas receitas no Amazon Linux é com uma instância da Amazon EC2 .

Este tópico mostra como executar receitas em uma EC2 instância da Amazon. Você usará o Test Kitchen e o Vagrant da mesma maneira como fez nas seções anteriores, com duas diferenças:

- O driver é `kitchen-ec2`, em vez do Vagrant.
- O `.kitchen.yml` arquivo do livro de receitas deve ser configurado com as informações necessárias para iniciar a EC2 instância da Amazon.

> **Note**
>
> Uma abordagem alternativa é usar o plug-in do Vagrant `vagrant-aws`. Para obter mais informações, consulte Vagrant AWS Provider.

Você precisará das credenciais da AWS para criar uma EC2 instância da Amazon. Caso não tenha uma conta da AWS, você pode obter uma da maneira a seguir.

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra a https://portal.aws.amazon.com/billing/inscrição.
2. Siga as instruções online.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, você pode visualizar a atividade atual da sua conta e gerenciar sua conta acessando https://aws.amazon.com/e escolhendo Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS AWS IAM Identity Center, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login AWS Management Consolecomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, insira a senha.

   Para obter ajuda ao fazer login usando o usuário-raiz, consulte Fazer login como usuário-raiz no Guia do usuário do Início de Sessão da AWS .

2. Habilite a autenticação multifator (MFA) para o usuário-raiz.

   Para obter instruções, consulte Habilitar um dispositivo de MFA virtual para seu usuário Conta da AWS raiz (console) no Guia do usuário do IAM.

Criar um usuário com acesso administrativo

1. Habilita o Centro de Identidade do IAM.

   Para obter instruções, consulte Habilitar o AWS IAM Identity Center no Guia do usuário do AWS IAM Identity Center .

2. No Centro de Identidade do IAM, conceda o acesso administrativo a um usuário.

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM no Guia AWS IAM Identity Center do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com o seu usuário do Centro de Identidade do IAM, use o URL de login enviado ao seu endereço de e-mail quando o usuário do Centro de Identidade do IAM foi criado.

  Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

Para obter instruções, consulte Criar um conjunto de permissões no Guia do usuário do AWS IAM Identity Center .

2. Atribua usuários a um grupo e, em seguida, atribua o acesso de autenticação única ao grupo.

   Para obter instruções, consulte Adicionar grupos no Guia do usuário do AWS IAM Identity Center .

Você deve criar um usuário do IAM com permissões para acessar a Amazon EC2 e salvar o acesso e as chaves secretas do usuário em um local seguro na sua estação de trabalho. O Test Kitchen usará essas credenciais para criar a instância. A maneira preferida de fornecer credenciais para o Test Kitchen é atribuir as chaves às variáveis de ambiente a seguir na estação de trabalho.

> **Warning**
>
> Os usuários do IAM têm credenciais de longo prazo, o que representa um risco de segurança. Para ajudar a reduzir esse risco, recomendamos que você forneça a esses usuários somente as permissões necessárias para realizar a tarefa e que você os remova quando não forem mais necessários.

- AWS_ACCESS_KEY — a chave de acesso do usuário, que será semelhante aAKIAIOSFODNN7EXAMPLE.
- AWS_SECRET_KEY — a chave secreta do seu usuário, que será semelhante awJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY.

Essa abordagem reduz as chances de comprometer acidentalmente a conta, por exemplo, fazendo upload de um projeto com as credenciais em um repositório público.

Para configurar o livro de receitas

1. Para usar o driver `kitchen-ec2`, você deve ter o pacote `ruby-dev` instalado no sistema. O comando de exemplo a seguir mostra como usar `aptitude` para instalar o pacote em um sistema Ubuntu.

   ```
   sudo aptitude install ruby1.9.1-dev
   ```

2. O driver `kitchen-ec2` é um gem, que você pode instalar da seguinte forma:

```
gem install kitchen-ec2
```

Dependendo de sua estação de trabalho, esse comando pode exigir `sudo`. Você também pode usar um gerenciador de ambiente Ruby, como o RVM. Este procedimento foi testado com a versão 0.8.0 do driver `kitchen-ec2`, mas há versões mais novas. Para instalar uma specific version, execute `gem install kitchen-ec2 -v <version number>`.

3. Você deve especificar um par de chaves Amazon EC2 SSH que o Test Kitchen possa usar para se conectar à instância. Se você não tiver um par de EC2 chaves da Amazon, consulte Amazon EC2 Key Pairs para obter informações sobre como criar um. Observe que o par de chaves deve pertencer a mesma região da AWS que a instância. O exemplo usa Oeste dos EUA (N. da Califórnia).

   Depois que você tiver selecionado um par de chaves, crie um subdiretório `opsworks_cookbooks` chamado `ec2_keys` e copie o arquivo de chave privada do par de chaves (`.pem`) para esse diretório. Colocar a chave privada em `ec2_keys` é apenas uma comodidade que simplifica um pouco o código; ela pode estar em qualquer lugar do sistema.
4. Crie um subdiretório de `opsworks_cookbooks` chamado `createdir-ec2` e navegue até ele.
5. Adicione um arquivo `metadata.rb` a `createdir-ec2`, com o seguinte conteúdo.

```
name "createdir-ec2"
version "0.1.0"
```

6. Initialize o Test Kitchen, conforme descrito em Exemplo 1: Instalação de pacotes. A seção a seguir descreve como configurar `.kitchen.yml`, o que é significativamente mais complicado para as EC2 instâncias da Amazon.
7. Adicione um subdiretório `recipes` a `createdir-ec2`.

Configurando .kitchen.yml para a Amazon EC2

Você configura `.kitchen.yml` com as informações que o `kitchen-ec2` driver precisa para iniciar uma EC2 instância da Amazon configurada adequadamente. Este é um exemplo de um arquivo `.kitchen.yml` para uma instância do Amazon Linux na região Oeste dos EUA (N. da Califórnia).

```
driver:
  name: ec2
```

```
  aws_ssh_key_id: US-East1
  region: us-west-1
  availability_zone: us-west-1c
  require_chef_omnibus: true
  security_group_ids: sg........
  subnet_id: subnet-.........
  associate_public_ip: true
  interface: dns

provisioner:
  name: chef_solo

platforms:
  -name: amazon
  driver:
    image_id: ami-xxxxxxxx
  transport:
    username: ec2-user
    ssh_key: ../ec2_keys/US-East1.pem

suites:
  - name: default
    run_list:
      - recipe[createdir-ec2::default]
    attributes:
```

Você pode usar as configurações padrão para as seções `provisioner` e `suites`, mas deve modificar as configurações padrão `driver` e `platforms`. Este exemplo usa uma lista mínima de configurações e aceita os valores padrão para o restante. Para obter uma lista completa de `kitchen-ec2` configurações, consulte Kitchen: :Ec2: A Test Kitchen Driver for Amazon. EC2

O exemplo define os atributos `driver` a seguir. Ele pressupõe que você tenha atribuído o acesso do usuário e as chaves secretas a variáveis do ambiente padrão, conforme abordado anteriormente. O driver usa essas chaves por padrão. Do contrário, você deve especificar explicitamente as chaves adicionando `aws_access_key_id` e `aws_secret_access_key` aos atributos `driver` e definir os valores de chave apropriados.

nome

(Obrigatório) Este atributo deve ser definido como `ec2`.

aws_ssh_key_id

(Obrigatório) O nome do par de chaves Amazon EC2 SSH, que é nomeado `US-East1` neste exemplo.

transport.ssh_key

(Obrigatório) O arquivo de chave privada (`.pem`) da chave especificada por você para `aws_ssh_key_id`. Para este exemplo, o arquivo será chamado `US-East1.pem` e ficará no diretório `../opsworks/ec2_keys`.

região

(Obrigatório) A região da AWS da instância. O exemplo usa Oeste dos EUA (N. da Califórnia), que é representado por `us-west-1`).

availability_zone

(Opcional) A zona de disponibilidade da instância. Caso você omita essa configuração, o Test Kitchen usa uma zona de disponibilidade padrão para a região especificada, `us-west-1b` para Oeste dos EUA (N. da Califórnia). No entanto, a região padrão talvez não esteja disponível para a conta. Neste caso, você deve especificar explicitamente uma zona de disponibilidade. Quando isso acontece, a conta usada para preparar os exemplos não dá suporte a `us-west-1b`. Assim, o exemplo especifica explicitamente `us-west-1c`.

require_chef_omnibus

Quando definida como `true`, essa configuração garante que o instalador omnibus seja usado para instalar `chef-client` em todas as instâncias da plataforma.

security_group_ids

(Opcional) Uma lista de grupos de segurança IDs a serem aplicados à instância. Essa configuração aplica o grupo de segurança `default` à instância. Certifique-se de que as regras de entrada do grupo de segurança permitam conexões SSH de entrada, ou o Test Kitchen não poderá se comunicar com a instância. Caso use o security grupo de segurança`default`, você talvez possa precisar editá-lo de acordo. Para obter mais informações, consulte Amazon EC2 Security Groups.

subnet_id

A ID da sub-rede de destino da instância, caso aplicável.

associate_public_ip

Você pode fazer com que a Amazon EC2 associe um endereço IP público à instância se quiser acessar a instância pela Internet.

interface

O tipo de configuração do nome de host usado por você para acessar a instância. Os valores válidos são `dns`, `public`, `private` ou `private_dns`. Caso você não especifique um valor para esse atributo, `kitchen-ec2` define a configuração do nome de host na ordem a seguir. Caso você omita este atributo, o tipo de configuração não é definido.

1. Nome DNS
2. Endereço IP público
3. Endereço IP privado
4. Nome DNS privado

> **Important**
>
> Em vez de usar as credenciais da conta para o acesso e as chaves secretas, você deve criar um usuário e fornecer essas credenciais ao Test Kitchen. Para obter mais informações, consulte Práticas recomendadas de gerenciamento de chaves de acesso da AWS.
> Tenha cuidado para não colocá-lo `.kitchen.yml` em um local acessível ao público, como enviá-lo para um repositório público GitHub ou do Bitbucket. Isso expõe as credenciais e pode comprometer a segurança da conta.

O driver `kitchen-ec2` dá suporte padrão às seguintes plataformas:

- ubuntu-10.04
- ubuntu-12.04
- ubuntu-12.10
- ubuntu-13.04
- ubuntu-13.10
- ubuntu-14.04
- centos-6.4
- debian-7.1.0

- windows-2012r2
- windows-2008r2

Caso você queira usar uma ou mais dessas plataformas, adicione os nomes de plataforma apropriados a `platforms`. O driver `kitchen-ec2` seleciona automaticamente uma AMI apropriada e gera um nome do usuário SSH. Você pode usar outras plataformas; este exemplo usa o Amazon Linux, mas é necessário especificar explicitamente os atributos `platforms` a seguir.

nome

O nome da plataforma. Este exemplo usa o Amazon Linux, de maneira que `name` seja definido como `amazon`.

driver

Os atributos `driver`, que incluem o seguinte:

- `image_id`: a AMI da plataforma, que deve pertencer à região especificada. O exemplo usa `ami-ed8e9284`, uma AMI do Amazon Linux da região Oeste dos EUA (N. da Califórnia).
- `transport.username`: o nome de usuário de SSH que o Test Kitchen usará para se comunicar com a instância.

  Use `ec2-user` para o Amazon Linux. Outros AMIs podem ter nomes de usuário diferentes.

Substitua o código em `.kitchen.yml` pelo exemplo e atribua valores apropriados a atributos específicos da conta, como `aws_access_key_id`.

Execução da receita

Este exemplo usa a receita de Iteração.

Para executar a receita

1. Crie um arquivo chamado `default.rb` com o código a seguir e o salve na pasta `recipes` do livro de receitas.

```
directory "/srv/www/shared" do
  mode 0755
  owner 'root'
```

```
  group 'root'
  recursive true
  action :create
end
```

2. Execute `kitchen converge` para executar a receita. Observe que esse comando levará mais tempo para ser concluído do que os exemplos anteriores devido ao tempo necessário para iniciar e inicializar uma EC2 instância da Amazon.
3. Acesse o EC2console da Amazon, selecione a região Oeste dos EUA (Norte da Califórnia) e clique em Instâncias no painel de navegação. Você verá a instância recém-criada na lista.
4. Execute `kitchen login` para fazer login na instância, assim como você tem feito com as instâncias em execução VirtualBox. Você verá os diretórios recém-criados em `/srv`. Você também pode usar o cliente SSH favorito para se conectar à instância.

Próximas etapas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este capítulo mostrou o passo a passo dos conceitos básicos para implementar os livros de receitas do Chef, mas há muito mais:

- Os exemplos mostraram como utilizar alguns dos recursos mais usados, mas há muitos outros.

  Para os recursos que foram abordados, os exemplos usaram apenas alguns dos atributos e das ações disponíveis. Para obter uma referência completa, consulte About Resources and Providers.
- Os exemplos usaram apenas os elementos do livro de receitas principais: `recipes`, `attributes`, `filese templates`.

  Os livros de receitas também podem incluir uma grande variedade de outros elementos, como `libraries`, `definitions` e `specs`. Para obter mais informações, consulte a Chef documentation.

- Os exemplos usaram o Test Kitchen somente como uma maneira prática de iniciar instâncias, executar receitas e fazer logon em instâncias.

  O Test Kitchen é basicamente uma plataforma de teste que você pode usar para executar uma grande variedade de testes nas receitas. Caso ainda não tenha feito isso, passe pelo restante do Test Kitchen walkthrough, que apresenta você aos recursos de teste.
- Implementando livros de receitas para pilhas AWS OpsWorksfornece alguns exemplos mais avançados e mostra como implementar livros de receitas para AWS OpsWorks Stacks.

Implementando livros de receitas para pilhas AWS OpsWorks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Conceitos básicos do livro de receitas lhe apresentou aos livros de receitas e às receitas. Os exemplos nessa seção foram simples por design e funcionarão em qualquer instância que ofereça suporte ao Chef, incluindo instâncias do AWS OpsWorks Stacks. Para implementar livros de receitas mais sofisticados para AWS OpsWorks Stacks, você normalmente precisa aproveitar ao máximo o ambiente AWS OpsWorks Stacks, que difere do Chef padrão de várias maneiras.

Este tópico descreve os princípios básicos da implementação de receitas para instâncias do AWS OpsWorks Stacks.

> **Note**
>
> Se você não estiver familiarizado com a implementação dos livros de receitas, inicie com Conceitos básicos do livro de receitas.

Tópicos

- Executando uma receita em uma instância AWS OpsWorks do Stacks Linux

- Executar uma receita na instância Windows
- Executando um PowerShell script do Windows
- Simular os atributos de implantação e configuração de pilha no Vagrant
- Usar valores de atributos de implantação e configuração de pilha
- Usar um livro de receitas externo em uma instância do Linux: Berkshelf
- Usar o SDK for Ruby: fazer download de arquivos do Amazon S3
- Instalar o softwares do Windows
- Sobrepor atributos integrados
- Substituir modelos integrados

Executando uma receita em uma instância AWS OpsWorks do Stacks Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O Test Kitchen e o Vagrant fornecem uma maneira simples e eficiente de implementar livros de receitas, mas para verificar se as receitas de um livro de receitas serão executadas corretamente na produção, você deve executá-las em uma instância do Stacks. AWS OpsWorks Este tópico descreve como instalar um livro de receitas personalizado em uma instância Linux do AWS OpsWorks Stacks e executar uma receita simples. O tópico também fornece algumas dicas para corrigir os bugs de receita eficientemente.

Para obter uma descrição de como executar receitas em instâncias do Windows, consulte Executar uma receita na instância Windows.

Tópicos

- Criar e executar a receita
- Executar a receita automaticamente

- Solucionar problemas e corrigir receitas

Criar e executar a receita

Primeiro, é necessário criar uma pilha. As instruções a seguir resumem o procedimento da criação de uma pilha neste exemplo. Para obter mais informações, consulte Criar uma nova pilha.

Para criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e clique em Add Stack (Adicionar pilha).
2. Especifique as seguintes configurações, aceite os padrões para as outras configurações e clique em Add Stack.

   - Nome — OpsTest
   - Chave SSH padrão — um par de EC2 chaves da Amazon

   Se você precisar criar um par de EC2 chaves da Amazon, consulte Amazon EC2 Key Pairs. Observe que o par de chaves deve pertencer a mesma região da AWS que a instância. O exemplo usa a região padrão, Oeste dos EUA (Oregon).
3. Clique em Add a layer e adicione uma camada personalizada à pilha com as seguintes configurações.

   - Nome — OpsTest
   - Nome curto: opstest

   Qualquer tipo de camada funcionará nas pilhas do Linux, mas o exemplo não exige nenhum dos pacotes instalados por outros tipos de camada, portanto, uma camada personalizada é a abordagem mais simples.
4. Adicione uma instância ininterrupta com as configurações padrão à camada e inicie-a.

Enquanto a instância está iniciando, o que geralmente demora vários minutos, você pode criar o livro de receitas. Este exemplo usa uma versão modificada de receita do Lógica condicional, que cria um diretório de dados cujo nome depende da plataforma.

Para configurar o livro de receitas

1. Crie um diretório em `opsworks_cookbooks` chamado `opstest` e navegue até ele.

2. Crie um arquivo `metadata.rb` com o seguinte conteúdo e salve-o em `opstest`.

```
name "opstest"
version "0.1.0"
```

3. Criar um diretório `recipes` em `opstest`.
4. Crie um arquivo `default.rb` com a seguinte receita e salve-o no diretório `recipes`.

```
Chef::Log.info("******Creating a data directory.******")

data_dir = value_for_platform(
  "centos" => { "default" => "/srv/www/shared" },
  "ubuntu" => { "default" => "/srv/www/data" },
  "default" => "/srv/www/config"
)

directory data_dir do
  mode 0755
  owner 'root'
  group 'root'
  recursive true
  action :create
end
```

Observe que a receita registra uma mensagem, designada `Chef::Log.info`. Você não está usando o Test Kitchen para este exemplo, então o `log` método não é muito útil. `Chef::Log.info`coloca a mensagem no log do Chef, que você pode ler após a conclusão da execução do Chef. AWS OpsWorks O Stacks fornece uma maneira fácil de visualizar esses registros, conforme descrito posteriormente.

> **Note**
>
> Os logs do Chef normalmente contêm uma grande quantidade de informações relativamente rotineiras e desinteressantes. Os caracteres '*' antes e depois da mensagem de texto facilitam sua localização.

5. Crie um arquivo `.zip` dos `opsworks_cookbooks`. Para instalar seu livro de receitas em uma instância do AWS OpsWorks Stacks, você deve armazená-lo em um repositório e fornecer

ao AWS OpsWorks Stacks as informações necessárias para baixar o livro de receitas para a instância. Você pode armazenar os livros de receitas em diversos tipos de repositório que possuem suporte. Este exemplo armazena um arquivo compactado contendo os livros de receitas em um bucket do Amazon S3. Para obter mais informações sobre repositórios de livros de receitas, consulte Repositórios de livro de receitas.

> **Note**
>
> Para simplificar, este exemplo apenas arquiva todo o diretório `opsworks_cookbooks`. No entanto, isso significa que o AWS OpsWorks Stacks baixará todos os livros de receitas `opsworks_cookbooks` para a instância, mesmo que você use apenas um deles. Para instalar somente o livro de receitas como exemplo, crie outro diretório pai e mova `opstest` para esse diretório. Em seguida, crie um arquivo `.zip` do diretório pai e use-o em vez de `opsworks_cookbooks.zip`.
> O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

6. Faça upload do arquivo em um bucket do Amazon S3, torne o arquivo público e registre o URL do arquivo.

Agora você pode instalar o livro de receitas e executar a receita.

Para executar a receita

1. Edite a pilha para habilitar livros de receitas personalizados, e especifique as seguintes configurações.

   - Tipo de repositório: Arquivamento do S3
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente

   Use os valores padrão nas outras configurações e clique em Save para atualizar a configuração da pilha.

2. Execute o comando de pilha Update Custom Cookbooks, que instala a versão atual de seus livros de receitas personalizados nas instâncias da pilha. Se uma versão mais antiga de seus livros de receitas estiver presente, este comando a sobrescreverá.

3. Execute a receita acionando o comando de stack Execute Recipes (Executar receitas) com a opção Recipes to execute (Receitas para executar) definida como **`opstest::default`**. Este comando inicia uma execução do Chef, com uma lista de execução que consiste em `opstest::default`.

Após a receita ser executada com sucesso, verifique isso.

Para verificar opstest

1. A primeira etapa é examinar o log do Chef. Clique em show na coluna Log da instância opstest1 para exibir o log. Role para baixo e veja sua mensagem de log próxima à parte inferior.

```
...
[2014-07-31T17:01:45+00:00] INFO: Storing updated cookbooks/opsworks_cleanup/
attributes/customize.rb in the cache.
[2014-07-31T17:01:45+00:00] INFO: Storing updated cookbooks/opsworks_cleanup/
metadata.rb in the cache.
[2014-07-31T17:01:46+00:00] INFO: ******Creating a data directory.******
[2014-07-31T17:01:46+00:00] INFO: Processing template[/etc/hosts] action create
 (opsworks_stack_state_sync::hosts line 3)
...
```

2. Utilize o SSH para fazer login na instância e listar o conteúdo de `/srv/www/`.

Se você seguiu todas as etapas, verá `/srv/www/config` em vez do diretório `/srv/www/shared` que estava esperando. A seção a seguir fornece algumas diretrizes para corrigir rapidamente tais bugs.

Executar a receita automaticamente

O comando Execute Recipes é uma maneira conveniente de testar receitas personalizadas, por isso ele é usado na maioria dos exemplos. No entanto, na prática, você normalmente executa receitas em pontos padrão do ciclo de vida de uma instância, como depois que a instância termina de inicializar ou quando você implanta um aplicativo. AWS OpsWorks O Stacks simplifica a execução de receitas em sua instância ao oferecer suporte a um conjunto de eventos de ciclo de vida para cada camada: configuração, configuração, implantação, desimplantação e desativação. Você pode fazer com que o AWS OpsWorks Stacks execute uma receita automaticamente nas instâncias de uma camada atribuindo a receita ao evento de ciclo de vida apropriado.

Normalmente, os diretórios são criados, assim que uma instância termina a inicialização, que corresponde à instalação do evento. Os procedimentos a seguir mostram como executar a receita de exemplo na configuração usando a mesma pilha criada anteriormente no exemplo. O mesmo procedimento pode ser usado para os outros eventos.

Para executar automaticamente uma receita na instalação

1. Escolha Camadas no painel de navegação e, em seguida, escolha o ícone de lápis ao lado do link Receitas da OpsTest camada.
2. Adicione **`opstest::default`** às receitas Setup (Instalação) da camada, clique em + para adicioná-lo à camada e selecione Save (Salvar) para salvar a configuração.
3. Escolha Instances, adicione outra instância à layer e a inicie.

   A instância deve ser designada `opstest2`. Depois que a inicialização for concluída, as AWS OpsWorks pilhas serão executadas. `opstest::default`
4. Quando a instância `opstest2` estiver online, verifique se `/srv/www/shared` está presente.

> **Note**
>
> Se você atribuiu as receitas para configurar ou implantar eventos, também é possível executá-las manualmente usando um comando de pilha (instalar e configurar) ou um comando de implante (Implantar) para acionar o evento. Observe que se você tiver várias receitas atribuídas a um evento, esses comandos executam todas elas.

Solucionar problemas e corrigir receitas

Se você não estiver recebendo os resultados esperados ou se suas receitas ainda não estão sendo executadas com sucesso, a solução de problemas começa normalmente, examinando o log do Chef. Ele contém uma descrição detalhada da execução e inclui quaisquer mensagens de log em linha de suas receitas. Os logs são particularmente úteis se sua receita simplesmente tiver falhado. Quando isso acontece, o Chef registra o erro, incluindo um rastreamento de pilha.

Se a receita tiver sido executada com sucesso como neste exemplo, o log do Chef não costuma ajuda muito. Neste caso, você pode descobrir o problema, examinando mais de perto a receita, especialmente as primeiras linhas:

```
Chef::Log.info("******Creating a data directory.******")

data_dir = value_for_platform(
  "centos" => { "default" => "/srv/www/shared" },
  "ubuntu" => { "default" => "/srv/www/data" },
  "default" => "/srv/www/config"
)
...
```

CentOS é um substituto razoável para o Amazon Linux em testes de receitas em Vagrant, mas agora você está executando em uma instância do Amazon Linux. O valor da plataforma para o Amazon Linux é `amazon`, que não está incluído na chamada `value_for_platform`. Portanto, a receita cria `/srv/www/config` por padrão. Para obter mais informações sobre solução de problemas, consulte Guia de depuração e solução de problemas.

Agora que você identificou o problema, atualize a receita e verifique a correção. É possível voltar aos arquivos originais, atualizar `default.rb`, carregar um novo arquivo no Amazon S3 etc. No entanto, esse processo pode ser um pouco tedioso e demorado. As instruções a seguir mostram uma abordagem muito mais rápida, especialmente útil para os bugs de receita simples, como aquele do exemplo: edite a receita na instância.

Para editar uma receita em uma instância

1. Use o SSH para fazer login na instância e, em seguida, execute `sudo su` para elevar seus privilégios. É necessário ter privilégios de root para acessar os diretórios do livro de receitas.
2. AWS OpsWorks O Stacks armazena seu livro de receitas`/opt/aws/opsworks/current/site-cookbooks`, então navegue até. `/opt/aws/opsworks/current/site-cookbooks/opstest/recipes`

   > **Note**
   >
   > AWS OpsWorks O Stacks também armazena uma cópia de seus livros de receitas em. `/opt/aws/opsworks/current/merged-cookbooks` Não edite aquele livro de receitas. Quando você executa a receita, o AWS OpsWorks Stacks copia o livro de receitas de `.../site-cookbooks` para`.../merged-cookbooks`, portanto, todas as alterações feitas `.../merged-cookbooks` serão sobrescritas.

3. Use um editor de texto na instância para editar `default.rb`, e substitua `centos` com `amazon`. Agora, as configurações devem ser semelhantes a essas.

```
Chef::Log.info("******Creating a data directory.******")

data_dir = value_for_platform(
  "amazon" => { "default" => "/srv/www/shared" },
  "ubuntu" => { "default" => "/srv/www/data" },
  "default" => "/srv/www/config"
)
...
```

Para verificar a correção, execute a receita executando o comando de pilha Execute Recipe novamente. Agora a instância deve ter um diretório `/srv/www/shared`. Se você precisar fazer outras alterações em uma receita, é possível executar Execute Recipe sempre que desejar; não é necessário interromper e reiniciar a instância cada vez que você executar o comando. Quando você estiver satisfeito com a receita funcionando corretamente, não se esqueça de atualizar o código na fonte de seu livro de receitas.

> **Note**
>
> Se você atribuiu sua receita a um evento de ciclo de vida para que o AWS OpsWorks Stacks a execute automaticamente, você sempre pode usar Execute Recipe para executar novamente a receita. Você também pode executar novamente a receita quantas vezes quiser sem reiniciar a instância usando o console AWS OpsWorks Stacks para acionar manualmente o evento apropriado. No entanto, esta abordagem executa todas as receitas do evento. Lembrete:
>
> - Use um comando de pilha para acionar a instalação ou configuração de eventos.
> - Use um comando de implante para acionar eventos de implante ou desimplante.

Executar uma receita na instância Windows

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico é basicamente uma versão abreviada do Executar uma receita em uma instância Linux, que mostra como executar uma receita em um stack do Windows. Recomendamos que você passe por Executar uma receita em uma instância Linux primeiro, porque isso fornece uma discussão mais detalhada, a qual é relevante principalmente para ambos tipos de sistemas operacionais.

Para obter uma descrição de como executar receitas em instâncias do AWS OpsWorks Stacks Linux, consulteExecutar uma receita em uma instância Linux.

Tópicos

- Habilitar o acesso ao RDP
- Criar e executar a receita
- Executar a receita automaticamente

Habilitar o acesso ao RDP

Antes de começar, se você ainda não tiver feito isso, configure um grupo de segurança com uma regra de entrada, a qual permita que suas instâncias tenham acesso ao RDP de entrada. Você precisará desse grupo quando criar a pilha.

Quando você cria a primeira pilha em uma região, o AWS OpsWorks Stacks cria um conjunto de grupos de segurança. Eles incluem um chamado algo como`AWS-OpsWorks-RDP-Server`, que o AWS OpsWorks Stacks anexa a todas as instâncias do Windows para permitir acesso ao RDP. No entanto, por padrão, esse grupo de segurança não tem regras. Assim, você deve adicionar uma regra de entrada para permitir o acesso RDP para suas instâncias.

Para permitir o acesso RDP

1. Abra o EC2console da Amazon, configure-o para a região da pilha e escolha Security Groups no painel de navegação.
2. Escolha AWS- OpsWorks -RDP-Server, escolha a guia Entrada e escolha Editar.
3. Adicione uma regra às seguintes configurações:
   - Tipo: RDP
   - Origem: os endereços IP de origem permitidos.

Normalmente, você permite solicitações RDP de entrada de seu endereço IP ou de uma faixa especificada de endereços IP (normalmente a faixa de endereços IP de sua empresa).

> **Note**
>
> Conforme descrito posteriormente, edite as permissões de usuário para autorizar o acesso de usuários frequentes ao RDP.

Para obter mais informações, consulte Login com RDP.

Criar e executar a receita

As instruções a seguir resumem o procedimento da criação de uma pilha neste exemplo. Para obter mais informações, consulte Criar uma nova pilha.

Criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e selecione Add Stack (Adicionar pilha). Especifique as seguintes configurações, aceite os padrões para as outras configurações e escolha Add Stack.
   - Nome — WindowsRecipeTest
   - Região: Oeste dos EUA (Oregon)

     Este exemplo funcionará em qualquer região, mas recomendamos o uso de Oeste dos EUA (Oregon) para tutoriais.
   - Sistema operacional padrão: Microsoft Windows Server 2012 R2
2. Clique em Add a layer e adicione uma camada personalizada à pilha com as seguintes configurações.
   - Nome — RecipeTest
   - Nome curto: recipetest
3. Adicione uma instância 24 horas por dia, 7 dias por semana, com configurações padrão à RecipeTest camada e inicie-a.

   AWS OpsWorks As pilhas são atribuídas automaticamente `AWS-OpsWorks-RDP-Server` a essa instância, o que permite que usuários autorizados façam login na instância.

4. Escolha Permissions e, em seguida, Edit, SSH/RDP e sudo/admin. Os usuários frequentes precisam dessa autorização além do grupo de segurança `AWS-OpsWorks-RDP-Server` para fazer login na instância.

> **Note**
> É possível também fazer login como administrador, mas isso requer um procedimento diferente. Para obter mais informações, consulte Login com RDP.

Enquanto a instância está iniciando, o que geralmente demora vários minutos, você pode criar o livro de receitas. A receita deste exemplo cria um diretório de dados e é basicamente a receita de Exemplo 3: Criação de diretórios, modificada para o Windows.

> **Note**
> Ao implementar livros de receitas para instâncias do AWS OpsWorks Stacks Windows, você usa uma estrutura de diretórios um pouco diferente da que usa ao implementar livros de receitas para instâncias do AWS OpsWorks Stacks Linux. Para obter mais informações, consulte Repositórios de livro de receitas.

Para configurar o livro de receitas

1. Crie um diretório chamado `windowstest` e navegue até ele.
2. Crie um arquivo `metadata.rb` com o seguinte conteúdo e salve-o em `windowstest`.

```
name "windowstest"
version "0.1.0"
```

3. Criar um diretório `recipes` em `windowstest`.
4. Crie um arquivo `default.rb` com a seguinte receita e salve-o no diretório `recipes`.

```
Chef::Log.info("******Creating a data directory.******")

directory 'C:\data' do
  rights :full_control, 'instance_name\username'
```

```
  inherits false
  action :create
end
```

Substitua *username* pelo nome do usuário.

5. Coloque o livro de receitas em um repositório.

   Para instalar seu livro de receitas em uma instância do AWS OpsWorks Stacks, você deve armazená-lo em um repositório e fornecer ao AWS OpsWorks Stacks as informações necessárias para baixar o livro de receitas para a instância. É possível armazenar livros de receitas do Windows como um arquivo compactado em um bucket do S3 ou em um repositório Git. Este exemplo usa um bucket do S3, portanto, você deve criar um arquivo .zip do diretório `windowstest`. Para obter mais informações sobre repositórios de livros de receitas, consulte Repositórios de livro de receitas.

6. Faça upload do arquivo em um bucket do S3, torne o arquivo público e registre o URL do arquivo. Você também pode usar um arquivo privado, mas um arquivo público é suficiente para este exemplo, além de ser mais fácil de trabalhar com ele.

   O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Agora você pode instalar o livro de receitas e executar a receita.

Para executar a receita

1. Edite a pilha para habilitar livros de receitas personalizados e especifique as seguintes configurações.

   - Tipo de repositório: Arquivamento do S3
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente

   Aceite os valores padrão para as outras configurações e escolha Save para atualizar a configuração da pilha.

2. Execute o comando de pilha Update Custom Cookbooks, que instala a versão atual de seus livros de receitas personalizados nas instâncias da pilha, inclusive em instâncias online. Se uma versão mais antiga de seus livros de receitas estiver presente, este comando a sobrescreverá.

3. Após a execução de Update Custom Cookbooks (Atualizar livros de receitas personalizados), execute a receita executando o comando de pilhaExecute Recipes (Executar receitas) com a opção Recipes to execute (Receitas para executar) definida como **`windowstest::default`**. Esse comando inicia uma execução do Chef com uma lista de execução que consiste na sua receita.

Após a receita ser executada com sucesso, verifique isso.

Para verificar windowstest

1. Examine o log do Chef. Escolha show na coluna Log da instância opstest1 para exibir o log. Role para baixo e veja sua mensagem de log próxima à parte inferior.

```
...
[2014-07-31T17:01:45+00:00] INFO: Storing updated cookbooks/opsworks_cleanup/
attributes/customize.rb in the cache.
[2014-07-31T17:01:45+00:00] INFO: Storing updated cookbooks/opsworks_cleanup/
metadata.rb in the cache.
[2014-07-31T17:01:46+00:00] INFO: ******Creating a data directory.******
[2014-07-31T17:01:46+00:00] INFO: Processing template[/etc/hosts] action create
 (opsworks_stack_state_sync::hosts line 3)
...
```

2. Escolha Instances e, em seguida, selecione rdp na coluna Actions da instância e solicite uma senha RDP com um tempo de expiração adequado. Copie o nome DNS, o nome de usuário e a senha. Em seguida, você pode usar essas informações com um cliente RDP, como o cliente do Windows Remote Desktop Connection, para fazer login na instância e verificar se `c:\data` existe. Para obter mais informações, consulte Login com RDP.

> **Note**
>
> Caso sua receita não esteja funcionando corretamente, consulte Solucionar problemas e corrigir receitas para obter dicas de solução de problemas; a maioria delas também se aplica a instâncias do Windows. Se você quiser testar sua correção editando a receita na instância, procure seu livro de receitas no `C:\chef\cookbooks` diretório, onde o AWS OpsWorks Stacks instala livros de receitas personalizados.

Executar a receita automaticamente

O comando Execute Recipes é uma maneira conveniente de testar receitas personalizadas, por isso ele é usado na maioria dos exemplos. No entanto, na prática, você normalmente executa receitas em pontos padrão do ciclo de vida de uma instância, como depois que a instância termina de inicializar ou quando você implanta um aplicativo. AWS OpsWorks O Stacks simplifica a execução de receitas em sua instância ao oferecer suporte a um conjunto de eventos de ciclo de vida para cada camada: configuração, configuração, implantação, desimplantação e desativação. Você pode fazer com que o AWS OpsWorks Stacks execute uma receita automaticamente nas instâncias de uma camada atribuindo a receita ao evento de ciclo de vida apropriado.

Normalmente, os diretórios são criados, assim que uma instância termina a inicialização, que corresponde à instalação do evento. Os procedimentos a seguir mostram como executar a receita de exemplo na configuração usando a mesma pilha criada anteriormente no exemplo. O mesmo procedimento pode ser usado para os outros eventos.

Para executar automaticamente uma receita na instalação

1. Escolha Camadas no painel de navegação e, em seguida, escolha o ícone de lápis ao lado do link Receitas da RecipeTest camada.
2. Adicione **`windowstest::default`** às receitas Setup (Instalação) da camada, escolha + para adicioná-lo à camada e selecione Save (Salvar) para salvar a configuração.
3. Escolha Instances, adicione outra instância à layer e a inicie.

   A instância deve ser designada `recipetest2`. Depois que a inicialização for concluída, as AWS OpsWorks pilhas serão executadas. `windowstest::default`
4. Quando a instância `recipetest2` estiver online, verifique se `c:\data` está presente.

> **Note**
>
> Se você tiver atribuído receitas aos eventos Setup, Configure ou Deploy, também poderá executá-las manualmente usando um comando de pilha (Setup e Configure) ou um comando de implantação (Deploy) para acionar o evento. Observe que se você tiver várias receitas atribuídas a um evento, esses comandos executam todas elas.

Executando um PowerShell script do Windows

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Estes exemplos supõem que você já concluiu o exemplo em Executar uma receita na instância Windows. Caso contrário, conclua esse exemplo primeiro. Especificamente, ele descreve como habilitar o acesso RDP para suas instâncias.

Uma maneira de fazer com que uma receita execute tarefas em uma instância do Windows, especialmente tarefas que não têm um recurso correspondente do Chef, é fazer com que a receita execute um script do Windows. PowerShell Esta seção apresenta o básico, descrevendo como usar um PowerShell script do Windows para instalar um recurso do Windows.

O `powershell_script`recurso executa PowerShell cmdlets do Windows em uma instância. O exemplo a seguir usa um WindowsFeature cmdlet Install- para instalar um visualizador XPS na instância.

As instruções a seguir resumem o procedimento da criação de uma pilha neste exemplo. Para obter mais informações, consulte Criar uma nova pilha.

Criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e selecione Add Stack (Adicionar pilha). Especifique as seguintes configurações, aceite os padrões para as outras configurações e clique em Add Stack.

   - Nome — PowerShellTest
   - Região: Oeste dos EUA (Oregon)

Este exemplo funcionará em qualquer região, mas recomendamos o uso de Oeste dos EUA (Oregon) para tutoriais.

- Sistema operacional padrão: Microsoft Windows Server 2012 R2

2. Clique em Add a layer e adicione uma camada personalizada à pilha com as seguintes configurações.

   - Nome — PowerShell
   - Nome curto: powershell

3. Adicione uma instância 24 horas por dia, 7 dias por semana, com configurações padrão na PowerShell camada e inicie-a.
4. Escolha Permissions e selecione Edit, SSH/RDP e sudo/admin. Você precisa dessa autorização além do grupo de segurança `AWS-OpsWorks-RDP-Server` para fazer login na instância como um usuário frequente.

Enquanto a instância está iniciando, o que geralmente demora vários minutos, você pode criar o livro de receitas. A receita deste exemplo cria um diretório de dados e é basicamente a receita de Exemplo 3: Criação de diretórios, modificada para o Windows.

Para configurar o livro de receitas

1. Crie um diretório chamado `powershell` e navegue até ele.
2. Crie um arquivo `metadata.rb` com o seguinte conteúdo e salve-o em `windowstest`.

```
name "powershell"
version "0.1.0"
```

3. Criar um diretório `recipes` em `powershell`.
4. Crie um arquivo `default.rb` com a seguinte receita e salve-o no diretório `recipes`.

```
Chef::Log.info("******Installing XPS.******")

powershell_script "Install XPS Viewer" do
  code <<-EOH
    Install-WindowsFeature XPS-Viewer
  EOH
```

```
  guard_interpreter :powershell_script
  not_if "(Get-WindowsFeature -Name XPS-Viewer).installed"
end
```

- O recurso `powershell_script` executa um cmdlet para instalar o visualizador XPS.

  Este exemplo executa apenas um cmdlet, mas o bloco `code` pode conter várias linhas de comando.
- O `guard_interpreter` atributo direciona o Chef a usar a versão de 64 bits do Windows PowerShell.
- O atributo de proteção `not_if` garante que o Chef não instale o recurso caso ele já tenha sido instalado.

5. Crie um arquivo `.zip` do diretório `powershell`.
6. Faça upload do arquivo em um bucket do Amazon S3, torne o arquivo público e registre o URL do arquivo. Você também pode usar um arquivo privado, mas um arquivo público é suficiente para este exemplo, além de ser mais fácil de trabalhar com ele.

   O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Agora você pode instalar o livro de receitas e executar a receita.

Para executar a receita

1. Edite a pilha para habilitar livros de receitas personalizados e especifique as seguintes configurações.

   - Tipo de repositório: Arquivamento do S3
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente

   Aceite os valores padrão para as outras configurações e escolha Save para atualizar a configuração da pilha.
2. Execute o comando de pilha Update Custom Cookbooks para instalar a versão atual de seus livros de receitas personalizados nas instâncias da pilha.

3. Após a execução de Update Custom Cookbooks (Atualizar livros de receitas personalizados), execute a receita executando o comando de stack Execute Recipes (Executar receitas) com a opção Recipes to execute (Receitas apra executar) definida como **`powershell::default`**.

> **Note**
>
> Este exemplo usa Execute Recipes por conveniência, mas normalmente você faz com que AWS OpsWorks as pilhas executem suas receitas automaticamente, atribuindo-as ao evento de ciclo de vida apropriado. Você pode executar essas receitas acionando manualmente o evento. Use um comando de stack para acionar eventos Setup e Configure, e um comando de pilha para acionar eventos Deploy e Undeploy.

Após a receita ser executada com sucesso, verifique isso.

Para verificar a receita do powershell

1. Examine o log do Chef. Clique em show na coluna Log da instância powershell1 para exibir o log. Role para baixo e veja sua mensagem de log próxima à parte inferior.

```
...
[2015-04-27T18:12:09+00:00] INFO: Storing updated cookbooks/powershell/metadata.rb
 in the cache.
[2015-04-27T18:12:09+00:00] INFO: ******Installing XPS.******
[2015-04-27T18:12:09+00:00] INFO: Processing powershell_script[Install XPS Viewer]
 action run (powershell::default line 3)
[2015-04-27T18:12:09+00:00] INFO: Processing powershell_script[Guard resource]
 action run (dynamically defined)
[2015-04-27T18:12:42+00:00] INFO: powershell_script[Install XPS Viewer] ran
 successfully
...
```

2. Use RDP para acessar a instância e abra o menu Start. O visualizador XPS deve estar listado com Windows Accessories.

Simular os atributos de implantação e configuração de pilha no Vagrant

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este tópico aplica-se apenas às instâncias do Linux. O Test Kitchen ainda não oferece suporte ao Windows, então você executará todos os exemplos do Windows em instâncias do AWS OpsWorks Stacks.

AWS OpsWorks O Stacks adiciona atributos de configuração e implantação da pilha ao objeto de nó de cada instância em sua pilha para cada evento do ciclo de vida. Esses atributos fornecem um snapshot da configuração da pilha, incluindo a configuração de cada camada e de suas instâncias online, a configuração de cada aplicativo implantado etc. Como esses atributos estão no objeto node, eles podem ser acessados por qualquer receita; a maioria das receitas para instâncias do AWS OpsWorks Stacks usa um ou mais desses atributos.

Uma instância executada em uma caixa Vagrant não é gerenciada pelo AWS OpsWorks Stacks, portanto, seu objeto de nó não inclui nenhuma configuração de pilha e atributos de implantação por padrão. No entanto, você pode adicionar um conjunto de atributos adequados ao ambiente do Test Kitchen. Em seguida, o Test Kitchen adiciona os atributos ao objeto node da instância, e suas receitas podem acessar os atributos da mesma forma que acessariam em uma instância do AWS OpsWorks Stacks.

Este tópico mostra como obter uma cópia de atributos de implantação e configuração de pilha apropriados, instalar os atributos em uma instância e acessá-los.

> **Note**
>
> Se você estiver usando o Test Kitchen para executar testes em suas receitas, fauxhai fornece uma maneira alternativa para simular JSON de configuração e implantação da pilha.

Para configurar o livro de receitas

1. Crie um subdiretório de `opsworks_cookbooks` chamado `printjson` e navegue até ele.
2. Inicialize e configure o Test Kitchen, conforme descrito em Exemplo 1: Instalação de pacotes.
3. Adicione dois subdiretórios a `printjson`: `recipes` e `environments`.

É possível simular atributos de implantação e configuração de pilha adicionando um arquivo de atributos ao seu livro de receitas com as definições apropriadas, mas uma abordagem melhor é usar o ambiente do Test Kitchen. Há duas abordagens básicas:

- Adicione definições de atributos a `.kitchen.yml`.

  Essa abordagem é útil se você tiver apenas alguns atributos. Para obter mais informações, consulte kitchen.yml.
- Definir os atributos em um arquivo de ambiente e fazer referência ao arquivo em `.kitchen.yml`.

  Essa abordagem normalmente é preferível para atributos de implantação e configuração de pilha, porque o arquivo de ambiente já está no formato JSON. Você pode obter uma cópia dos atributos no formato JSON de uma instância adequada do AWS OpsWorks Stacks e simplesmente colá-la. Todos os exemplos usam um arquivo de ambiente.

A maneira mais simples de criar atributos de implantação e configuração de pilhapara seu livro de receitas é criar uma pilha devidamente configurada e copiar os atributos resultantes de uma instância como JSON. Para manter seu arquivo de ambiente do Test Kitchen gerenciável, edite o JSON para ter apenas os atributos de que suas receitas precisam. Os exemplos neste capítulo são baseados na pilha de Conceitos básicos das pilhas Linux do Chef 11, que é uma pilha de servidor de aplicativo PHP simples com um load balancer, servidores de aplicativos PHP e um servidor de banco de dados MySQL.

Para criar um JSON de configuração e implantação de pilha

1. Crie MyStack conforme descrito emConceitos básicos das pilhas Linux do Chef 11, incluindo a implantação do SimplePHPApp. Se preferir, omita a segunda instância PHP App Server solicitada em Etapa 4: Dimensionar MyStack; os exemplos não usam esses atributos.
2. Se você ainda não tiver feito isso, inicie a instância `php-app1` e faça login com SSH.
3. Na janela do terminal, execute o seguinte comando agent cli:

   ```
   sudo opsworks-agent-cli get_json
   ```

   Esse comando imprime os atributos de implantação e configuração de pilha mais recentes da instância na janela do terminal no formato JSON.
4. Copie o JSON em um arquivo `.json` e salve-o em um local conveniente na sua estação de trabalho. Os detalhes dependem de seu cliente SSH. Por exemplo, se você estiver usando PuTTY no Windows, execute o comando `Copy All to Clipboard`, que copia todo o texto na janela do terminal para a área de transferência do Windows. Em seguida, cole o conteúdo em um arquivo `.json` e edite o arquivo para remover texto adicional.
5. Edite o MyStack JSON conforme necessário. Os atributos de implantação e configuração de pilha são numerosos, mas os livros de receitas normalmente usam apenas um pequeno subconjunto deles. Para manter seu arquivo de ambiente gerenciável, edite o JSON para que ele mantenha a estrutura original, mas contenha apenas os atributos que seus livros de receitas efetivamente usam.

   Este exemplo usa uma versão altamente editada do MyStack JSON que inclui apenas dois `['opsworks']['stack']` atributos, `['id]` e. `['name']` Crie uma versão editada do MyStack JSON parecida com a seguinte:

   ```
   {
     "opsworks": {
       "stack": {
         "name": "MyStack",
         "id": "42dfd151-6766-4f1c-9940-ba79e5220b58",
       },
     },
   }
   ```

Para inserir esse JSON no objeto de nó da instância, é necessário adicioná-lo a um ambiente do Test Kitchen.

Para adicionar atributos de implantação e configuração de pilha ao ambiente do Test Kitchen

1. Crie um arquivo de ambiente chamado `test.json` com o seguinte conteúdo e salve-o na pasta `environments` do livro de receitas.

```
{
  "default_attributes": {
    "opsworks" : {
      "stack" : {
        "name" : "MyStack",
        "id" : "42dfd151-6766-4f1c-9940-ba79e5220b58"
      }
    }
  },
  "chef_type" : "environment",
  "json_class" : "Chef::Environment"
}
```

   O arquivo de ambiente tem os seguintes elementos:

   - `default_attributes`: os atributos padrão no formato JSON.

     Esses atributos são adicionados ao objeto de nó com o tipo de atributo `default`, que é o tipo usado por todos os atributos JSON de configuração e implantação de pilha. Este exemplo usa a versão editada do JSON de configuração e implantação da pilha mostrado anteriormente.
   - `chef_type`: defina esse elemento como `environment`.
   - `json_class`: defina esse elemento como `Chef::Environment`.
2. Edite `.kitchen.yml` para definir o ambiente do Test Kitchen da seguinte maneira.

```
---
driver:
  name: vagrant

provisioner:
  name: chef_solo
```

```
  environments_path: ./environments

platforms:
  - name: ubuntu-12.04

suites:
  - name: printjson
    provisioner:
      solo_rb:
        environment: test
    run_list:
      - recipe[printjson::default]
    attributes:
```

Para definir o ambiente, adicione os seguintes elementos ao `.kitchen.yml` padrão criado por `kitchen init`.

provisioner

Adicione os seguintes elementos.

- `name`: defina esse elemento como `chef_solo`.

  Para replicar o ambiente AWS OpsWorks Stacks mais de perto, você pode usar o modo local do cliente Chef em vez do Chef solo. O modo local é uma opção do cliente do Chef que usa uma versão leve do servidor do Chef (Chef Zero) que é executada localmente na instância e não em um servidor remoto. Ele permite que suas receitas usem os recursos de servidor do Chef, como pesquisa ou recipientes de dados, sem a conexão a um servidor remoto.

- `environments_path`: o subdiretório do livro de receitas que contém o arquivo de ambiente, `./environments`, neste exemplo.

suites:provisioner

Adicione um elemento `solo_rb` com um conjunto de elementos `environment` ao nome do arquivo de ambiente, menos a extensão .json. Este exemplo define `environment` como `test`.

3. Crie um arquivo de receitas designado `default.rb` com o seguinte conteúdo e salve-o no diretório `recipes` do livro de receitas.

```
log "Stack name: #{node['opsworks']['stack']['name']}"
log "Stack id: #{node['opsworks']['stack']['id']}"
```

Essa receita apenas registra os dois valores de configuração e implantação da pilha que você adicionou ao ambiente. Embora a receita esteja sendo executada localmente no Virtual Box, você faz referência a esses atributos usando a mesma sintaxe de nó que usaria se a receita estivesse sendo executada em uma instância do AWS OpsWorks Stacks.

4. Executar `kitchen converge`. Você deve ver algo semelhante à seguinte saída de log.

```
...
Converging 2 resources
Recipe: printjson::default
  * log[Stack name: MyStack] action write[2014-07-01T23:14:09+00:00] INFO:
 Processing log[Stack name: MyStack] action write (printjson::default line 1)

[2014-07-01T23:14:09+00:00] INFO: Stack name: MyStack

  * log[Stack id: 42dfd151-6766-4f1c-9940-ba79e5220b58] action
 write[2014-07-01T23:14:09+00:00] INFO: Processing log[Stack id:
 42dfd151-6766-4f1c-9940-ba79e5220b58] action write (printjson::default line 2)

[2014-07-01T23:14:09+00:00] INFO: Stack id: 42dfd151-6766-4f1c-9940-ba79e5220b58

...
```

Usar valores de atributos de implantação e configuração de pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As receitas, muitas vezes, precisam de informações sobre a configuração da pilha ou dos aplicativos implantados. Por exemplo, talvez você precise de uma lista de endereços IP da pilha para criar um

arquivo de configuração ou de um diretório de implantação do aplicativo para criar um diretório de log. Em vez de armazenar esses dados em um servidor central, o AWS OpsWorks Stacks instala um conjunto de atributos de configuração e implantação da pilha no objeto de nó de cada instância para cada evento do ciclo de vida. Esses atributos representam o estado atual da pilha, incluindo os aplicativos implantados. As receitas podem então obter os dados de que precisam do objeto de nó.

> **Note**
>
> Às vezes, os aplicativos precisam de informações do objeto de nó, como os valores de atributos de implantação e configuração de pilha. No entanto, um aplicativo não pode acessar o objeto de nó. Para fornecer dados do objeto de nó a um aplicativo, implemente uma receita que recupera as informações necessárias do objeto de nó e as coloca em um arquivo em um formato conveniente. O aplicativo pode então ler os dados do arquivo. Para obter mais informações e um exemplo, consulte Transmissão de dados para aplicativos.

As receitas podem obter valores de atributos de implantação e configuração de pilha do objeto de nó da seguinte maneira.

- Diretamente, usando o nome totalmente qualificado de um atributo.

  Use essa abordagem com qualquer pilha do Linux, mas não com pilhas do Windows.
- Com a pesquisa do Chef, que você pode usar para consultar o objeto de nó para valores de atributo.

  Use essa abordagem com pilhas do Windows e pilhas do Linux do Chef 11.10.

> **Note**
>
> Com pilhas do Linux, use o agente CLI para obter uma cópia dos pilhas de implantação e configuração de pilha de uma instância no formato JSON. Para obter mais informações, consulte Simular os atributos de implantação e configuração de pilha no Vagrant.

Tópicos

- Obter valores de atributo diretamente
- Obter valores de atributo com a pesquisa do Chef

Obter valores de atributo diretamente

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta abordagem funciona apenas para pilhas do Linux.

Simular os atributos de implantação e configuração de pilha no Vagrant mostra como obter dados de configuração e implantação de pilha usando a sintaxe de nó para fazer referência diretamente aos atributos específicos. Às vezes, essa é a melhor abordagem. No entanto, muitos atributos são definidos em coleções ou listas cujos conteúdos e nomes variam de uma pilha para outra e ao longo do tempo para uma determinada pilha. Por exemplo, o atributo `deploy` contém uma lista de atributos de aplicativo, que são designados com o nome abreviado do aplicativo. A lista, incluindo os nomes de atributos de aplicativo, geralmente varia de uma pilha para outra e até mesmo entre implantações.

Geralmente, ela é útil e, às vezes, indispensável para obter os dados necessários ao enumerar atributos em uma lista ou uma coleção. Por exemplo, suponha que você queira saber os endereços IP públicos das instâncias de sua pilha. Essas informações estão no atributo `['opsworks']['layers']`, que é definido como uma tabela de hash que contém um elemento para cada uma das camadas da pilha, designadas com o nome abreviado da camada. Cada elemento da camada é definido como uma tabela de hash que contém os atributos da camada, um dos quais é `['instances']`. Esse elemento, por sua vez, é definido como outra tabela de hash que contém um atributo para cada uma das instâncias da camada, designada com o nome abreviado da instância. Cada atributo da instância é definido como outra tabela de hash que contém os atributos da instância, incluindo `['ip']`, que representa o endereço IP público. Se você estiver tendo problemas para visualizar isso, o procedimento a seguir inclui um exemplo no formato JSON.

Este exemplo mostra como obter os dados do JSON de configuração e implantação de pilha para as camadas de uma pilha.

Para configurar o livro de receitas

1. Crie um diretório em `opsworks_cookbooks` chamado `listip` e navegue até ele.
2. Inicialize e configure o Test Kitchen, conforme descrito em Exemplo 1: Instalação de pacotes.
3. Adicione dois diretórios a `listip`: `recipes` e `environments`.
4. Crie uma versão JSON editada dos atributos de MyStack configuração e implantação que contenha os atributos relevantes. Ela deve ser parecida com a seguinte.

```
{
  "opsworks": {
    "layers": {
      "php-app": {
        "name": "PHP App Server",
        "id": "efd36017-ec42-4423-b655-53e4d3710652",
        "instances": {
          "php-app1": {
            "ip": "192.0.2.0"
          }
        }
      },
      "db-master": {
        "name": "MySQL",
        "id": "2d8e0b9a-0d29-43b7-8476-a9b2591a7251",
        "instances": {
          "db-master1": {
            "ip": "192.0.2.5"
          }
        }
      },
      "lb": {
        "name": "HAProxy",
        "id": "d5c4dda9-2888-4b22-b1ea-6d44c7841193",
        "instances": {
          "lb1": {
            "ip": "192.0.2.10"
          }
        }
      }
    }
  }
```

```
}
```

5. Crie um arquivo de ambiente chamado `test.json`, cole o JSON de exemplo em `default_attributes` e salve o arquivo na pasta `environments` do livro de receitas. O arquivo deve ser semelhante ao seguinte (para resumir, a maior parte do JSON de exemplo é representado por reticências).

```
{
  "default_attributes" : {
    "opsworks": {
      "layers": {
        ...
      }
    }
  },
  "chef_type" : "environment",
  "json_class" : "Chef::Environment"
}
```

6. Substitua o texto em `.kitchen.yml` pelo seguinte conteúdo.

```
---
driver:
  name: vagrant

provisioner:
  name: chef_zero
  environments_path: ./environment

platforms:
  - name: ubuntu-12.04

suites:
  - name: listip
    provisioner:
      client_rb:
        environment: test
    run_list:
      - recipe[listip::default]
    attributes:
```

Depois que o livro de receitas estiver configurado, você poderá usar a seguinte receita para registrar a camada IDs.

```
node['opsworks']['layers'].each do |layer, layerdata|
  log "#{layerdata['name']} : #{layerdata['id']}"
end
```

A receita enumera as camadas em `['opsworks']['layers']` e registra o nome e o ID de cada camada.

Para executar a receita de registro de ID de camada

1. Crie um arquivo chamado `default.rb` com a receita de exemplo e salve-o no diretório `recipes`.
2. Executar `kitchen converge`.

A parte relevante da saída deve ser semelhante ao conteúdo a seguir.

```
Recipe: listip::default
  * log[PHP App Server : efd36017-ec42-4423-b655-53e4d3710652] action
 write[2014-07-17T22:56:19+00:00] INFO: Processing log[PHP App Server : efd36017-
ec42-4423-b655-53e4d3710652] action write (listip::default line 4)
[2014-07-17T22:56:19+00:00] INFO: PHP App Server : efd36017-ec42-4423-b655-53e4d3710652



  * log[MySQL : 2d8e0b9a-0d29-43b7-8476-a9b2591a7251] action
 write[2014-07-17T22:56:19+00:00] INFO: Processing log[MySQL : 2d8e0b9a-0d29-43b7-8476-
a9b2591a7251] action write (listip::default line 4)
[2014-07-17T22:56:19+00:00] INFO: MySQL : 2d8e0b9a-0d29-43b7-8476-a9b2591a7251


  * log[HAProxy : d5c4dda9-2888-4b22-b1ea-6d44c7841193] action
 write[2014-07-17T22:56:19+00:00] INFO: Processing log[HAProxy : d5c4dda9-2888-4b22-
b1ea-6d44c7841193] action write (listip::default line 4)
[2014-07-17T22:56:19+00:00] INFO: HAProxy : d5c4dda9-2888-4b22-b1ea-6d44c7841193
```

Para listar os endereços IP das instâncias, você precisa de um ciclo aninhado, como o seguinte.

```
node['opsworks']['layers'].each do |layer, layerdata|
  log "#{layerdata['name']} : #{layerdata['id']}"
  layerdata['instances'].each do |instance, instancedata|
    log "Public IP: #{instancedata['ip']}"
  end
end
```

O ciclo interno itera nas instâncias de cada camada e registra os endereços IP.

Para executar a receita de registro de IP da instância

1. Substitua o código em `default.rb` pela receita de exemplo.
2. Execute `kitchen converge` para executar a receita.

A parte relevante da saída deve ser semelhante ao conteúdo a seguir.

```
  * log[PHP App Server : efd36017-ec42-4423-b655-53e4d3710652] action
 write[2014-07-17T23:09:34+00:00] INFO: Processing log[PHP App Server : efd36017-
ec42-4423-b655-53e4d3710652] action write (listip::default line 2)
[2014-07-17T23:09:34+00:00] INFO: PHP App Server : efd36017-ec42-4423-b655-53e4d3710652



  * log[Public IP: 192.0.2.0] action write[2014-07-17T23:09:34+00:00] INFO: Processing
 log[Public IP: 192.0.2.0] action write (listip::default line 4)
[2014-07-17T23:09:34+00:00] INFO: Public IP: 192.0.2.0


  * log[MySQL : 2d8e0b9a-0d29-43b7-8476-a9b2591a7251] action
 write[2014-07-17T23:09:34+00:00] INFO: Processing log[MySQL : 2d8e0b9a-0d29-43b7-8476-
a9b2591a7251] action write (listip::default line 2)
[2014-07-17T23:09:34+00:00] INFO: MySQL : 2d8e0b9a-0d29-43b7-8476-a9b2591a7251


  * log[Public IP: 192.0.2.5] action write[2014-07-17T23:09:34+00:00] INFO: Processing
 log[Public IP: 192.0.2.5] action write (listip::default line 4)
[2014-07-17T23:09:34+00:00] INFO: Public IP: 192.0.2.5
```

```
  * log[HAProxy : d5c4dda9-2888-4b22-b1ea-6d44c7841193] action
 write[2014-07-17T23:09:34+00:00] INFO: Processing log[HAProxy : d5c4dda9-2888-4b22-
b1ea-6d44c7841193] action write (listip::default line 2)
[2014-07-17T23:09:34+00:00] INFO: HAProxy : d5c4dda9-2888-4b22-b1ea-6d44c7841193


  * log[Public IP: 192.0.2.10] action write[2014-07-17T23:09:34+00:00] INFO: Processing
 log[Public IP: 192.0.2.10] action write (listip::default line 4)
[2014-07-17T23:09:34+00:00] INFO: Public IP: 192.0.2.10
```

Quando concluir, execute `kitchen destroy`; o próximo tópico usa um novo livro de receitas.

> **Note**
>
> Um dos motivos mais comuns para enumerar uma coleção de JSON de configuração e implantação de pilha é para obter dados para um determinado aplicativo implantado, por exemplo, o respectivo diretório de implantação. Para obter um exemplo, consulte Receitas de implantação.

Obter valores de atributo com a pesquisa do Chef

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta abordagem está disponível para pilhas do Windows e do Linux do Chef 11.10.

A obtenção de valores de atributo de implantação e configuração de pilha diretamente do objeto de nó pode ser complicada e não pode ser usada com pilhas do Windows. Uma abordagem alternativa é usar a pesquisa do Chef para consultar os atributos de interesse. Se você estiver familiarizado com

o Chef Server, descobrirá que a pesquisa do Chef funciona de maneira um pouco diferente com o AWS OpsWorks Stacks. Como o AWS OpsWorks Stacks usa chef-client no modo local, a pesquisa do Chef depende de uma versão local do servidor Chef chamada chef-zero, para que a pesquisa opere nos dados armazenados localmente no objeto nó da instância, em vez de em um servidor remoto.

Na prática, restringir a pesquisa a dados armazenados localmente geralmente não importa porque o objeto de nó em uma instância do AWS OpsWorks Stacks inclui a configuração da pilha e os atributos de implantação. Eles contêm a maioria, senão todos, os dados que as receitas normalmente obteriam do servidor Chef e usam os mesmos nomes, então você geralmente pode usar o código de pesquisa escrito para o Chef Server em instâncias do AWS OpsWorks Stacks sem modificação. Para obter mais informações, consulte Utilização da pesquisa do Chef.

Veja a seguir a estrutura básica de uma consulta de pesquisa:

```
result = search(:search_index, "key:pattern")
```

- O índice de pesquisa especifica quais atributos se aplicam à consulta e determina o tipo de objeto que será retornado.
- A chave especifica o nome do atributo.
- O padrão especifica quais valores do atributo você quer recuperar.

  É possível consultar valores de atributo específicos ou usar caracteres curinga para consultar um intervalo de valores.
- O resultado é uma lista de objetos que satisfazem a consulta, sendo que cada um deles é uma tabela de hash contendo vários atributos relacionados.

  Por exemplo, se você usar o índice de pesquisa `node`, a consulta retornará uma lista de objetos de instância, um para cada instância que atende à consulta. Cada objeto é uma tabela de hash que contém um conjunto de atributos que definem a configuração da instância, como o nome de host e o endereço IP.

Por exemplo, a consulta a seguir usa o índice de pesquisa `node`, que é um índice do Chef padrão que se aplica às instâncias da pilha (ou nós, segundo a terminologia do Chef). Ele procura instâncias com o nome de host `myhost`.

```
result = search(:node, "hostname:myhost")
```

A pesquisa retorna uma lista de objetos de instância cujo nome de host é `myhost`. Se você quiser o sistema operacional da primeira instância, por exemplo, ele seria representado como `result[0][:os]`. Se a consulta retornar vários objetos, você poderá enumerá-los para recuperar as informações necessárias.

Os detalhes de como usar a pesquisa em uma receita são diferentes se você estiver usando uma pilha do Linux ou do Windows. Os tópicos a seguir fornecem exemplos para os dois tipos de pilha.

Tópicos

- Usar a pesquisa em uma pilha do Linux
- Usar a pesquisa em uma pilha do Windows

Usar a pesquisa em uma pilha do Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este exemplo é baseado em uma pilha do Linux com um único servidor de aplicativo PHP. Ele usa uma pesquisa do Chef para obter o endereço IP público do servidor e coloca o endereço em um arquivo no diretório `/tmp`. Ele recupera basicamente as mesmas informações do objeto de nó como em Obter valores de atributo diretamente , mas o código é muito mais simples e não depende dos detalhes da estrutura de atributos de implantação e configuração de pilha.

As instruções a seguir resumem o procedimento para criar a pilha neste exemplo. Para obter mais informações, consulte Criar uma nova pilha.

> **Note**
>
> Se você ainda não executou uma receita personalizada em uma instância do AWS OpsWorks Stacks, primeiro veja o Executar uma receita em uma instância Linux exemplo.

Criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e clique em Add Stack (Adicionar pilha).
2. Especifique as seguintes configurações, aceite os padrões para as outras configurações e clique em Add Stack.

   - Nome: SearchJSON
   - Chave SSH padrão — um par de EC2 chaves da Amazon

   Se você precisar criar um par de EC2 chaves da Amazon, consulte Amazon EC2 Key Pairs. Observe que o par de chaves deve pertencer a mesma região da AWS que a instância. O exemplo usa a região Oeste dos EUA (Oregon).
3. Clique em Adicionar uma a camada e adicione uma camada do PHP App Server à pilha com configurações padrão.
4. Adicione uma instância ininterrupta com as configurações padrão à camada e inicie-a.

Para configurar o livro de receitas

1. Crie um diretório em `opsworks_cookbooks` chamado `searchjson` e navegue até ele.
2. Crie um arquivo `metadata.rb` com o seguinte conteúdo e salve-o em `opstest`.

```
name "searchjson"
version "0.1.0"
```

3. Criar um diretório `recipes` em `searchjson`.
4. Crie um arquivo `default.rb` com a seguinte receita e salve-o no diretório `recipes`.

```
phpserver = search(:node, "layers:php-app").first
Chef::Log.info("**********The public IP address is: '#{phpserver[:ip]}'**********")

file "/tmp/ip_addresses" do
  content "#{phpserver[:ip]}"
  mode 0644
  action :create
end
```

As pilhas do Linux oferecem suporte somente ao índice de pesquisa `node`. A receita usa esse índice para obter uma lista de instâncias na camada `php-app`. Como a layer tem somente uma instância, a receita atribui a primeira a `phpserver`. Se a camada tiver várias instâncias, você poderá enumerá-las para recuperar as informações necessárias. Cada item da lista é uma tabela de hash contendo um conjunto de atributos de instância. O atributo `ip` é definido como o endereço IP público da instância, então, você pode representar o endereço no código de receita subsequente como `phpserver[:ip]`.

Depois de adicionar uma mensagem ao log do Chef, a receita usa um recurso `file` para criar um arquivo denominado `ip_addresses`. O atributo `content` é definido como uma representação de string `phpserver[:ip]`. Quando o Chef cria `ip_addresses`, ele adiciona essa string ao arquivo.

5. Crie um arquivo `.zip` de `opsworks_cookbooks`, faça upload do arquivo em um bucket do Amazon S3, torne o arquivo público e registre o URL do arquivo. Para obter mais informações sobre repositórios de livros de receitas, consulte Repositórios de livro de receitas.

   O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Agora você pode instalar o livro de receitas e executar a receita.

Para executar a receita

1. Edite a pilha para habilitar livros de receitas personalizados, e especifique as seguintes configurações.

   - Tipo de repositório: Arquivamento HTTP
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente

   Use os valores padrão nas outras configurações e clique em Save para atualizar a configuração da pilha.

2. Edite a configuração da camada personalizada e `searchjson::defaultatribua` ao evento de configuração da camada. AWS OpsWorks As pilhas executarão a receita após a inicialização da instância ou se você acionar explicitamente o evento de configuração.

3. Execute o comando de pilha Update Custom Cookbooks, que instala a versão atual de seu repositório de livro de receitas personalizado nas instâncias da pilha. Se uma versão mais antiga do repositório estiver presente, esse comando a substituirá.
4. Execute a receita executando o comando de pilha Setup, que aciona um evento Setup na instância e executa `searchjson::default`. Deixe a Running command setup page aberta.

Após a execução com êxito da receita, você pode verificá-la.

Para verificar searchjson

1. A primeira etapa é examinar no log do Chef o evento Setup mais recente. Na Running command setup page, clique em show na coluna Log da instância php-app1 para exibir o log. Role para baixo até localizar sua mensagem de log, geralmente na metade do arquivo, que será semelhante à seguinte.

```
...
[2014-09-05T17:08:41+00:00] WARN: Previous
 bash[logdir_existence_and_restart_apache2]: ...
[2014-09-05T17:08:41+00:00] WARN: Current
 bash[logdir_existence_and_restart_apache2]: ...
[2014-09-05T17:08:41+00:00] INFO: **********The public IP address is:
 '192.0.2.0'**********
[2014-09-05T17:08:41+00:00] INFO: Processing directory[/etc/sysctl.d] action create
 (opsworks_initial_setup::sysctl line 1)
...
```

2. Use SSH fazer login na instância e liste o conteúdo de `/tmp`, que deve incluir um arquivo chamado `ip_addresses` que contém o endereço IP.

Usar a pesquisa em uma pilha do Windows

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks fornece duas opções para usar a pesquisa em pilhas do Windows.

- O índice de pesquisa `node`, que pode ser usado para consultar um conjunto de atributos do Chef padrão.

  Se você tiver receitas existentes com código de pesquisa que usa`node`, elas geralmente funcionarão nas pilhas do AWS OpsWorks Stacks sem modificação.
- Um conjunto adicional de índices de pesquisa que podem ser usados para consultar conjuntos de atributos específicos do AWS OpsWorks Stacks e alguns atributos padrão.

  Esses índices são discutidos em Usando índices de pesquisa AWS OpsWorks específicos de pilhas no Windows Stacks.

Recomendamos usar `node` para recuperar informações padrão, como nomes de host ou endereços IP. Essa abordagem mantém as receitas consistentes com a prática padrão do Chef. Use os índices de pesquisa do AWS OpsWorks Stacks para recuperar informações específicas do Stacks. AWS OpsWorks

Tópicos

- Usar o índice de pesquisa de nó em pilhas do Windows
- Usando índices de pesquisa AWS OpsWorks específicos de pilhas no Windows Stacks

Usar o índice de pesquisa de nó em pilhas do Windows

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este exemplo supõe que você já tenha concluído o exemplo em Executar uma receita na instância Windows. Caso contrário, conclua esse exemplo primeiro. Especificamente, ele descreve como habilitar o acesso RDP para suas instâncias.

Este exemplo usa uma pilha do Windows com uma única camada personalizada e uma instância. Ele usa a pesquisa do Chef com o índice de pesquisa `node` para obter o endereço IP público do servidor e coloca o endereço em um arquivo no diretório `C:\tmp`. As instruções a seguir resumem o procedimento para criar a pilha neste exemplo. Para obter mais informações, consulte Criar uma nova pilha.

Criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e selecione Add Stack (Adicionar pilha).
2. Especifique as seguintes configurações, aceite os padrões para as outras configurações e escolha Add Stack.
   - Nome — NodeSearch
   - Região: Oeste dos EUA (Oregon)

     Este exemplo funcionará em qualquer região, mas recomendamos o uso de Oeste dos EUA (Oregon) para tutoriais.
   - Sistema operacional padrão: Microsoft Windows Server 2012 R2
3. Clique em Add a layer e adicione uma camada personalizada à pilha com as seguintes configurações.
   - Nome — IPTest
   - Nome curto: iptest
4. Adicione uma instância t2.micro 24 horas por dia, 7 dias por semana, com configurações padrão na IPTest camada e inicie-a. Ela será chamada iptest1.

   AWS OpsWorks As pilhas são atribuídas automaticamente `AWS-OpsWorks-RDP-Server` a essa instância, o que permite que usuários autorizados façam login na instância.

5. Escolha Permissions e selecione Edit, SSH/RDP e sudo/admin. Os usuários frequentes precisam dessa autorização além do grupo de segurança `AWS-OpsWorks-RDP-Server` para fazer login na instância.

> **Note**
>
> Também é possível fazer login como administrador, mas isso requer um procedimento diferente. Para obter mais informações, consulte Login com RDP.

Para configurar o livro de receitas

1. Crie um diretório chamado `nodesearch` e navegue até ele.
2. Crie um arquivo `metadata.rb` com o seguinte conteúdo e salve-o em `opstest`.

```
name "nodesearch"
version "0.1.0"
```

3. Criar um diretório `recipes` em `nodesearch`.
4. Crie um arquivo `default.rb` com a seguinte receita e salve-o no diretório `recipes`.

```
directory 'C:\tmp' do
  rights :full_control, 'Everyone'
  recursive true
  action :create
end

windowsserver = search(:node, "hostname:iptest*").first
Chef::Log.info("**********The public IP address is:
 '#{windowsserver[:ipaddress]}'**********")

file 'C:\tmp\addresses.txt' do
  content "#{windowsserver[:ipaddress]}"
  rights :full_control, 'Everyone'
  action :create
end
```

A receita faz o seguinte:

1. Usa um recurso de diretório para criar um diretório `C:\tmp` para o arquivo.

   Para obter mais informações sobre esse recurso, consulte Exemplo 3: Criação de diretórios.

2. Usa a pesquisa do Chef com o índice de pesquisa `node` para obter uma lista de nós (instâncias) com um nome de host que começa com `iptest`.

   Se você usar o tema padrão, que cria nomes de host anexando números inteiros ao nome curto da camada, essa consulta retornará todas as instâncias na camada. IPTest Como a camada tem somente uma instância, a receita atribui a primeira a `windowsserver`. Para várias instâncias, você pode obter a lista completa e, em seguida, enumerá-las.

3. Adiciona uma mensagem com o endereço IP ao log do Chef para essa execução.

   O objeto `windowsserver` é uma tabela de hash cujo atributo `ipaddress` é definido como o endereço IP público da instância, então, você pode representar esse endereço no código de receita subsequente como `windowsserver[:ipaddress]`. A receita insere a string correspondente na mensagem e a adiciona ao log do Chef.

4. Usa o recurso `file` para criar um arquivo com o endereço IP chamado `C:\tmp\addresses.txt`.

   O atributo `content` do recurso especifica o conteúdo a ser adicionado ao arquivo, que, neste caso, é o endereço IP público.

5. Crie um arquivo `.zip` de `nodesearch`, faça upload do arquivo em um bucket do S3, torne o arquivo público e registre o URL do arquivo.

   O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Agora você pode instalar o livro de receitas e executar a receita.

Para instalar o livro de receitas e executar a receita

1. Edite a pilha para habilitar livros de receitas personalizados e especifique as seguintes configurações.

   - Tipo de repositório: Arquivamento do S3
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente

Aceite os valores padrão para as outras configurações e escolha Save para atualizar a configuração da pilha.

2. Execute o comando de pilha Update Custom Cookbooks, que instala a versão atual de seus livros de receitas personalizados nas instâncias da pilha, inclusive em instâncias online. Se uma versão mais antiga de seus livros de receitas estiver presente, este comando a sobrescreverá.
3. Após a execução de Update Custom Cookbooks (Atualizar livros de receitas personalizados), execute a receita executando o comando de pilhaExecute Recipes (Executar receitas) com a opção Recipes to execute (Receitas apra executar) definida como **`nodesearch::default`**. Esse comando inicia uma execução do Chef com uma lista de execução que consiste na sua receita. Deixe a página execute_recipes aberta.

Após a execução com êxito da receita, você pode verificá-la.

Para verificar nodesearch

1. Examine no log do Chef o evento execute_recipes mais recente. Na Running command execute_recipes page, escolha show na coluna Log da instância iptest1 para exibir o log. Role para baixo para localizar sua mensagem de log na parte inferior, que será semelhante ao seguinte.

```
...
[2015-05-13T18:55:47+00:00] INFO: Storing updated cookbooks/nodesearch/recipes/
default.rb in the cache.
[2015-05-13T18:55:47+00:00] INFO: Storing updated cookbooks/nodesearch/metadata.rb
 in the cache.
[2015-05-13T18:55:47+00:00] INFO: **********The public IP address is:
 '192.0.0.1'**********
[2015-05-13T18:55:47+00:00] INFO: Processing directory[C:\tmp] action create
 (nodesearch::default line 1)
[2015-05-13T18:55:47+00:00] INFO: Processing file[C:\tmp\addresses.txt] action
 create (nodesearch::default line 10)
...
```

2. Use o RDP para fazer login na instância e examine o conteúdo de `C:\tmp\addresses.txt`.

Usando índices de pesquisa AWS OpsWorks específicos de pilhas no Windows Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este exemplo supõe que você já tenha concluído o exemplo em Executar uma receita na instância Windows. Caso contrário, conclua esse exemplo primeiro. Especificamente, ele descreve como habilitar o acesso RDP para suas instâncias.

AWS OpsWorks O Stacks fornece os seguintes índices de pesquisa, além de: `node`

- `aws_opsworks_stack`: configuração da pilha.
- `aws_opsworks_layer`: as configurações de camada da pilha.
- `aws_opsworks_instance`: as configurações de instância da pilha.
- `aws_opsworks_app`: as configurações do aplicativo da pilha.
- `aws_opsworks_user`: as configurações de usuário da pilha.
- `aws_opsworks_rds_db_instance`: as informações de conexão para instâncias RDS registradas.

Esses índices incluem alguns atributos padrão do Chef, mas são destinados principalmente à recuperação de atributos específicos do AWS OpsWorks Stacks. Por exemplo, `aws_opsworks_instance` inclui um atributo `status` que fornece o status da instância, como `online`.

> **Note**
>
> Recomenda-se usar `node` quando possível para manter suas receitas consistentes com o uso padrão do Chef. Para obter um exemplo, consulte Usar o índice de pesquisa de nó em pilhas do Windows.

Este exemplo mostra como usar os índices do AWS OpsWorks Stacks para recuperar o valor de um AWS OpsWorks atributo específico do Stacks. Ele é baseado em uma pilha simples do Windows com uma camada personalizada que tem uma instância. Ele usa a pesquisa do Chef para obter o ID de AWS OpsWorks pilhas da instância e coloca os resultados no registro do Chef.

As instruções a seguir resumem o procedimento da criação de uma pilha neste exemplo. Para obter mais informações, consulte Criar uma nova pilha.

Criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e selecione + Stack (+ pilha). Especifique as seguintes configurações, aceite os padrões para as outras configurações e escolha Add Stack.
   - Nome — IDSearch
   - Região: Oeste dos EUA (Oregon)

     Este exemplo funcionará em qualquer região, mas recomendamos o uso de Oeste dos EUA (Oregon) para tutoriais.
   - Sistema operacional padrão: Microsoft Windows Server 2012 R2
2. Clique em Add a layer e adicione uma camada personalizada à pilha com as seguintes configurações.
   - Nome — IDCheck
   - Nome curto: idcheck
3. Adicione uma instância t2.micro 24 horas por dia, 7 dias por semana, com configurações padrão na IDCheck camada e inicie-a. Ela será chamada iptest1.

   AWS OpsWorks As pilhas são atribuídas automaticamente `AWS-OpsWorks-RDP-Server` a essa instância. Habilitar o acesso ao RDPexplica como adicionar uma regra de entrada a esse grupo de segurança que permite que usuários autorizados façam login na instância.

4. Escolha Permissions e, em seguida, Edit, SSH/RDP e sudo/admin. Os usuários frequentes precisam dessa autorização além do grupo de segurança `AWS-OpsWorks-RDP-Server` para fazer login na instância.

> **Note**
> É possível também fazer login como administrador, mas isso requer um procedimento diferente. Para obter mais informações, consulte Login com RDP.

Para configurar o livro de receitas

1. Crie um diretório chamado `idcheck` e navegue até ele.
2. Crie um arquivo `metadata.rb` com o seguinte conteúdo e salve-o em `opstest`.

```
name "idcheck"
version "0.1.0"
```

3. Crie um diretório `recipes` em `idcheck` e adicione um arquivo `default.rb` ao diretório que contém a seguinte receita.

```
windowsserver = search(:aws_opsworks_instance, "hostname:idcheck*").first
Chef::Log.info("**********The public IP address is:
 '#{windowsserver[:instance_id]}'**********")
```

A receita usa a pesquisa do Chef com um índice de pesquisa `aws_opsworks_instance` para obter os atributos da instância de cada instância na pilha com um nome de host que começa com `idcheck`. Se você usar o tema padrão, que cria nomes de host anexando números inteiros ao nome curto da camada, essa consulta retornará todas as instâncias na camada. IDCheck Como a camada tem somente uma instância, a receita atribui a primeira a `windowsserver`. Para várias instâncias, você pode obter a lista completa e, em seguida, enumerá-las.

A receita aproveita o fato de que há somente uma instância na pilha com esse nome de host, portanto, o primeiro resultado é o correto. Caso sua pilha tenha várias instâncias, uma pesquisa em outros atributos pode retornar mais de um resultado. Para obter uma lista de atributos de instância, consulte Data bag da instância (aws_opsworks_instance).

Os atributos da instância são basicamente uma tabela de hash, e o AWS OpsWorks Stacks ID da instância é atribuído ao `instance_id` atributo, então você pode se referir ao ID como. `windowsserver[:instance_id]` A receita insere a string correspondente na mensagem e a adiciona ao log do Chef.

4. Crie um arquivo `.zip` do livro de receitas `ipaddress`, faça upload do arquivo em um bucket do Amazon S3 e registre o URL do arquivo. Para obter mais informações sobre repositórios de livros de receitas, consulte Repositórios de livro de receitas.

   O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Agora você pode instalar o livro de receitas e executar a receita.

Para instalar o livro de receitas e executar a receita

1. Edite a pilha para habilitar livros de receitas personalizados e especifique as seguintes configurações.

   - Tipo de repositório: Arquivamento do S3
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente

   Aceite os valores padrão para as outras configurações e escolha Save para atualizar a configuração da pilha.

2. Execute o comando de pilha Update Custom Cookbooks, que instala a versão atual de seus livros de receitas personalizados nas instâncias da pilha, inclusive em instâncias online. Se uma versão mais antiga de seus livros de receitas estiver presente, este comando a sobrescreverá.
3. Após a execução de Update Custom Cookbooks (Atualizar livros de receitas personalizados), execute a receita executando o comando de pilhaExecute Recipes (Executar receitas) com a opção Recipes to execute (Receitas para executar) definida como **`idcheck::default`**. Esse comando inicia uma execução do Chef com uma lista de execução que consiste na sua receita. Deixe a página execute_recipes aberta.

Após a execução com êxito da receita, você pode verificá-la examinando no log do Chef o evento execute_recipes mais recente. Na Running command execute_recipes page, escolha show na coluna

Log da instância iptest1 para exibir o log. Role para baixo para localizar sua mensagem de log na parte inferior, que será semelhante ao seguinte.

```
...
[2015-05-13T20:03:47+00:00] INFO: Storing updated cookbooks/nodesearch/recipes/
default.rb in the cache.
[2015-05-13T20:03:47+00:00] INFO: Storing updated cookbooks/nodesearch/metadata.rb in
 the cache.
[2015-05-13T20:03:47+00:00] INFO: **********The instance ID is: 'i-8703b570'**********
[2015-05-13T20:03:47+00:00] INFO: Chef Run complete in 0.312518 seconds
...
```

Usar um livro de receitas externo em uma instância do Linux: Berkshelf

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> O Berkshelf está disponível apenas para pilhas do Linux do Chef 11.10.

Antes de iniciar a implementação de um livro de receitas, consulte Chef Community Cookbooks, que contém livros de receitas criados por membros da comunidade do Chef para diversas finalidades. Muitos desses livros de receitas podem ser usados com AWS OpsWorks Stacks sem modificações, então você pode aproveitá-los para algumas de suas tarefas em vez de implementar todo o código sozinho.

Para usar um livro de receitas externo em uma instância, você precisa de uma maneira de instalá-lo e gerenciar todas as dependências. A abordagem preferível é implementar um livro de receitas que suporte um gerenciador de dependência chamado Berkshelf. O Berkshelf funciona em EC2 instâncias da Amazon, incluindo instâncias AWS OpsWorks Stacks, mas também foi projetado para

funcionar com Test Kitchen e Vagrant. No entanto, o uso no Vagrant é um pouco diferente do que no AWS OpsWorks Stacks, então este tópico inclui exemplos para ambas as plataformas. Para obter mais informações sobre como usar o Berkshelf, consulte Berkshelf.

Tópicos

- Usar o Berkshelf com o Test Kitchen e o Vagrant
- Usando o Berkshelf com Stacks AWS OpsWorks

Usar o Berkshelf com o Test Kitchen e o Vagrant

Este exemplo mostra como usar o Berkshelf para instalar o livro de receitas da comunidade getting-started e executar sua receita, que instala um pequeno arquivo de texto em seu diretório inicial na instância.

Para instalar o Berkshelf e inicializar um livro de receitas

1. Em sua estação de trabalho, instale o Berkshelf gem da seguinte maneira.

   ```
   gem install berkshelf
   ```

   Dependendo de sua estação de trabalho, esse comando pode exigir `sudo`. Você também pode usar um gerenciador de ambiente Ruby, como o RVM. Para verificar se o Berkshelf foi instalado com êxito, execute `berks --version`.

2. O livro de receitas para este tópico se chama external_cookbook. Use o Berkshelf para criar um livro de receitas inicializado em vez de adotar a abordagem manual dos tópicos anteriores. Para fazer isso, vá até o diretório `opsworks_cookbooks` e execute o seguinte comando.

   ```
   berks cookbook external_cookbook
   ```

   O comando cria o diretório `external_cookbook` e vários subdiretórios padrão do Chef e do Test Kitchen, inclusive `recipes` e `test`. O comando também cria versões padrão de diversos arquivos padrão, incluindo:

   - `metadata.rb`
   - Arquivos de configuração para Vagrant, Test Kitchen e Berkshelf
   - Uma receita `default.rb` vazia no diretório `recipes`

> **Note**
>
> Não é necessário executar o comando `kitchen init`; o comando `berks cookbook` lida com essas tarefas.

3. Executar `kitchen converge`. Neste ponto, o livro de receitas recém-criado não faz nada de interessante, mas converge.

> **Note**
>
> Você também pode usar `berks init` para inicializar um livro de receitas existente para usar o Berkshelf.

Para usar o Berkshelf para gerenciar as dependências externas de um livro de receitas, o diretório raiz do livro de receitas deve conter um `Berksfile`, que é um arquivo de configuração que especifica como o Berkshelf deve gerenciar dependências. Quando você usou `berks cookbook` para criar o livro de receitas `external_cookbook`, ele criou um `Berksfile` com o seguinte conteúdo.

```
source "https://supermarket.chef.io"
metadata
```

Este arquivo tem as seguintes declarações:

- `source`: o URL da origem de um livro de receitas.

  Um Berksfile pode ter qualquer número de declarações `source`, sendo que cada uma delas especifica uma origem padrão para livros de receitas dependentes. Se você não especificar explicitamente a origem de um livro de receitas, o Berkshelf procura um livro de receitas com o mesmo nome nos repositórios padrão. O Berksfile padrão inclui um único atributo `source` que especifica o repositório de receitas da comunidade. Esse repositório contém o livro de receitas getting-started, por isso, você pode deixar a linha inalterada.
- `metadata`: direciona o Berkshelf a incluir as dependências de livro de receitas que são declaradas no arquivo `metadata.rb` do livro de receitas.

Também é possível declarar um livro de receitas dependentes no Berksfile incluindo um atributo `cookbook`, como discutido posteriormente.

Há duas maneiras de declarar uma dependência de livro de receitas:

- Incluindo uma declaração `cookbook` no Berksfile.

  Essa é a abordagem usada pelo AWS OpsWorks Stacks. Por exemplo, para especificar o livro de receitas getting-started usado neste exemplo, inclua `cookbook "getting-started"` no Berksfile. Em seguida, o Berkshelf procura um livro de receitas com esse nome nos repositórios padrão. Você também pode usar `cookbook` para especificar explicitamente uma origem de livro de receitas e, até mesmo, uma versão específica. Para obter mais informações, consulte Berkshelf.
- Incluindo uma declaração `metadata` no Berksfile e declarando a dependência em `metadata.rb`.

  Essa declaração instrui o Berkshelf a incluir dependências de livro de receitas declaradas em `metadata.rb`. Por exemplo, para declarar uma dependência de getting-started, adicione uma declaração `depends 'getting-started'` ao arquivo `metadata.rb` do livro de receitas.

Este exemplo usa a primeira abordagem, para manter a consistência com AWS OpsWorks as pilhas.

Para instalar o livro de receitas getting-started

1. Edite o Berksfile padrão para substituir a declaração `metadata` por uma declaração `cookbook` para `getting-started`. O conteúdo deve ser semelhante ao seguinte.

   ```
   source "https://supermarket.chef.io"

   cookbook 'getting-started'
   ```

2. Execute `berks install`, que faz o download do livro de receitas getting-started do repositório de livros de receitas da comunidade para o diretório Berkshelf de sua estação de trabalho, que normalmente é `~/.berkshelf`. Geralmente, esse diretório é simplesmente chamado de o Berkshelf. Procure no diretório `cookbooks` do Berkshelf e você verá o diretório do livro de receitas , que será nomeado como `getting-started-0.4.0`getting-started-.

3. Substitua `external_cookbook::default` na lista de execução `.kitchen.yml` por `getting-started::default`. Este exemplo não executa receitas de external_cookbook. Ela é apenas uma maneira de usar o livro de receitas getting-started. O arquivo `.kitchen.yml` agora deve ser semelhante ao conteúdo a seguir.

```
---
driver:
  name: vagrant

provisioner:
  name: chef_solo

platforms:
  - name: ubuntu-12.04

suites:
  - name: default
    run_list:
      - recipe[getting-started::default]
    attributes:
```

4. Execute `kitchen converge` e use `kitchen login` para fazer login na instância. O diretório de login deve conter um arquivo chamado `chef-getting-started.txt` com conteúdo semelhante ao seguinte:

```
Welcome to Chef!

This is Chef version 11.12.8.
Running on ubuntu.
Version 12.04.
```

O Test Kitchen instala livros de receitas no diretório `/tmp/kitchen/cookbooks` da instância. Se você listar o conteúdo do diretório, verá dois livros de receitas: external_cookbook e getting-started.

5. Execute `kitchen destroy` para desligar a instância. O próximo exemplo usa uma instância do AWS OpsWorks Stacks.

Usando o Berkshelf com Stacks AWS OpsWorks

AWS OpsWorks O Stacks suporta opcionalmente as pilhas Berkshelf for Chef 11.10. Para usar o Berkshelf com sua pilha, faça o seguinte.

- Habilite o Berkshelf na pilha.

  AWS OpsWorks Em seguida, o Stacks trata dos detalhes da instalação do Berkshelf nas instâncias do stack.
- Adicione um Berksfile ao diretório raiz do repositório do seu livro de receitas.

  O Berksfile deve conter as declarações `source` e `cookbook` para todos os livros de receitas dependentes.

Quando o AWS OpsWorks Stacks instala seu repositório de livros de receitas personalizados em uma instância, ele usa o Berkshelf para instalar os livros de receitas dependentes que são declarados no Berksfile do repositório. Para obter mais informações, consulte Utilização do Berkshelf.

Este exemplo mostra como usar o Berkshelf para instalar o livro de receitas da comunidade de introdução em uma instância do Stacks. AWS OpsWorks Ele também instala uma versão do livro de receitas personalizado createfile, que cria um arquivo em um diretório especificado. Para obter mais informações sobre o funcionamento do createfile, consulte Instalação de um arquivo usando um livro de receitas.

> **Note**
>
> Se esta é a primeira vez que você instala um livro de receitas personalizado em uma pilha do AWS OpsWorks Stacks, primeiro veja o exemplo. Executar uma receita em uma instância Linux

Comece criando uma pilha, como resumido a seguir. Para obter mais informações, consulte Criar uma nova pilha.

Criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e clique em Add Stack (Adicionar pilha).

2. Especifique as seguintes configurações, aceite os padrões para as outras configurações e clique em Add Stack.

   - Nome — BerksTest
   - Chave SSH padrão — um par de EC2 chaves da Amazon

   Se você precisar criar um par de EC2 chaves da Amazon, consulte Amazon EC2 Key Pairs. Observe que o par de chaves deve pertencer a mesma região da AWS que a instância. O exemplo usa a região padrão, Oeste dos EUA (Oregon).

3. Clique em Add a layer e adicione uma camada personalizada à pilha com as seguintes configurações.

   - Nome — BerksTest
   - Nome curto: berkstest

   Neste exemplo, é possível usar qualquer tipo de layer. No entanto, o exemplo não exige nenhum dos pacotes instalados pelas outras camadas, portanto, uma camada personalizada é a abordagem mais simples.

4. Adicione uma instância 24 horas por dia, 7 dias por semana à BerksTest camada com as configurações padrão, mas não a inicie ainda.

Com o AWS OpsWorks Stacks, os livros de receitas devem estar em um repositório remoto com uma estrutura de diretórios padrão. Em seguida, você fornece as informações de download para o AWS OpsWorks Stacks, que baixa automaticamente o repositório para cada uma das instâncias da pilha na inicialização. Para simplificar, o repositório deste exemplo é um arquivo público do Amazon S3, AWS OpsWorks mas o Stacks também oferece suporte a arquivos HTTP, repositórios Git e repositórios Subversion. Para obter mais informações, consulte Repositórios de livro de receitas.

O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Para criar o repositório do livro de receitas

1. Criar um diretório `opsworks_cookbooks` no seu diretório `berkstest_cookbooks`. Se preferir, crie esse diretório em qualquer lugar que considere conveniente, porque você vai carregá-lo em um repositório.
2. Adicione um arquivo chamado Berksfile a `berkstest_cookbooks` com o seguinte conteúdo.

```
source "https://supermarket.chef.io"

cookbook 'getting-started'
```

   Esse arquivo declara a dependência do livro de receitas getting-started e instrui o Berkshelf a fazer download dele no site do livro de receitas da comunidade.

3. Adicione um diretório `createfile` a `berkstest_cookbooks`, que contém o seguinte.
   - Um arquivo `metadata.rb` com o seguinte conteúdo.

```
name "createfile"
version "0.1.0"
```

   - Um diretório `files/default` que contém um arquivo `example_data.json` com o seguinte conteúdo.

```
{
  "my_name" : "myname",
  "your_name" : "yourname",
  "a_number" : 42,
  "a_boolean" : true
}
```

     O nome e o conteúdo do arquivo são arbitrários. A receita simplesmente copia o arquivo para o local especificado.

   - Um diretório `recipes` que contém um arquivo `default.rb` com o seguinte código de receita.

```
directory "/srv/www/shared" do
  mode 0755
  owner 'root'
  group 'root'
  recursive true
  action :create
end

cookbook_file "/srv/www/shared/example_data.json" do
  source "example_data.json"
  mode 0644
  action :create_if_missing
end
```

Essa receita cria `/srv/www/shared` e copia `example_data.json` nesse diretório a partir do diretório `files` do livro de receitas.

4. Crie um arquivo `.zip` de `berkstest_cookbooks`, faça upload do arquivo em um bucket do Amazon S3, torne o arquivo público e registre o URL do arquivo.

Agora você pode instalar os livros de receitas e executar a receita.

Para instalar os livros de receitas e executar as receitas

1. Edite a pilha para habilitar livros de receitas personalizados, e especifique as seguintes configurações.

   - Tipo de repositório: Arquivamento HTTP
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente
   - Gerenciar Berkshelf: Sim

   As duas primeiras configurações fornecem ao AWS OpsWorks Stacks as informações necessárias para baixar o repositório do livro de receitas para suas instâncias. A última configuração habilita o suporte ao Berkshelf, que faz download do livro de receitas getting-started para a instância. Aceite os valores padrão para as outras configurações e clique em Save para atualizar a configuração da pilha.

2. Edite a BerksTest camada para adicionar as seguintes receitas ao evento de ciclo de vida de configuração da camada.

- `getting-started::default`
- `createfile::default`

3. Inicie a instância. O evento de configuração ocorre após a conclusão da inicialização da instância. AWS OpsWorks O Stacks então instala o repositório do livro de receitas, usa o Berkshelf para baixar o livro de receitas de introdução e executa a configuração da camada e implementa receitas, incluindo e. `getting-started::default` `createfile::default`
4. Depois que a instância estiver online, use SSH para fazer login. Você deverá ver o seguinte

   - `/srv/www/shared` deve conter `example_data.json`.
   - `/root` deve conter `chef-getting-started.txt`.

     AWS OpsWorks O Stacks executa receitas como root, portanto, o getting start instala o arquivo no diretório em vez de no seu `/root` diretório inicial.

Usar o SDK for Ruby: fazer download de arquivos do Amazon S3

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Há algumas tarefas, como interagir com serviços da AWS, que não podem ser tratadas com recursos do Chef. Por exemplo, algumas vezes é preferível armazenar arquivos remotamente e deixar a receita baixá-los para a instância. Você pode usar o recurso remote_file para fazer download de arquivos de servidores remotos. No entanto, se quiser armazenar seus arquivos em um bucket do Amazon S3, `remote_file` poderá fazer download desses arquivos somente se a ACL permitir a operação.

As receitas podem usar o AWS SDK para Ruby para acessar a maioria dos serviços da AWS. Este tópico mostra como usar o SDK for Ruby para fazer download de um arquivo de um bucket do S3.

> **Note**
>
> Para obter mais informações sobre como usar o AWS SDK para Ruby para lidar com criptografia e descriptografia, consulte AWS:: S3:: S3Object. O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Tópicos

- Usar o SDK for Ruby em uma Instância do Vagrant
- Usando o SDK para Ruby em uma instância do Stacks AWS OpsWorks Linux
- Usando o SDK para Ruby em uma instância do Windows AWS OpsWorks Stacks

Usar o SDK for Ruby em uma Instância do Vagrant

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico descreve como uma receita em execução em uma instância do Vagrant pode usar o AWS SDK para Ruby para fazer download de um arquivo do Amazon S3. Antes de começar, você deve primeiro ter um conjunto de AWS credenciais — uma chave de acesso e uma chave de acesso secreta — que permitam que a receita acesse o Amazon S3.

> **Important**
>
> Recomendamos enfaticamente que você não use credenciais da conta raiz para essa finalidade. Em vez disso, crie um usuário com uma política adequada e forneça essas credenciais à receita.
>
> Tenha cuidado para não colocar credenciais, nem mesmo credenciais de usuário do IAM, em um local acessível ao público, por exemplo, fazendo o upload de um arquivo contendo as

credenciais para um repositório público ou do Bitbucket. GitHub Isso expõe as credenciais e pode comprometer a segurança da conta.
As receitas executadas em uma EC2 instância EC2 da Amazon podem usar uma abordagem ainda melhor, uma função do IAM, conforme descrito emUsando o SDK para Ruby em uma instância do Stacks AWS OpsWorks Linux.
O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Se você ainda não tem um usuário do apropriado, crie um usuário conforme os procedimentos a seguir. Para obter mais informações, consulte O que é o IAM.

> **Warning**
>
> Os usuários do IAM têm credenciais de longo prazo, o que representa um risco de segurança. Para ajudar a reduzir esse risco, recomendamos que você forneça a esses usuários somente as permissões necessárias para realizar a tarefa e que você os remova quando não forem mais necessários.

Para criar um usuário do IAM

1. Faça login no AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Usuários e, se necessário, escolha Adicionar usuários para criar um novo usuário administrativo.
3. Na página Definir permissões, selecione Anexar políticas diretamente.
4. Digite **S3** na caixa de pesquisa Políticas de permissão para exibir as políticas do Amazon S3.

   Escolha Amazon ReadOnlyAccess S3. Se preferir, você pode especificar uma política que conceda permissões mais amplas, como o AmazonS3 FullAccess, mas a prática padrão é conceder somente as permissões necessárias. Nesse caso, a receita só baixará um arquivo, portanto, o acesso somente leitura é suficiente.
5. Escolha Próximo.
6. Escolha Criar usuário

7. seguir, crie chaves de acesso para seu usuário. Para obter mais informações sobre a criação de chaves de acesso, consulte Gerenciar chaves de acesso para usuários do IAM no Guia do usuário do IAM.

Em seguida, forneça um arquivo para download. Este exemplo pressupõe que você colocará um arquivo chamado `myfile.txt` em um bucket do S3 recém-criado, chamado `cookbook_bucket`.

Para fornecer um arquivo para download

1. Crie um arquivo chamado `myfile.txt` com o seguinte texto e salve-o em um local conveniente na sua estação de trabalho.

```
This is the file that you just downloaded from Amazon S3.
```

2. No console do Amazon S3, crie um bucket chamado `cookbook_bucket` na região Padrão e faça upload de `myfile.txt` no bucket.

Defina o livro de receitas da seguinte maneira.

Para configurar o livro de receitas

1. Crie um diretório em `opsworks_cookbooks` chamado `s3bucket` e navegue até ele.
2. Inicialize e configure o Test Kitchen, conforme descrito em Exemplo 1: Instalação de pacotes.
3. Substitua o texto em `.kitchen.yml` pelo seguinte conteúdo.

```
---
driver:
  name: vagrant

provisioner:
  name: chef_solo
  environments_path: ./environments

platforms:
  - name: ubuntu-14.04

suites:
  - name: s3bucket
    provisioner:
```

```
      solo_rb:
        environment: test
    run_list:
      - recipe[s3bucket::default]
    attributes:
```

4. Adicione dois diretórios a `s3bucket`: `recipes` e `environments`.
5. Crie um arquivo de ambiente chamado `test.json` em que a seguinte seção `default_attributes` substitui os valores `access_key` e `secret_key` pelas chaves correspondentes para o usuário do . Salve o arquivo na pasta `environments` do livro de receitas.

```
{
  "default_attributes" : {
    "cookbooks_101" : {
      "access_key": "AKIAIOSFODNN7EXAMPLE",
      "secret_key" : "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
    }
  },
  "chef_type" : "environment",
  "json_class" : "Chef::Environment"
}
```

Há diversas maneiras de fornecer credenciais a uma receita em execução em uma instância. A principal consideração é limitar as chances de expor acidentalmente as chaves e comprometer a segurança de sua conta. Por esse motivo, não recomendamos usar valores de chave explícitos em seu código. O exemplo coloca os valores de chave no objeto de nó, o que permite que a receita faça referência a eles usando a sintaxe de nó em vez de expor valores literais. Você deve ter privilégios de raiz para acessar o objeto de nó, o que limita a possibilidade de que as chaves sejam expostas. Para obter mais informações, consulte Práticas recomendadas de gerenciamento de chaves de acesso da AWS.

> **Note**
>
> Observe que o exemplo usa atributos aninhados, com `cookbooks_101` como o primeiro elemento. Essa prática limita a chance de um conflito de nomes caso haja outros atributos `access_key` ou `secret_key` no objeto de nó.

A receita a seguir faz download de `myfile.text` no bucket `cookbook_bucket`.

```
gem_package "aws-sdk ~> 3" do
  action :install
end

ruby_block "download-object" do
  block do
    require 'aws-sdk'

    s3 = Aws::S3::Client.new(
          :access_key_id => "#{node['cookbooks_101']['access_key']}",
          :secret_access_key => "#{node['cookbooks_101']['secret_key']}")

    myfile = s3.bucket['cookbook_bucket'].objects['myfile.txt']
    Dir.chdir("/tmp")
    File.open("myfile.txt", "w") do |f|
      f.write(myfile.read)
      f.close
    end
  end
  action :run
end
```

A primeira parte da receita instala o SDK for Ruby, que é um pacote de gem. O recurso gem_package instala gems que serão usados por receitas ou outros aplicativos.

> **Note**
>
> Em geral, sua instância tem duas instâncias Ruby, que normalmente têm versões diferentes. Uma é uma instância dedicada que é usada pelo cliente do Chef. A outra é usada por aplicativos e receitas em execução na instância. É importante entender essa distinção ao instalar pacotes de gem, porque há dois recursos para instalar gems: gem_package e chef_gem. Se o pacote de gem for usado por aplicativos ou receitas, instale-o com `gem_package`; `chef_gem` é apenas para pacotes de gem usados pelo cliente Chef.

O restante da receita é um recurso ruby_block que contém o código Ruby, que faz download do arquivo. Talvez você pense que, como uma receita é um aplicativo Ruby, é possível colocar o código diretamente na receita. No entanto, uma execução do Chef compila todo esse código antes de

executar qualquer recurso. Se você colocar o código de exemplo diretamente na receita, o Ruby tentará resolver a instrução `require 'aws-sdk'` antes de executar o recurso `gem_package`. Como o SDK for Ruby ainda não foi instalado, a compilação falhará.

O código em um recurso `ruby_block` não é compilado até que esse recurso seja executado. Neste exemplo, o recurso `ruby_block` é executado após o recurso `gem_package` ter concluído a instalação do SDK for Ruby, então, o código será executado com êxito.

O código no `ruby_block` funciona da seguinte maneira.

1. Cria um novo objeto `Aws::S3`, que fornece a interface do serviço.

   As chaves de acesso e as chaves secretas são especificadas por referência aos valores armazenados no objeto de nó.
2. Chama a associação `bucket.objects` do objeto `S3`, que retorna um objeto `Aws::S3::Object` chamado `myfile` que representa `myfile.txt`.
3. Usa `Dir.chdir` para definir o diretório de trabalho como `/tmp`.
4. Abre um arquivo chamado `myfile.txt`, grava o conteúdo de `myfile` no arquivo e fecha o arquivo.

Para executar a receita

1. Crie um arquivo chamado `default.rb` com a receita de exemplo e salve-o no diretório `recipes`.
2. Executar `kitchen converge`.
3. Execute `kitchen login` para fazer login na instância e execute `ls /tmp`. Você deverá ver `myfile.txt`, junto com vários arquivos e diretórios do Test Kitchen.

   ```
   vagrant@s3bucket-ubuntu-1204:~$ ls /tmp
   install.sh  kitchen  myfile.txt  stderr
   ```

   Você também pode executar `cat /tmp/myfile.txt` para verificar se o conteúdo do arquivo está correto.

Quando terminar, execute `kitchen destroy` para encerrar a instância.

Usando o SDK para Ruby em uma instância do Stacks AWS OpsWorks Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico descreve como usar o SDK for Ruby AWS OpsWorks em uma instância do Stacks Linux para baixar um arquivo de um bucket do Amazon S3. AWS OpsWorks O Stacks instala automaticamente o SDK for Ruby em todas as instâncias do Linux. No entanto, quando você cria o objeto de cliente de um serviço, deve fornecer um conjunto adequado de credenciais da AWS `AWS::S3.new` ou o equivalente para outros serviços.

O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Usar o SDK for Ruby em uma Instância do Vagrant mostra como reduzir o risco de expor suas credenciais armazenando as credenciais no objeto de nó e mencionando os atributos em seu código de receita. Quando você executa receitas em uma EC2 instância da Amazon, você tem uma opção ainda melhor, uma função do IAM.

Um perfil do IAM; trabalha de maneira muito parecida com um usuário do IAM. Ela tem uma política associada que concede permissões para usar os diversos serviços da AWS. No entanto, você atribui uma função a uma EC2 instância da Amazon e não a um indivíduo. Os aplicativos em execução na instância podem, por sua vez, adquirir as permissões concedidas pela política associada. Com uma função, as credenciais nunca aparecem em seu código, sequer indiretamente. Este tópico descreve como você pode usar uma função do IAM para executar a receita Usar o SDK for Ruby em uma Instância do Vagrant em uma EC2 instância da Amazon.

Você pode executar essa receita com o Test Kitchen usando o driver kitchen-ec2, conforme descrito em Exemplo 9: Usando EC2 instâncias da Amazon. No entanto, instalar o SDK for Ruby EC2 em instâncias da Amazon é um pouco complicado e não é algo com que você precise se AWS OpsWorks preocupar para Stacks. Todas as instâncias do AWS OpsWorks Stacks Linux têm o SDK

for Ruby instalado por padrão. Para simplificar, o exemplo usa, portanto, uma instância do AWS OpsWorks Stacks.

A primeira etapa é configurar o perfil do IAM. Este exemplo usa a abordagem mais simples, que é usar a EC2 função da Amazon que o AWS OpsWorks Stacks cria quando você cria sua primeira pilha. Ela se chama `aws-opsworks-ec2-role`. No entanto, o AWS OpsWorks Stacks não anexa uma política a essa função, portanto, por padrão, ele não concede permissões.

Você deve anexar a política `AmazonS3ReadOnlyAccess` ao perfil `aws-opsworks-ec2-role` para conceder as permissões apropriadas. Para obter mais informações sobre como associar uma política a um perfil, consulte Adicionar permissões de identidade do IAM (console) no Guia do usuário do IAM.

Você especifica a função quando cria ou atualiza uma pilha. Configure uma pilha com uma camada personalizada, conforme descrito em Executar uma receita em uma instância Linux, com uma adição. Na página Add Stack, confirme se o perfil de instância padrão do IAM está definido como aws-opsworks-ec2 funções. AWS OpsWorks As pilhas então atribuirão essa função a todas as instâncias da pilha.

O procedimento para configurar o livro de receitas é semelhante ao usado para Executar uma receita em uma instância Linux. A seguir, há um breve resumo; consulte esse exemplo para obter detalhes.

Para configurar o livro de receitas

1. Crie um diretório chamado `s3bucket_ops` e navegue até ele.
2. Crie um arquivo `metadata.rb` com o seguinte conteúdo e salve-o em `s3bucket_ops`.

```
name "s3bucket_ops"
version "0.1.0"
```

3. Criar um diretório `recipes` em `s3bucket_ops`.
4. Crie um arquivo `default.rb` com a seguinte receita e salve-o no diretório `recipes`.

```
Chef::Log.info("******Downloading a file from Amazon S3.******")

ruby_block "download-object" do
  block do
    require 'aws-sdk'
```

```
    s3 = AWS::S3.new

    myfile = s3.buckets['cookbook_bucket'].objects['myfile.txt']
    Dir.chdir("/tmp")
    File.open("myfile.txt", "w") do |f|
      f.syswrite(myfile.read)
      f.close
    end
  end
  action :run
end
```

5. Crie um arquivo `.zip` de `s3bucket_ops` e carregue o arquivo em um bucket do Amazon S3. Para simplificar, torne o arquivo público e registre o URL do arquivo para uso posterior. Você também pode armazenar seus livros de receitas em um arquivo Amazon S3 privado ou em vários outros tipos de repositório. Para obter mais informações, consulte Repositórios de livro de receitas.

Essa receita é semelhante à usada pelo exemplo anterior, com as seguintes exceções.

- Como o AWS OpsWorks Stacks já instalou o SDK for Ruby, `chef_gem` o recurso foi excluído.
- A receita não passa credenciais para `AWS::S3.new`.

  As credenciais são atribuídas automaticamente ao aplicativo com base na função da instância.
- A receita usa `Chef::Log.info` para adicionar uma mensagem ao log do Chef.

Crie uma pilha para este exemplo da seguinte maneira. Você também pode usar uma pilha existente do Windows. Basta atualizar os livros de receitas, como será descrito mais tarde.

Para criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e clique em Add Stack (Adicionar pilha).
2. Especifique as seguintes configurações, aceite os padrões para as outras configurações e clique em Add Stack.
   - Nome: RubySDK
   - Chave SSH padrão — um par de EC2 chaves da Amazon

Se você precisar criar um par de EC2 chaves da Amazon, consulte Amazon EC2 Key Pairs. Observe que o par de chaves deve pertencer a mesma região da AWS que a instância. O exemplo usa a região padrão, Oeste dos EUA (Oregon).

3. Clique em Add a layer e adicione uma camada personalizada à pilha com as seguintes configurações.

   - Nome: S3Download
   - Nome curto: s3download

   Qualquer tipo de camada funcionará nas pilhas do Linux, mas o exemplo não exige nenhum dos pacotes instalados por outros tipos de camada, portanto, uma camada personalizada é a abordagem mais simples.

4. Adicione uma instância ininterrupta com as configurações padrão à camada e inicie-a.

Agora você pode instalar e executar a receita

Para executar a receita

1. Edite a pilha para habilitar livros de receitas personalizados, e especifique as seguintes configurações.

   - Tipo de repositório: Arquivamento HTTP
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente.

   Use os valores padrão nas outras configurações e clique em Save para atualizar a configuração da pilha.

2. Execute o comando de pilha Update Custom Cookbooks, que instala a versão atual de seus livros de receitas personalizados nas instâncias da pilha. Se uma versão mais antiga de seus livros de receitas estiver presente, este comando a sobrescreverá.

3. Execute a receita acionando o comando de stack Execute Recipes (Executar receitas) com a opção Recipes to execute (Receitas para executar) definida como **`s3bucket_ops::default`**. Este comando inicia uma execução do Chef, com uma lista de execução que consiste em `s3bucket_ops::default`.

> **Note**
>
> Normalmente, você faz com que AWS OpsWorks as pilhas executem suas receitas automaticamente, atribuindo-as ao evento de ciclo de vida apropriado. Você pode executar essas receitas acionando manualmente o evento. Use um comando de stack para acionar eventos Setup e Configure, e um comando de pilha para acionar eventos Deploy e Undeploy.

Após a receita ser executada com sucesso, verifique isso.

Para verificar s3bucket_ops

1. A primeira etapa é examinar o log do Chef. Sua pilha deve ter uma instância chamada opstest1. Na página Instances, clique em show na coluna Log da instância para exibir o log do Chef. Role para baixo para encontrar a mensagem de log próxima à parte inferior.

```
...
[2014-07-31T17:01:45+00:00] INFO: Storing updated cookbooks/opsworks_cleanup/
attributes/customize.rb in the cache.
[2014-07-31T17:01:45+00:00] INFO: Storing updated cookbooks/opsworks_cleanup/
metadata.rb in the cache.
[2014-07-31T17:01:46+00:00] INFO: ******Downloading a file from Amazon S3.******
[2014-07-31T17:01:46+00:00] INFO: Processing template[/etc/hosts] action create
 (opsworks_stack_state_sync::hosts line 3)
...
```

2. Utilize o SSH para fazer login na instância e listar o conteúdo de `/tmp`.

Usando o SDK para Ruby em uma instância do Windows AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este exemplo supõe que você já tenha concluído o exemplo em Executar uma receita na instância Windows. Caso contrário, conclua esse exemplo primeiro. Especificamente, ele descreve como habilitar o acesso RDP para suas instâncias.
> O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Este tópico descreve como usar o AWS SDK para Rubyem uma instância do AWS OpsWorks Stacks Windows para baixar um arquivo de um bucket do S3.

Se um aplicativo Ruby precisar acessar um recurso da AWS, forneça a ela um conjunto de credenciais da AWS com as permissões apropriadas. Para receitas, sua melhor opção para fornecer credenciais da AWS é usar uma função AWS Identity and Access Management (IAM). Uma função do IAM funciona como um usuário do IAM: ela tem uma política anexada que concede permissões para usar os vários AWS serviços. No entanto, você atribui uma função a uma instância do Amazon Elastic Compute Cloud (Amazon EC2) em vez de a um indivíduo. Os aplicativos em execução na instância podem, por sua vez, adquirir as permissões concedidas pela política associada. Com uma função, as credenciais nunca aparecem em seu código, sequer indiretamente.

A primeira etapa é configurar o perfil do IAM. Este exemplo usa a abordagem mais simples, que é usar a EC2 função da Amazon que o AWS OpsWorks Stacks cria quando você cria sua primeira pilha. Ela se chama `aws-opsworks-ec2-role`. No entanto, o AWS OpsWorks Stacks não anexa uma política a essa função, portanto, por padrão, ele não concede permissões.

Você deve anexar a política `AmazonS3ReadOnlyAccess` ao perfil `aws-opsworks-ec2-role` para conceder as permissões apropriadas. Para obter mais informações sobre como associar uma política a um perfil, consulte Adicionar permissões de identidade do IAM (console) no Guia do usuário do IAM.

Você especifica a função quando cria ou atualiza uma pilha. Configure uma pilha com uma camada personalizada, conforme descrito em Executar uma receita na instância Windows, com uma adição.

Na página Add Stack, confirme se o perfil de instância padrão do IAM está definido como aws-opsworks-ec2 funções. AWS OpsWorks As pilhas então atribuirão essa função a todas as instâncias da pilha.

O procedimento para configurar o livro de receitas é semelhante ao usado para Executar uma receita em uma instância Linux. A seguir, há um breve resumo; consulte esse exemplo para obter detalhes.

Para configurar o livro de receitas

1. Crie um diretório chamado `s3bucket_ops` e navegue até ele.
2. Crie um arquivo `metadata.rb` com o seguinte conteúdo e salve-o em `s3bucket_ops`.

```
name "s3download"
version "0.1.0"
```

3. Criar um diretório `recipes` em `s3download`.
4. Crie um arquivo `default.rb` com a seguinte receita e salve-o no diretório `recipes`. *`windows-cookbooks`*Substitua pelo nome do bucket do S3 que você usará para armazenar o arquivo a ser baixado.

```
Chef::Log.info("******Downloading an object from S3******")

chef_gem "aws-sdk-s3" do
  compile_time false
  action :install
end

ruby_block "download-object" do
  block do
    require 'aws-sdk-s3'

    Aws.use_bundled_cert!

    s3_client = Aws::S3::Client.new(region:'us-west-2')

    s3_client.get_object(bucket: 'windows-cookbooks',
                         key: 'myfile.txt',
                         response_target: '/chef/myfile.txt')
  end
```

```
  action :run
end
```

5. Crie um arquivo `.zip` de `s3download` e carregue o arquivo em um bucket do S3. Torne o público o arquivo e registre o URL para uso posterior.
6. Crie um arquivo de texto chamado `myfile.txt` e carregue em um bucket do S3. Este é o arquivo que sua receita baixará para que você possa usar qualquer bucket.

A receita executa as seguintes tarefas.

1: Instalação do SDK for Ruby v2.

O exemplo usa o SDK for Ruby para carregar o objeto. No entanto, o AWS OpsWorks Stacks não instala esse SDK em instâncias do Windows, então a primeira parte da receita usa um `chef_gem`recurso para lidar com essa tarefa. Utilize esse recurso para instalar gems para uso pelo Chef, que inclui receitas.

2: Faça download do arquivo.

A terceira parte da receita usa um recurso `ruby_block` para executar o código SDK for Ruby v2 para baixar `myfile.txt` de um bucket do S3 chamado *`windows-cookbooks`* para o diretório `/chef` da instância. Altere *`windows-cookbooks`* para o nome do bucket que contém `myfile.txt`.

> **Note**
>
> Uma receita é um aplicativo Ruby, por isso, você pode colocar código Ruby no corpo da receita; ele não precisa estar em um recurso `ruby_block`. No entanto, o Chef executa o código Ruby no corpo da receita primeiro, seguido por cada recurso, em ordem. Para este exemplo, se você colocar o código do download no corpo da receita, ele falhará porque depende do SDK for Ruby e o recurso `chef_gem` que instala o SDK ainda não foi executado. O código no recurso `ruby_block` é executado quando o recurso é executado, o que acontece depois que o recurso `chef_gem` instala o SDK for Ruby.

Crie uma pilha para este exemplo da seguinte maneira. Você também pode usar uma pilha existente do Windows. Basta atualizar os livros de receitas, como será descrito mais tarde.

Criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e selecione Add Stack (Adicionar pilha). Especifique as seguintes configurações, aceite os padrões para as outras configurações e escolha Add Stack.

   - Nome: S3Download
   - Região: Oeste dos EUA (Oregon)

     Este exemplo funcionará em qualquer região, mas recomendamos o uso de Oeste dos EUA (Oregon) para tutoriais.
   - Sistema operacional padrão: Microsoft Windows Server 2012 R2
2. Clique em Add a layer e adicione uma camada personalizada à pilha com as seguintes configurações.

   - Nome: S3Download
   - Nome curto: s3download
3. Adicione uma instância ininterrupta com as configurações padrão à camada S3Download e inicie-a.

Agora você pode instalar e executar a receita

Para executar a receita

1. Edite a pilha para habilitar livros de receitas personalizados, e especifique as seguintes configurações.

   - Tipo de repositório: Arquivamento do S3.
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente.

   Aceite os valores padrão para as outras configurações e escolha Save para atualizar a configuração da pilha.
2. Execute o comando de pilha Update Custom Cookbooks, que instala a versão mais recente de seu livro de receitas personalizado nas instâncias online da pilha. Se uma versão mais antiga de seus livros de receitas estiver presente, este comando a sobrescreverá.
3. Execute a receita acionando o comando de stack Execute Recipes (Executar receitas) com a opção Recipes to execute (Receitas para executar) definida como **`s3download::default`**.

Este comando inicia uma execução do Chef, com uma lista de execução que consiste em `s3download::default`.

> **Note**
>
> Normalmente, você faz com que AWS OpsWorks as pilhas executem suas receitas automaticamente, atribuindo-as ao evento de ciclo de vida apropriado. Essas receitas também podem ser executadas acionando o evento manualmente. Use um comando de stack para acionar eventos Setup e Configure, e um comando de pilha para acionar eventos Deploy e Undeploy.

Após a receita ser executada com sucesso, verifique isso.

Para verificar s3download

1. A primeira etapa é examinar o log do Chef. Sua pilha deve ter uma instância designada s3download1. Na página Instances, escolha show na coluna Log da instância para exibir o log do Chef. Role para baixo para encontrar a mensagem de log próximo ao fim.

```
...
[2015-05-01T21:11:04+00:00] INFO: Loading cookbooks [s3download@0.0.0]
[2015-05-01T21:11:04+00:00] INFO: Storing updated cookbooks/s3download/recipes/
default.rb in the cache.
[2015-05-01T21:11:04+00:00] INFO: ******Downloading an object from S3******
[2015-05-01T21:11:04+00:00] INFO: Processing chef_gem[aws-sdk] action install
 (s3download::default line 3)
[2015-05-01T21:11:05+00:00] INFO: Processing ruby_block[download-object] action run
 (s3download::default line 8)
...
```

2. Use o RDP para fazer login na instância e examine o conteúdo de `c:\chef`.

Instalar o softwares do Windows

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Estes exemplos supõem que você já concluiu o exemplo em Executar uma receita na instância Windows. Caso contrário, conclua esse exemplo primeiro. Especificamente, ele descreve como habilitar o acesso RDP para suas instâncias.

As instâncias do Windows iniciam com o Windows Server 2012 R2 Standard, portanto, é necessário instalar alguns softwares. Os detalhes dependem do tipo de software.

- Os recursos do Windows são componentes opcionais do sistema, incluindo .NET Framework e Internet Information Services (IIS), que você pode baixar para sua instância.
- Os softwares de terceiros normalmente são comercializados com um pacote de instalação, como um arquivo MSI, que você deve baixar para a instância e, em seguida, executar.

  Alguns softwares da Microsoft também fornecem um pacote de instalação.

Esta seção descreve como implementar livros de receitas para instalar pacotes e recursos do Windows. Ele também apresenta o livro de receitas do Windows do Chef, que contém os recursos e as funções auxiliares que simplificam a implementação das receitas para as instâncias do Windows.

Tópicos

- Instalar um recurso do Windows: IIS
- Instalar um pacote em uma instância do Windows

Instalar um recurso do Windows: IIS

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Os recursos do Windows são um conjunto de componentes opcionais do sistema, incluindo .NET Framework e Internet Information Services (IIS). Este tópico descreve como implementar um livro de receitas para instalar um atributo comum, o Internet Information Services (IIS).

> **Note**
>
> Instalar um pacote mostra como instalar softwares fornecidos em um pacote de instalação, como um arquivo MSI, que você deve baixar para a instância e executar. Livros de receitas do IIS

Executar uma receita na instância Windows mostra como usar um recurso `powershell_script` para instalar um recurso do Windows. Este exemplo mostra uma abordagem alternativa: usar o recurso do livro de receitas do Windows`windows_feature` do Chef. Este livro de receitas contém um conjunto de recursos que usa o Deployment Image Servicing and Management para executar várias tarefas no Windows, incluindo o recurso de instalação.

> **Note**
>
> O Chef também tem um livro de receitas do IIS, que você pode usar para gerenciar o IIS. Para obter mais informações, consulte livro de receitas do IIS.

Para configurar o livro de receitas

1. Acesse o GitHub repositório de livros de receitas do Windows e baixe o `windows` livro de receitas.

   Este exemplo supõe que você fará download do repositório `windows` como um arquivo .zip, mas também poderá clonar o repositório se preferir.

2. Acesse o GitHub repositório de livros de receitas chef_handler e baixe o livro de receitas. `chef-handler`

O livro de receitas `windows` depende de `chef_handler`; você não o usará diretamente. Este exemplo supõe que você fará download do repositório `chef_handler` como um arquivo .zip, mas também poderá clonar o repositório se preferir.

3. Extraia os livros de receitas `windows` e `chef_handler` em diretórios no seu diretório de livros de receitas chamados `windows` e `chef_handler`, respectivamente.
4. Crie um diretório em seu diretório de livros de receitas chamado `install-iis` e navegue até ele.
5. Adicione um arquivo `metadata.rb` a `install-iis`, com o seguinte conteúdo.

```
name "install-iis"
version "0.1.0"

depends "windows"
```

A diretriz `depends` permite que você use os recursos do livro de receitas `windows` em suas receitas.

6. Crie um diretório `recipes` em `install-iis` e adicione um arquivo chamado `default.rb` ao diretório que contém o seguinte código de receita.

```
%w{ IIS-WebServerRole IIS-WebServer }.each do |feature|
  windows_feature feature do
    action :install
  end
end

service 'w3svc' do
  action [:start, :enable]
end
```

A receita usa o recurso `windows_feature` do livro de receitas do `windows` para instalar o seguinte:

1. A função do Servidor Web do IIS.
2. O Servidor Web do IIS.

Em seguida, a receita usa um recurso `service` para iniciar e ativar o serviço do IIS (W3SVC).

> **Note**
>
> Para obter uma lista completa de recursos do Windows disponíveis, use o RDP para fazer login na instância, abra uma janela de prompt de comando, e execute o comando a seguir. Observe que a lista é bastante extensa.
>
> ```
> dism /online /Get-Features
> ```

7. Crie um arquivo `.zip` que contenha os livros de receitas `install-iis`, `chef_handler` e `windows` e carregue o arquivo em um bucket do S3. Torne público o arquivo e registre o URL para uso posterior. Este exemplo supõe que o nome do arquivo seja `install-iis.zip`. Para obter mais informações, consulte Repositórios de livro de receitas.

   O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Crie uma pilha para este exemplo da seguinte maneira. Use também uma pilha existente do Windows. Basta atualizar os livros de receitas, como será descrito mais tarde.

Criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e selecione Add Stack (Adicionar pilha). Especifique as seguintes configurações, aceite os padrões para as outras configurações e escolha Add Stack.

   - Nome: InstallIIS
   - Região: Oeste dos EUA (Oregon)

     Este exemplo funcionará em qualquer região, mas recomendamos o uso de Oeste dos EUA (Oregon) para tutoriais.
   - Sistema operacional padrão: Microsoft Windows Server 2012 R2

2. Clique em Add a layer e adicione uma camada personalizada à pilha com as seguintes configurações.

- Nome: IIS
- Nome curto: iis

3. Adicione uma instância ininterrupta com as configurações padrão à camada IIS e inicie-a.

Agora você pode instalar o livro de receitas e executar a receita

Para instalar o livro de receitas e executar a receita

1. Edite a pilha para habilitar livros de receitas personalizados, e especifique as seguintes configurações.

   - Tipo de repositório: Arquivamento do S3
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente.

   Aceite os valores padrão para as outras configurações e escolha Save para atualizar a configuração da pilha.

2. Execute o comando de pilha Update Custom Cookbooks, que instala a versão mais recente dos seus livros de receitas personalizado nas instâncias online da pilha. Se uma versão mais antiga de seus livros de receitas estiver presente, este comando a sobrescreverá.

3. Execute a receita acionando o comando de stack Execute Recipes (Executar receitas) com a opção Recipes to execute (Receitas para executar) definida como **`install-iis::default`**. Este comando inicia uma execução do Chef, que executa as receitas especificadas.

   > **Note**
   >
   > Este exemplo usa Execute Recipes por conveniência, mas normalmente você faz com que AWS OpsWorks as pilhas executem suas receitas automaticamente, atribuindo-as ao evento de ciclo de vida apropriado. Você pode executar essas receitas acionando manualmente o evento. Use um comando de stack para acionar eventos Setup e Configure, e um comando de pilha para acionar eventos Deploy e Undeploy.

4. Para verificar a instalação, use RDP para conectar à instância e abrir o Windows Explorer. O sistema de arquivos agora tem um diretório `C:\inetpub`. Se você verificar a lista de serviços no aplicativo do painel de controle de ferramentas administrativas, o IIS deve estar próximo à parte inferior. No entanto, ele será chamado World Wide Web Publishing Service, em vez de IIS.

Instalar um pacote em uma instância do Windows

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este exemplo supõe que você já tenha concluído o exemplo em Executar uma receita na instância Windows. Caso contrário, conclua esse exemplo primeiro. Especificamente, ele descreve como habilitar o acesso RDP para suas instâncias.

Caso seu software venha com um pacote de instalação, como um MSI, baixe o arquivo para a instância e execute-o. Este exemplo mostra como implementar um livro de receitas para instalar um pacote MSI, o runtime do Python, incluindo como definir variáveis de ambiente. Para obter mais informações sobre como instalar recursos do Windows, como o IIS, consulte Instalar um recurso do Windows: IIS.

Para configurar o livro de receitas

1. Crie um diretório chamado `installpython` e navegue até ele.
2. Adicione um arquivo `metadata.rb` a `installpython`, com o seguinte conteúdo.

   ```
   name "installpython"
   version "0.1.0"
   ```

3. Adicione os diretórios `recipes` e `files` a `installpython` e um diretório `default` aos arquivos.
4. Baixe um pacote Python de Python Releases for Windows para o diretório `files\default` do livro de receitas. Este exemplo instala a versão Windows x86- do 3.5.0a3, que usa um instalador chamado `python-3.4.3.amd64.msi`python-64.msi.

5. Adicione um arquivo chamado `default.rb` ao diretório `recipes` com o seguinte código de receita.

```
directory 'C:\tmp' do
  rights :full_control, 'Everyone'
  recursive true
  action :create
end

cookbook_file 'C:\tmp\python-3.4.3.amd64.msi' do
  source "python-3.4.3.amd64.msi"
  rights :full_control, 'Everyone'
  action :create
end

windows_package 'python' do
  source 'C:\tmp\python-3.4.3.amd64.msi'
  action :install
end

env "PATH" do
  value 'c:\python34'
  delim ";"
  action :modify
end
```

A receita faz o seguinte:

1. Usa um recurso de diretório para criar um diretório `C:\tmp`.

   Para obter mais informações sobre esse recurso, consulte Exemplo 3: Criação de diretórios.

2. Usa um recurso cookbook_file para copiar os arquivos do instalador do diretório `files\default` do livro de receitas para `C:\tmp`.

   Para obter mais informações sobre esse recurso, consulte Instalação de um arquivo usando um livro de receitas.

3. Usa um recurso windows_package para executar o instalador MSI, que instala o Python em `c:\python34`.

O instalador cria os diretórios necessários e instala os arquivos, mas não modifica a variável de ambiente `PATH` do sistema.

4. Usa um recurso env para adicionar `c:\python34` ao caminho do sistema.

   Use o recurso env para definir variáveis de ambiente. Nesse caso, a receita permite que você execute facilmente os scripts de Python a partir da linha de comando adicionando `c:\python34` ao caminho.

   - O nome do recurso especifica o nome da variável de ambiente. Neste exemplo, é `PATH`.
   - O atributo `value` especifica o valor da variável. Neste exemplo, `c:\\python34` (você precisa recuar o caractere \).
   - A ação `:modify` acrescenta o valor especificado ao valor atual da variável.
   - O atributo `delim` especifica um delimitador que separa o novo valor do valor existente, que é `;` neste exemplo.

6. Crie um arquivo `.zip` de `installpython`, carregue-o em um bucket do S3 e torne-o público. Registre o URL do arquivo para uso posterior. Para obter mais informações, consulte Repositórios de livro de receitas.

   O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Crie uma pilha para este exemplo da seguinte maneira. Use também uma pilha existente do Windows. Basta atualizar os livros de receitas, como será descrito mais tarde.

Criar uma pilha

1. Abra o console do AWS OpsWorks Stacks e selecione Add Stack (Adicionar pilha). Especifique as seguintes configurações, aceite os padrões para as outras configurações e escolha Add Stack.

   - Nome — InstallPython
   - Região: Oeste dos EUA (Oregon)

     Este exemplo funcionará em qualquer região, mas recomendamos o uso de Oeste dos EUA (Oregon) para tutoriais.
   - Sistema operacional padrão: Microsoft Windows Server 2012 R2

2. Clique em Add a layer e adicione uma camada personalizada à pilha com as seguintes configurações.
   - Nome: Python
   - Nome curto: python
3. Adicione uma instância ininterrupta com as configurações padrão à camada do Python e inicie-a.

Depois que a instância estiver online, instale o livro de receitas e execute a receita

Para instalar o livro de receitas e executar a receita

1. Edite a pilha para habilitar livros de receitas personalizados, e especifique as seguintes configurações.
   - Tipo de repositório: Arquivamento do S3.
   - URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente.

   Aceite os valores padrão para as outras configurações e escolha Save para atualizar a configuração da pilha.
2. Execute o comando de pilha Update Custom Cookbooks, que instala a versão mais recente dos seus livros de receitas personalizado nas instâncias online da pilha. Se houver uma versão mais antiga de seu livro de receitas, este comando a substituirá.
3. Execute a receita acionando o comando de stack Execute Recipes (Executar receitas) com a opção Recipes to execute (Receitas para executar) definida como **`installpython::default`**. Este comando inicia uma execução do Chef, com uma lista de execução que consiste em `installpython::default`.

   > **Note**
   >
   > Este exemplo usa Execute Recipes por conveniência, mas normalmente você faz com que AWS OpsWorks as pilhas executem suas receitas automaticamente, atribuindo-as ao evento de ciclo de vida apropriado. Você pode executar essas receitas acionando manualmente o evento. Use um comando de stack para acionar eventos Setup e Configure, e um comando de pilha para acionar eventos Deploy e Undeploy.
4. Para verificar a instalação, use RDP para conectar à instância e abrir o Windows Explorer.

- O sistema de arquivos agora tem um diretório `C:\Python34`.
- Se você executar `path` a partir da linha de comando, ele deve ser semelhante a: `PATH=c:\python34;C:\Windows\system32;...`
- Se você executar `python --version` a partir da linha de comando, ele deve retornar `Python 3.4.3`.

Sobrepor atributos integrados

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este tópico se aplica apenas a pilhas do Linux. Você não pode substituir atributos integrados em pilhas do Windows.

AWS OpsWorks O Stacks instala um conjunto de livros de receitas integrados em cada instância. Muitos dos livros de receitas integrados oferecem suporte a camadas integradas e seus arquivos de atributo definem uma variedade de configurações padrão do sistema e de aplicativo, como as definições de configuração do servidor Apache. Ao colocar essas configurações em arquivos de atributo, você pode personalizar muitas definições de configuração, substituindo o atributo integrado correspondente de uma das seguintes maneiras:

- Defina o atributo em JSON personalizado.

  Essa abordagem tem a vantagem de ser simples e flexível. No entanto, você deve inserir manualmente o JSON personalizado, portanto, não há nenhuma maneira eficiente de gerenciar as definições de atributos.

- Implemente um livro de receitas personalizado e defina o atributo em um arquivo de atributo `customize.rb`.

  Essa abordagem é menos flexível do que usar o JSON personalizado, mas é mais eficiente, pois você pode colocar livros de receitas personalizados sob o controle de código-fonte.

Este tópico descreve como usar um arquivo de atributo de livro de receitas personalizado para substituir atributos integrados, usando o servidor Apache como exemplo. Para obter mais informações sobre como substituir atributos com o JSON personalizado, consulte Usar JSON personalizado. Para consultar uma discussão geral sobre como substituir atributos, consulte Sobrepor atributos.

> **Note**
>
> A substituição de atributos é a maneira preferida de personalizar as definições de configuração, mas as configurações não são sempre representadas por atributos. Neste caso, você pode personalizar o arquivo de configuração substituindo o modelo que as receitas integradas usam para criar o arquivo de configuração. Para obter um exemplo, consulte Substituir modelos integrados.

Os atributos integrados normalmente representam valores nos arquivos de modelo que as receitas de configuração usam para criar arquivos de configuração. Por exemplo, uma das receitas Setup do `apache2`, `default.rb`, usa o modelo `apache2.conf.erb` para criar o principal arquivo de configuração de servidor do Apache, `httpd.conf` (Amazon Linux) ou `apache2.conf` (Ubuntu). O exemplo a seguir é um trecho do arquivo de modelo:

```
...
#
# MaxKeepAliveRequests: The maximum number of requests to allow
# during a persistent connection. Set to 0 to allow an unlimited amount.
# We recommend you leave this number high, for maximum performance.
#
MaxKeepAliveRequests <%= node[:apache][:keepaliverequests] %>
#
# KeepAliveTimeout: Number of seconds to wait for the next request from the
# same client on the same connection.
#
```

```
KeepAliveTimeout <%= node[:apache][:keepalivetimeout] %>
##
## Server-Pool Size Regulation (MPM specific)
##

...
```

A configuração `KeepAliveTimeout` neste exemplo é o valor do atributo `[:apache][:keepalivetimeout]` O valor padrão desse atributo é definido no `apache2` arquivo de atributo `apache.rb` do livro de receitas, conforme o trecho a seguir:

```
...
# General settings
default[:apache][:listen_ports] = [ '80','443' ]
default[:apache][:contact] = 'ops@example.com'
default[:apache][:log_level] = 'info'
default[:apache][:timeout] = 120
default[:apache][:keepalive] = 'Off'
default[:apache][:keepaliverequests] = 100
default[:apache][:keepalivetimeout] = 3
...
```

> **Note**
>
> Para obter mais informações sobre atributos integrados comuns, consulte Atributos de livros de receitas integrados.

Para oferecer suporte à substituição de atributos integrados, todos os livros de receitas integrados contêm um arquivo de atributos `customize.rb`, que é incorporado a todos os módulos usando uma diretriz `include_attribute`. Os arquivos `customize.rb` dos livros de receitas integrados não contêm definições de atributos e não têm qualquer efeito sobre os atributos integrados. Para substituir os atributos integrados, crie um livro de receitas personalizado com o mesmo nome do livro de receitas integrado e coloque suas definições de atributo em um arquivo de atributos cujo nome também é `customize.rb`. Esse arquivo tem prioridade sobre a versão integrada e é incluído em todos os módulos relacionados. Se você definir qualquer atributo integrado em seu `customize.rb`, ele substituirá o atributo integrado correspondente.

Este exemplo mostra como substituir o atributo integrado `[:apache][:keepalivetimeout]` para definir seu valor como 5, em vez de 3. Você pode usar uma abordagem semelhante para qualquer atributo integrado. No entanto, tenha cuidado em relação a quais atributos você substitui. Por exemplo, substituir atributos no namespace `opsworks` pode causar problemas para algumas receitas integradas.

> **Important**
>
> Não substitua atributos integrados modificando uma cópia do arquivo de atributos integrados. Por exemplo, você pode colocar uma cópia de `apache.rb` na pasta `apache2/attributes` do seu livro de receitas personalizado e modificar algumas de suas configurações. No entanto, este arquivo tem prioridade sobre a versão integrada, e as receitas integradas passarão a usar sua versão do `apache.rb`. Se o AWS OpsWorks Stacks modificar posteriormente o `apache.rb` arquivo incorporado, as receitas não receberão os novos valores, a menos que você atualize manualmente sua versão. Ao usar`customize.rb`, você substitui somente os atributos especificados; as receitas integradas continuam obtendo automaticamente up-to-date valores para cada atributo que você não substituiu.

Para iniciar, crie um livro de receitas personalizado.

Para criar o livro de receitas

1. No seu diretório `opsworks_cookbooks`, crie um diretório de livros de receitas chamado `apache2` e navegue até ele.

   Para substituir atributos integrados, o livro de receitas personalizado devem ter o mesmo nome que o livro de receitas integrado. Neste exemplo, é `apache2`.

2. No diretório `apache2`, crie um diretório `attributes`.
3. Adicione um arquivo chamado `customize.rb` ao o diretório `attributes` e use-o para definir os atributos integrados do livro de receitas que você deseja substituir. Para este exemplo, o arquivo deve conter o seguinte:

   ```
   normal[:apache][:keepalivetimeout] = 5
   ```

> **Important**
>
> Para substituir um atributo integrado, um atributo personalizado deve ser do tipo `normal` ou superior e ter exatamente o mesmo nome de nó que o atributo integrado correspondente. O tipo `normal` garante que o atributo personalizado tenha prioridade sobre os atributos integrados, que são todos os atributos do tipo `default`. Para obter mais informações, consulte Precedência de atributo.

4. Crie um arquivo `.zip` de `opsworks_cookbooks` chamado `opsworks_cookbooks.zip` e carregue o arquivo em um bucket do Amazon Simple Storage Service (Amazon S3). Para simplificar, torne o arquivo público. Registre o URL para uso posterior. Você também pode armazenar seus livros de receitas em um arquivo privado do Amazon S3 ou em vários outros tipos de repositório. Para obter mais informações, consulte Repositórios de livro de receitas.

   O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Para usar o atributo personalizado, crie uma pilha e instale o livro de receitas.

Para usar o atributo personalizado

1. Abra o console do AWS OpsWorks Stacks e selecione Add Stack (Adicionar pilha).
2. Especifique as seguintes configurações padrão.

   - Nome — ApacheConfig
   - Região: Oeste dos EUA (Oregon)

     Você pode colocar sua pilha em qualquer região, mas recomendamos Oeste dos EUA (Oregon) para tutoriais.
   - Chave SSH padrão — Um par de EC2 chaves

     Se você precisar criar um par de EC2 chaves, consulte Amazon EC2 Key Pairs. Observe que o par de chaves deve pertencer a mesma região da AWS como a pilha.

   Escolha Advanced>>, defina Use custom Chef cookbooks como Yes e especifique as seguintes configurações.

- Tipo de repositório: Arquivamento HTTP
- URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente

Aceite os valores padrão para as outras configurações e, em seguida, escolha Add Stack para criar a pilha.

> **Note**
>
> Este exemplo usa o sistema operacional padrão, o Amazon Linux. Se preferir, use o Ubuntu. A única diferença é que, nos sistemas Ubuntu, a configuração integrada da receita produz um arquivo de configuração com as mesmas configurações chamado `apache2.conf` e o coloca no diretório `/etc/apache2`.

3. Clique em Adicionar uma camada e adicione uma camada Java App Server à pilha com as seguintes configurações.
4. Adicione uma instância ininterrupta com as configurações padrão à camada e inicie a instância.

   Uma instância t2.micro é suficiente para este exemplo.
5. Depois que a instância estiver online, use SSH para fazer login. O arquivo `httpd.conf` está no diretório `/etc/httpd/conf`. Se você examinar o arquivo, verá sua configuração personalizada `KeepAliveTimeout`. O restante das configurações terá os valores padrão do arquivo integrado `apache.rb`. A parte relevante do `httpd.conf` deve ser semelhante a:

```
...
#
# KeepAliveTimeout: Number of seconds to wait for the next request from the
# same client on the same connection.
#
KeepAliveTimeout 5
...
```

Substituir modelos integrados

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este tópico se aplica apenas a pilhas do Linux. Você não pode substituir modelos integrados em pilhas do Windows.

As receitas integradas do AWS OpsWorks Stacks usam modelos para criar arquivos em instâncias, principalmente arquivos de configuração para servidores, como o Apache. Por exemplo, as receitas `apache2` usam o modelo `apache2.conf.erb` para criar o arquivo de configuração principal do servidor Apache, `httpd.conf` (Amazon Linux) ou `apache2.conf` (Ubuntu).

A maioria das definições de configuração nesses modelos é representada por atributos. Portanto, a melhor maneira de personalizar um arquivo de configuração é substituir os atributos integrados apropriados. Para obter um exemplo, consulte Sobrepor atributos integrados. No entanto, se as configurações que você deseja personalizar não estiverem representadas por atributos integrados ou não estiverem no modelo, você deve substituir o modelo em si. Este tópico descreve como substituir um modelo integrado para especificar uma definição de configuração do Apache.

Você pode fornecer respostas de erro personalizadas para o Apache, adicionando as configurações `ErrorDocument` ao arquivo `httpd.conf`. `apache2.conf.erb` contém apenas alguns exemplos comentados, como mostrado a seguir:

```
...
#
# Customizable error responses come in three flavors:
# 1) plain text 2) local redirects 3) external redirects
```

```
#
# Some examples:
#ErrorDocument 500 "The server made a boo boo."
#ErrorDocument 404 /missing.html
#ErrorDocument 404 "/cgi-bin/missing_handler.pl"
#ErrorDocument 402 http://www.example.com/subscription_info.html
...
```

Como essas configurações são comentários codificados, você não pode especificar valores personalizados substituindo atributos, deve substituir o modelo propriamente dito. No entanto, ao contrário dos atributos, não é possível substituir partes específicas de um arquivo de modelo. Você deve criar um livro de receitas personalizado com o mesmo nome que a versão integrada, copiar o arquivo de modelo para o mesmo subdiretório e modificar o arquivo conforme o necessário. Este tópico mostra como substituir o `apache2.conf.erb` para obter uma resposta ao erro 500. Para ver uma discussão geral sobre a substituição de modelos, consulte Uso de modelos personalizados.

> **Important**
>
> Quando você substitui um modelo integrado, as receitas integradas usam sua versão personalizada do modelo, em vez da versão integrada. Se o AWS OpsWorks Stacks atualizar o modelo integrado, o modelo personalizado ficará fora de sincronia e poderá não funcionar corretamente. AWS OpsWorks O Stacks não faz essas alterações com frequência e, quando um modelo muda, o AWS OpsWorks Stacks lista as alterações e oferece a opção de atualizar para uma nova versão. Recomendamos que você monitore as alterações no repositório do AWS OpsWorks Stacks e atualize manualmente seu modelo personalizado, conforme o necessário. Observe que o repositório tem uma ramificação específica para cada versão do Chef com suporte, então verifique se você está na ramificação correta.

Para iniciar, crie um livro de receitas personalizado.

Para criar o livro de receitas

1. Crie um `opsworks_cookbooks` diretório em seu diretório de livros de receitas chamado `apache2` e navegue até ele. Para substituir modelos integrados, o livro de receitas personalizado deve ter o mesmo nome que o livro de receitas integrado. Neste exemplo, é `apache2`.

> **Note**
>
> Se você já tiver concluído a demonstração Sobrepor atributos integrados, use o mesmo livro de receitas `apache2` neste exemplo e pule a Etapa 2.

2. Crie um arquivo `metadata.rb` com o conteúdo a seguir. Em seguida, salve-o no diretório `apache2` .

```
name "apache2"
version "0.1.0"
```

3. No diretório `apache2`, crie um diretório `templates/default`.

> **Note**
>
> O diretório `templates/default` funciona para as instâncias do Amazon Linux, que usam o modelo `apache2.conf.erb` padrão. As instâncias Ubuntu 14.04 usam um modelo `apache2.conf.erb` específico do sistema operacional, que está no diretório `templates/ubuntu-14.04`. Se você quiser que a personalização se aplique às instâncias do Ubuntu 14.04, substitua esse modelo também.

4. Copie o modelo `apache2.conf.erb` integrado no diretório `templates/default`. Abra o arquivo de modelo, exclua a linha `ErrorDocument 500` e forneça uma mensagem de erro personalizada, como o exemplo a seguir:

```
...
ErrorDocument 500 "A custom error message."
#ErrorDocument 404 /missing.html
...
```

5. Crie um arquivo `.zip` de `opsworks_cookbooks` chamado `opsworks_cookbooks.zip` e, em seguida, faça upload dele em um bucket do Amazon Simple Storage Service (Amazon S3). Para simplificar, torne o arquivo público. Registre o URL do arquivo para uso posterior. Você também pode armazenar seus livros de receitas em um arquivo privado do Amazon S3 ou em vários outros tipos de repositório. Para obter mais informações, consulte Repositórios de livro de receitas.

O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

> **Note**
>
> Para simplificar, este exemplo adiciona uma mensagem de erro fixada ao modelo. Para alterá-la, você deve modificar o modelo e reinstalar o livro de receitas. Para ter maior flexibilidade, defina um atributo personalizado padrão para a string de erro no arquivo de atributo `customize.rb` do livro de receitas personalizado e atribua o valor do atributo ao `ErrorDocument 500`. Por exemplo, se você nomear o atributo `[:apache][:custom][:error500]`, a linha correspondente em `apache2.conf.erb` deverá ser semelhante a:
>
> ```
> ...
> ErrorDocument 500 <%= node[:apache][:custom][:error500] %>
> #ErrorDocument 404 /missing.html
> ...
> ```
>
> Em seguida, você pode alterar a mensagem de erro personalizada a qualquer momento, substituindo `[:apache][:custom][:error500]`. Se você utilizar o JSON personalizado para substituir o atributo, não será necessário mexer no mesmo livro de receitas.

Para usar o modelo personalizado, crie uma pilha e instale o livro de receitas.

Para usar o modelo personalizado

1. Abra o console do AWS OpsWorks Stacks e selecione Add Stack (Adicionar pilha).
2. Especifique as seguintes configurações padrão:
   - Nome — ApacheTemplate
   - Região: Oeste dos EUA (Oregon)
   - Chave SSH padrão — um par de chaves Amazon Elastic Compute Cloud EC2 (Amazon)

     Se você precisar criar um par de EC2 chaves da Amazon, consulte Amazon EC2 Key Pairs. Observe que o par de chaves deve pertencer a mesma região da AWS que a instância.

Escolha Advanced>> e escolha Use custom Chef cookbooks para especificar as seguintes configurações:

- Tipo de repositório: Arquivamento HTTP
- URL do repositório: o URL do arquivo do livro de receitas que você registrou anteriormente

Aceite os valores padrão para as outras configurações e, em seguida, escolha Add Stack para criar a pilha.

3. Clique em Adicionar uma camada e, em seguida, adicione uma camada Java App Server à pilha com as seguintes configurações.
4. Adicione uma instância ininterrupta com as configurações padrão à camada e inicie a instância.

   Uma instância t2.micro é suficiente para este exemplo.
5. Depois que a instância estiver online, use SSH para fazer login. O arquivo `httpd.conf` está no diretório `/etc/httpd/conf`. O arquivo deve conter sua configuração personalizada `ErrorDocument`, que deve ser semelhante a:

```
...
# Some examples:
ErrorDocument 500 "A custom error message."
#ErrorDocument 404 /missing.html
#ErrorDocument 404 "/cgi-bin/missing_handler.pl"
#ErrorDocument 402 http://www.example.com/subscription_info.html
...
```

Balanceamento de carga de uma camada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks fornece duas opções de balanceamento de carga, Elastic Load Balancing HAProxye, que normalmente são usadas para balancear a carga nas instâncias de uma camada de servidor de aplicativos. Este tópico descreve os benefícios e limitações de cada uma para ajudar você a decidir qual opção escolher ao adicionar o balanceamento de carga para uma camada. Em alguns casos, a melhor abordagem é usar as duas opções.

Terminação SSL

A HAProxy camada incorporada não lida com a terminação SSL; você deve encerrar a SSL nos servidores. A vantagem dessa abordagem é que o tráfego é criptografado até chegar aos servidores. No entanto, os servidores devem lidar com a descriptografia, o que aumenta a carga do servidor. Além disso, é necessário colocar os certificados SSL em servidores de aplicativos, que são mais acessíveis aos usuários.

Com o Elastic Load Balancing, é possível encerrar o SSL no balanceador de carga. Isso reduz a carga nos servidores de aplicativos, mas o tráfego entre o balanceador de carga e o servidor não é criptografado. O Elastic Load Balancing também permite encerrar o SSL no servidor, mas isso é algo complicado de configurar.

Escalabilidade

Se o tráfego de entrada exceder a capacidade de um balanceador de HAProxy carga, você deverá aumentar sua capacidade manualmente.

O Elastic Load Balancing escala automaticamente para processar o tráfego. Para garantir que um balanceador de carga do Elastic Load Balancing tenha capacidade suficiente para suportar a carga esperada quando ele for lançado on-line, é possível preparar o balanceador de carga.

Falha no balanceamento de carga

Se a instância que hospeda seu HAProxy servidor falhar, todo o site poderá ficar off-line até que você possa reiniciar a instância.

O Elastic Load Balancing é mais resistente a falhas do que. HAProxy Por exemplo, ele provisiona nós de balanceamento de carga em cada zona de disponibilidade que tem EC2 instâncias registradas. Se o serviço em uma região for interrompido, os outros nós permanecem suportando o tráfego de entrada. Para obter mais informações, consulte Elastic Load Balancing Concepts.

Intervalo ocioso

Ambos os load balancers encerram uma conexão se um servidor estiver inativo por mais de um tempo limite de ociosidade especificado.

- HAProxy — O valor do tempo limite de inatividade não tem um limite superior.
- Elastic Load Balancing: o valor padrão do intervalo ocioso é de 60 segundos, com máximo de 3.600 segundos (60 minutos).

O limite de tempo ocioso do Elastic Load Balancing é suficiente para a maioria das finalidades. Recomendamos usar HAProxy se você precisar de um tempo limite de inatividade mais longo. Por exemplo:

- Uma conexão HTTP de longa duração que é usada para enviar notificações.
- Uma interface administrativa usada para executar tarefas que podem demorar mais de 60 minutos.

Mapeamento com base em URL

Você pode querer ter um load balancer encaminhar uma solicitação recebida para um servidor específico com base na solicitação do URL. Por exemplo, suponha que você tem um grupo de dez servidores de aplicativos que fornece suporte a um aplicativo de comércio eletrônico online. Oito dos servidores processam o catálogo e dois processam pagamentos. Direcione todas as solicitações HTTP relacionadas ao pagamento para os servidores de pagamento, com base no URL solicitado. Nesse caso, você direcionaria tudo o URLs que incluísse “pagamento” ou “finalização da compra” para um dos servidores de pagamento.

Com HAProxy, você pode usar o mapeamento baseado em URL para direcionar o conteúdo URLs de uma string especificada para servidores específicos. Para usar o mapeamento baseado em URL com o AWS OpsWorks Stacks, você deve criar um arquivo de HAProxy configuração personalizado substituindo o `haproxy-default.erb` modelo no livro de receitas integrado. `haproxy` Para obter mais informações, consulte o Manual de HAProxy configuração Uso de modelos personalizados e. Não é permitido usar o mapeamento baseado em URL para solicitações HTTPS. Uma solicitação HTTPS é criptografada, portanto, não HAProxy há como examinar a URL da solicitação.

O Elastic Load Balancing tem suporte limitado para mapeamento de URL. Para obter mais informações, consulte Listener Configurations for Elastic Load Balancing. (Configurações do ouvinte para Elastic Load Balancing).

Recomendação: Recomendamos usar o Elastic Load Balancing para balanceamento de carga, a menos que você tenha requisitos que só possam ser atendidos por. HAProxy Nesse caso, a melhor abordagem pode ser combinar os dois usando o Elastic Load Balancing como um balanceador de

carga front-end que distribui o tráfego de entrada para um conjunto de servidores. HAProxy Para fazer isso:

- Configure uma HAProxy instância em cada uma das zonas de disponibilidade da sua pilha para distribuir solicitações aos servidores de aplicativos da zona.
- Atribua as HAProxy instâncias a um balanceador de carga do Elastic Load Balancing, que então distribui as solicitações recebidas para os balanceadores de carga. HAProxy

Essa abordagem permite que você use HAProxy o mapeamento baseado em URL para distribuir diferentes tipos de solicitações aos servidores de aplicativos apropriados. No entanto, se um dos HAProxy servidores ficar off-line, o site continuará funcionando porque o balanceador de carga do Elastic Load Balancing distribui automaticamente o tráfego de entrada para os servidores íntegros. HAProxy Observe que você deve usar o Elastic Load Balancing como balanceador de carga front-end; um HAProxy servidor não pode distribuir solicitações para outros servidores. HAProxy

Migrando do Chef Server para o Stacks AWS OpsWorks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Como o AWS OpsWorks Stacks é baseado no Chef, a migração do Chef Server para o AWS OpsWorks Stacks é relativamente simples. Este tópico oferece orientações para modificar o código do Chef Server para que ele funcione com o AWS OpsWorks Stacks.

> **Note**
>
> Não recomendamos a migração para pilhas que usam versões do Chef anteriores à 11.10, que são baseadas somente em Chef e não são compatíveis com pesquisa ou recipientes de dados.

Tópicos

- Funções de mapeamento para camadas
- Utilização de recipientes de dados
- Utilização da pesquisa do Chef
- Gerenciamento de livros de receitas e receitas
- Como usar ambientes do Chef

Funções de mapeamento para camadas

O Chef Server usa funções para representar e gerenciar instâncias com a mesma finalidade e configuração, como um conjunto de instâncias em que cada uma hospeda um servidor de aplicativos Java. Uma camada do AWS OpsWorks Stacks tem essencialmente a mesma finalidade que uma função do Chef. Uma camada é um modelo para criar um conjunto de instâncias do Amazon Elastic Compute Cloud EC2 (Amazon) com a mesma configuração, pacotes instalados, procedimento de implantação de aplicativos e assim por diante.

AWS OpsWorks As pilhas incluem um conjunto de camadas integradas para vários tipos de servidor de aplicativos, um balanceador de HAProxy carga, um mestre de banco de dados MySQL e um mestre de monitoramento Ganglia. Por exemplo, a camada Java App Server incorporada é um esquema para a criação de instâncias que hospedam um servidor Tomcat.

Para migrar para o AWS OpsWorks Stacks, você precisa associar cada função a uma camada que forneça funcionalidade equivalente. Para algumas funções, você pode simplesmente usar uma das camadas integradas. Outras funções podem exigir vários níveis de personalização. Inicie examinando a funcionalidade das camadas incorporadas, incluindo as receitas associadas a cada uma, para ver se alguma fornece pelo menos algumas das funcionalidades da função. Para obter mais informações sobre as camadas integradas, consulte Camadas e AWS OpsWorks Referência de camadas de pilhas. Para examinar as receitas integradas, consulte o GitHub repositório público AWS OpsWorks Stacks.

Como continuar depende do quanto você consegue corresponder uma camada para cada função, como do seguinte modo.

Uma camada integrada é compatível com todas as funcionalidades da função

> Você pode usar a camada integrada diretamente, com pequenas personalizações, se necessário. Por exemplo, se um perfil é compatível com um servidor Tomcat, as receitas da camada Java App Server podem já processar todas as tarefas do perfil, talvez com uma pequena personalização. Por exemplo, você pode fazer as receitas incorporadas da camada usarem

definições de configuração do Apache ou Tomcat personalizadas substituindo os atributos ou modelos apropriados.

Uma camada integrada é compatível com algumas funcionalidades da função, mas não com todas

Você pode usar uma camada integrada estendendo a camada. Isso normalmente envolve a implementação de receitas personalizadas para oferecer compatibilidade à funcionalidade ausente e a atribuição das receitas aos eventos de ciclo de vida da camada. Por exemplo, suponha que sua função instale um servidor Redis nas mesmas instâncias que hospedam um servidor Tomcat. Você pode estender a camada Java App Server para que ela corresponda à funcionalidade do perfil implementando uma receita personalizada para instalar o Redis nas instâncias da camada e atribuindo a receita ao evento de configuração da camada.

Nenhuma camada integrada é compatível de maneira adequada com a funcionalidade da função

Implemente uma camada personalizada. Por exemplo, suponha que sua função seja compatível com um servidor de banco de dados MongoDB, que não é compatível com nenhuma das camadas integradas. Você pode fornecer essa compatibilidade com a implementação de receitas para instalar os pacotes necessários, configurar o servidor, e assim por diante, e atribuir as receitas a uma camada personalizada de eventos de ciclo de vida. Normalmente, você pode usar pelo menos algumas das receitas da função para essa finalidade. Para obter mais informações sobre como implementar uma camada personalizada, consulte Criação de uma camada de servidor Tomcat personalizada.

Utilização de recipientes de dados

O Chef Server permite que você passe dados definidos pelo usuário para as receitas usando recipientes de dados.

- Você armazena os dados com seus livros de receitas e o Chef os instala em cada instância.
- Você pode usar recipientes de dados criptografados para dados confidenciais, como senhas.

AWS OpsWorks O Stacks suporta sacos de dados; as receitas podem recuperar os dados usando exatamente o mesmo código do Chef Server. No entanto, a compatibilidade tem as seguintes limitações e diferenças:

- Os recipientes de dados são compatíveis apenas no Chef 11.10 Linux e pilhas posteriores.

  As pilhas do Windows e do Linux que executam versões anteriores do Chef não são compatíveis com recipientes de dados.

- Você não armazena recipientes de dados no repositório do livro de receitas.

  Em vez disso, você usa JSON personalizado para gerenciar os dados dos recipientes de dados.
- AWS OpsWorks O Stacks não suporta pacotes de dados criptografados.

  Se você precisar armazenar dados confidenciais de maneira criptografada, como senhas ou certificados, recomendamos armazená-los em um bucket do S3 privado. Você pode então criar uma receita personalizada que usa o Amazon SDK para Ruby para recuperar os dados. Para obter um exemplo, consulte Usar o SDK for Ruby.

Para obter mais informações, consulte Utilização de recipientes de dados.

Utilização da pesquisa do Chef

O Chef Server armazena informações de configuração da pilha, como endereços IP e configurações de função, no servidor. As receitas usam a pesquisa do Chef para recuperar esses dados. AWS OpsWorks O Stacks usa uma abordagem um pouco diferente. Por exemplo, as pilhas do Linux do Chef 11.10 são baseadas no modo local do cliente do Chef, uma opção de cliente do Chef que executa uma versão leve do Chef Server (muitas vezes chamada de Chef Zero) localmente na instância. O Chef Zero é compatível com a pesquisa em relação aos dados armazenados no objeto do nó da instância.

Em vez de armazenar dados da pilha em um servidor remoto, o AWS OpsWorks Stacks adiciona um conjunto de atributos de configuração e implantação da pilha ao objeto de nó de cada instância para cada evento do ciclo de vida. Esses atributos representam um snapshot da configuração da pilha. Eles usam a mesma sintaxe do Chef Server e representam a maioria dos dados que as receitas precisam recuperar do servidor.

Muitas vezes, você não precisa modificar o código dependente da pesquisa de suas receitas para Stacks. AWS OpsWorks Como a pesquisa do Chef opera no objeto do nó, que inclui a configuração da pilha e os atributos de implantação, as consultas de pesquisa no AWS OpsWorks Stacks geralmente funcionam exatamente como funcionam com o Chef Server.

A principal exceção é causada pelo fato de que os atributos de configuração e implantação da pilha contêm somente dados que a AWS OpsWorks Stacks conhece quando instala os atributos na instância. Se você criar ou modificar um atributo localmente em uma instância específica, essas alterações não se propagarão de volta para AWS OpsWorks as pilhas e não serão incorporadas aos atributos de configuração e implantação da pilha que estão instalados nas outras instâncias. Você

pode usar a pesquisa para recuperar o valor do atributo apenas para essa instância. Para obter mais informações, consulte Utilização da pesquisa do Chef.

Para compatibilidade com o Chef Server, o AWS OpsWorks Stacks adiciona um conjunto de `role` atributos ao objeto do nó, cada um contendo um dos atributos da camada da pilha. Se a receita usa `roles` como uma chave de pesquisa, você não precisa alterar o código de pesquisa. A consulta retorna automaticamente os dados para a camada correspondente. Por exemplo, ambas as consultas a seguir retornam os atributos da camada `php-app`.

```
phpserver = search(:node, "layers:php-app").first
```

```
phpserver = search(:node, "roles:php-app").first
```

Gerenciamento de livros de receitas e receitas

AWS OpsWorks O Stacks e o Chef Server lidam com livros de receitas e receitas de forma um pouco diferente. Com o Chef Server:

- Você fornece todos os livros de receitas, implementando-os você mesmo ou usando livros de receitas da comunidade.
- Você armazena livros de receitas no servidor.
- Você executa receitas manualmente ou em uma programação regular.

Com AWS OpsWorks pilhas:

- AWS OpsWorks O Stacks fornece um ou mais livros de receitas para cada uma das camadas integradas. Esses livros de receitas processam tarefas padrão, como instalação e configuração de um software da camada integrada e implantação de aplicativos.

  Para processar tarefas que não são realizadas pelos livros de receitas incorporados, você adiciona livros de receitas personalizados à pilha ou usa livros de receitas da comunidade.
- Você armazena livros de receitas do AWS OpsWorks Stacks em um repositório remoto, como um bucket S3 ou um repositório Git.

  Para obter mais informações, consulte Como armazenar livros de receitas.

- Você pode executar receitas manualmente, mas normalmente AWS OpsWorks as pilhas executam receitas para você em resposta a um conjunto de eventos do ciclo de vida que ocorrem em pontos-chave durante o ciclo de vida de uma instância.

  Para obter mais informações, consulte Execução de receitas.
- AWS OpsWorks Stacks suporta Berkshelf somente em pilhas do Chef 11.10. Se você usa Berkshelf para gerenciar as dependências dos livros de receitas, não pode usar as pilhas que executam o Chef 11.4 ou versões anteriores.

  Para obter mais informações, consulte Utilização do Berkshelf.

Tópicos

- Como armazenar livros de receitas
- Execução de receitas

Como armazenar livros de receitas

Com o Chef Server, você armazena os livros de receitas no servidor e os implanta do servidor para as instâncias. Com o AWS OpsWorks Stacks, você armazena livros de receitas em um repositório: um arquivo S3 ou HTTP ou um repositório Git ou Subversion. Você especifica as informações que o AWS OpsWorks Stacks precisa para baixar o código do repositório para as instâncias de uma pilha ao instalar livros de receitas.

Para migrar do Chef Server, você precisa colocar os livros de receitas em um desses repositórios. Para obter informações sobre como estruturar um repositório de livros de receitas, consulte Repositórios de livro de receitas.

Execução de receitas

No AWS OpsWorks Stacks, cada camada tem um conjunto de eventos de ciclo de vida — configuração, configuração, implantação, desimplantação e desligamento — cada um deles ocorre em um ponto-chave durante o ciclo de vida de uma instância. Para executar uma receita personalizada, você normalmente a atribui ao evento apropriado na camada apropriada. Quando o evento ocorre, o AWS OpsWorks Stacks executa as receitas associadas. Por exemplo, o evento de Instalação ocorre após uma instância terminar a inicialização, de modo que você normalmente atribui receitas a esse evento que executam tarefas como instalar e configurar pacotes e iniciar serviços.

Você pode executar receitas manualmente usando o comando de pilha Executar receitas. Esse comando é útil para desenvolvimento e testes, mas você também pode usá-lo para executar receitas que não são mapeadas para um evento de ciclo de vida. Você também pode usar o comando Executar receitas para acionar manualmente os eventos Instalar e Configurar.

Além do console AWS OpsWorks Stacks, você pode usar o AWS CLI SDKsou executar receitas. Essas ferramentas são compatíveis com todas as ações da API do AWS OpsWorks Stacks, mas são mais simples de usar do que a API. Use o comando da CLI create-deployment para acionar um evento de ciclo de vida, que executa todas as receitas associadas. Você também pode usar esse comando para executar uma ou mais receitas sem acionar um evento. O código SDK equivalente depende da linguagem específica, mas geralmente é semelhante ao comando CLI.

Os exemplos a seguir descrevem duas maneiras de usar o comando do CLI `create-deployment` para automatizar a implantação de aplicativos.

- Implemente seu aplicativo em uma programação regular adicionando uma camada personalizada com uma única instância à pilha.

  Adicione uma receita Instalar personalizada à camada que cria um trabalho `cron` na instância para executar o comando em uma programação especificada. Para obter um exemplo de como usar uma receita para criar um trabalho `cron`, consulte Execução de trabalhos cron em instâncias do Linux.
- Adicione uma tarefa ao pipeline de integração contínuo que usa o comando do CLI `create-deployment` para implementar o aplicativo.

Como usar ambientes do Chef

AWS OpsWorks O Stacks não suporta ambientes Chef; `node.chef_environment` sempre retorna`_default`.

AWS OpsWorks Referência de camadas de pilhas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Cada instância que o AWS OpsWorks Stacks implanta deve ser membro de pelo menos uma camada, que define a função de uma instância na pilha e controla os detalhes da instalação e configuração da instância, instalação de pacotes, implantação de aplicativos e assim por diante. Para obter mais informações sobre como usar as AWS OpsWorks pilhas para criar e gerenciar camadas, consulteCamadas.

A descrição de cada camada inclui uma lista das receitas integradas que o AWS OpsWorks Stacks executa para cada um dos eventos do ciclo de vida da camada. Essas receitas são armazenadas em https://github.com/aws/opsworks-cookbooks. Observe que as listas incluem somente as receitas que são executadas diretamente pelo AWS OpsWorks Stacks. Essas receitas às vezes executam receitas dependentes, que não estão relacionadas. Para ver a lista completa de receitas de um determinado evento, incluindo as receitas personalizadas e dependentes, leia atentamente a lista de execução do log do Chef do ciclo de vida do evento apropriado.

Tópicos

- HAProxy Referência de camada
- HAProxy AWS OpsWorks Camada de pilhas
- Referência da camada MySQL
- Camada MySQL OpsWorks
- Guia de referência das camadas do servidor de aplicativos
- Layers de servidor de aplicativos
- Referência da camada de ECS Cluster
- Guia de referência da camada personalizada
- Guia de referência de outras camadas
- Outras camadas

HAProxy Referência de camada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

Uma HAProxy camada usa HAProxy—um balanceador de TCP/HTTP carga confiável de alto desempenho— para fornecer serviços de proxy e balanceamento de carga de alta disponibilidade para aplicativos baseados em TCP e HTTP. Ela é especialmente útil para websites que devem fazer o crawling com cargas muito altas e, ao mesmo tempo, exijam persistência ou processamento na camada 7.

HAProxy monitora o tráfego e exibe as estatísticas e a integridade das instâncias associadas em uma página da web. Por padrão, o URI é http://*DNSName*/haproxy? stats, onde *DNSName* é o nome DNS da HAProxy instância.

Short name: lb

Compatibilidade: uma HAProxy camada é compatível com as seguintes camadas: custom, db-master e memcached.

Portas abertas: HAProxy permite acesso público às portas 22 (SSH), 80 (HTTP) e 443 (HTTPS).

Autoassign Elastic IP addresses: opção ativada por padrão

Default EBS volume: não

Grupo de segurança padrão: AWS-OpsWorks-LB-Server

Configuração: Para configurar uma HAProxy camada, você deve especificar o seguinte:

- URI de verificação de saúde (padrão: http://*DNSName*/).
- URI de estatísticas (padrão: http://*DNSName*/haproxy? estatísticas).
- Estatísticas de senhas (opcional).

- Método de verificação de integridade (opcional). Por padrão, HAProxy usa o método HTTP OPTIONS. Você também pode especificar GET ou HEAD.
- Habilitar estatísticas (opcional)
- Portas. Por padrão, o AWS OpsWorks Stacks é configurado HAProxy para lidar com tráfego HTTP e HTTPS. Você pode configurar HAProxy para lidar com apenas um ou outro substituindo o modelo de configuração do Chef,`haproxy.cfg.erb`.

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies
- ebs
- opsworks_ganglia::client
- haproxy

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- agent_version
- haproxy::configure

Deploy recipes:

- deploy::default
- haproxy::configure

Shutdown recipes:

- opsworks_shutdown::default

- haproxy::stop

Instalação:

- AWS OpsWorks O Stacks usa o instalador de pacotes da instância HAProxy para instalar em seus locais padrão.
- Você deve configurar o syslog para direcionar os arquivos de log para um local especificado. Para obter mais informações, consulte HAProxy.

HAProxy AWS OpsWorks Camada de pilhas

> **Note**
>
> Essa camada está disponíveis apenas para o Chef 11 e pilhas anteriores com base em Linux.

A camada AWS OpsWorks Stacks é uma HAProxy camada AWS OpsWorks Stacks que fornece um modelo para instâncias que hospedam um HAProxyservidor — um balanceamento de carga confiável de alto desempenho. TCP/HTTP Uma pequena instância é normalmente suficiente para processar todo o tráfego do servidor dos aplicativos.

> **Note**
>
> As pilhas estão limitadas a uma única região. Para distribuir seu aplicativo em várias regiões, é necessário criar uma pilha separada para cada região.

Para criar uma HAProxy camada

1. No painel de navegação, clique em Layers.
2. Na página Layers, clique em Add a Layer ou + Layer. Em Tipo de camada, selecione HAProxy.

A camada tem as seguintes definições de configuração, as quais são opcionais.

HAProxy estatísticas

Se a camada coleta e exibe estatísticas. O valor padrão é Yes.

Statistics URL

A cominho do URL da página estática. O URL completo é http://*DNSNameStatisticsPath*, onde *DNSName* está o nome DNS da instância associada. O *StatisticsPath* valor padrão é /haproxy? stats, que corresponde a algo como: http://ec2-54-245-151-7.us-west-2.compute.amazonaws.com/haproxy?stats.

Statistics user name

O nome de usuário da página de estatísticas, que você deve fornecer para visualizar a página de estatísticas. O valor padrão é “opsworks”.

Statistics password

A senha da página de estatísticas, que você deve fornecer para visualizar a página de estatísticas. O valor padrão é uma string gerada aleatoriamente.

Health check URL

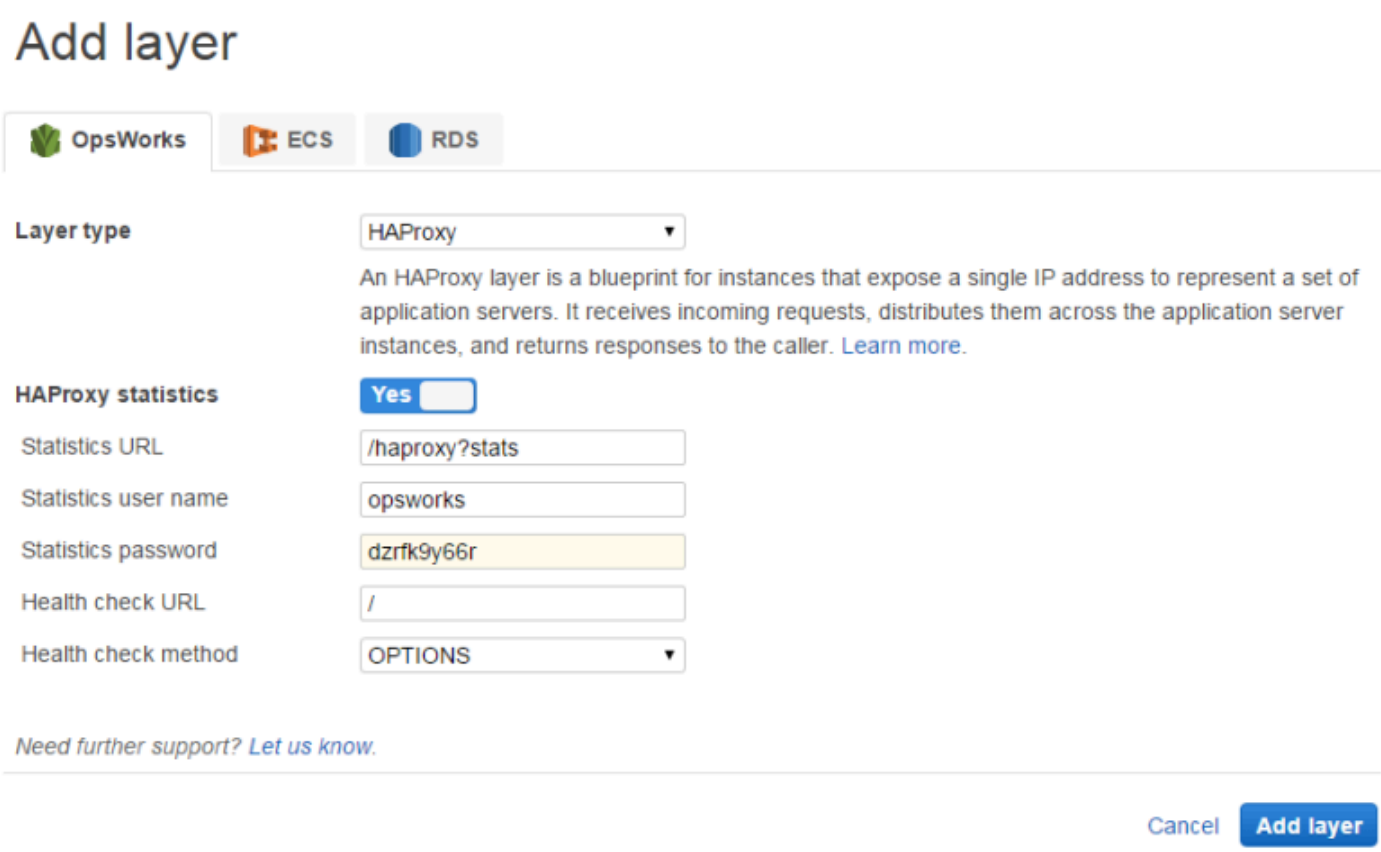
O sufixo do URL da verificação de saúde. HAProxy usa essa URL para chamar periodicamente um método HTTP em cada instância do servidor de aplicativos para determinar se a instância está funcionando. Se a verificação de integridade falhar, HAProxy interrompe o roteamento do tráfego para a instância até que ela seja reiniciada, manualmente ou por meio de recuperação automática. O valor padrão para o sufixo do URL é “/”, que corresponde à página inicial da instância do servidor: http://*DNSName*/.

Health check method

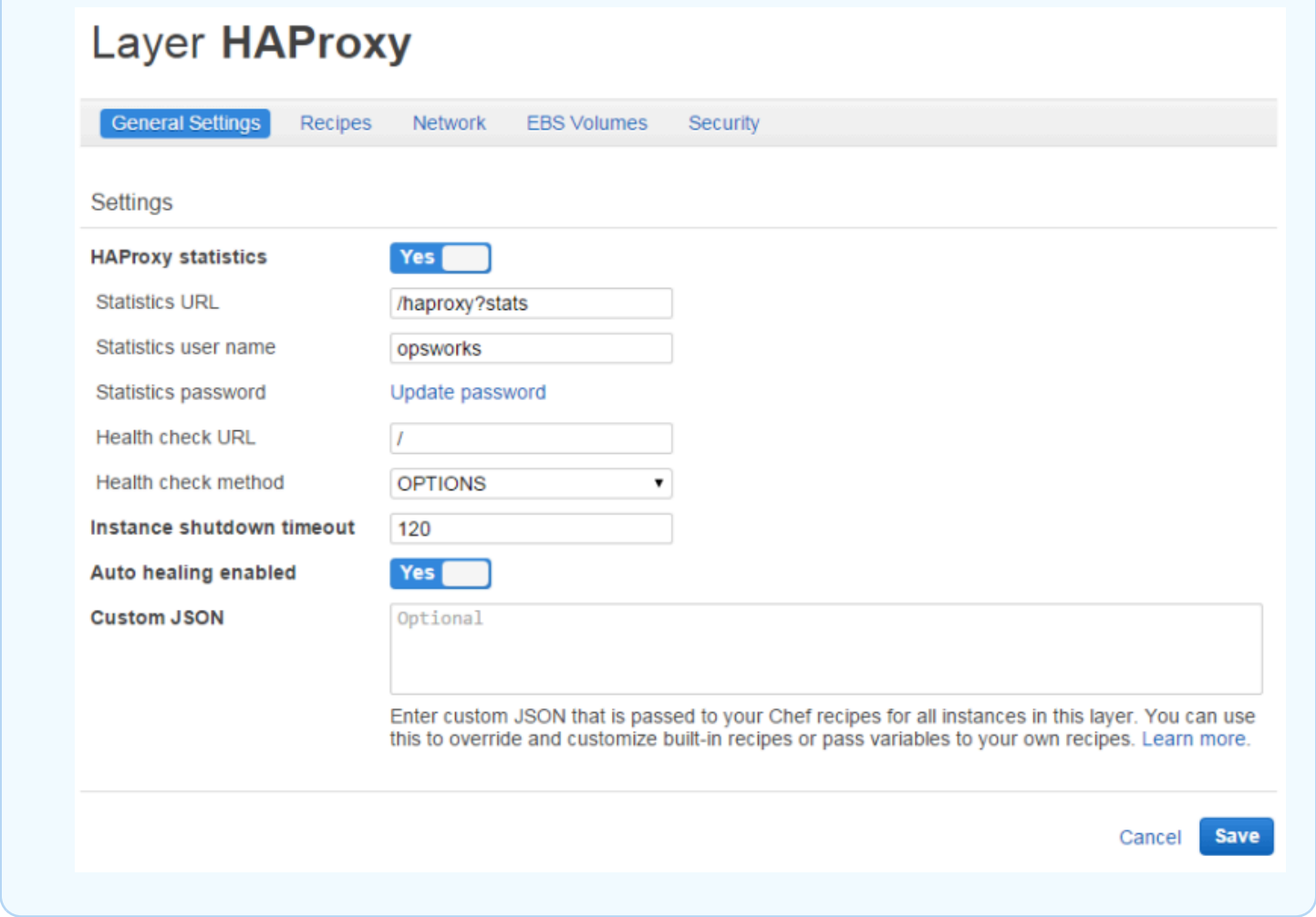
Um método HTTP para ser usado a fim de verificar se as instâncias estão funcionando. O valor padrão é OPTIONS, e você também pode especificar GET ou HEAD. Para obter mais informações, consulte httpchk.

Grupos de segurança personalizados

Essa configuração aparece se você optar por não associar automaticamente um grupo de segurança embutido do AWS OpsWorks Stacks às suas camadas. Você deve especificar qual grupo de segurança associar com a camada. Certifique-se de que o grupo tenha as configurações corretas para permitir o tráfego entre as camadas. Para obter mais informações, consulte Criar uma nova pilha.



> **Note**
>
> Grave a senha para uso posterior; o AWS OpsWorks Stacks não permite que você visualize a senha depois de criar a camada. No entanto, você pode atualizar a senha acessando a página Edit da camada e clicando em Update password na guia General Settings.



Como a HAProxy camada funciona

Por padrão, HAProxy faz o seguinte:

- Detecta solicitações nas portas HTTP e HTTPS.

  Você pode configurar HAProxy para ouvir apenas na porta HTTP ou HTTPS substituindo o modelo de configuração do Chef,`haproxy.cfg.erb`.
- Direciona o tráfego de entrada para instâncias que fazem parte de qualquer camada do servidor do aplicativo.

  Por padrão, o AWS OpsWorks Stacks é configurado HAProxy para distribuir tráfego para instâncias que são membros de qualquer camada do servidor de aplicativos. Você poderia, por exemplo, ter uma pilha com as camadas Rails App Server e PHP App Server, e um HAProxy mestre distribui o tráfego para as instâncias em ambas as camadas. É possível configurar o roteamento padrão utilizando uma receita personalizada.

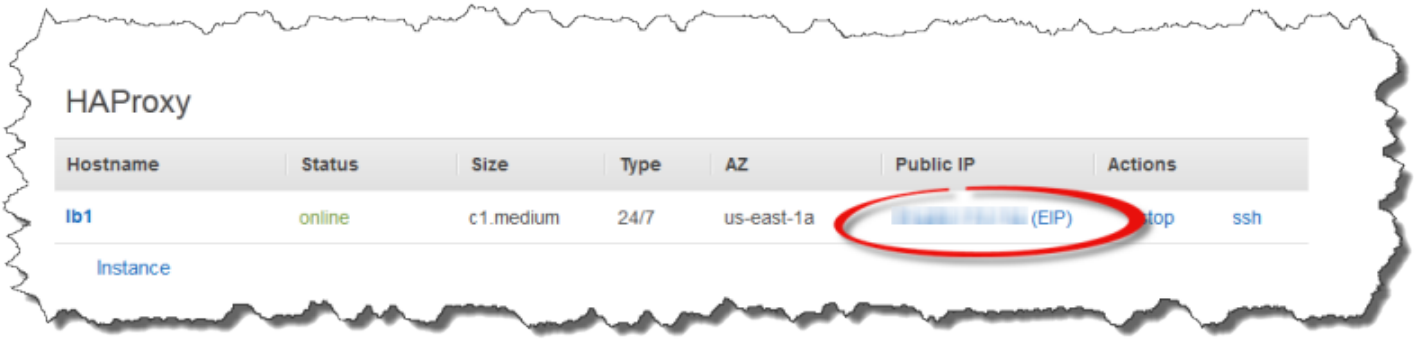
- Direciona o tráfego em diversas Zonas de disponibilidade.

  Se uma Zona de disponibilidade for desativada, o load balancer direciona o tráfego de entrada para instâncias em outras zonas para que o aplicativo continue a ser executado sem interrupção. Por esse motivo, uma prática recomendada é distribuir os servidores de aplicativos em várias Zonas de disponibilidade.
- Executa periodicamente o método de verificação de integridade em cada instância do servidor do aplicativo avaliar sua integridade.

  Se o método não retornar dentro de um período de tempo limite especificado, presume-se que a instância falhou e HAProxy interrompe o roteamento de solicitações para a instância. AWS OpsWorks O Stacks também fornece uma forma de substituir automaticamente as instâncias com falha. Para obter mais informações, consulte Como usar a correção automática. Você pode alterar o método de verificação de integridade ao criar a camada.
- Coleta as estatísticas e, como opção, as exibe em uma página da Web.

> **Important**
>
> Para que a verificação de integridade funcione corretamente com o método OPÇÕES padrão, o aplicativo deve retornar um código de status 2xx ou 3xx.

Por padrão, quando você adiciona uma instância a uma HAProxy camada, o AWS OpsWorks Stacks atribui a ela um endereço IP elástico para representar o aplicativo, que é público para o mundo. Como o endereço IP elástico da instância HAProxy é o único URL exposto publicamente do aplicativo, não é necessário criar e gerenciar nomes de domínio público para as instâncias do servidor de aplicativo subjacente. Você pode obter o endereço na página Instâncias e examinar o endereço IP público da instância, como a ilustração a seguir mostra. Um endereço seguido de (EIP) é um endereço IP elástico. Para obter mais informações sobre endereços IP elásticos, consulte Endereços IP elásticos (EIP).



Quando você interrompe uma HAProxy instância, o AWS OpsWorks Stacks retém o endereço IP elástico e o reatribui à instância quando você a reinicia. Se você excluir uma HAProxy instância, por padrão, o AWS OpsWorks Stacks excluirá o endereço IP da instância. Para manter o endereço desmarque a opção Excluir o IP elástico da instância, como mostrado na ilustração a seguir.

Essa opção afeta o que acontece ao adicionar uma nova instância à camada para substituir uma instância excluída:

- Se você manteve o endereço IP elástico da instância excluída, o AWS OpsWorks Stacks atribui o endereço à nova instância.
- Caso contrário, o AWS OpsWorks Stacks atribuirá um novo endereço IP elástico à instância e você deverá atualizar as configurações do registrador de DNS para mapear para o novo endereço.

Quando as instâncias do servidor de aplicativos ficam on-line ou off-line, manualmente ou como consequência da escalabilidade automática ou da autorrecuperação; a configuração do balanceador de carga deve ser atualizada para direcionar o tráfego até o conjunto atual de instâncias on-line. Essa tarefa é processada automaticamente pelas receitas internas da camada:

- Quando novas instâncias são ativadas, o AWS OpsWorks Stacks aciona um evento de ciclo de vida do Configure. As receitas de configuração integradas da HAProxy camada atualizam a configuração do balanceador de carga para que ela também distribua solicitações para qualquer nova instância do servidor de aplicativos.
- Quando as instâncias ficam off-line ou uma instância falha em uma verificação de saúde, o AWS OpsWorks Stacks também aciona um evento de ciclo de vida do Configure. As receitas de HAProxy configuração atualizam a configuração do balanceador de carga para rotear o tráfego somente para as instâncias on-line restantes.

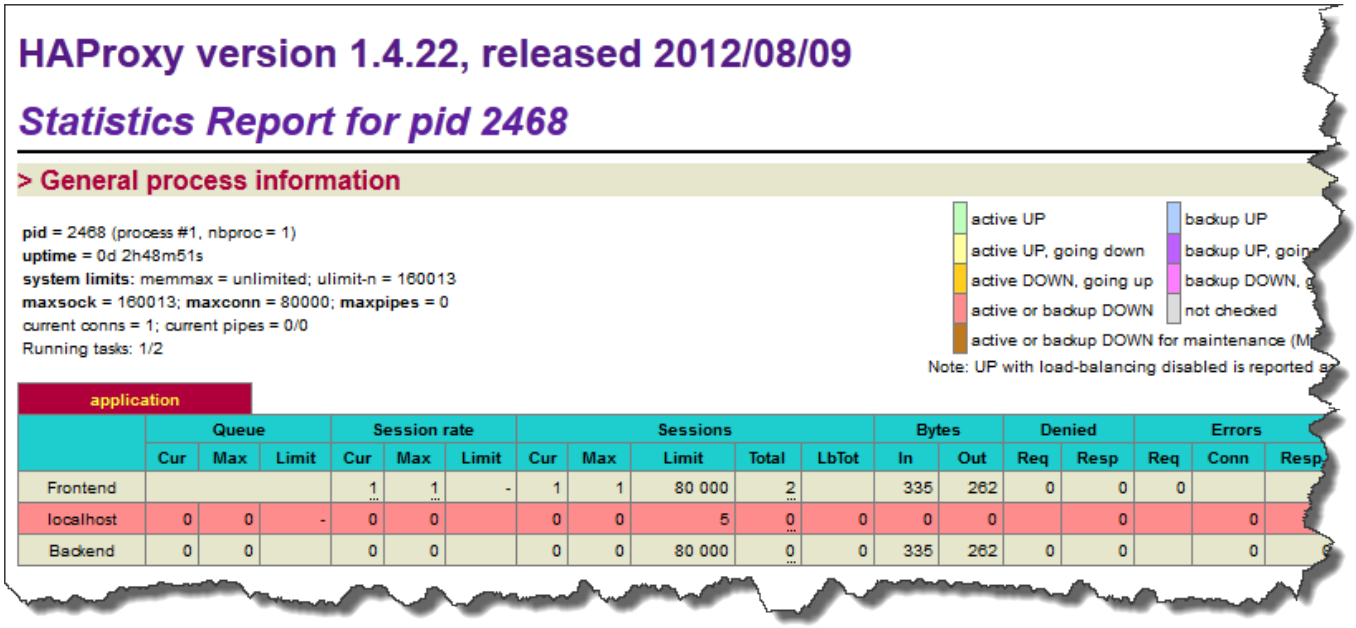
Finalmente, você também pode usar um domínio personalizado com a HAProxy camada. Para obter mais informações, consulte Usando domínios predefinidos.

Página de estatísticas

Se você tiver ativado a página de estatísticas, ela HAProxy exibirá uma página contendo uma variedade de métricas no URL especificado.

Para ver HAProxy estatísticas

1. Obtenha o nome DNS público da HAProxy instância na página Detalhes da instância e copie-o.
2. Na página Camadas, clique em HAProxy para abrir a página de detalhes da camada.
3. Obtenha o URL das estatísticas nos detalhes da camada e anexe-os ao nome DNS público. Por exemplo: **`http://ec2-54-245-102-172.us-west-2.compute.amazonaws.com/haproxy?stats`** para isso.
4. Cole o URL da etapa anterior no navegador e use o nome do usuário e a senha especificada ao criar a camada para abrir a página de estatísticas.



HAProxy version 1.4.22, released 2012/08/09

Statistics Report for pid 2468

> General process information

pid = 2468 (process #1, nbproc = 1)
uptime = 0d 2h48m51s
system limits: memmax = unlimited; ulimit-n = 160013
maxsock = 160013; **maxconn** = 80000; **maxpipes** = 0
current conns = 1; current pipes = 0/0
Running tasks: 1/2

active UP | backup UP
active UP, going down | backup UP, goin
active DOWN, going up | backup DOWN, g
active or backup DOWN | not checked
active or backup DOWN for maintenance (M
Note: UP with load-balancing disabled is reported a

application

	Queue			Session rate			Sessions					Bytes		Denied		Errors		
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Total	LbTot	In	Out	Req	Resp	Req	Conn	Resp
Frontend				1	1	-	1	1	80 000	2		335	262	0	0	0		
localhost	0	0	-	0	0		0	0	5	0	0	0	0		0		0	
Backend	0	0		0	0		0	0	80 000	0	0	335	262	0	0		0	

Referência da camada MySQL

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada MySQL suporta o MySQL, um sistema de gerenciamento de banco de dados relacional amplamente usado. AWS OpsWorks O Stacks instala a versão mais recente disponível, que depende do sistema operacional. Se você adicionar uma instância do MySQL, as informações de acesso necessárias são fornecidas às camadas do servidor de aplicativos. Você deve gravar receitas personalizadas do Chef para instalar configurações mestre – mestre ou mestre – subordinado.

Short name: db-master

Compatibilidade: a camada MySQL é compatível com as seguintes camadas: personalizada, lb, memcached, monitoring-master, nodejs-app, php-app, rails-app e web.

Portas abertas: a camada MySQL permite o acesso público à porta 22 (SSH) e a todas as portas dos servidores da web, servidores personalizados e servidores de aplicativos Rails, PHP e Node.js da pilha.

Autoassign Elastic IP addresses: opção desativada por padrão

Default EBS volume: sim em `/vol/mysql`

Grupo de segurança padrão: AWS-OpsWorks-DB-Master -Server

Configuração: para configurar uma camada MySQL, você deve especificar o seguinte:

- Senha de usuário raiz
- Mecanismo do MySQL

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies
- ebs
- opsworks_ganglia::client
- mysql::server
- dependencies
- deploy::mysql

Configure recipes:

- opsworks_ganglia::configure-client

- ssh_users
- agent_version
- deploy::mysql

Deploy recipes:

- deploy::default
- deploy::mysql

Shutdown recipes:

- opsworks_shutdown::default
- mysql::stop

Instalação:

- AWS OpsWorks O Stacks usa o instalador de pacotes da instância para instalar o MySQL e seus arquivos de log em seus locais padrão. Para obter mais informações, consulte a Documentação do MySQL.

Camada MySQL OpsWorks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Essa camada está disponíveis apenas para o Chef 11 e pilhas anteriores com base em Linux.

Uma OpsWorks camada MySQL fornece um modelo para EC2 instâncias da Amazon que funcionam como um banco de dados mestre do MySQL. Uma receita integrada cria um banco de dados para cada aplicativo implantado a uma camada de servidor do aplicativo. Por exemplo, se você implanta um aplicativo PHP “myapp”, a receita cria um banco de dados “myapp”.

A camada MySQL personalizada possui as seguintes definições de configuração.

MySQL root user password

(Obrigatório) A raiz da senha do usuário.

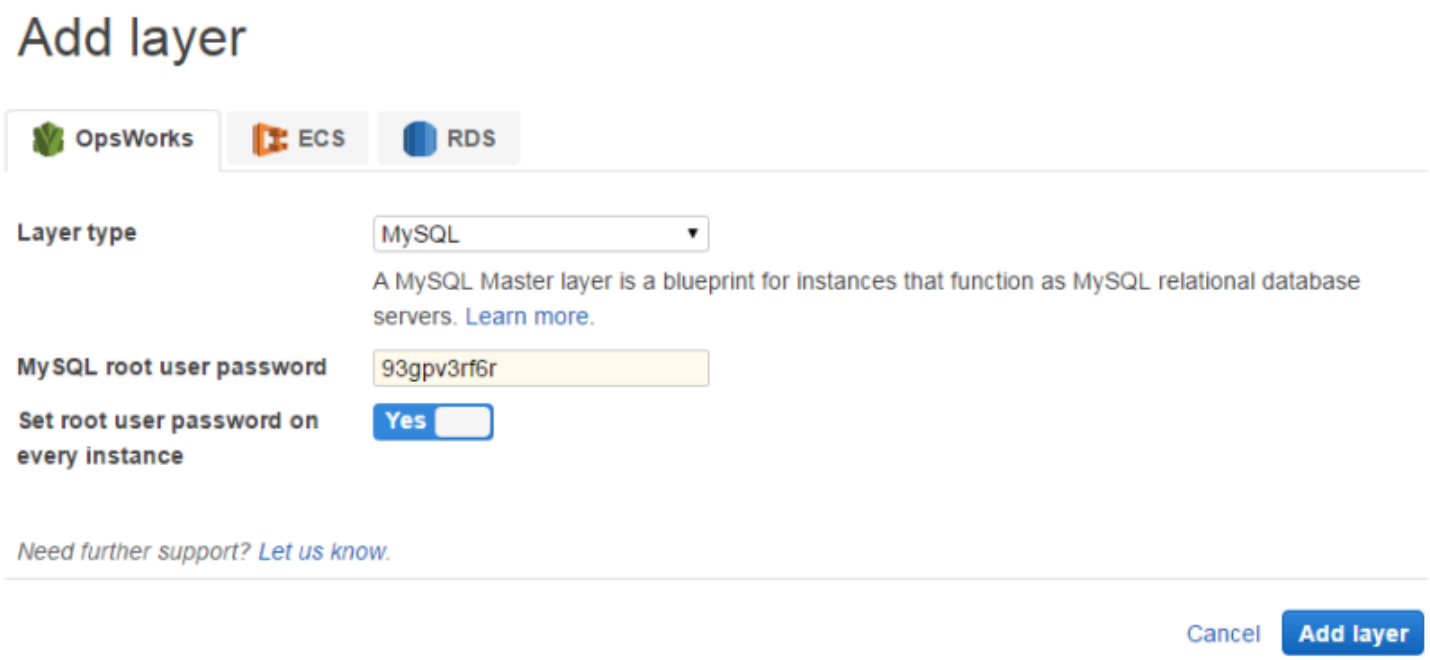
Set root user password on every instance

(Opcional) Se a senha do usuário raiz está incluída na configuração da pilha e os atributos de implantação que estão instalados em cada instância na stack.pilha A configuração padrão é Sim.

Se você definir esse valor como Não, o AWS OpsWorks Stacks passará a senha raiz somente para as instâncias do servidor de aplicativos.

Grupos de segurança personalizados

(Opcional) Um grupo de segurança personalizado a ser associado com a camada. Para obter mais informações, consulte Criar uma nova pilha.



É possível adicionar uma ou mais instâncias à camada, cada uma representando um banco de dados mestre separado do MySQL. Em seguida, anexe uma instância a um aplicativo, que instala as informações de conexão necessárias nos servidores de aplicativos do aplicativo. Em seguida, o aplicativo pode usar as informações de conexão para conectar ao servidor do banco de dados da instância.

Guia de referência das camadas do servidor de aplicativos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks O Stacks oferece suporte a vários aplicativos diferentes e servidores de páginas da web estáticas.

Tópicos

- Referência de camada do AWS Flow (Ruby)
- Referência da camada do Java App Server
- Referência da camada do Node.js App Server
- Referência de camada de servidor de aplicativos PHP
- Referência de camada do Rails App Server
- Referência estática da camada do servidor Web

Referência de camada do AWS Flow (Ruby)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

Um camada AWS Flow (Ruby) fornece um esquema para as instâncias que hospedam as atividades e os operadores de fluxo de trabalho do Amazon Simple Workflow Service.

Nome curto: aws-flow-ruby

Compatibilidade: uma camada do AWS Flow (Ruby) é compatível com PHP App Server, MySQL, Memcached, Ganglia e camadas personalizadas.

Open ports: nenhuma.

Função do IAM: aws-opsworks-ec 2- role-with-swf é a função padrão do AWS Flow (Ruby) que o AWS OpsWorks Stacks cria para você, se solicitada.

Autoassign Elastic IP addresses: opção desativada por padrão

Default EBS Volume: não

Grupo de segurança padrão: AWS-OpsWorks-AWS-Flow -Ruby-Server

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies
- ebs
- opsworks_ganglia::client
- opsworks_aws_flow_ruby::setup

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- mysql::client
- agent_version
- opsworks_aws_flow_ruby::configure

Deploy recipes:

- deploy::default
- implantar: aws-flow-ruby

Undeploy recipes:

- implantar: aws-flow-ruby-undeploy

Shutdown recipes:

- opsworks_shutdown::default

Referência da camada do Java App Server

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada Java App Server fornece suporte ao servidor de aplicativos Apache Tomcat 7.0.

Short name: java-app

Compatibilidade: a camada Java App Server é compatível com as seguintes camadas: personalizada, db-master e memcached.

Portas abertas: uma camada Java App Server permite o acesso público às portas 22 (SSH), 80 (HTTP), 443 (HTTPS), e a todas as portas de balanceadores de carga.

Autoassign Elastic IP addresses: opção desativada por padrão

Default EBS Volume: não

Grupo de segurança padrão: AWS-OpsWorks-Java-App -Server

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies
- ebs
- opsworks_ganglia::client
- opsworks_java::setup

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- agent_version
- opsworks_java::configure

Deploy recipes:

- deploy::default
- deploy::java

Undeploy recipes:

- deploy::java-undeploy

Shutdown recipes:

- opsworks_shutdown::default
- deploy::java-stop

Instalação:

- Tomcat é instalado em `/usr/share/tomcat7`.
- Para obter mais informações sobre como produzir arquivos de log, consulte Registro em log no Tomcat.

Referência da camada do Node.js App Server

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada Node.js App Server fornece suporte ao servidor de aplicativos Node.js, que é uma plataforma para implementação de servidores de aplicativos de redes altamente escaláveis. Os programas são escritos usando o modo assíncrono JavaScript orientado por eventos I/O para minimizar a sobrecarga e maximizar a escalabilidade.

Short name: nodejs-app

Compatibilidade: uma camada Node.js App Server é compatível com as seguintes camadas: personalizada, db-master, memcached e monitoring-master.

Portas abertas: uma camada Node.js App Server permite o acesso público às portas 22 (SSH), 80 (HTTP), 443 (HTTPS), e a todas as portas de balanceadores de carga.

Autoassign Elastic IP addresses: opção desativada por padrão

Default EBS volume: não

Grupo de segurança padrão: AWS-OpsWorks-nodejs-App -Server

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies
- ebs
- opsworks_ganglia::client
- opsworks_nodejs
- opsworks_nodejs::npm

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- agent_version
- opsworks_nodejs::configure

Deploy recipes:

- deploy::default
- opsworks_nodejs
- opsworks_nodejs::npm

- deploy::nodejs

Undeploy recipes:

- deploy::nodejs-undeploy

Shutdown recipes:

- opsworks_shutdown::default
- deploy::nodejs-stop

Instalação:

- Node.js é instalado em `/usr/local/bin/node`.
- Para obter mais informações sobre como criar arquivos de log, consulte Como criar logs no node.js no site Nodejitsu.

Node.js application configuration:

- O arquivo principal executado pelo Node.js deve se chamar `server.js` e residir no diretório raiz do aplicativo implantado.
- O aplicativo Node.js deve ser definido para escutar na porta 80 (ou na porta 443, se for aplicável).

> **Note**
>
> Os aplicativos Node.js que executam Express normalmente usam o código a seguir para definir a porta de escuta, onde `process.env.PORT` representa a porta padrão e é definida com 80:
>
> ```
> app.set('port', process.env.PORT || 3000);
> ```
>
> Com o AWS OpsWorks Stacks, você deve especificar explicitamente a porta 80, da seguinte forma:
>
> ```
> app.set('port', 80);
> ```

Referência de camada de servidor de aplicativos PHP

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada PHP App Server fornece suporte ao servidor de aplicativos PHP usando o Apache2 com mod_php.

Short name: php-app

Compatibilidade: uma camada PHP App Server é compatível com as seguintes camadas: personalizada, db-master, memcached, monitoring-master e rails-app.

Portas abertas: uma camada PHP App Server permite o acesso público às portas 22 (SSH), 80 (HTTP), 443 (HTTPS), e a todas as portas de balanceadores de carga.

Autoassign Elastic IP addresses: opção desativada por padrão

Default EBS volume: não

Grupo de segurança padrão: AWS-OpsWorks-PHP-App -Server

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies

- ebs
- opsworks_ganglia::client
- mysql::client
- dependencies
- mod_php5_apache2

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- agent_version
- mod_php5_apache2::php
- php::configure

Deploy recipes:

- deploy::default
- deploy::php

Undeploy recipes:

- deploy::php-undeploy

Shutdown recipes:

- opsworks_shutdown::default
- apache2::stop

Instalação:

- AWS OpsWorks O Stacks usa o instalador de pacotes da instância para instalar o Apache2, o mod_php e os arquivos de log associados em seus locais padrão. Para obter mais informações sobre a instalação, consulte Apache. Para obter mais informações sobre conexão, consulte Arquivos de log.

Referência de camada do Rails App Server

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada Rails App Server fornece suporte ao servidor de aplicativos Ruby on Rails.

Short name: rails-app

Compatibilidade: uma camada Rails App Server é compatível com as seguintes camadas: personalizada, db-master, memcached, monitoring-master e php-app.

Portas: uma camada Rails App Server permite o acesso público às portas 22 (SSH), 80 (HTTP), 443 (HTTPS), e a todas as portas de balanceadores de carga.

Autoassign Elastic IP addresses: opção desativada por padrão

Default EBS volume: não

Grupo de segurança padrão: AWS-OpsWorks-Rails-App -Server

Configuração: para configurar uma camada Rails App Server, você deve especificar o seguinte:

- Versão do Ruby
- Pilha do Rails
- Versão do Rubygems
- Se deseja instalar e gerenciar o Bundler
- A versão do Bundler

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies
- ebs
- opsworks_ganglia::client
- apache2 apache2::mod_deflate
- passenger_apache2
- passenger_apache2::mod_rails
- passenger_apache2::rails

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- agent_version
- rails::configure

Deploy recipes:

- deploy::default
- deploy::rails

Undeploy recipes:

- deploy::rails-undeploy

Shutdown recipes:

- opsworks_shutdown::default
- apache2::stop

Instalação:

- AWS OpsWorks O Stacks usa o instalador de pacotes da instância para instalar o Apache2 com mod_passenger, mod_rails e os arquivos de log associados em seus locais padrão. Para obter mais informações sobre a instalação, consulte Phusion Passenger. Para obter mais informações sobre conexão, consulte Arquivos de log.

Referência estática da camada do servidor Web

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada Static Web Server serve páginas HTML estáticas, que podem incluir código do lado do cliente, como. JavaScript Ela é baseada no Nginx, que é um HTTP de código aberto, com proxy reverso e servidor de proxy de e-mail.

Short name: web

Compatibilidade: a camada Static Web Server é compatível com as seguintes camadas: personalizada, db-master e memcached.

Portas abertas: uma camada Static Web Server permite o acesso público às portas 22 (SSH), 80 (HTTP), 443 (HTTPS), e a todas as portas de balanceadores de carga.

Autoassign Elastic IP addresses: opção desativada por padrão

Default EBS volume: não

Grupo de segurança padrão: AWS-OpsWorks-Web-Server

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies
- ebs
- opsworks_ganglia::client
- nginx

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- agent_version

Deploy recipes:

- deploy::default
- deploy::web

Undeploy recipes:

- deploy::web-undeploy

Shutdown recipes:

- opsworks_shutdown::default
- nginx::stop

Instalação:

- Nginx é instalado em `/usr/sbin/nginx`.

- Os arquivos de log do Nginx ficam em `/var/log/nginx`.

Layers de servidor de aplicativos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Essas layers estão disponíveis apenas para o Chef 11 e pilhas anteriores com base no Linux.

AWS OpsWorks O Stacks oferece suporte a vários servidores de aplicativos diferentes, nos quais o "aplicativo" inclui páginas da web estáticas. Cada tipo de servidor tem uma camada AWS OpsWorks Stacks separada, com receitas integradas que tratam da instalação do servidor de aplicativos e de quaisquer pacotes relacionados em cada uma das instâncias da camada, da implantação de aplicativos e assim por diante. Por exemplo, a camada Java App Server instala vários pacotes, incluindo Apache Tomcat e OpenJDK, e implanta aplicativos Java para cada uma das instâncias da camada.

O procedimento básico para usar layers de servidor de aplicativo é o seguinte:

1. Crie um dos tipos de layer App Server disponíveis.
2. Adicione uma ou mais instâncias à layer.
3. Crie aplicativos e implante-os nas instâncias. Para obter mais informações, consulte Apps.
4. (Opcional) Se a layer tem várias instâncias, você pode adicionar um load balancer para distribuir o tráfego de entrada entre as instâncias. Para obter mais informações, consulte HAProxy AWS OpsWorks Camada de pilhas.

Tópicos

- Camada AWS Flow (Ruby)
- Camada de AWS OpsWorks pilhas do servidor de aplicativos Java
- Camada de AWS OpsWorks pilhas do servidor de aplicativos Node.js
- Camada de AWS OpsWorks pilhas do servidor de aplicativos PHP
- Camada de pilhas do servidor de aplicativos AWS OpsWorks Rails
- Camada estática de AWS OpsWorks pilhas de servidores Web

Camada AWS Flow (Ruby)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

Uma camada do AWS Flow (Ruby) é uma camada AWS OpsWorks Stacks que fornece um modelo para instâncias que hospedam atividades e trabalhadores de fluxo de trabalho do Amazon SWF. Os operadores são implementados usando o AWS Flow Framework para Ruby, que é uma estrutura de programação que simplifica o processo de implantação de um aplicativo assíncrono distribuído ao mesmo tempo que oferece todos os benefícios do Amazon SWF. É ideal para a implantação de aplicativos que atendam a uma ampla gama de cenários, incluindo processos de negócios, codificação de mídia, tarefas de longa duração e processamento em segundo plano.

A camada AWS Flow (Ruby) inclui as seguintes configurações.

RubyGems versão

A versão Gem da estrutura.

Versão do Bundler

A versão do Bundler.

EC2 Perfil da instância

Um perfil de EC2 instância da Amazon definido pelo usuário para ser usado pelas instâncias da camada. Esse perfil deve conceder permissões para os aplicativos em execução nas instâncias da camada para acessar o Amazon SWF.

Se sua conta não tiver um perfil adequado, você pode selecionar Novo perfil com acesso SWF para que o AWS OpsWorks Stacks atualize o perfil ou você mesmo pode atualizá-lo usando o console do IAM. Em seguida, você pode usar o perfil atualizado para todos as layers subsequentes do AWS Flow. A seguir, uma breve descrição de como criar o perfil usando o console do IAM. Para obter mais informações, consulte Gerenciamento de identidade e acesso no Amazon Simple Workflow Service.

Criação de um perfil para instâncias do AWS Flow (Ruby)

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. Escolha Políticas no painel de navegação e escolha Criar política para criar uma nova política gerenciada pelo cliente.
3. Para Serviço, escolha SWF.
4. Em Ações, escolha todas as ações SWF (swf:*).
5. Para Nome do recurso da Amazon (ARN), insira o ARN que especifica quais domínios do Amazon SWF os trabalhadores podem acessar. Escolha **`All resources`** para fornecer acesso a todos os domínios.
6. Escolha Próximo.
7. Opcionalmente, insira uma tag para identificar a política.
8. Escolha Próximo.
9. Quando terminar, escolha Criar política.
10. Escolha Perfis no painel de navegação Criar perfil.
11. Especifique o nome do perfil e escolha Próxima etapa. Você não pode alterar o nome após criar a função.
12. Escolha o serviço da AWS e, em seguida, escolha EC2.
13. Escolha Próximo.

14. Na lista de Políticas de permissões, escolha a política que você criou anteriormente.
15. Escolha Próximo.
16. Forneça um nome de função e escolha Create role (Criar função). Você não pode alterar o nome após criar a função.
17. Especifique esse perfil ao criar uma camada do AWS Flow (Ruby) em AWS OpsWorks Stacks.

Camada de AWS OpsWorks pilhas do servidor de aplicativos Java

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada Java App Server é uma camada AWS OpsWorks Stacks que fornece um esquema para instâncias que funcionam como servidores de aplicativos Java. Essa camada é baseada no Apache Tomcat 7.0 e no Open JDK 7. AWS OpsWorks O Stacks também instala a biblioteca de conectores Java, que permite que aplicativos Java usem um `DataSource` objeto JDBC para se conectar a um armazenamento de dados de back-end.

Instalação: o Tomcat é instalado em `/usr/share/tomcat7`.

A página Add Layer oferece as seguintes opções de configuração:

Opções de Java VM

Você pode usar essa configuração para especificar as opções de Java VM personalizadas. Não há opções padrão. Por exemplo, um conjunto comum de opções é `-Djava.awt.headless=true -Xmx128m -XX:+UseConcMarkSweepGC`. Se você usa Java VM Options, certifique-se de passar um conjunto válido de opções; AWS OpsWorks Stacks não

valida a string. Se você tentar passar uma opção inválida, a inicialização do servidor Tomcat normalmente falhará, o que causará a falha da configuração. Se isso acontecer, você poderá examinar a configuração do log do Chef da instância para obter mais detalhes. Para obter mais informações sobre como visualizar e interpretar os logs do Chef, consulte Logs do Chef.

Os grupos de segurança personalizados

Essa configuração aparece se você optar por não associar automaticamente um grupo de segurança embutido do AWS OpsWorks Stacks às suas camadas. Você deve especificar qual grupo de segurança associar com a camada. Para obter mais informações, consulte Criar uma nova pilha.

Elastic Load Balancer

Você pode anexar um balanceador de carga do Elastic Load Balancing às instâncias de camada. Para obter mais informações, consulte Camada Elastic Load Balancing.

Você pode especificar outras configurações usando atributos personalizados JSON ou um arquivo de atributos personalizado. Para obter mais informações, consulte Configuração personalizada.

> **Important**
>
> Se seu aplicativo Java usa SSL, recomendamos que você desative, SSLv3 se possível, para resolver as vulnerabilidades descritas em CVE-2014-3566. Para obter mais informações, consulte Desativação SSLv3 para servidores Apache.

Tópicos

- Desativação SSLv3 para servidores Apache
- Configuração personalizada
- Implantar aplicativos Java

Desativação SSLv3 para servidores Apache

Para desativar SSLv3, você deve modificar a `SSLProtocol` configuração do `ssl.conf` arquivo do servidor Apache. Para fazer isso, você deve substituir o modelo de arquivo integrado do livro de receita do apache2 `ssl.conf.erb`, que a configuração de receitas da camada Java App Server usa para criar o `ssl.conf`. Os detalhes dependem de qual sistema operacional você especifica para as

instâncias da layer. O seguinte resume as modificações necessárias para os sistemas Amazon Linux e Ubuntu. SSLv3 é automaticamente desativado para sistemas Red Hat Enterprise Linux (RHEL). Para obter mais informações sobre como substituir um modelo integrado, consulte Uso de modelos personalizados.

Amazon Linux

O arquivo `ssl.conf.erb` para esses sistemas operacionais está no diretório `apache2` do livro de receitas `apache2/templates/default/mods`. A tabela a seguir mostra a parte relevante do arquivo integrado.

```
...
#SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL

# enable only secure protocols: SSLv3 and TLSv1.2, but not SSLv2
SSLProtocol all -SSLv2
</IfModule>
```

Substitua `ssl.conf.erb` e modifique a configuração `SSLProtocol` da seguinte forma.

```
...
#SSLCipherSuite ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL

# enable only secure protocols: SSLv3 and TLSv1.2, but not SSLv2
SSLProtocol all -SSLv3 -SSLv2
</IfModule>
```

Ubuntu 14.04 LTS

O arquivo `ssl.conf.erb` para esse sistema operacional está no diretório `apache2` do livro de receitas `apache2/templates/ubuntu-14.04/mods`. A tabela a seguir mostra a parte relevante do arquivo integrado.

```
...
# The protocols to enable.
# Available values: all, SSLv3, TLSv1.2
# SSL v2 is no longer supported
SSLProtocol all
```

```
...
```

Altere essa configuração da seguinte forma.

```
...
# The protocols to enable.
# Available values: all, SSLv3, TLSv1.2
# SSL v2 is no longer supported
SSLProtocol all -SSLv3 -SSLv2
...
```

Configuração personalizada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks Stacks expõe configurações adicionais como atributos integrados, que estão todos no namespace. `opsworks_java` Você pode usar um JSON personalizado ou um arquivo de atributos personalizado para substituir os atributos integrados e especificar valores personalizados. Por exemplo, as versões JVM e Tomcat são representadas pelos atributos integrados `jvm_version` e `java_app_server_version`, ambos definidos para 7. Você pode usar um JSON personalizado ou um arquivo de atributos personalizado para definir um ou ambos para 6. O exemplo a seguir usa um JSON personalizado para definir ambos os atributos para 6:

```
{
  "opsworks_java": {
    "jvm_version": 6,
    "java_app_server_version" : 6
  }
}
```

Para obter mais informações, consulte Usar JSON personalizado.

Outro exemplo de configuração personalizada é a instalação de um JDK personalizado substituindo os atributos `use_custom_pkg_location`, `custom_pkg_location_url_debiane` `custom_pkg_location_url_rhel`.

> **Note**
>
> Se você substituir os livros de receita integrados, precisará atualizar os componentes.

Para obter mais informações sobre atributos e como substitui-los, consulte Sobrepor atributos. Para obter uma lista de atributos integrados, consulte Atributos opsworks_java.

Implantar aplicativos Java

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Os tópicos a seguir descrevem como implantar aplicativos em instâncias de camada Java App Server. Os exemplos são para aplicativos JSP, mas você pode usar o mesmo procedimento para instalar outros tipos de aplicativo Java.

Você pode implantar páginas JSP de qualquer um dos repositórios suportados. Se você quiser implantar arquivos WAR, observe que o AWS OpsWorks Stacks extrai automaticamente arquivos WAR que são implantados de um arquivo Amazon S3 ou HTTP, mas não de um repositório Git ou Subversion. Se você quer usar Git ou Subversion para arquivos WAR, pode executar uma das seguintes ações:

- Armazenar o arquivo extraído no repositório.
- Armazenar o arquivo WAR no repositório e usar um hook de implantação do Chef para extrair o arquivo, conforme descrito no exemplo a seguir.

Você pode usar ganchos de implantação do Chef para executar aplicativos Ruby fornecidos pelo usuário em uma instância em qualquer um dos quatro estágios de implantação. O nome do aplicativo determina o estágio. Veja a seguir um exemplo de um aplicativo Ruby chamado `before_migrate.rb`, que extrai um arquivo WAR implantado a partir de um repositório Git ou Subversion. O nome associa o aplicativo com o hook de implantação do Checkout para que ele seja executado no início da operação de implantação, após a verificação do código, mas antes da migração. Para obter mais informações sobre como usar esse exemplo, consulte Uso de hooks de implantação do Chef.

```
::Dir.glob(::File.join(release_path, '*.war')) do |archive_file|
  execute "unzip_#{archive_file}" do
    command "unzip #{archive_file}"
    cwd release_path
  end
end
```

> **Note**
>
> Quando você implanta uma atualização em um aplicativo JSP, o Tomcat pode não reconhecer a atualização e, em vez disso, continuar a executar a versão do aplicativo existente. Isso pode acontecer, por exemplo, se você implantar seu aplicativo como um arquivo.zip que contém apenas uma página JSP. Para garantir que o Tomcat execute a última versão implementada, o diretório raiz do projeto deve incluir um diretório WEB-INF que contenha um arquivo `web.xml`. Um arquivo `web.xml` pode conter uma variedade de conteúdos, mas o conteúdo a seguir é suficiente para garantir que o Tomcat reconheça as atualizações e execute a última versão implementada do aplicativo. Você não precisa alterar a versão para cada atualização. O Tomcat reconhecerá a atualização mesmo se a versão não tiver sido alterada.
>
> ```
> <context-param>
>  <param-name>appVersion</param-name>
>  <param-value>0.1</param-value>
> </context-param>
> ```

Tópicos

- implantar um aplicativo JSP
- A implantação de um JSP App com um banco de dados de back-end

implantar um aplicativo JSP

Para implantar um aplicativo JSP, especifique o nome e as informações do repositório. Você também pode especificar, opcionalmente, os domínios e as configurações de SSL. Para obter mais informações sobre como criar um aplicativo, consulte Adição de aplicativos. O procedimento a seguir mostra como criar e implantar uma simples página JSP a partir de um arquivo do Amazon S3 público. Para obter informações sobre como usar outros tipos de repositório, incluindo arquivos do Amazon S3 privados, consulte Origem do aplicativo.

O exemplo a seguir mostra a página JSP, que exibe algumas informações do sistema.

```
<%@ page import="java.net.InetAddress" %>
<html>
<body>
<%
    java.util.Date date = new java.util.Date();
    InetAddress inetAddress = InetAddress.getLocalHost();
%>
The time is
<%
    out.println( date );
    out.println("<br>Your server's hostname is "+inetAddress.getHostName());
%>
<br>
</body>
</html>
```

> **Note**
>
> O procedimento a seguir pressupõe que você já está familiarizado com os conceitos básicos de criação de pilhas, a adição de instâncias em layers e assim por diante. Se você é novo no AWS OpsWorks Stacks, você deve ver Conceitos básicos das pilhas Linux do Chef 11 primeiro.

Para implantar uma página JSP a partir de um arquivo do Amazon S3

1. Crie uma pilha com uma camada Java App Server, adicione uma instância 24/7 à camada e inicie-a.
2. Copie o código em um arquivo chamado `simplejsp.jsp`, coloque o arquivo em uma pasta chamada `simplejsp` e crie um arquivo `.zip` da pasta. Os nomes são arbitrários; você pode usar os nomes que quiser nos arquivos ou pastas. Você também pode usar outros tipos de arquivo, incluindo arquivo gzip, bzip2, tarball ou Java WAR. Observe que o AWS OpsWorks Stacks não suporta arquivos tar não compactados. Para implantar múltiplas páginas JSP, inclua-as no mesmo arquivo.
3. Carregue o arquivo para um bucket do Amazon S3 e torne o arquivo público. Copie o URL do arquivo para uso posterior. Para obter mais informações sobre como criar buckets e fazer upload de arquivos, consulte Comece a usar o Amazon Simple Storage Service.
4. Adicione um aplicativo na pilha e especifique as seguintes configurações:

   - Nome: `SimpleJSP`
   - App type – `Java`
   - Tipo de repositório – `Http Archive`
   - URL do repositório: o URL do Amazon S3 do arquivo compactado.

   Use os valores padrão para as configurações restantes e clique em Add App para criar o aplicativo.
5. Implante o aplicativo na instância Java App Server.

Agora você pode acessar o URL do aplicativo e visualizá-lo. Se você não tiver especificado um domínio, pode construir um URL usando o endereço IP público da instância ou o nome DNS público. Para obter o endereço IP público ou o nome DNS público de uma instância, acesse o console AWS OpsWorks Stacks e clique no nome da instância na página Instâncias para abrir a página de detalhes.

O restante do URL depende do nome curto do aplicativo, que é um nome em minúsculas que o AWS OpsWorks Stacks gera a partir do nome do aplicativo que você especificou ao criar o aplicativo. Por exemplo, o nome curto do SimpleJSP é simplejsp. Você pode obter um nome curto de aplicativo de na página de detalhes.

- Se o nome curto é `root`, você pode usar `http://public_DNS/appname.jsp` ou `http://public_IP/appname.jsp`.
- Caso contrário, você pode usar `http://public_DNS/app_shortname/appname.jsp` ou `http://public_IP/app_shortname/appname.jsp`.

Se você especificou um domínio para o aplicativo, o URL é `http://domain/appname.jsp`.

O URL para o exemplo seria algo similar a `http://192.0.2.0/simplejsp/simplejsp.jsp`.

Se você quer implantar vários aplicativos para a mesma instância, não deve usar `root` como nome curto. Isso pode causar conflitos de URL que impedem o bom funcionamento do aplicativo. Em vez disso, atribua um nome de domínio diferente para cada aplicativo.

A implantação de um JSP App com um banco de dados de back-end

As páginas JSP podem usar um objeto JDBC `DataSource` para se conectar a um banco de dados de back-end. Você pode criar e implantar um aplicativo usando o procedimento da seção anterior, com uma etapa adicional para configurar a conexão.

A página JSP seguinte mostra como se conectar a um objeto `DataSource`.

```
<html>
  <head>
    <title>DB Access</title>
  </head>
  <body>
    <%@ page language="java" import="java.sql.*,javax.naming.*,javax.sql.*" %>
    <%
      StringBuffer output = new StringBuffer();
      DataSource ds = null;
      Connection con = null;
      Statement stmt = null;
      ResultSet rs = null;
      try {
        Context initCtx = new InitialContext();
        ds = (DataSource) initCtx.lookup("java:comp/env/jdbc/mydb");
        con = ds.getConnection();
        output.append("Databases found:<br>");
        stmt = con.createStatement();
        rs = stmt.executeQuery("show databases");
```

```
        while (rs.next()) {
          output.append(rs.getString(1));
          output.append("<br>");
        }
      }
      catch (Exception e) {
        output.append("Exception: ");
        output.append(e.getMessage());
        output.append("<br>");
      }
      finally {
        try {
          if (rs != null) {
            rs.close();
          }
          if (stmt != null) {
            stmt.close();
          }
          if (con != null) {
            con.close();
          }
        }
        catch (Exception e) {
          output.append("Exception (during close of connection): ");
          output.append(e.getMessage());
          output.append("<br>");
        }
      }
    %>
    <%= output.toString() %>
  </body>
</html>
```

AWS OpsWorks O Stacks cria e inicializa o `DataSource` objeto, o vincula a um nome lógico e registra o nome em um serviço de nomenclatura Java Naming and Directory Interface (JNDI). O nome lógico completo é `java:comp/env/`*`user-assigned-name`*. Você deve especificar a parte do nome atribuída ao usuário adicionando os atributos de JSON personalizado aos atributos de configuração e implantação da pilha para definir o atributo `['opsworks_java']['datasources']`, conforme descrito a seguir.

Para implantar uma página JSP que se conecta a um banco de dados MySQL

1. Crie uma pilha com uma camada Java App Server, adicione uma instância ininterrupta em cada camada e inicie-a.
2. Adicione uma layer de banco de dados na pilha. Os detalhes dependem de qual banco de dados você usa.

   Para usar uma instância do MySQL para o exemplo, adicione uma camada MySQL à pilha, adicione uma instância 24/7 na camada e inicie-a.

   Para usar uma instância Amazon RDS (MySQL) para o exemplo:

   - Especifique um mecanismo de banco de dados MySQL para a instância.
   - Atribua os grupos de segurança AWS- OpsWorks -DB-Master-Server () ***security_group_id*** e AWS- OpsWorks -Java-App-Server () à instância. *security_group_id* AWS OpsWorks O Stacks cria esses grupos de segurança para você quando você cria sua primeira pilha na região.
   - Crie um banco de dados chamado `simplejspdb`.
   - Verifique se nome de usuário e a senha principais não contêm & ou outros caracteres que possam causar um erro no Tomcat.

     Especificamente durante o startup, o Tomcat deve analisar o arquivo de contexto do aplicativo da Web, um arquivo XML que inclui o nome de usuário e a senha principais. Se a string incluir um caractere &, o analisador XML o tratará como uma entidade XML malformada e lançará uma exceção de análise, que impedirá a inicialização do Tomcat. Para obter mais informações sobre o arquivo de contexto do aplicativo da Web, consulte tomcat::context.
   - Adicione um driver MySQL na camada Java App Server.
   - Registre a instância RDS com sua pilha.

   Para obter mais informações sobre como usar instâncias do Amazon RDS com AWS OpsWorks Stacks, consulte. Camada de serviços do Amazon RDS
3. Copie o código de exemplo em um arquivo chamado `simplejspdb.jsp`, coloque o arquivo em uma pasta chamada `simplejspdb` e crie um arquivo `.zip` da pasta. Os nomes são arbitrários; você pode usar os nomes que quiser nos arquivos ou pastas. Você também pode usar outros tipos de arquivo, incluindo gzip, bzip2 ou tarball. Para implantar múltiplas páginas JSP, inclua-

as no mesmo arquivo. Para obter mais informações sobre como implantar aplicativos de outros tipos de repositório, consulte Origem do aplicativo.

4. Carregue o arquivo para um bucket do Amazon S3 e torne o arquivo público. Copie o URL do arquivo para uso posterior. Para obter mais informações sobre como criar buckets e fazer upload de arquivos, consulte Comece a usar o Amazon Simple Storage Service.
5. Adicione um aplicativo na pilha e especifique as seguintes configurações:

   - Nome: `SimpleJSPDB`
   - App type – `Java`
   - Tipo de fonte de dados — OpsWorks(para uma instância do MySQL) ou RDS (para uma instância do Amazon RDS).
   - Instância de banco de dados — A instância do MySQL que você criou anteriormente, que normalmente é chamada de db-master1 (mysql), ou a instância do Amazon RDS, que será nomeada (mysql). ***DB_instance_name***
   - Database name – `simplejspdb`.
   - Tipo de repositório – `Http Archive`
   - URL do repositório: o URL do Amazon S3 do arquivo compactado.

   Use os valores padrão para as configurações restantes e clique em Add App para criar o aplicativo.
6. Adicione os seguintes atributos de JSON personalizado aos atributos da configuração da pilha, em que simplejspdb é o nome curto do aplicativo.

```
{
  "opsworks_java": {
    "datasources": {
      "simplejspdb": "jdbc/mydb"
    }
  }
}
```

   AWS OpsWorks O Stacks usa esse mapeamento para gerar um arquivo de contexto com as informações necessárias do banco de dados.

Para obter mais informações sobre como adicionar atributos de JSON personalizado aos atributos de configuração da pilha, consulte Usar JSON personalizado.

7. Implante o aplicativo na instância Java App Server.

Agora você pode usar o URL do aplicativo para visualizá-lo. Para obter uma descrição de como construir o URL, consulte implantar um aplicativo JSP.

O URL para o exemplo seria algo similar a `http://192.0.2.0/simplejspdb/simplejspdb.jsp`.

> **Note**
>
> O atributo `datasources` pode conter vários atributos. Cada atributo é nomeado com um nome curto de aplicativo e definido para a parte apropriada do nome atribuída ao usuário de um nome lógico. Se você tiver vários aplicativos, pode usar nomes lógicos individuais, o que exige um JSON personalizado parecido com o seguinte.
>
> ```
> {
>  "opsworks_java": {
>  "datasources": {
>  "myjavaapp": "jdbc/myappdb",
>  "simplejsp": "jdbc/myjspdb",
>  ...
>  }
>  }
> }
> ```

Camada de AWS OpsWorks pilhas do servidor de aplicativos Node.js

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada do servidor de aplicativos Node.js é uma camada de AWS OpsWorks pilhas que fornece um plano para instâncias que funcionam como servidores de aplicativos Node.js. AWS OpsWorks O Stacks também instala o Express, então as instâncias da camada suportam aplicativos padrão e Express.

Instalação: o Node.js é instalado em `/usr/local/bin/node`.

A página Add Layer oferece as seguintes opções de configuração:

Versão do Node.js

Para ver uma lista das versões suportadas no momento, consulte AWS OpsWorks Sistemas operacionais de pilha.

Os grupos de segurança personalizados

Essa configuração aparece se você optar por não associar automaticamente um grupo de segurança embutido do AWS OpsWorks Stacks às suas camadas. Você deve especificar qual grupo de segurança associar com a camada. Para obter mais informações, consulte Criar uma nova pilha.

Elastic Load Balancer

Você pode anexar um balanceador de carga do Elastic Load Balancing às instâncias de camada.

> **Important**
>
> Se seu aplicativo Node.js usa SSL, recomendamos que você desative, SSLv3 se possível, para resolver as vulnerabilidades descritas em CVE-2015-8027. Para fazer isso, você deve definir Node.js version como `0.12.9`.

Implantar aplicativos Node.js

Para obter uma descrição detalhada de como implementar um aplicativo Node.js simples no AWS OpsWorks Stacks e implantá-lo em uma stack, consulte Criação da sua primeira stack do Node.js. Em geral, os aplicativos Node.js para o AWS OpsWorks Stacks devem atender às seguintes condições:

- O arquivo principal deve ser chamado `server.js` e residir no diretório raiz do aplicativo implantado.
- Os aplicativos Express devem incluir um arquivo `package.json` no diretório raiz do aplicativo.
- Por padrão, o aplicativo deve escutar na porta 80 (HTTP) ou na porta 443 (HTTPS).

  É possível escutar em outras portas, mas o grupo de segurança integrado da camada Node.js App Server, AWS- OpsWorks -NodeJS-App-Server, permite tráfego de entrada de usuários somente para as portas 80, 443 e 22 (SSH). Para permitir o tráfego de entrada de usuário em outras portas, crie um grupo de segurança com regras de entrada apropriadas e atribua-o à camada do Node.js App Server. Não modifique as regras de entrada editando o grupo de segurança integrado. Cada vez que você cria uma pilha, o AWS OpsWorks Stacks substitui os grupos de segurança integrados pelas configurações padrão, portanto, todas as alterações feitas serão perdidas.

> **Note**
>
> AWS OpsWorks Stacks define a variável de ambiente PORT como 80 (padrão) ou 443 (se você habilitar o SSL), para que você possa usar o código a seguir para escutar as solicitações.
>
> ```
> app.listen(process.env.PORT);
> ```

Se você configurar um aplicativo Node.js para oferecer suporte a SSL, deverá especificar a chave e os certificados. AWS OpsWorks O Stacks coloca os dados de cada instância do servidor de aplicativos como arquivos separados no `/srv/www/`*`app_shortname`*`/shared/config` diretório, da seguinte maneira.

- `ssl.crt`: o certificado SSL.
- `ssl.key`: a chave SSL.

- `ssl.ca`: o certificado da cadeia, se você tiver especificado um.

O aplicativo pode obter a chave SSL e os certificados a partir desses arquivos.

Camada de AWS OpsWorks pilhas do servidor de aplicativos PHP

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada PHP App Server é uma camada AWS OpsWorks Stacks que fornece um modelo para instâncias que funcionam como servidores de aplicativos PHP. A camada PHP App Server é baseada no Apache2 com `mod_php` e não tem opções de configuração padrão. As versões PHP e Apache dependem de qual sistema operacional você especifica para as instâncias da layer.

Sistema operacional	Versão da PHP	Versão do Apache
Amazon Linux 2018.03	5.3	2.2
Amazon Linux 2017.09	5.3	2.2
Amazon Linux 2017.03	5.3	2.2
Amazon Linux 2016.09	5.3	2.2
Amazon Linux 2016.03	5.3	2.2
Amazon Linux 2015.09	5.3	2.2

Sistema operacional	Versão da PHP	Versão do Apache
Amazon Linux 2015.03	5.3	2.2
Amazon Linux 2014.09	5.3	2.2
Ubuntu 14.04 LTS	5.5	2.4

Instalação: o AWS OpsWorks Stacks usa o instalador de pacotes da instância para instalar o Apache2 e `mod_php` em seus locais padrão. Para obter mais informações sobre a instalação, consulte Apache.

A página Add Layer oferece as seguintes opções de configuração:

Os grupos de segurança personalizados

Essa configuração aparece se você optar por não associar automaticamente um grupo de segurança embutido do AWS OpsWorks Stacks às suas camadas. Você deve especificar qual grupo de segurança associar com a camada. Para obter mais informações, consulte Criar uma nova pilha.

Elastic Load Balancer

Você pode anexar um balanceador de carga do Elastic Load Balancing às instâncias de camada.

Você pode modificar algumas configurações do Apache usando um JSON personalizado ou um arquivo de atributos personalizado. Para obter mais informações, consulte Sobrepor atributos. Para uma lista dos atributos do Apache que podem ser substituídos, consulte Atributos apache2.

Para obter um exemplo de como implantar um aplicativo PHP, incluindo como conectar o aplicativo a um banco de dados de back-end, consulte Conceitos básicos das pilhas Linux do Chef 11.

> **Important**
>
> Se seu aplicativo PHP usa SSL, recomendamos que você desative, SSLv3 se possível, para resolver as vulnerabilidades descritas em CVE-2014-3566. Para fazer isso, você deve modificar a `SSLProtocol` configuração no arquivo `ssl.conf` do servidor Apache. Para obter mais informações sobre como modificar essa configuração, consulte Desativação SSLv3 para servidores Apache.

Camada de pilhas do servidor de aplicativos AWS OpsWorks Rails

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada Rails App Server é uma camada AWS OpsWorks Stacks que fornece um plano para instâncias que funcionam como servidores de aplicativos Rails.

Instalação: o AWS OpsWorks Stacks usa o instalador de pacotes da instância para instalar os pacotes do servidor em seus locais padrão. Para obter mais informações sobre Apache/Passenger instalação, consulte Phusion Passenger. Para obter mais informações sobre conexão, consulte Arquivos de log. Para obter mais informações sobre Nginx/Unicorn instalação, consulte Unicorn.

A página Add Layer fornece as seguintes opções de configuração, todas delas opcionais.

Versão do Ruby

A versão do Ruby que será usada pelos aplicativos. O valor padrão é 2.3.

Você também pode especificar sua versão do Ruby preferida ao substituir o atributo `[:opsworks][:ruby_version]`.

> **Note**
>
> AWS OpsWorks O Stacks instala um pacote Ruby separado para ser usado pelas receitas e pelo agente da instância. Para obter mais informações, consulte Versões do Ruby.

Pilha do Rails

A pilha padrão do Rails é o Apache2 com o Phusion Passenger. Você também pode usar o Nginx com o Unicorn.

> **Note**
>
> Se você usar o Nginx e o Unicorn, deve adicionar o gem unicorn no Gemfile do aplicativo, como no exemplo a seguir:
>
> ```
> source 'https://rubygems.org'
> gem 'rails', '3.2.15'
> ...
> # Use unicorn as the app server
> gem 'unicorn'
> ...
> ```

Versão do Passenger

Se tiver especificado Apache2/Passenger, você deverá especificar a versão do Passenger. O valor padrão é 5.0.28.

Versão do Rubygems

A versão padrão do Rubygems é 2.5.1

Instalar e gerenciar o Bundler

Permite que você opte por instalar e gerenciar o Bundler. O valor padrão é Yes.

Versão do Bundler

A versão padrão do Bundler é 1.12.5.

Os grupos de segurança personalizados

Essa configuração aparece se você optar por não associar automaticamente um grupo de segurança embutido do AWS OpsWorks Stacks às suas camadas. Você deve especificar qual grupo de segurança associar com a camada. Para obter mais informações, consulte Criar uma nova pilha.

Elastic Load Balancer

Você pode anexar um balanceador de carga do Elastic Load Balancing às instâncias de camada.

Você pode modificar algumas configurações usando um JSON personalizado ou um arquivo de atributos personalizado. Para obter mais informações, consulte Sobrepor atributos. Para obter uma lista dos atributos do Apache, Nginx, Passenger e Phusion Unicorn que podem ser substituídos, consulte Atributos de livros de receitas integrados.

> **Important**
>
> Se seu aplicativo Ruby on Rails usa SSL, recomendamos que você desative, SSLv3 se possível, para resolver as vulnerabilidades descritas em CVE-2014-3566. Para obter mais informações, consulte Desabilitando SSLv3 para servidores Rails.

Tópicos

- Desabilitando SSLv3 para servidores Rails
- Conectar-se a um banco de dados
- Implantar aplicativos Ruby on Rails

Desabilitando SSLv3 para servidores Rails

SSLv3 Para desativar os servidores Rails, atualize a configuração da versão Ruby da camada para 2.1 ou superior, o que instala o Ruby 2.1.4 ou superior como a versão que os aplicativos usam.

- Atualize a configuração Ruby Version (Versão do Ruby) para 2.1 ou superior.
- Atualize o arquivo de configuração para a pilha do Rails, como a seguir.

Apache com Phusion Passenger

Atualize a `SSLProtocol` configuração do arquivo do servidor Apache `ssl.conf` do, conforme descrito em Desativação SSLv3 para servidores Apache.

Nginx com Unicorn

Adicione uma diretiva explícita `ssl_protocols` para arquivo do servidor Nginx `nginx.conf`. Para desativar SSLv3, substitua o arquivo de modelo do livro de receitas `nginx.conf.erb`

nginx embutido, que as receitas de configuração da camada Rails App Server usam para `nginx.conf` criar, e adicione a seguinte diretiva:

```
ssl_protocols TLSv1.2;
```

Para obter mais informações sobre como configurar o `nginx.conf`, consulte Configurar servidores HTTPS. Para obter mais informações sobre como substituir um modelo integrado, consulte Uso de modelos personalizados.

Conectar-se a um banco de dados

Quando você implanta um aplicativo, o AWS OpsWorks Stacks cria um novo `database.yml` arquivo usando informações dos `deploy`atributos do aplicativo. Se você anexar uma instância do MySQL ou do Amazon RDS ao aplicativo, o AWS OpsWorks Stacks adiciona as informações de conexão aos `deploy` atributos, para que contenha `database.yml` automaticamente os dados de conexão corretos.

Se um aplicativo não tiver um banco de dados anexado, por padrão, o AWS OpsWorks Stacks não adiciona nenhuma informação de conexão aos `deploy` atributos e não criado`database.yml`. Se você quiser usar um banco de dados diferente, pode usar um JSON personalizado para adicionar atributos de banco de dados aos `deploy` atributos do aplicativo com as informações de conexão. Os atributos estão todos abaixo`["deploy"]["`*`appshortname`*`"]["database"]`, onde *`appshortname`* está o nome curto do aplicativo, que o AWS OpsWorks Stacks gera a partir do nome do aplicativo. Os valores especificados no JSON personalizado substituem as configurações padrão. Para obter mais informações, consulte Adição de aplicativos.

AWS OpsWorks Stacks incorpora os seguintes valores de `[:...][:database]`atributos em. `database.yml` Os atributos necessários dependem do banco de dados específico, mas você deve ter um `host` atributo ou AWS OpsWorks as pilhas não serão criadas`database.yml`.

- `[:adapter] (String)`: o adaptador de banco de dados, como `mysql`.
- `[:database]` (String): o nome do banco de dados.
- `[:encoding]` (String): a codificação, que geralmente é definida como `utf8`.
- `[:host]` (String): o URL do host, como `railsexample.cdlqlk5uwd0k.us-west-2.rds.amazonaws.com`.
- `[:reconnect]` (Boolean): caso o aplicativo deva se reconectar se a conexão não existir mais.

- `[:password]` (String): a senha do banco de dados.
- `[:port]` (telefone). : o número da porta do banco de dados. Use esse atributo para substituir o número de porta padrão, que é definido pelo adaptador.
- `[:username]` (String): o nome de usuário do banco de dados.

O exemplo a seguir mostra um JSON personalizado para um aplicativo cujo nome curto é myapp.

```
{
  "deploy" : {
    "myapp" : {
      "database" : {
        "adapter" : "adapter",
        "database" : "databasename",
        "host" : "host",
        "password" : "password",
        "port" : portnumber
        "reconnect" : true/false,
        "username" : "username"
      }
    }
  }
}
```

Para obter informações sobre como especificar um JSON personalizado, consulte Usar JSON personalizado. Para ver o modelo usado para criar `database.yml` (`database.yml.erb`), acesse o repositório integrado do livro de receita.

Implantar aplicativos Ruby on Rails

Você pode implantar aplicativos Ruby on Rails a partir de qualquer um dos repositórios suportados. A seguir, mostramos como implantar um exemplo de aplicativo Ruby on Rails em um servidor executando uma pilha Apache/Passenger Rails. O código de exemplo é armazenado em um GitHub repositório público, mas o procedimento básico é o mesmo para os outros repositórios compatíveis. Para obter mais informações sobre como criar e implantar aplicativos, consulte Apps. Para ver o código do exemplo, que inclui comentários extensos, acesse https://github.com/awslabs/opsworks-demo-rails-photo-share-app.

Para implantar um aplicativo Ruby on Rails a partir de um repositório GitHub

1. Crie uma pilha com uma camada do Rails App Server Apache/Passenger como pilha Rails, adicione uma instância 24 horas por dia, 7 dias por semana à camada e inicie-a.
2. Depois que a instância estiver online, adicione um aplicativo à pilha e especifique as configurações a seguir:

   - Name – Qualquer nome que você quiser; o exemplo usa `PhotoPoll`.

     AWS OpsWorks O Stacks usa esse nome para fins de exibição e gera um nome curto para uso interno e para identificar o aplicativo na configuração da pilha e nos atributos de implantação. Por exemplo, o nome PhotoPoll curto é photopoll.
   - App type – Ruby on Rails.
   - Rails environment – Os ambientes disponíveis são determinados pelo aplicativo.

     O aplicativo de exemplo tem três: **`development`**, **`test`** e produção**`production`**. Para este exemplo, defina o ambiente como **`development`**. Consulte o código de exemplo para obter descrições de cada ambiente.
   - URL do repositório: qualquer um dos tipos de repositório suportados. Especifique o `Git` para este exemplo
   - Repository URL – O repositório de implantação do código.

     Para este exemplo, defina o URL como **`git://github.com/awslabs/opsworks-demo-rails-photo-share-app`**.

   Use os valores padrão para as configurações restantes e clique em Add App para criar o aplicativo.
3. Implante o aplicativo na instância Rails App Server.
4. Quando a implantação for concluída, vá à página Instâncias e clique no endereço IP público da instância do Rails App Server. Você deve ver o seguinte:

Camada estática de AWS OpsWorks pilhas de servidores Web

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada Static Web Server é uma camada AWS OpsWorks Stacks que fornece um modelo para instâncias servirem páginas HTML estáticas, que podem incluir scripts do lado do cliente. Essa layer é baseada no Nginx.

Instalação: o Nginx é instalado em `/usr/sbin/nginx`.

A página Add Layer oferece as seguintes opções de configuração:

Os grupos de segurança personalizados

Essa configuração aparece se você optar por não associar automaticamente um grupo de segurança embutido do AWS OpsWorks Stacks às suas camadas. Você deve especificar qual grupo de segurança associar com a camada. Para obter mais informações, consulte Criar uma nova pilha.

Elastic Load Balancer

Você pode anexar um balanceador de carga do Elastic Load Balancing às instâncias de camada.

Você pode modificar algumas configurações do Nginx usando um JSON personalizado ou um arquivo de atributos personalizado. Para obter mais informações, consulte Sobrepor atributos. Para uma lista dos atributos do Apache que podem ser substituídos, consulte Atributos nginx.

> **Important**
>
> Se seu aplicativo web usa SSL, recomendamos que você desative, SSLv3 se possível, para resolver as vulnerabilidades descritas em CVE-2014-3566.
>
> Para desativar SSLv3, você deve modificar o arquivo do servidor Nginx. `nginx.conf` Para isso, substitua o modelo de arquivo integrado do livro de receita do nginx `nginx.conf.erb` que a configuração de receitas da camada Rails App Server usa para criar o `nginx.conf` e adicione a seguinte diretiva:
>
> ```
> ssl_protocols TLSv1.2;
> ```
>
> Para obter mais informações sobre como configurar o `nginx.conf`, consulte Configurar servidores HTTPS. Para obter mais informações sobre como substituir um modelo integrado, consulte Uso de modelos personalizados.

Referência da camada de ECS Cluster

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

Uma camada do ECS Cluster representa um cluster do Amazon Elastic Container Service (Amazon ECS) e simplifica o gerenciamento do cluster.

Short name: ecs-cluster

Compatibilidade: uma camada de serviços do Amazon ECS é compatível apenas com camadas personalizadas

Open ports: o ECS Cluster permite o acesso público à porta 22 (SSH)

Autoassign Elastic IP addresses: opção desativada por padrão

Default EBS volume: não

Grupo de segurança padrão: AWS-OpsWorks-ECS-Cluster

Configuração: para configurar uma camada ECS Cluster, você deve especificar o seguinte:

- Se vai atribuir endereços IP públicos ou endereços IP elásticos às instâncias de contêiner
- O perfil da instância para as instâncias de contêiner

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies
- ebs
- opsworks_ganglia::client
- opsworks_ecs::setup

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- mysql::client

- agent_version
- opsworks_ecs::configure

Deploy recipes:

- deploy::default
- opsworks_ecs::deploy

Undeploy recipes:

- opsworks_ecs::undeploy

Shutdown recipes:

- opsworks_shutdown::default
- opsworks_ecs::shutdown

Instalação:

- AWS OpsWorks O Stacks usa o instalador de pacotes da instância para instalar o Docker em seus locais padrão
- O log do evento de instalação do Chef registra se o agente do Amazon ECS foi instalado com êxito. Caso contrário, os registros fornecidos pelo AWS OpsWorks Stacks não incluem informações de registro de erros do Amazon ECS. Para obter mais informações sobre como lidar com erros do Amazon ECS, consulte Solução de problemas do Amazon ECS.

Guia de referência da camada personalizada

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Se as camadas padrão não se adequarem aos seus requisitos, você poderá criar uma camada personalizada. Uma pilha pode ter várias camadas personalizadas. Por padrão, a camada personalizada executa um conjunto limitado de receitas padrão que fornece suporte às funcionalidades básicas. Você implementa a funcionalidade principal da layer implementando um conjunto de receitas personalizadas do Chef em cada um dos eventos de ciclo de vida apropriado para instalar e configurar o software da layer, e assim por diante. As receitas personalizadas seguem as receitas padrão do AWS OpsWorks Stacks para cada evento.

Short name: definido pelo usuário; cada layer personalizada em uma pilha deve ter um nome abreviado diferente

Open ports: por padrão, uma camada de servidor personalizado oferece acesso público às portas 22 (SSH), 80 (HTTP), 443 (HTTPS), e a todas as portas das camadas dos servidores de aplicativos Rails e PHP pilha

Autoassign Elastic IP Addresses: opção desativada por padrão

Default EBS volume: não

Grupo de segurança padrão: AWS-OpsWorks-Custom-Server

Compatibility: as camadas personalizadas são compatíveis com as seguintes camadas: personalizada, db-master, lb, memcached, monitoring-master, nodejs-app, php-app, rails-app e web

Configuration: para configurar uma camada personalizada, você deve especificar o seguinte:

- O nome da camada
- O nome abreviado da camada que identifica a camada nas receitas do Chef, e deve usar apenas caracteres de a-z e números

Para pilhas do Linux, a camada personalizada usa as receitas a seguir.

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client

- dependencies
- ebs
- opsworks_ganglia::client

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- agent_version

Deploy recipes:

- deploy::default

Shutdown recipes:

- opsworks_shutdown::default

Guia de referência de outras camadas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks As pilhas também suportam as seguintes camadas.

Tópicos

- Referência da camada Ganglia
- Referência da camada do Memcached

Referência da camada Ganglia

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

A camada Ganglia fornece suporte ao Ganglia, um sistema distribuído de monitoramento que gerencia o armazenamento e a visualização das métricas de instâncias. Ela foi projetada para funcionar com topologias de instâncias hierárquicas, o que a torna particularmente útil para grupos de instâncias. O Ganglia tem dois componentes básicos:

- Um cliente de baixo custo, que é instalado em cada instância na pilha e envia métricas para o mestre.
- Um mestre, que coleta as métricas dos clientes e as armazena em um volume do Amazon EBS. Ele também exibe as métricas em uma página da web.

AWS OpsWorks O Stacks tem um agente de monitoramento Ganglia em cada instância que ele gerencia. Quando você adiciona uma camada Ganglia à sua pilha e a inicia, os agentes do Ganglia em cada instância relatam as métricas à instância do Ganglia. Para usar o Ganglia, adicione uma camada Ganglia com uma instância à pilha. Você pode acessar os dados efetuando login no back-end do Ganglia no endereço IP do mestre. Você pode fornecer definições de métrica adicionais gravando receitas do Chef.

Short name: monitoring-master

Compatibilidade: uma camada Ganglia é compatível com as seguintes camadas: custom, db-master, memcached, php-app e rails-app.

Open ports: Load-Balancer permite o acesso público às portas 22 (SSH), 80 (HTTP) e 443 (HTTPS).

Autoassign Elastic IP addresses: opção desativada por padrão

Default EBS volume: sim em `/vol/ganglia`

Grupo de segurança padrão: AWS-OpsWorks-Monitoring-Master -Server

Configuração: para configurar uma camada Ganglia, você deve especificar o seguinte:

- O URI que fornece acesso aos gráficos de monitoramento. O valor padrão é http://*DNSName*/ganglia, onde *DNSName* está o nome DNS da instância do Ganglia.
- Um nome de usuário e senha que controlam o acesso às estatísticas de monitoramento.

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies
- ebs
- opsworks_ganglia::client
- opsworks_ganglia::server

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- agent_version
- opsworks_ganglia::configure-server

Deploy recipes:

- deploy::default

- opsworks_ganglia::configure-server
- opsworks_ganglia::deploy

Shutdown recipes:

- opsworks_shutdown::default
- apache2::stop

Instalação:

- O cliente Ganglia é instalado em: `/etc/ganglia`.
- O front-end da Web Ganglia é instalado em: `/usr/share/ganglia-webfrontend`.
- O logtailer Ganglia é instalado em: `/usr/share/ganglia-logtailer`.

Referência da camada do Memcached

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta layer está disponível apenas para stacks baseadas em Linux.

Memcached é um sistema distribuído de armazenamento em cache de memória para dados arbitrários. Ele acelera websites armazenando strings e objetos como chaves e valores na RAM, para reduzir o número de vezes que uma fonte de dados externa precisa ser lida.

Para usar o Memcached em uma pilha, crie uma camada Memcached e adicione uma ou mais instâncias, que funcionam como servidores Memcached. As instâncias instalam automaticamente o

Memcached, e as outras instâncias da pilha podem acessar e usar os servidores do Memcached. Se você usa uma camada do Rails App Server, o AWS OpsWorks Stacks coloca automaticamente um arquivo de `memcached.yml` configuração no diretório de configuração de cada instância na camada. Você pode obter o número da porta e do servidor Memcached neste arquivo.

Short name: memcached

Compatibilidade: a camada Memcached é compatível com as seguintes camadas: personalizada, db-master, lb, monitoring-master, nodejs-app, php-app, rails-app e web.

Portas abertas: a camada Memcached permite o acesso público à porta 22 (SSH) e a todas as portas dos servidores da web, servidores personalizados e servidores de aplicativos Rails, PHP e Node.js da pilha.

Autoassign Elastic IP addresses: opção desativada por padrão

Default EBS volume: não

Grupo de segurança padrão: AWS-OpsWorks-Memcached-Server

Para configurar uma camada Memcached, você deve especificar o tamanho do cache, em MB.

Setup recipes:

- opsworks_initial_setup
- ssh_host_keys
- ssh_users
- mysql::client
- dependencies
- ebs
- opsworks_ganglia::client
- memcached

Configure recipes:

- opsworks_ganglia::configure-client
- ssh_users
- agent_version

Deploy recipes:

- deploy::default

Shutdown recipes:

- opsworks_shutdown::default
- memcached::stop

Instalação:

- AWS OpsWorks O Stacks usa o instalador de pacotes da instância para instalar o Memcached e seus arquivos de log em seus locais padrão.

Outras camadas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Essas layers estão disponíveis apenas para o Chef 11 e pilhas anteriores com base no Linux.

AWS OpsWorks O Stacks também suporta as camadas Ganglia e Memcached.

Tópicos

- Camada Ganglia
- Memcached

Camada Ganglia

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Essa camada está disponíveis apenas para o Chef 11 e pilhas anteriores com base em Linux.

AWS OpsWorks O Stacks envia todas as suas métricas de instância e volume para a Amazon CloudWatch, facilitando a visualização de gráficos e a configuração de alarmes para ajudá-lo a solucionar problemas e realizar ações automatizadas com base no estado de seus recursos. Você também pode usar a camada Ganglia AWS OpsWorks Stacks para opções adicionais de monitoramento de aplicativos, como armazenar as métricas escolhidas.

A camada Ganglia é um esquema para uma instância que monitora sua pilha usando monitoramento distribuído Ganglia. Uma pilha normalmente tem apenas uma instância Ganglia. A camada Ganglia inclui as seguintes definições opcionais de configuração:

Ganglia URL

O caminho URL de estatística. O URL completo é http://*DNSNameURLPath*, onde *DNSName* está o nome DNS da instância associada. O *URLPath* valor padrão é “/ganglia”, que corresponde a algo como: http://ec2-54-245-151-7.us-west-2.compute.amazonaws.com/ganglia.

Ganglia user name

Um nome de usuário para a página da Web de estatística. Você deve fornecer o nome de usuário ao visualizar a página. O valor padrão é “opsworks”.

Ganglia password

Uma senha que controla o acesso à página da web de estatística. Você deve fornecer a senha quando visualizar a página. O valor padrão é uma string gerada aleatoriamente.

> **Note**
>
> Grave a senha para uso posterior; o AWS OpsWorks Stacks não permite que você visualize a senha depois de criar a camada. No entanto, você pode atualizar a senha acessando a página de Edição da camada e clicando em Update password.

Grupos de segurança personalizados

Essa configuração aparece se você optar por não associar automaticamente um grupo de segurança embutido do AWS OpsWorks Stacks às suas camadas. Você deve especificar qual grupo de segurança associar com a camada. Para obter mais informações, consulte Criar uma nova pilha.

Elastic Load Balancer

Você pode anexar um balanceador de carga do Elastic Load Balancing às instâncias de camada.

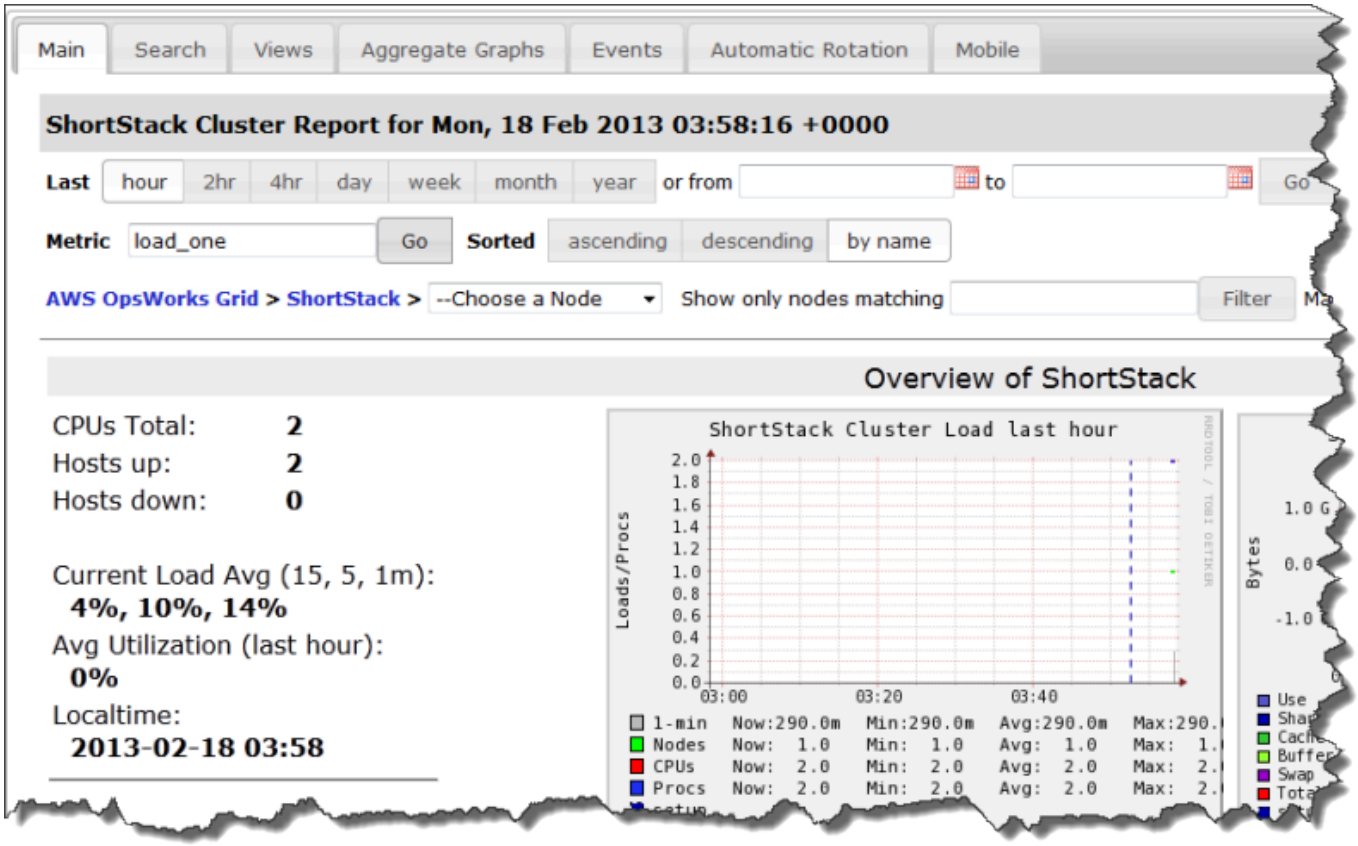
> **Important**
>
> Se sua pilha incluir uma camada Ganglia, recomendamos que você desabilite, SSLv3 se possível, essa camada para resolver as vulnerabilidades descritas em CVE-2014-3566. Para fazer isso, você deve substituir o modelo do servidor Apache `ssl.conf.erb` para modificar a configuração `SSLProtocol`. Para obter detalhes, consulte Desativação SSLv3 para servidores Apache.

Veja as estatísticas do Ganglia

AWS OpsWorks As receitas do Stacks instalam um cliente Ganglia de baixa sobrecarga em cada instância. Se a pilha incluir uma camada Ganglia, o cliente Ganglia começa a relatar automaticamente ao Ganglia assim que a instância fica on-line. O Ganglia usa os dados do cliente para computar uma variedade de estatísticas e exibe os resultados graficamente na página da Web de estatística.

Para visualizar estatísticas Ganglia

1. Na página Camadas, clique em Ganglia para abrir a página de detalhes da camada.
2. No painel de navegação, clique em Instances. Em Ganglia, clique no nome da instância.
3. Copie o nome Public DNS da instância.
4. Use o nome DNS para criar o URL de estatísticas, que terá uma aparência semelhante a: http://ec2-54-245-151-7.us-west-2.compute.amazonaws.com/ganglia.
5. Cole o URL completo em seu navegador, navegue até a página e insira o nome de usuário Ganglia e a senha para exibir a página. Veja a seguir um exemplo.



Memcached

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esta camada está disponível apenas para o Chef 11 e as pilhas anteriores com base em Linux.

Uma camada Memcached é uma camada AWS OpsWorks Stacks que fornece um plano para instâncias que funcionam como servidores Memcached — um sistema distribuído de armazenamento em cache de memória para dados arbitrários. A camada Memcached inclui as seguintes configurações.

Allocated memory (MB)

(Opcional) A quantidade de cache de memória (em MB) para cada uma das instâncias da camada. O padrão é 512 MB.

Grupos de segurança personalizados

Essa configuração aparece se você optar por não associar automaticamente um grupo de segurança embutido do AWS OpsWorks Stacks às suas camadas. Você deve especificar qual grupo de segurança associar com a camada. Para obter mais informações, consulte Criar uma nova pilha.

Componentes do livro de receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Um livro de receitas geralmente inclui os seguintes componentes básicos:

- Os arquivos de atributo contêm um conjunto de atributos que representam valores a serem usados por receitas e modelos.
- Os arquivos de modelo são modelos que as receitas usam para criar outros arquivos, como arquivos de configuração.

  Os arquivos modelo normalmente permitem que você modifique o arquivo de configuração substituindo os atributos, o que pode ser feito sem a necessidade de tocar no livro de receitas, em vez de reescrever um arquivo de configuração. De acordo com a prática padrão, sempre que considerar alterar um arquivo de configuração em uma instância, mesmo que seja ligeiramente, deve-se usar um arquivo de modelo.
- Os arquivos de receita são aplicativos Ruby que definem tudo o que é necessário para configurar um sistema, incluindo a criação e a configuração de pastas, a instalação e a configuração de pacotes, serviços de inicialização, e assim por diante.

Os livros de receita não precisam ter todos os três componentes. As abordagens de personalização mais simples exigem apenas o atributo ou os arquivos de modelo. Além disso, os livros de receitas podem, incluir opcionalmente, outros tipos de arquivo, como definições ou especificações.

Esta seção descreve os três componentes de receitas padrão. Para obter mais informações, especialmente sobre como implementar receitas, consulte Opscode.

Tópicos

- Atributos
- Modelos
- Fórmulas

Atributos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Receitas e modelos dependem de uma variedade de valores, como definições de configuração. Em vez de codificar esses valores diretamente em receitas ou modelos, você pode criar um arquivo de atributo com um atributo que represente cada valor. Em seguida, use os atributos em suas receitas ou em seus modelos, em vez de valores explícitos. A vantagem de usar atributos é que seus valores podem ser substituídos sem precisar tocar no livro de receitas. Por esse motivo, os atributos sempre devem ser usados para definir os seguintes tipos de valores:

- Valores que podem ser diferentes no tempo ou de uma pilha para outra, por exemplo, nomes de usuário.

  Se você codificar esses valores, você deve alterar a receita ou o modelo cada vez que você precisar alterar um valor. Ao usar atributos para definir estes valores, você pode usar os mesmos livros de receitas para cada pilha e apenas substituir os atributos apropriados.
- Valores confidenciais, como senhas ou chaves secretas.

  Colocar valores confidenciais explícitos em seu livro de receitas pode aumentar o risco de exposição. Em vez disso, defina atributos com valores fictícios e os substitua para definir os valores reais. A melhor maneira de substituir tais atributos é com o JSON personalizado. Para obter mais informações, consulte Usar JSON personalizado.

Para obter mais informações sobre atributos e como substituí-los, consulte Sobrepor atributos.

O exemplo a seguir é uma parte de um exemplo de arquivo de atributo.

```
...
default["apache"]["listen_ports"] = [ '80','443' ]
default["apache"]["contact"] = 'ops@example.com'
default["apache"]["timeout"] = 120
default["apache"]["keepalive"] = 'Off'
default["apache"]["keepaliverequests"] = 100
default["apache"]["keepalivetimeout"] = 3
default["apache"]["prefork"]["startservers"] = 16
default["apache"]["prefork"]["minspareservers"] = 16
default["apache"]["prefork"]["maxspareservers"] = 32
default["apache"]["prefork"]["serverlimit"] = 400
default["apache"]["prefork"]["maxclients"] = 400
default["apache"]["prefork"]["maxrequestsperchild"] = 10000
...
```

AWS OpsWorks Stacks define atributos usando a seguinte sintaxe:

```
node.type["attribute"]["subattribute"]["..."]=value
```

Dois pontos (:) podem ser usados da seguinte forma:

```
node.type[:attribute][:subattribute][:...]=value
```

Uma definição de atributo tem os seguintes componentes:

`node.`

O prefixo `node.` é opcional e geralmente omitido, como se pode ver no exemplo.

`type`

O tipo determina se o atributo pode ser substituído. AWS OpsWorks Os atributos das pilhas geralmente usam um dos seguintes tipos:

- `default` é o tipo mais usado, pois ele permite que o atributo seja substituído.
- `normal`define um atributo que substitui um dos valores padrão do atributo AWS OpsWorks Stacks.

> **Note**
>
> O Chef suporta tipos adicionais, que não são necessários para o AWS OpsWorks Stacks, mas podem ser úteis para o seu projeto. Para obter mais informações, consulte Sobre atributos.

`attribute name`

O nome do atributo usa a sintaxe padrão do nó do Chef, `[:attribute][:subattribute][...]`. Qualquer nome pode ser usado para seus atributos. No entanto, como discutido em Sobrepor atributos, os atributos personalizados do livro de receitas são mesclados no objeto de nó da instância, juntamente com os atributos de configuração e de implantação da pilha e a ferramenta Ohai do Chef. Os nomes de configuração normalmente usados como porta ou usuário aparecem em uma vários livros de receitas.

Para evitar colisões de nome, a convenção é criar nomes de atributo qualificados com pelo menos dois elementos, como se pode ver no exemplo. O primeiro elemento deve ser exclusivo e é

normalmente baseado em um nome de produto, como Apache. Ele é seguido por um ou mais sub-atributos que identificam o valor específico, como `[:user]` ou `[:port]`. Podem ser usados quantos sub-atributos forem necessários para seu projeto.

`value`

Um atributo pode ser configurado para os seguintes tipos de valores:

- Uma string, como `default[:apache][:keepalive] = 'Off'`.
- Um número (sem aspas) como `default[:apache][:timeout] = 120`.
- Um valor Booliano, que pode ser `true` ou `false` (sem aspas).
- Uma lista de valores, como `default[:apache][:listen_ports] = [ '80','443' ]`

O arquivo de atributo é um aplicativo Ruby, portanto, você também pode usar a sintaxe do nó e os operadores lógicos para atribuir valores com base em outros atributos. Para obter mais informações sobre como definir atributos, consulte Sobre atributo s. Para ver exemplos de arquivos de atributos funcionais, consulte os livros de receitas integrados do AWS OpsWorks Stacks em https://github.com/aws/ opsworks-cookbooks.

Modelos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Muitos pacotes são configurados, criando um arquivo de configuração e colocando-o no diretório apropriado. Você pode incluir um arquivo de configuração em seu livro de receitas e copiá-lo para o diretório apropriado, mas uma abordagem mais flexível é que suas receitas criem o arquivo de configuração a partir de um modelo. Uma vantagem de um modelo é que atributos podem ser usados para definir os valores do modelo. Por exemplo, isso permite que um arquivo de configuração seja modificado sem tocar no livro de receitas, usando o JSON personalizado para substituir os valores de atributo apropriados.

Um modelo tem essencialmente o mesmo conteúdo e a mesma estrutura do arquivo associado. Veja um arquivo de exemplo, `httpd.conf`.

```
ServerRoot "<%= node[:apache][:dir] %>"
<% if node[:platform] == "debian" || node[:platform] == "ubuntu" -%>
  LockFile /var/lock/apache2/accept.lock
<% else -%>
   LockFile logs/accept.lock
<% end -%>
PidFile <%= node[:apache][:pid_file] %>
Timeout <%= node[:apache][:timeout] %>
KeepAlive <%= node[:apache][:keepalive] %>
MaxKeepAliveRequests <%= node[:apache][:keepaliverequests] %>
KeepAliveTimeout <%= node[:apache][:keepalivetimeout] %>
<IfModule mpm_prefork_module>
    StartServers          <%= node[:apache][:prefork][:startservers] %>
    MinSpareServers       <%= node[:apache][:prefork][:minspareservers] %>
    MaxSpareServers       <%= node[:apache][:prefork][:maxspareservers] %>
    ServerLimit           <%= node[:apache][:prefork][:serverlimit] %>
    MaxClients            <%= node[:apache][:prefork][:maxclients] %>
    MaxRequestsPerChild   <%= node[:apache][:prefork][:maxrequestsperchild] %>
</IfModule>
...
```

O exemplo a seguir é o `httpd.conf` arquivo que foi gerado para uma instância do Ubuntu:

```
ServerRoot "/etc/httpd"
LockFile logs/accept.lock
PidFile /var/run/httpd/httpd.pid
Timeout 120
KeepAlive Off
MaxKeepAliveRequests 100
KeepAliveTimeout 3
<IfModule mpm_prefork_module>
    StartServers          16
    MinSpareServers       16
    MaxSpareServers       32
    ServerLimit           400
    MaxClients            400
    MaxRequestsPerChild   10000
</IfModule>
```

```
...
```

Grande parte do texto do modelo é simplesmente copiado do modelo para o `httpd.conf`arquivo No entanto, o conteúdo `<%= ... %>` é tratado da seguinte forma:

- O Chef substitui `<%= node[:attribute][:sub_attribute][:...]%>` para o valor do atributo.

  Por exemplo, `StartServers <%= node[:apache][:prefork][:startservers] %>` transforma-se em `StartServers 16` no `httpd.conf`.
- Você pode usar `<%if-%>, <%else-%>, and <%end-%>` para selecionar um valor condicionalmente.

  O exemplo define um caminho de arquivo diferente para `accept.lock`, dependendo da plataforma.

> **Note**
>
> Você não está limitado aos atributos em seu livro de receitas de arquivos de atributo. Pode ser usado qualquer atributo no nó da instância do objeto. Gerado por uma ferramenta do Chef, chamada Ohai, e também incorporado ao nó do objeto. Para obter mais informações sobre atributos, consulte Sobrepor atributos.

Para obter mais informações sobre os modelos, incluindo como incorporar o código Ruby, consulte Sobre modelos.

Fórmulas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Receitas são aplicações Ruby que definem a configuração de um sistema. Elas instalam pacotes, criam arquivos de configuração a partir de modelos, executam comandos de shell, criam arquivos e diretórios, e assim por diante. Normalmente, você faz com que AWS OpsWorks as pilhas executem receitas automaticamente quando um evento de ciclo de vida ocorre na instância, mas você também pode executá-las explicitamente a qualquer momento usando o comando Execute Recipes stack. Para obter mais informações, consulte Sobre receitas.

Uma receita geralmente consiste em grande parte de uma série de recursos, cada um dos quais representa o estado desejado de um aspecto do sistema. Cada recurso inclui um conjunto de atributos que define o estado desejado e especifica qual ação deve ser executada. O Chef associa cada recurso a um provedor apropriado que executa a ação. Para obter mais informações, consulte Referência de recursos e provedores.

Um recurso `package` ajuda a gerenciar pacotes de software em instâncias do Linux. O exemplo a seguir instala o pacote do Apache.

```
...
package 'apache2' do
  case node[:platform]
  when 'centos','redhat','fedora','amazon'
    package_name 'httpd'
  when 'debian','ubuntu'
    package_name 'apache2'
  end
  action :install
end
...
```

O Chef usa o provedor de pacote apropriado para a plataforma. Muitas vezes, um valor é atribuído aos recursos de atributo, mas é possível usar as operações lógicas de Ruby para executar atribuições condicionais. O exemplo mostra um operador `case` que utiliza `node[:platform]` para identificar a instância do sistema operacional e define o atributo `package_name` apropriadamente. Os atributos podem ser inseridos em uma receita, usando a sintaxe do nó padrão do Chef e o Chef a substitui com o valor associado. Além dos atributos de receita, qualquer atributo pode ser utilizado no nó do objeto.

Após determinar o nome do pacote, o segmento de código termina com uma ação `install` que instala o pacote. Outras ações para este recurso incluem `upgrade` e `remove`. Para obter mais informações, consulte: pacote.

Muitas vezes é útil separar tarefas complexas de instalação e configuração em uma ou mais subtarefas, cada uma implementada como uma receita separada para executar sua receita primária no momento apropriado. O exemplo a seguir mostra a linha de código que segue o exemplo anterior:

```
include_recipe 'apache2::service'
```

Para que uma receita execute uma receita criança, use a palavra-chave `include_recipe` seguida pelo nome da receita. As receitas são identificadas através da utilização da sintaxe padrão do Chef *`CookbookName`*`::`*`RecipeName`*, em que *`RecipeName`* omite a extensão `.rb`.

> **Note**
>
> Uma declaração `include_recipe` executa efetivamente a receita nessa etapa na receita principal. No entanto, o que de fato acontece é que o Chef substitui cada declaração `include_recipe` com o código especificado da receita, antes que ele execute a receita principal.

Um recurso `directory` representa um diretório, como aquele que contém os arquivos do pacote. O seguinte recurso `default.rb` cria um diretório de log do Linux.

```
directory node[:apache][:log_dir] do
    mode 0755
    action :create
end
```

O diretório de log é definido em um dos arquivos de atributo de receitas. O recurso especifica o modo do diretório como 0755, e usa uma ação `create` para criar o diretório. Para obter mais informações, consulte diretório. Esse recurso também pode ser usado com instâncias do Windows.

O recurso `execute` representa comandos, tais como comandos de shell ou scripts. O exemplo a seguir gera arquivos de module.load.

```
execute 'generate-module-list' do
  if node[:kernel][:machine] == 'x86_64'
    libdir = 'lib64'
```

```
  else
    libdir = 'lib'
  end
  command "/usr/local/bin/apache2_module_conf_generate.pl /usr/#{libdir}/httpd/
modules /etc/httpd/mods-available"
  action :run
end
```

O primeiro recurso determina o tipo de CPU. `[:kernel][:machine]` é outro atributo automático que o Chef gera para representar diversas propriedades do sistema, nesse caso, o tipo de CPU. Em seguida, ele especifica o comando, um script Perl e usa uma ação `run` para executar o script, o que gera arquivos de module.load. Para obter mais informações, consulte execute.

Um recurso `template` representa um arquivo, normalmente um arquivo de configuração, que deve ser gerado a partir de um dos arquivos de modelo do livro de receitas. O exemplo a seguir cria um arquivo de configuração `httpd.conf` a partir do modelo `apache2.conf.erb`, discutido em Modelos.

```
template 'apache2.conf' do
  case node[:platform]
  when 'centos','redhat','fedora','amazon'
    path "#{node[:apache][:dir]}/conf/httpd.conf"
  when 'debian','ubuntu'
    path "#{node[:apache][:dir]}/apache2.conf"
  end
  source 'apache2.conf.erb'
  owner 'root'
  group 'root'
  mode 0644
  notifies :restart, resources(:service => 'apache2')
end
```

O recurso determina o nome e a localização do arquivo gerado, baseando-se na instância do sistema operacional. Em seguida, ele especifica `apache2.conf.erb` como o modelo a ser usado para gerar o arquivo e define o proprietário do arquivo, o grupo, e o modo. Ele executa a ação `notify` para notificar o recurso `service` que representa o servidor Apache para reiniciar o servidor. Para obter mais informações, consulte modelo.

Atributos de implantação e configuração de pilha: Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este tópico inclui os atributos de implantação e configuração de pilhas usadas mais comumente e sua sintaxe de nó associada. Ele é organizado em torno da estrutura do namespace da configuração de pilha usada pelas pilhas do Linux. Observe que os mesmos nomes de atributo são, às vezes, usados para finalidades diferentes e ocorrem em diferentes namespaces. Por exemplo, `id` pode se referir a um ID de pilha, de camada, de aplicação e assim por diante, portanto você precisa do nome totalmente qualificado para usar o valor do atributo. Uma maneira conveniente de visualizar esses dados é como um objeto JSON. Para ver exemplos, consulte Configuração de pilha e atributos de implantação.

> **Note**
>
> Em instâncias Linux, o AWS OpsWorks Stacks instala esse objeto JSON em cada instância, além de adicionar os dados ao objeto node. Você pode recuperá-lo usando o comando `get_json` do agente da CLI.

Tópicos

- Atributos opsworks
- Atributos opsworks_custom_cookbooks
- Atributos dependencies
- Atributos ganglia
- Atributos mysql
- Atributos passenger
- Atributos opsworks_bundler
- Atributos deploy

- Outros atributos de nível superior

Atributos opsworks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O elemento `opsworks`, às vezes denominado como namespace `opsworks`, contém um conjunto de atributos que definem a configuração básica da pilha.

> **Important**
>
> Substituir os valores de atributo no namespace `opsworks` não é recomendado. Isso pode fazer com que as receitas integradas falhem.

Tópicos

- aplicações
- Atributos instance
- Atributos layers
- Atributos rails_stack
- Atributos pilha
- Outros atributos opsworks de nível superior

aplicações

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Contém uma lista de objetos incorporados, um para cada aplicativo que existe para a pilha. Cada objeto incorporado contém os seguintes atributos que descrevem a configuração da aplicação.

> **Note**
>
> A sintaxe de nó geral desses atributos é como a seguinte, em que *i* especifica o índice de lista baseado em zero da instância.
>
> ```
> node["opsworks"]["applications"]["i"]["attribute_name"]
> ```

application_type

O tipo da aplicação (string). Os valores possíveis são:

- `php`: aplicação PHP
- `rails`: uma aplicação Ruby on Rails
- `java`: uma aplicação Java
- `nodejs`: uma aplicação Node.js
- `web`: uma página HTML estática
- `other`: todos os outros tipos de aplicação

```
node["opsworks"]["applications"]["i"]["application_type"]
```

nome

O nome de exibição definido pelo usuário, como `"SimplePHP"` (string).

```
node["opsworks"]["applications"]["i"]["name"]
```

slug_name

Um nome curto, que é um nome totalmente em minúsculas, como o gerado `"simplephp"` a OpsWorks partir do nome do aplicativo (string).

```
node["opsworks"]["applications"]["i"]["slug_name"]
```

Atributos instance

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O atributo `instance` contém um conjunto de atributos que especificam a configuração dessa instância.

arquitetura	availability_zone	backends
aws_instance_id	hostname	id
instance_type	ip	camadas
private_dns_name	private_ip	public_dns_name
região		

arquitetura

A arquitetura da instância, como `"i386"` (string).

```
node["opsworks"]["instance"]["architecture"]
```

availability_zone

A zona de disponibilidade da instância, como `"us-west-2a"` (string).

```
node["opsworks"]["instance"]["availability_zone"]
```

backends

O número de processos da web de back-end (string). Ele determina, por exemplo, o número de conexões simultâneas que HAProxy serão encaminhadas para um back-end do Rails. O valor padrão depende da memória e do número de núcleos da instância.

```
node["opsworks"]["instance"]["backends"]
```

aws_instance_id

O ID da EC2 instância (string).

```
node["opsworks"]["instance"]["aws_instance_id"]
```

hostname

O nome do host, como `"php-app1"` (string).

```
node["opsworks"]["instance"]["hostname"]
```

id

O ID da instância, que é um GUID AWS OpsWorks gerado pelo Stacks que identifica exclusivamente a instância (string).

```
node["opsworks"]["instance"]["id"]
```

instance_type

O tipo de instância, como `"c1.medium"` (string).

```
node["opsworks"]["instance"]["instance_type"]
```

ip

O endereço IP público (string).

```
node["opsworks"]["instance"]["ip"]
```

camadas

Uma lista dos camadas da instância, que são identificados pelos nomes curtos, como `"lb"` ou `"db-master"` (lista de string).

```
node["opsworks"]["instance"]["layers"]
```

private_dns_name

O nome DNS privado (string).

```
node["opsworks"]["instance"]["private_dns_name"]
```

private_ip

O endereço IP privado (string).

```
node["opsworks"]["instance"]["private_ip"]
```

public_dns_name

O nome DNS público (string).

```
node["opsworks"]["instance"]["public_dns_name"]
```

região

A região da AWS, como `"us-west-2"` (string).

```
node["opsworks"]["instance"]["region"]
```

Atributos layers

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O atributo `layers` contém um conjunto de atributos de camada, um para cada uma das camadas da pilha, que são nomeadas com o nome curto da camada, como `php-app`. Uma pilha pode ter no máximo uma de cada das camadas integradas, cujos nomes curtos são como o seguinte:

- `db-master`: Camada MySQL
- `java-app`: Camada Java App Server
- `lb`: HAProxy camada
- `monitoring-master`: Camada Ganglia
- `memcached`: Camada Memcached
- `nodejs-app`: Camada Node.js App Server
- `php-app`: Camada PHP App Server
- `rails-app`: Camada Rails App Server
- `web`: Camada Static Web Server

Uma pilha pode conter qualquer número de camadas personalizadas, que têm nomes curtos definidos pelo usuário.

Cada atributo de camada contém os seguintes atributos:

- id
- Instâncias
- name

id

O ID da camada, que é um GUID gerado OpsWorks e que identifica exclusivamente a camada (string).

```
node["opsworks"]["layers"]["layershortname"]["id"]
```

instâncias

O elemento `instances` contém um conjunto de atributos instance, um para cada uma das instâncias online da camada. Eles são nomeados com o nome de host da instância, como `php-app1`.

> **Note**
>
> O elemento `instances` contém somente as instâncias que estão no estado online quando os atributos específicos de implantação e configuração de pilha são criados.

Cada elemento instance contém os seguintes atributos:

availability_zone	aws_instance_id	backends
booted_at	created_at	elastic_ip
instance_type	ip	private_ip
public_dns_name	private_dns_name	região
status		

availability_zone

A zona de disponibilidade, como `"us-west-2a"` (string).

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["availability_zone"]
```

aws_instance_id

O ID da EC2 instância (string).

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["aws_instance_id"]
```

backends

O número de processos da web de back-end (número). Ele determina, por exemplo, o número de conexões simultâneas que HAProxy serão encaminhadas para um back-end do Rails. O valor padrão depende da memória e do número de núcleos da instância.

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["backends"]
```

booted_at

A hora em que a EC2 instância foi inicializada, usando o formato UTC:mm:ss+HH:mm yyyy-mm-dddThh (string). Por exemplo, `"2013-10-01T08:35:22+00:00"` corresponde a 08:35:22, em 1 de out. de 2013, sem diferença de fuso horário. Para obter mais informações, consulte ISO 8601.

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["booted_at"]
```

created_at

A hora em que a EC2 instância foi criada, usando o formato UTC:mm:ss+HH:mm yyyy-mm-dddThh (string). Por exemplo, `"2013-10-01T08:35:22+00:00"` corresponde a 08:35:22, em 1 de out. de 2013, sem diferença de fuso horário. Para obter mais informações, consulte ISO 8601.

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["created_at"]
```

elastic_ip

O endereço IP elástico, que é definido como nulo se a instância não tiver um (string).

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["elastic_ip"]
```

instance_type

O tipo de instância, como `"c1.medium"` (string).

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["instance_type"]
```

ip

O endereço IP público (string).

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["ip"]
```

private_ip

O endereço IP privado (string).

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["private_ip"]
```

public_dns_name

O nome DNS público (string).

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["public_dns_name"]
```

private_dns_name

O nome DNS privado (string).

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["private_dns_name"]
```

região

A região da AWS, como `"us-west-2"` (string).

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["region"]
```

status

O status (string). Os valores possíveis são:

- "requested"
- "booting"
- "running_setup"
- "online"
- "setup_failed"
- "start_failed"
- "terminating"
- "terminated"
- "stopped"
- "connection_lost"

```
node["opsworks"]["layers"]["layershortname"]["instances"]["instancehostname"]
["status"]
```

nome

O nome da camada, que é usado para representar a camada do console (string). Ele pode ser definido pelo e não é necessariamente exclusivo.

```
node["opsworks"]["layers"]["layershortname"]["name"]
```

Atributos rails_stack

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

nome

Especifica a pilha do Rails e é definido como `"apache_passenger"` ou `"nginx_unicorn"` (string).

```
node["opsworks"]["rails_stack"]["name"]
```

fórmula

A receita associada, que depende se você utiliza Passenger ou Unicorn (string):

- Unicorn: `"unicorn::rails"`
- Passenger: `"passenger_apache2::rails"`

```
node["opsworks"]["rails_stack"]["recipe"]
```

restart_command

O comando de reinicialização, que depende se você utiliza Passenger ou Unicorn (string):

- Unicorn: `"../../shared/scripts/unicorn clean-restart"`
- Passenger: `"touch tmp/restart.txt"`

serviço

O nome do serviço, que depende se você utiliza Passenger ou Unicorn (string):

- Unicorn: `"unicorn"`
- Passenger: `"apache2"`

```
node["opsworks"]["rails_stack"]["service"]
```

Atributos pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Os atributos `stack` especificam alguns aspectos da configuração de pilha, como as configurações da camada de serviço.

- elb-load-balancers
- id
- nome
- rds_instances
- vpc_id

elb-load-balancers

Contém uma lista de objetos incorporados, um para cada load balancer Elastic Load Balancing na pilha. Cada objeto incorporado contém os seguintes atributos que descrevem a configuração do load balancer.

> **Note**
>
> A sintaxe de nó geral desses atributos é como a seguinte, em que *i* especifica o índice de lista baseado em zero da instância.
>
> ```
> node["opsworks"]["stack"]["elb-load-balancers"]["i"]["attribute_name"]
> ```

dns_name

O nome DNS do load balancer (string).

```
node["opsworks"]["stack"]["elb-load-balancers"]["i"]["dns_name"]
```

nome

O nome do load balancer (string).

```
node["opsworks"]["stack"]["elb-load-balancers"]["i"]["name"]
```

layer_id

O ID da camada à qual o load balancer está anexado (string).

```
node["opsworks"]["stack"]["elb-load-balancers"]["i"]["layer_id"]
```

id

O ID da pilha (string).

```
node["opsworks"]["stack"]["id"]
```

nome

O nome da pilha (string).

```
node["opsworks"]["stack"]["name"]
```

rds_instances

Contém uma lista de objetos incorporados, um para cada instância do Amazon RDS que está registrada com a stack. Cada objeto incorporado contém um conjunto de atributos que definem a configuração da instância. Você especifica esses valores quando usa o console do Amazon RDS ou a API para criar a instância. Você também pode usar o console Amazon RDS ou a API para editar algumas das configurações depois que a instância for criada. Para obter mais informações, consulte a documentação do Amazon RDS.

> **Note**
>
> A sintaxe de nó geral desses atributos é como a seguinte, em que *i* especifica o índice de lista baseado em zero da instância.
>
> ```
> node["opsworks"]["stack"]["rds_instances"]["i"]["attribute_name"]
> ```

Se a sua pilha tiver várias instâncias do Amazon RDS, a seguir há um exemplo de como usar uma instância específica em uma receita.

```
if my_rds = node["opsworks"]["stack"]["rds_instances"].select{|rds_instance|
 rds_instance["db_instance_identifier"] == ‘db_id’ }.first
  template “/etc/rds.conf” do
    source "rds.conf.erb"
    variables :address => my_rds["address"]
  end
```

```
end
```

address	allocated_storage	arn
auto_minor_version_upgrade	availability_zone	backup_retention_period
db_instance_class	db_instance_identifier	db_instance_status
db_name	db_parameter_groups	db_security_groups
db_user	engine	instance_create_time
license_model	multi_az	option_group_memberships
porta	preferred_backup_window	preferred_maintenance_window
publicly_accessible	read_replica_db_instance_identifiers	região
status_infos	vpc_security_groups	

address

O URL das instâncias, como `opsinstance.ccdvt3hwog1a.us-west-2.rds.amazonaws.com` (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["address"]
```

allocated_storage

O armazenamento alocado, em GB (número).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["allocated_storage"]
```

arn

O Nome de região da Amazon (ARN) da instância (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["arn"]
```

auto_minor_version_upgrade

Se atualizações de versão secundárias devem ser aplicadas automaticamente (booliano).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["auto_minor_version_upgrade"]
```

availability_zone

A Zona de disponibilidade da instância, como `us-west-2a` (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["availability_zone"]
```

backup_retention_period

O período de retenção de backup, em dias (número).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["backup_retention_period"]
```

db_instance_class

A classe de instância de banco de dados, como `db.m1.small` (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["db_instance_class"]
```

db_instance_identifier

O DB instance identifier definido pelo usuário (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["db_instance_identifier"]
```

db_instance_status

O status da instância (string). Para obter mais informações, consulte Instância de banco de dados.

```
node["opsworks"]["stack"]["rds_instances"]["i"]["db_instance_status"]
```

db_name

O nome do banco de dados definido pelo usuário (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["db_name"]
```

db_parameter_groups

Os parameter groups de banco de dados da instância, que contêm uma lista de objetos incorporados, um para cada grupo de parâmetro. Para obter mais informações, consulte Trabalhar com parameter groups de banco de dados. Cada objeto contém os seguintes atributos:

db_parameter_group_name

O nome do grupo (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["db_parameter_groups"][j"]
["db_parameter_group_name"]
```

parameter_apply_status

O status de aplicação (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["db_parameter_groups"][j"]
["parameter_apply_status"]
```

db_security_groups

Os grupos de segurança do banco de dados da instância, que contêm uma lista de objetos incorporados, um para cada grupo de segurança. Para obter mais informações, consulte Trabalhar com grupos de segurança de banco de dados. Cada objeto contém os seguintes atributos

db_security_group_name

O nome do grupo de segurança (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["db_security_groups"][j"]
["db_security_group_name"]
```

status

O status (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["db_security_groups"][j"]
["status"]
```

db_user

O nome do usuário mestre definido pelo usuário (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["db_user"]
```

engine

O mecanismo de banco de dados, como `mysql(5.6.13)` (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["engine"]
```

instance_create_time

A hora de criação da instância, como `2014-04-15T16:13:34Z` (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["instance_create_time"]
```

license_model

O modelo de licença da instância, como `general-public-license` (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["license_model"]
```

multi_az

Se a Implantação Multi-AZ está habilitada (booliano).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["multi_az"]
```

option_group_memberships

As associações de grupo de opção da instância, que contêm uma lista dos objetos incorporados, um para cada grupo de opção. Para obter mais informações, consulte Trabalhar com grupos de opções. Cada objeto contém os seguintes atributos:

option_group_name

O nome do grupo (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["option_group_memberships"]
[j"]["option_group_name"]
```

status

O status do grupo (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["option_group_memberships"]
[j"]["status"]
```

porta

A porta do servidor do banco de dados (número).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["port"]
```

preferred_backup_window

O período de backup diário preferencial, como `06:26-06:56` (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["preferred_backup_window"]
```

preferred_maintenance_window

O período de manutenção semanal preferencial, como `thu:07:13-thu:07:43` (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["preferred_maintenance_window"]
```

publicly_accessible

Se o banco de dados é acessível publicamente (booliano).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["publicly_accessible"]
```

read_replica_db_instance_identifiers

Uma lista dos identificadores de instância de réplica de leitura (lista de string). Para obter mais informações, consulte Trabalhar com réplicas de leitura.

```
node["opsworks"]["stack"]["rds_instances"]["i"]
["read_replica_db_instance_identifiers"]
```

região

A região da AWS, como `us-west-2` (string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["region"]
```

status_infos

Uma lista das informações de status (lista de string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["status_infos"]
```

vpc_security_groups

Uma lista dos grupos de segurança da VPC (lista de string).

```
node["opsworks"]["stack"]["rds_instances"]["i"]["vpc_security_groups"]
```

vpc_id

O ID da VPC (string). Esse valor é `null` se a instância não estiver em uma VPC.

```
node["opsworks"]["stack"]["vpc_id"]
```

Outros atributos opsworks de nível superior

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção contém os atributos `opsworks` que não têm atributos filhos.

atividade

A atividade que está associada com os atributos, como `deploy` (string).

```
node["opsworks"]["activity"]
```

agent_version

A versão do OpsWorks agente da instância (string).

```
node["opsworks"]["agent_version"]
```

deploy_chef_provider

O provedor de implantação do Chef, que influencia a estrutura de diretório de uma aplicação implantada (string). Você pode definir esse atributo como um dos seguintes:

- `Branch`
- `Revision`
- `Timestamped` (valor padrão)

```
node["opsworks"]["deploy_chef_provider"]
```

ruby_stack

A pilha do Ruby (string). A configuração padrão é a versão empresarial (`ruby_enterprise`). Para a versão MRI, defina esse atributo como `ruby`.

```
node["opsworks"]["ruby_stack"]
```

ruby_version

A versão do Ruby que será usada pelas aplicações (string). Você pode usar esse atributo para especificar somente as versões principais e secundárias. Você deve usar o atributo ["ruby"] adequado para especificar a versão do patch. Para obter mais informações sobre como especificar uma versão, incluindo exemplos, consulte Versões do Ruby. Para saber com mais detalhes como o AWS OpsWorks Stacks determina a versão do Ruby, consulte o arquivo de atributos integrados, ruby.rb.

```
node["opsworks"]["ruby_version"]
```

run_cookbook_tests

Se você deve executar minitest-chef-handlertestes em seus livros de receitas do Chef 11.4 (booleano).

```
node["opsworks"]["run_cookbook_tests"]
```

sent_at

Quando esse comando foi enviado para a instância (número).

```
node["opsworks"]["sent_at"]
```

implantação

Se esses atributos forem associados a uma atividade de implantação, `deployment` será definido como o ID de implantação, um GUID gerado pelo AWS OpsWorks Stacks que identifica exclusivamente a implantação (string). Caso contrário, o atributo é definido como nulo.

```
node["opsworks"]["deployment"]
```

Atributos opsworks_custom_cookbooks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Contém atributos que especificam os livros de receitas personalizados da pilha.

habilitado

Se os livros de receitas personalizados estão ativados (Booliano).

```
node["opsworks_custom_cookbooks"]["enabled"]
```

recipes

Uma lista das receitas que devem ser executadas para esse comando, incluindo as receitas personalizadas, usando o formado *cookbookname*::*recipename* (lista de string).

```
node["opsworks_custom_cookbooks"]["recipes"]
```

Atributos dependencies

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Contém vários atributos que são relacionados ao `update_dependencies`comando de stackExecuta comandos de pilha.

gem_binary

A localização dos binários Gems (string).

upgrade_debs

Se deve ser feito o upgrade dos pacotes Debs (booliano).

update_debs

Se os pacotes Debs devem ser atualizados (booliano).

Atributos ganglia

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Contém um atributo web que contém vários atributos que especificam como acessar a página da web de estatísticas do Ganglia:

password

A senha necessária para acessar a página de estatísticas (string).

```
node["ganglia"]["web"]["password"]
```

url

O caminho do URL da página de estatísticas, como `"/ganglia"` (string). O URL completo é `http://DNSNameURLPath`, onde *DNSName* é o nome DNS da instância associada.

```
node["ganglia"]["web"]["url"]
```

usuário

O nome do usuário necessário para acessar a página de estatísticas (string).

```
node["ganglia"]["web"]["user"]
```

Atributos mysql

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Contém um conjunto de atributos que especificam a configuração do servidor do banco de dados do MySQL.

clientes

Uma lista de endereços IP clientes (lista de string).

```
node["mysql"]["clients"]
```

server_root_password

A senha raiz (string).

```
node["mysql"]["server_root_password"]
```

Atributos passenger

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Contém um conjunto de atributos que especificam a configuração do Phusion Passenger.

gem_bin

A localização dos RubyGems binários, como `"/usr/local/bin/gem"` (string).

```
node["passenger"]["gem_bin"]
```

max_pool_size

O tamanho máximo do grupo (número).

```
node["passenger"]["max_pool_size"]
```

ruby_bin

A localização dos arquivos binários do Ruby, como `"/usr/local/bin/ruby"`.

```
node["passenger"]["ruby_bin"]
```

version

A versão do Passenger (string).

```
node["passenger"]["version"]
```

Atributos opsworks_bundler

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Contém elementos que especificam o suporte do Bundler.

manage_package

Se o Bundler deve ser instalado e gerenciado (booliano).

```
node["opsworks_bundler"]["manage_package"]
```

version

A versão do Bundler (string).

```
node["opsworks_bundler"]["version"]
```

Atributos deploy

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Se os atributos forem associados a um evento Deploy ou um comando de pilha Execute Recipes, o `deploy` atributo conterá um atributo para cada aplicação que for implantada, nomeado de acordo com o nome curto da aplicação. Cada atributo app contém os seguintes atributos:

aplicação	application_type	auto_bundle_on_deploy
banco de dados	deploy_to	domínios
document_root	environment_variables	grupo
keep_releases	memcached	migrar
mounted_at	purge_before_symlink	rails_env
restart_command	scm	ssl_certificate
ssl_certificate_ca	ssl_certificate_key	ssl_support
pilha	symlink_before_migrate	symlinks
usuário		

aplicação

O nome slug da aplicação, como `"simplephp"` (string).

```
node["deploy"]["appshortname"]["application"]
```

application_type

O tipo da aplicação (string). Os valores possíveis são:

- `java`: uma aplicação Java
- `nodejs`: uma aplicação Node.js
- `php`: uma aplicação PHP
- `rails`: uma aplicação Ruby on Rails
- `web`: uma página HTML estática
- `other`: todos os outros tipos de aplicação

```
node["deploy"]["appshortname"]["application_type"]
```

auto_bundle_on_deploy

Para aplicações Rails, se for necessário executar o Bundler durante a implantação (booliano).

```
node["deploy"]["appshortname"]["auto_bundle_on_deploy"]
```

banco de dados

Contém as informações necessárias para a conexão do banco de dados da aplicação. Se o aplicativo tiver uma camada de banco de dados anexada, o AWS OpsWorks Stacks atribuirá automaticamente os valores apropriados a esses atributos.

adapter

O adaptador do banco de dados, como `mysql` (string).

```
node["deploy"]["appshortname"]["database"]["adapter"]
```

banco de dados

O nome do banco de dados, que geralmente é o nome slug da aplicação, como `"simplephp"` (string).

```
node["deploy"]["appshortname"]["database"]["database"]
```

data_source_provider

A fonte de dados: `mysql` ou `rds` (string).

```
node["deploy"]["appshortname"]["database"]["data_source_provider"]
```

host

O endereço IP do host do banco de dados (string).

```
node["deploy"]["appshortname"]["database"]["host"]
```

password

A senha do banco de dados (string).

```
node["deploy"]["appshortname"]["database"]["password"]
```

porta

A porta do banco de dados (número).

```
node["deploy"]["appshortname"]["database"]["port"]
```

reconnect

Para aplicações Rails, se a aplicação deve se reconectar caso a conexão não exista mais (booliano).

```
node["deploy"]["appshortname"]["database"]["reconnect"]
```

username

O nome do usuário (string).

```
node["deploy"]["appshortname"]["database"]["username"]
```

deploy_to

Onde o aplicativo será implantado, como `"/srv/www/simplephp"` (string).

```
node["deploy"]["appshortname"]["deploy_to"]
```

domínios

Uma lista dos domínios da aplicação (lista de string).

```
node["deploy"]["appshortname"]["domains"]
```

document_root

A raiz do documento, se você especificar uma raiz não padrão, ou nulo se você usar a raiz padrão (string).

```
node["deploy"]["appshortname"]["document_root"]
```

environment_variables

Uma coleção de até vinte atributos que representam as variáveis do ambiente especificadas pelo usuário, definidas para a aplicação. Para obter mais informações sobre como definir as variáveis do ambiente de uma aplicação, consulte Adição de aplicativos. Cada nome de atributo é definido como um nome de variável do ambiente e o valor correspondente é definido como o valor da variável, para que você possa usar a sintaxe a seguir para fazer referência a um valor específico.

```
node["deploy"]["appshortname"]["environment_variables"]["variable_name"]
```

grupo

O grupo da aplicação (string).

```
node["deploy"]["appshortname"]["group"]
```

keep_releases

O número de implantações de aplicativos que o AWS OpsWorks Stacks armazenará (número). Este atributo controla o número de vezes que você pode reverter uma aplicação. Por padrão, ele é definido para o valor global, deploy_keep_releases , que tem um valor padrão de 5. Você pode substituir `keep_releases` para especificar o número de implantações armazenadas para uma aplicação específica.

```
node["deploy"]["appshortname"]["keep_releases"]
```

memcached

Contém dois atributos que definem a configuração do Memcached.

host

O endereço IP da instância do servidor do Memcached (string).

```
node["deploy"]["appshortname"]["memcached"]["host"]
```

porta

A porta que o servidor do Memcached está utilizando para escutar (número).

```
node["deploy"]["appshortname"]["memcached"]["port"]
```

migrar

Se as migrações devem ser executadas para aplicações Rails (booliano).

```
node["deploy"]["appshortname"]["migrate"]
```

mounted_at

O ponto de montagem da aplicação, se você especificar um ponto de montagem não padrão, ou nulo se você usar o ponto de montagem padrão (string).

```
node["deploy"]["appshortname"]["mounted_at"]
```

purge_before_symlink

Para aplicações Rails, um conjunto de caminhos a serem limpos antes da criação de symlinks (lista de string).

```
node["deploy"]["appshortname"]["purge_before_symlink"]
```

rails_env

Para instâncias do Rails App Server, o ambiente Rails, como `"production"` (string).

```
node["deploy"]["appshortname"]["rails_env"]
```

restart_command

Um comando a ser executado quando a aplicação é reiniciada, como `"echo 'restarting app'"`.

```
node["deploy"]["appshortname"]["restart_command"]
```

scm

Contém um conjunto de atributos que especificam as informações OpsWorks usadas para implantar o aplicativo a partir de seu repositório de controle de origem. Os atributos variam de acordo com o tipo de repositório.

password

Para repositórios privados, a senha, e nulo para repositórios públicos (string). Para buckets privados do Amazon S3, o atributo é definido como a chave secreta.

```
node["deploy"]["appshortname"]["scm"]["password"]
```

repositório

O URL do repositório, como `"git://github.com/amazonwebservices/opsworks-demo-php-simple-app.git"` (string).

```
node["deploy"]["appshortname"]["scm"]["repository"]
```

revisão

Se o repositório tiver várias ramificações, o atributo especificará a versão ou a ramificação da aplicação, como `"version1"` (string). Caso contrário, ele será definido como nulo.

```
node["deploy"]["appshortname"]["scm"]["revision"]
```

scm_type

O tipo de repositório (string). Os valores possíveis são:

- `"git"`: um repositório Git
- `"svn"`: um repositório Subversion
- `"s3"`: um bucket do Amazon S3
- `"archive"`: um arquivo HTTP
- `"other"`: outro tipo de repositório

```
node["deploy"]["appshortname"]["scm"]["scm_type"]
```

ssh_key

Uma implantação de chave SSH para acessar repositórios Git privados e nulo para repositórios públicos (string).

```
node["deploy"]["appshortname"]["scm"]["ssh_key"]
```

usuário

Para repositórios privados, o nome do usuário, e nulo para repositórios públicos (string). Para buckets privados do Amazon S3, o atributo é definido como a chave de acesso.

```
node["deploy"]["appshortname"]["scm"]["user"]
```

ssl_certificate

Se você ativou o suporte a SSL, o certificado SSL da aplicação. Caso contrário, nulo (string).

```
node["deploy"]["appshortname"]["ssl_certificate"]
```

ssl_certificate_ca

Se o SSL está ativado, um atributo para especificar uma chave de autoridade de certificado intermediária ou autenticação do cliente (string).

```
node["deploy"]["appshortname"]["ssl_certificate_ca"]
```

ssl_certificate_key

Se você ativou o suporte a SSL, a chave privada do SSL da aplicação. Caso contrário, nulo (string).

```
node["deploy"]["appshortname"]["ssl_certificate_key"]
```

ssl_support

Se há suporte a SSL (booliano).

```
node["deploy"]["appshortname"]["ssl_support"]
```

pilha

Contém um atributo booliano, `needs_reload`, que especifica se é necessário recarregar o servidor da aplicação durante a implantação.

```
node["deploy"]["appshortname"]["stack"]["needs_reload"]
```

symlink_before_migrate

Para aplicações Rails, contém symlinks que devem ser criados antes de executar migrações como pares `"link":"target"`.

```
node["deploy"]["appshortname"]["symlink_before_migrate"]
```

symlinks

Contém os symlinks da implantação como pares `"link":"target"`.

```
node["deploy"]["appshortname"]["symlinks"]
```

usuário

O usuário da aplicação (string).

```
node["deploy"]["appshortname"]["user"]
```

Outros atributos de nível superior

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção contém atributos de configuração de pilha de nível superior sem atributos filhos.

Atributos rails

Contém um atributo max_pool_size que especifica o tamanho máximo do grupo do servidor (número). O valor do atributo é definido pelo AWS OpsWorks Stacks e depende do tipo de instância, mas você pode substituí-lo usando JSON personalizado ou um arquivo de atributo personalizado.

```
node["rails"]["max_pool_size"]
```

Atributos recipes

Uma lista de receitas integradas que foram executadas por essa atividade, usando o formato "*cookbookname*::*recipename*" (lista de string).

```
node["recipes"]
```

Atributos opsworks_rubygems

Contém um elemento de versão que especifica a RubyGems versão (string).

```
node["opsworks_rubygems"]["version"]
```

Atributos languages

Contém um atributo para cada linguagem instalada. Leva o nome da linguagem, como ruby. O atributo é um objeto que contém um atributo, como ruby_bin, que especifica a pasta de instalação, como `"/usr/bin/ruby"` (string).

Atributos ssh_users

Contém um conjunto de atributos, cada um dos quais descreve um dos usuários que receberam permissões SSH. Cada atributo é nomeado com o ID Unix de um usuário. AWS OpsWorks O Stacks gera uma ID exclusiva para cada usuário na faixa de 2.000 a 4.000, por exemplo`"2001"`, e cria um usuário com essa ID em cada instância. Como AWS OpsWorks reserva o intervalo de 2.000 a 4.000, os usuários que você cria fora AWS OpsWorks (usando receitas de livros de receitas ou importando usuários AWS OpsWorks do IAM, por exemplo) podem fazer com UIDs que sejam substituídos pelo AWS OpsWorks Stacks para outro usuário. Como prática recomendada, crie usuários e gerencie seus acessos no console do AWS OpsWorks Stacks. Se você criar usuários fora do AWS OpsWorks Stacks, use *UnixID* valores maiores que 4000.

Cada atributo contém os seguintes atributos:

email

O endereço de e-mail do usuário do (string).

```
node["ssh_users"]["UnixID"]["email"]
```

public_key

A chave SSH pública do usuário do (string).

```
node["ssh_users"]["UnixID"]["public_key"]
```

sudoer

Se o usuário do tem permissões de sudo (booliano).

```
node["ssh_users"]["UnixID"]["sudoer"]
```

nome

O nome do usuário do (string).

```
node["ssh_users"]["UnixID"]["name"]
```

Atributos de livros de receitas integrados

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> A maioria desses atributos estão disponíveis apenas em pilhas do Linux.

A maioria das receitas integradas possui um ou mais arquivos de atributos que definem várias configurações. Você pode acessar essas configurações em suas receitas personalizadas e usar seu JSON personalizado para substituí-las. Normalmente, você precisa acessar ou substituir atributos que controlam a configuração das várias tecnologias de servidor suportadas pelo AWS OpsWorks Stacks. Esta seção resume esses atributos. Os arquivos de atributos completos e as receitas e modelos associados estão disponíveis em https://github.com/aws/opsworks-cookbooks.git.

> **Note**
>
> Todos os atributos de receitas integrados são do tipo `default`.

Tópicos

- Atributos apache2
- Atributos deploy
- Atributos haproxy
- Atributos do memcached
- Atributos mysql
- Atributos nginx
- Atributos opsworks_berkshelf
- Atributos opsworks_java
- Atributos passenger_apache2
- Atributos ruby
- Atributos unicorn

Atributos apache2

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esses atributos estão disponíveis apenas em pilhas do Linux.

Os atributos apache2 especificam a configuração do servidor HTTP Apache. Para obter mais informações, consulte Funcionalidades principais do Apache. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

binary	contact	deflate_types
dir	document_root	grupo
hide_info_headers	icondir	init_script
keepalive	keepaliverequests	keepalivetimeout
lib_dir	libexecdir	listen_ports
log_dir	Atributos logrotate	pid_file
Atributos prefork	serversignature	servertokens
timeout	traceenable	usuário
version	Atributos worker	

binary

A localização do arquivo binário (string) do Apache. O valor padrão é `'/usr/sbin/httpd'`.

```
node[:apache][:binary]
```

contact

Um e-mail de contato (string). O valor padrão é um endereço fictício `'ops@example.com'`.

```
node[:apache][:contact]
```

deflate_types

Define o `mod_deflate` para habilitar a compactação de determinados tipos MIME, se eles forem compatíveis com o navegador (lista de strings). O valor padrão é:

```
['application/javascript',
 'application/json',
 'application/x-javascript',
 'application/xhtml+xml',
 'application/xml',
 'application/xml+rss',
 'text/css',
 'text/html',
 'text/javascript',
 'text/plain',
 'text/xml']
```

> **Warning**
>
> A compactação pode introduzir riscos à segurança. Para desabilitar completamente a compactação, defina este atributo da seguinte forma:
>
> ```
> node[:apache][:deflate_types] = []
> ```

```
node[:apache][:deflate_types]
```

dir

O diretório raiz do servidor (string). Os valores padrão são:

- Amazon Linux e Red Hat Enterprise Linux (RHEL): `'/etc/httpd'`
- Ubuntu: `'/etc/apache2'`

```
node[:apache][:dir]
```

document_root

A raiz do documento (string). Os valores padrão são:

- Amazon Linux e RHEL: `'/var/www/html'`
- Ubuntu: `'/var/www'`

```
node[:apache][:document_root]
```

grupo

O nome do grupo (string). Os valores padrão são:

- Amazon Linux e RHEL: `'apache'`
- Ubuntu: `'www-data'`

```
node[:apache][:group]
```

hide_info_headers

Determina se vai omitir a versão e as informações do módulo de cabeçalhos do HTTP (`'true'`/`'false'`) (string). O valor padrão é `'true'`.

```
node[:apache][:hide_info_headers]
```

icondir

O ícone do diretório (string). Os valores padrão são:

- Amazon Linux e RHEL: `'/var/www/icons/'`
- Ubuntu: `'/usr/share/apache2/icons'`

```
node[:apache][:icondir]
```

init_script

O script de inicialização (string). Os valores padrão são:

- Amazon Linux e RHEL: `'/etc/init.d/httpd'`
- Ubuntu: `'/etc/init.d/apache2'`

```
node[:apache][:init_script]
```

keepalive

Determina se vai habilitar as conexões de keep-alive (string). Os valores possíveis são `'On'` e `'Off'` (string). O valor padrão é `'Off'`.

```
node[:apache][:keepalive]
```

keepaliverequests

O número máximo de solicitações de keep-alive que o Apache irá processar simultaneamente (número). O valor padrão é `100`.

```
node[:apache][:keepaliverequests]
```

keepalivetimeout

O tempo que o Apache aguarda por uma solicitação antes de fechar a conexão (número). O valor padrão é `3`.

```
node[:apache][:keepalivetimeout]
```

lib_dir

O diretório que contém as bibliotecas de código objeto (string). Os valores padrão são:

- Amazon Linux (x86): `'/usr/lib/httpd'`
- Amazon Linux (x64) e RHEL: `'/usr/lib64/httpd'`
- Ubuntu: `'/usr/lib/apache2'`

```
node[:apache][:lib_dir]
```

libexecdir

O diretório que contém os programas executáveis (string). Os valores padrão são:

- Amazon Linux (x86): `'/usr/lib/httpd/modules'`
- Amazon Linux (x64) e RHEL: `'/usr/lib64/httpd/modules'`
- Ubuntu: `'/usr/lib/apache2/modules'`

```
node[:apache][:libexecdir]
```

listen_ports

Uma lista das portas que o servidor escuta (lista de strings). O valor padrão é `[ '80','443' ]`.

```
node[:apache][:listen_ports]
```

log_dir

O diretório de log (string). Os valores padrão são:

- Amazon Linux e RHEL: `'/var/log/httpd'`
- Ubuntu: `'/var/log/apache2'`

```
node[:apache][:log_dir]
```

Atributos logrotate

Esses atributos especificam como fazer a rotação dos arquivos de log.

delaycompress

Determina se vai adiar a compactação de um arquivo de log fechado até o início do próximo ciclo de rotação (`'true'`/`'false'`) (string). O valor padrão é `'true'`.

```
node[:apache][:logrotate][:delaycompress]
```

grupo

O grupo dos arquivos de log (string). O valor padrão é `'adm'`.

```
node[:apache][:logrotate][:group]
```

modo

O modo dos arquivos de log (string). O valor padrão é `'640'`.

```
node[:apache][:logrotate][:mode]
```

proprietário

O proprietário dos arquivos de log (string). O valor padrão é `'root'`.

```
node[:apache][:logrotate][:owner]
```

rotate

O número de ciclos de rotação antes que um arquivo de log fechado seja removido (string). O valor padrão é `'30'`.

```
node[:apache][:logrotate][:rotate]
```

programar

A programação da rotação (string). Os valores possíveis são:

- `'daily'`
- `'weekly'`
- `'monthly'`

O valor padrão é `'daily'`.

```
node[:apache][:logrotate][:schedule]
```

pid_file

O arquivo que contém o ID de processo do daemon (string). Os valores padrão são:

- Amazon Linux e RHEL: `'/var/run/httpd/httpd.pid'`
- Ubuntu: `'/var/run/apache2.pid'`

```
node[:apache][:pid_file]
```

Atributos prefork

Esses atributos especificam a configuração anterior à bifurcação.

maxclients

O número máximo de solicitações simultâneas que será atendido (número). O valor padrão é `400`.

> **Note**
>
> Use este atributo apenas para instâncias que estão executando o Amazon Linux, ou o RHEL. Se suas instâncias estão executando Ubuntu 14.04 LTS, use maxrequestworkers.

```
node[:apache][:prefork][:maxclients]
```

maxrequestsperchild

O número máximo de solicitações que um processo de servidor filho irá processar (número). O valor padrão é `10000`.

```
node[:apache][:prefork][:maxrequestsperchild]
```

maxrequestworkers

O número máximo de solicitações simultâneas que será atendido (número). O valor padrão é `400`.

> **Note**
>
> Use este atributo apenas para instâncias que estão executando Ubuntu 14.04 LTS. Se suas instâncias estão executando o Amazon Linux, ou RHEL, use maxclients .

```
node[:apache][:prefork][:maxrequestworkers]
```

maxspareservers

O número máximo de processos de servidores filho ociosos (número). O valor padrão é `32`.

```
node[:apache][:prefork][:maxspareservers]
```

minspareservers

O número mínimo de processos de servidores filho ociosos (número). O valor padrão é `16`.

```
node[:apache][:prefork][:minspareservers]
```

serverlimit

O número máximo de processos que podem ser configurados (número). O valor padrão é `400`.

```
node[:apache][:prefork][:serverlimit]
```

startservers

O número de processos de servidores filho a serem criados na inicialização (número). O valor padrão é 16.

```
node[:apache][:prefork][:startservers]
```

serversignature

Especifica se e como configurar uma rodapé inferior para documentos gerados pelo servidor (string). Os valores possíveis são `'On'`, `'Off'` e `'Email'`). O valor padrão é `'Off'`.

```
node[:apache][:serversignature]
```

servertokens

Especifica que tipo de informação sobre a versão do servidor está incluída no cabeçalho de resposta (string):

- `'Full'`: Informações completas. Por exemplo, Server: Apache/2.4.2 (Unix) PHP/4.2.2 /1.2 MyMod
- `'Prod'`: Nome do produto. Por exemplo, Servidor: Apache
- `'Major'`: Versão principal. Por exemplo, Servidor: Apache/2
- `'Minor'`: Versões principal e secundária. Por exemplo, Servidor: Apache/2.4
- `'Min'`: Versão mínima. Por exemplo, Servidor: Apache/2.4.2
- `'OS'`: Versão com o sistema operacional. Por exemplo, Servidor: Apache/2.4.2 (Unix)

O valor padrão é `'Prod'`.

```
node[:apache][:servertokens]
```

timeout

A quantidade de tempo que o Apache espera I/O (número). O valor padrão é `120`.

```
node[:apache][:timeout]
```

traceenable

Determina se vai habilitar as solicitações de `TRACE` (string). Os valores possíveis são `'On'` e `'Off'`. O valor padrão é `'Off'`.

```
node[:apache][:traceenable]
```

usuário

O nome do usuário (string). Os valores padrão são:

- Amazon Linux e RHEL: `'apache'`
- Ubuntu: `'www-data'`

```
node[:apache][:user]
```

version

A versão do Apache (string). Os valores padrão são:

- Amazon Linux: `2.2`
- Ubuntu 14.04 LTS: `2.4`
- RHEL: `2.4`

```
node[:apache][:version]
```

Atributos worker

Esses atributos especificam a configuração do processo de operador.

startservers

O número de processos de servidores filho a serem criados na inicialização (número). O valor padrão é `4`.

```
node[:apache][:worker][:startservers]
```

maxclients

O número máximo de solicitações simultâneas que será atendido (número). O valor padrão é `1024`.

```
node[:apache][:worker][:maxclients]
```

maxsparethreads

O número máximo de threads ociosos (número). O valor padrão é `192`.

```
node[:apache][:worker][:maxsparethreads]
```

minsparethreads

O número mínimo de threads ociosos (número). O valor padrão é `64`.

```
node[:apache][:worker][:minsparethreads]
```

threadsperchild

O número de threads por processo filho (número). O valor padrão é `64`.

```
node[:apache][:worker][:threadsperchild]
```

maxrequestsperchild

O número máximo de solicitações que um processo de servidor filho irá processar (número). O valor padrão é `10000`.

```
node[:apache][:worker][:maxrequestsperchild]
```

Atributos deploy

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

O arquivo de atributos integrados `deploy.rb` do livro de receitas de implantação define os seguintes atributos no `opsworks` namespace. Para obter mais informações sobre a implantação de diretórios, consulte Receitas de implantação. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

deploy_keep_releases

Uma configuração global para o número de implantações de aplicativos que o AWS OpsWorks Stacks armazenará (número). O valor padrão é 5. Este valor controla o número de vezes que você pode reverter um aplicativo.

```
node[:opsworks][:deploy_keep_releases]
```

grupo

(Somente para Linux) A configuração do `group` para o diretório de implantação do aplicativo (string). O valor padrão depende do sistema operacional da instância:

- Para instâncias do Ubuntu, o valor padrão é `www-data`.
- Para instâncias do Amazon Linux ou RHEL que são membros de uma camada do App Server que usa Nginx e Unicorn, o valor padrão é `nginx`.
- Para todas as outras instâncias do Amazon Linux ou RHEL, o valor padrão é `apache`.

```
node[:opsworks][:deploy_user][:group]
```

usuário

(Somente para Linux) A configuração do `user` para o diretório de implantação do aplicativo (string). O valor padrão é `deploy`.

```
node[:opsworks][:deploy_user][:user]
```

Atributos haproxy

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes

migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esses atributos estão disponíveis apenas em pilhas do Linux.

Os `haproxyatributos` especificam a configuração HAProxy do servidor. Para obter mais informações, consulte HAProxyDocs. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

balance	check_interval	client_timeout
connect_timeout	default_max_connections	global_max_connections
health_check_method	health_check_url	queue_timeout
http_request_timeout	maxcon_factor_nodejs_app	maxcon_factor_nodejs_app_ssl
maxcon_factor_php_app	maxcon_factor_php_app_ssl	maxcon_factor_rails_app
maxcon_factor_rails_app_ssl	maxcon_factor_static	maxcon_factor_static_ssl
retries	server_timeout	stats_url
stats_user		

balance

O algoritmo usado por um load balancer para selecionar um servidor (string). O valor padrão é `'roundrobin'`. As outras opções são:

- 'static-rr'
- 'leastconn'
- 'source'

- 'uri'
- 'url_param'
- 'hdr(name)'
- 'rdp-cookie'
- 'rdp-cookie(name)'

Para obter mais informações sobre esses argumentos, consulte balance.

```
node[:haproxy][:balance]
```

check_interval

O intervalo de tempo de verificação de integridade (string). O valor padrão é `'10s'`.

```
node[:haproxy][:check_interval]
```

client_timeout

O tempo máximo que um cliente pode ficar inativo (string). O valor padrão é `'60s'`.

```
node[:haproxy][:client_timeout]
```

connect_timeout

O tempo máximo que HAProxy aguardará até que uma tentativa de conexão com o servidor seja bem-sucedida (string). O valor padrão é `'10s'`.

```
node[:haproxy][:connect_timeout]
```

default_max_connections

O número máximo padrão de conexões (string). O valor padrão é `'80000'`.

```
node[:haproxy][:default_max_connections]
```

global_max_connections

O número máximo de conexões (string). O valor padrão é `'80000'`.

```
node[:haproxy][:global_max_connections]
```

health_check_method

O método de verificação de integridade (string). O valor padrão é `'OPTIONS'`.

```
node[:haproxy][:health_check_method]
```

health_check_url

O caminho de URL usado para verificar a integridade dos servidores (string). O valor padrão é `'/'`.

```
node[:haproxy][:health_check_url ]
```

queue_timeout

O tempo máximo de espera por uma conexão livre (string). O valor padrão é `'120s'`.

```
node[:haproxy][:queue_timeout]
```

http_request_timeout

O tempo máximo que HAProxy aguardará por uma solicitação HTTP completa (string). O valor padrão é `'30s'`.

```
node[:haproxy][:http_request_timeout]
```

retries

O número de novas tentativas após uma falha de conexão do servidor (string). O valor padrão é `'3'`.

```
node[:haproxy][:retries]
```

server_timeout

O tempo máximo que um cliente pode ficar inativo (string). O valor padrão é `'60s'`.

```
node[:haproxy][:server_timeout]
```

stats_url

O caminho de URL para a página de estatísticas (string). O valor padrão é `'/haproxy?stats'`.

```
node[:haproxy][:stats_url]
```

stats_user

O nome do usuário da página de estatísticas (string). O valor padrão é `'opsworks'`.

```
node[:haproxy][:stats_user]
```

Os `maxcon` atributos representam um multiplicador do fator de carga usado para calcular o número máximo de conexões que HAProxy permite back-ends. Por exemplo, suponha que você tenha um servidor de aplicativos Rails em uma pequena instância com um `backend` valor de 4, o que significa que o AWS OpsWorks Stacks configurará quatro processos Rails para essa instância. Se você usar o `maxcon_factor_rails_app` valor padrão de 7, HAProxy manipulará 28 (4*7) conexões com o servidor Rails.

maxcon_factor_nodejs_app

O fator maxcon para um servidor de aplicativo Node.js (número). O valor padrão é `10`.

```
node[:haproxy][:maxcon_factor_nodejs_app]
```

maxcon_factor_nodejs_app_ssl

O fator maxcon para um servidor de aplicativo Node.js com SSL (número). O valor padrão é `10`.

```
node[:haproxy][:maxcon_factor_nodejs_app_ssl]
```

maxcon_factor_php_app

O fator maxcon para um servidor de aplicativo PHP (número). O valor padrão é `10`.

```
node[:haproxy][:maxcon_factor_php_app]
```

maxcon_factor_php_app_ssl

O fator maxcon para um servidor de aplicativo PHP com SSL (número). O valor padrão é `10`.

```
node[:haproxy][:maxcon_factor_php_app_ssl]
```

maxcon_factor_rails_app

O fator maxcon para um servidor de aplicativo Rails (número). O valor padrão é 7.

```
node[:haproxy][:maxcon_factor_rails_app]
```

maxcon_factor_rails_app_ssl

O fator maxcon para um servidor de aplicativo Rails com SSL (número). O valor padrão é 7.

```
node[:haproxy][:maxcon_factor_rails_app_ssl]
```

maxcon_factor_static

O fator maxcon para um servidor da web estático (número). O valor padrão é 15.

```
node[:haproxy][:maxcon_factor_static]
```

maxcon_factor_static_ssl

O fator maxcon para um servidor da web estático com SSL (número). O valor padrão é 15.

```
node[:haproxy][:maxcon_factor_static_ssl]
```

Atributos do memcached

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esses atributos estão disponíveis apenas em pilhas do Linux.

Os atributos `memcached` especificam a configuração de servidor Memcached. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

memória	max_connections	pid_file
porta	start_command	stop_command
usuário		

memória

O máximo de memória para uso, em MB (número). O valor padrão é `512`.

```
node[:memcached][:memory]
```

max_connections

O número máximo de conexões (string). O valor padrão é `'4096'`.

```
node[:memcached][:max_connections]
```

pid_file

O arquivo que contém o ID de processo do daemon (string). O valor padrão é `'var/run/memcached.pid'`.

```
node[:memcached][:pid_file]
```

porta

A porta de escuta (número). O valor padrão é `11211`.

```
node[:memcached][:port]
```

start_command

O comando de iniciação (string). O valor padrão é `'/etc/init.d/memcached start'`.

```
node[:memcached][:start_command]
```

stop_command

O comando de interrupção (string). O valor padrão é `'/etc/init.d/memcached stop'`.

```
node[:memcached][:stop_command]
```

usuário

O usuário (string). O valor padrão é `'nobody'`.

```
node[:memcached][:user]
```

Atributos mysql

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esses atributos estão disponíveis apenas em pilhas do Linux.

Os atributos `mysql` especificam a configuração principal do MySQL. Para obter mais informações, consulte Variáveis do sistema do servidor. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

basedir	bind_address	clientes
conf_dir	confd_dir	datadir
grants_path	mysql_bin	mysqladmin_bin

pid_file	porta	root_group
server_root_password	socket	Atributos tunable

basedir

O diretório de base (string). O valor padrão é `'/usr'`.

```
node[:mysql][:basedir]
```

bind_address

O endereço que o MySQL atende (string). O valor padrão é `'0.0.0.0'`.

```
node[:mysql][:bind_address]
```

clientes

Uma lista de clientes (lista de strings).

```
node[:mysql][:clients]
```

conf_dir

O diretório que contém o arquivo de configuração (string). Os valores padrão são:

- Amazon Linux e RHEL: `'/etc'`
- Ubuntu: `'/etc/mysql'`

```
node[:mysql][:conf_dir]
```

confd_dir

O diretório que contém os arquivos de configuração adicionais (string). O valor padrão é `'/etc/mysql/conf.d'`.

```
node[:mysql][:confd_dir]
```

datadir

O diretório de dados (string). O valor padrão é `'/var/lib/mysql'`.

```
node[:mysql][:datadir]
```

grants_path

O local da tabela de concessões (string). O valor padrão é `'/etc/mysql_grants.sql'`.

```
node[:mysql][:grants_path]
```

mysql_bin

O local dos arquivos binários do mysql (string). O valor padrão é `'/usr/bin/mysql'`.

```
node[:mysql][:mysql_bin]
```

mysqladmin_bin

O local do mysqladmin (string). O valor padrão é `'/usr/bin/mysqladmin'`.

```
node[:mysql][:mysqladmin_bin]
```

pid_file

O arquivo que contém o ID de processo do daemon (string). O valor padrão é `'/var/run/mysqld/mysqld.pid'`.

```
node[:mysql][:pid_file]
```

porta

A porta que o servidor escuta (número). O valor padrão é `3306`.

```
node[:mysql][:port]
```

root_group

O grupo raiz (string). O valor padrão é `'root'`.

```
node[:mysql][:root_group]
```

server_root_password

A senha raiz do servidor (string). O valor padrão é gerado aleatoriamente.

```
node[:mysql][:server_root_password]
```

socket

A localização do arquivo de soquete (string). O valor padrão é `'/var/lib/mysql/mysql.sock'`. Os valores padrão são:

- Amazon Linux e RHEL: `'/var/lib/mysql/mysql.sock'`
- Ubuntu: `'/var/run/mysqld/mysqld.sock'`

```
node[:mysql][:socket]
```

Atributos tunable

Os atributos ajustáveis são usados para ajustar o desempenho.

back_log	innodb_additional_mem_pool_size	innodb_buffer_pool_size
innodb_flush_log_at_trx_commit	innodb_lock_wait_timeout	key_buffer
log_slow_queries	long_query_time	max_allowed_packet
max_connections	max_heap_table_size	net_read_timeout
net_write_timeout	query_cache_limit	query_cache_size
query_cache_type	thread_cache_size	thread_stack
wait_timeout	table_cache	

back_log

O número máximo de solicitações pendentes (string). O valor padrão é `'128'`.

```
node[:mysql][:tunable][:back_log]
```

innodb_additional_mem_pool_size

O tamanho do grupo que o Innodb usa para armazenar estruturas de dados internas (string). O valor padrão é `'20M'`.

```
node[:mysql][:tunable][:innodb_additional_mem_pool_size]
```

innodb_buffer_pool_size

O tamanho do grupo de buffers do Innodb (string). O valor do atributo é definido pelo AWS OpsWorks Stacks e depende do tipo de instância, mas você pode substituí-lo usando JSON personalizado ou um arquivo de atributo personalizado.

```
node[:mysql][:tunable][:innodb_buffer_pool_size]
```

innodb_flush_log_at_trx_commit

A frequência com que o Innodb libera o buffer de log (string). O valor padrão é `'2'`. Para obter mais informações, consulte innodb_flush_log_at_trx_commit.

```
node[:mysql][:tunable][:innodb_flush_log_at_trx_commit]
```

innodb_lock_wait_timeout

O tempo máximo, em segundos, que uma transação Innodb aguarda por um bloqueio de linha (string). O valor padrão é `'50'`.

```
node[:mysql][:tunable][:innodb_lock_wait_timeout]
```

key_buffer

O tamanho do buffer de índices (string). O valor padrão é `'250M'`.

```
node[:mysql][:tunable][:key_buffer]
```

log_slow_queries

A localização do arquivo de log de consulta lenta (string). O valor padrão é `'/var/log/mysql/mysql-slow.log'`.

```
node[:mysql][:tunable][:log_slow_queries]
```

long_query_time

O tempo, em segundos, necessário para indicar uma consulta como uma consulta longa (string). O valor padrão é `'1'`.

```
node[:mysql][:tunable][:long_query_time]
```

max_allowed_packet

O tamanho do pacote máximo permitido (string). O valor padrão é `'32M'`.

```
node[:mysql][:tunable][:max_allowed_packet]
```

max_connections

O número máximo de conexões de clientes simultâneas (string). O valor padrão é `'2048'`.

```
node[:mysql][:tunable][:max_connections]
```

max_heap_table_size

O tamanho máximo de tabelas `MEMORY` criadas pelo usuário (string). O valor padrão é `'32M'`.

```
node[:mysql][:tunable][:max_heap_table_size]
```

net_read_timeout

O tempo de espera, em segundos, para obter mais dados em uma conexão (string). O valor padrão é `'30'`.

```
node[:mysql][:tunable][:net_read_timeout]
```

net_write_timeout

O tempo de espera, em segundos, para a gravação de um bloco em uma conexão (string). O valor padrão é `'30'`.

```
node[:mysql][:tunable][:net_write_timeout]
```

query_cache_limit

O tamanho máximo de uma consulta individual armazenada em cache (string). O valor padrão é `'2M'`.

```
node[:mysql][:tunable][:query_cache_limit]
```

query_cache_size

O tamanho do cache de consulta (string). O valor padrão é `'128M'`.

```
node[:mysql][:tunable][:query_cache_size]
```

query_cache_type

O tipo de cache de consulta (string). Os valores possíveis são:

- `'0'`: Não faz o armazenamento em cache nem a recuperação de dados armazenados em cache.
- `'1'`: Instruções de cache que não começam com `SELECT SQL_NO_CACHE`.
- `'2'`: Instruções de cache que começam com `SELECT SQL_CACHE`.

O valor padrão é `'1'`.

```
node[:mysql][:tunable][:query_cache_type]
```

thread_cache_size

O número de threads de cliente que são armazenados em cache para reutilização (string). O valor padrão é `'8'`.

```
node[:mysql][:tunable][:thread_cache_size]
```

thread_stack

O tamanho da pilha para cada thread (string). O valor padrão é `'192K'`.

```
node[:mysql][:tunable][:thread_stack]
```

wait_timeout

O tempo de espera, em segundos, por uma conexão não interativa. O valor padrão é `'180'` (string).

```
node[:mysql][:tunable][:wait_timeout]
```

table_cache

O número de tabelas abertas (string). O valor padrão é `'2048'`.

```
node[:mysql][:tunable][:table_cache]
```

Atributos nginx

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esses atributos estão disponíveis apenas em pilhas do Linux.

Os atributos `nginx` especificam a configuração do Nginx. Para obter mais informações, consulte o Índice de diretivas. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

binary	dir	gzip
gzip_comp_level	gzip_disable	gzip_http_version
gzip_proxied	gzip_static	gzip_types

gzip_vary	keepalive	keepalive_timeout
log_dir	usuário	server_names_hash_bucket_size
worker_processes	worker_connections	

binary

A localização dos arquivos binários do Nginx (string). O valor padrão é `'/usr/sbin/nginx'`.

```
node[:nginx][:binary]
```

dir

A localização dos arquivos como por exemplo, os arquivos de configuração (string). O valor padrão é `'/etc/nginx'`.

```
node[:nginx][:dir]
```

gzip

Determina se a compactação gzip está habilitada (string). Os valores possíveis são `'on'` e `'off'`. O valor padrão é `'on'`.

> **Warning**
>
> A compactação pode introduzir riscos à segurança. Para desabilitar completamente a compactação, defina este atributo da seguinte forma:
>
> ```
> node[:nginx][:gzip] = 'off'
> ```

```
node[:nginx][:gzip]
```

gzip_comp_level

O nível de compactação, que varia entre 1 a 9, sendo que 1 representa o menor nível de compactação (string). O valor padrão é `'2'`.

```
node[:nginx][:gzip_comp_level]
```

gzip_disable

Desabilita a compactação gzip para os agentes de usuário especificados (string). O valor é uma expressão regular e o valor padrão é `'MSIE [1-6].(?!.*SV1)'`.

```
node[:nginx][:gzip_disable]
```

gzip_http_version

Habilita a compactação gzip para a versão de HTTP especificada (string). O valor padrão é `'1.0'`.

```
node[:nginx][:gzip_http_version]
```

gzip_proxied

Determina se e como compactar as respostas às solicitações de proxy. Os valores possíveis são (string):

- `'off'`: não compactar solicitações encaminhadas por proxy
- `'expired'`: compactar se o cabeçalho Expire impedir o armazenamento em cache
- `'no-cache'`: compactar se o cabeçalho Cache-Control está definido como "no-cache"
- `'no-store'`: compactar se o cabeçalho Cache-Control está definido como "no-store"
- `'private'`: compactar se o cabeçalho Cache-Control está definido como "private"
- `'no_last_modified'`: compactar se Last-Modified não está definido
- `'no_etag'`: compacte se a solicitação não tiver um cabeçalho ETag
- `'auth'`: compactar se a solicitação inclui um cabeçalho Authorization
- `'any'`: compactar todas as solicitações encaminhadas por proxy

O valor padrão é `'any'`.

```
node[:nginx][:gzip_proxied]
```

gzip_static

Determina se o módulo estático gzip está habilitado (string). Os valores possíveis são `'on'` e `'off'`. O valor padrão é `'on'`.

```
node[:nginx][:gzip_static]
```

gzip_types

Uma lista de tipos MIME para compactação (lista de strings). O valor padrão é `['text/plain', 'text/html', 'text/css', 'application/x-javascript', 'text/xml', 'application/xml', 'application/xml+rss', 'text/javascript']`.

```
node[:nginx][:gzip_types]
```

gzip_vary

Determina se vai habilitar um cabeçalho de resposta `Vary:Accept-Encoding` (string). Os valores possíveis são `'on'` e `'off'`. O valor padrão é `'on'`.

```
node[:nginx][:gzip_vary]
```

keepalive

Determina se vai habilitar uma conexão de keep-alive (string). Os valores possíveis são `'on'` e `'off'`. O valor padrão é `'on'`.

```
node[:nginx][:keepalive]
```

keepalive_timeout

O tempo máximo, em segundos, em que uma conexão de keep-alive permanece aberta (número). O valor padrão é `65`.

```
node[:nginx][:keepalive_timeout]
```

log_dir

A localização dos arquivos de log (string). O valor padrão é `'/var/log/nginx'`.

```
node[:nginx][:log_dir]
```

usuário

O usuário (string). Os valores padrão são:

- Amazon Linux e RHEL: `'www-data'`

- Ubuntu: `'nginx'`

```
node[:nginx][:user]
```

server_names_hash_bucket_size

O tamanho do bucket para tabelas de hash de nomes de servidores, que pode ser definido com `32`, `64`ou `128` (número). O valor padrão é `64`.

```
node[:nginx][:server_names_hash_bucket_size]
```

worker_processes

O número de processos de operadores (número). O valor padrão é `10`.

```
node[:nginx][:worker_processes]
```

worker_connections

O número máximo de conexões de operadores (número). O valor padrão é `1024`. O número máximo de clientes é definido como `worker_processes * worker_connections`.

```
node[:nginx][:worker_connections]
```

Atributos opsworks_berkshelf

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esses atributos estão disponíveis apenas em pilhas do Linux.

Os atributos `opsworks_berkshelf` especificam a configuração do Berkshelf. Para obter mais informações, consulte Berkshelf. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

depurar

Determina se vai incluir as informações de depuração do Berkshelf no log do Chef (booliano). O valor padrão é `false`.

```
node['opsworks_berkshelf]['debug']
```

Atributos opsworks_java

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esses atributos estão disponíveis apenas em pilhas do Linux.

Os atributos `opsworks_java` especificam a configuração do servidor Tomcat. Para obter mais informações, consulte a Referência de configuração do Apache Tomcat. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

datasources	java_app_server_version	java_shared_lib_dir
Atributos jvm_pkg	custom_pkg_location_url_debian	java_home_basedir
custom_pkg_location_url_rhel	use_custom_pkg_location	jvm_options

jvm_version	Atributos tomcat	

datasources

Um conjunto de atributos que definem os nomes de recursos da JNDI (string). Para obter mais informações sobre como usar este atributo, consulte A implantação de um JSP App com um banco de dados de back-end. O valor padrão é um hash vazio que pode ser preenchido com os mapeamentos personalizados entre os nomes curtos dos aplicativos e os nomes da JNDI. Para obter mais informações, consulte A implantação de um JSP App com um banco de dados de back-end.

```
node['opsworks_java']['datasources']
```

java_app_server_version

A versão do servidor de aplicativos Java (número). O valor padrão é 7. Você pode substituir esse atributo e especificar a versão 6. Se você instalar um JDK não padrão, este atributo será ignorado.

```
node['opsworks_java']['java_app_server_version']
```

java_shared_lib_dir

O diretório das bibliotecas compartilhadas Java (string). O valor padrão é `/usr/share/java`.

```
node['opsworks_java']['java_shared_lib_dir']
```

Atributos jvm_pkg

Um conjunto de atributos que você pode substituir para instalar um JDK não padrão.

use_custom_pkg_location

Determina se vai instalar um JDK personalizado em vez do OpenJDK (booliano). O valor padrão é `false`.

```
node['opsworks_java']['jvm_pkg']['use_custom_pkg_location']
```

custom_pkg_location_url_debian

A localização do pacote JDK a ser instalado nas instâncias do Ubuntu (string). O valor padrão é `'http://aws.amazon.com/'`, que é simplesmente um valor de inicialização sem significado próprio. Se você deseja instalar um JDK não padrão, deve substituir este atributo e configurá-lo com o URL apropriado.

```
node['opsworks_java']['jvm_pkg']['custom_pkg_location_url_debian']
```

custom_pkg_location_url_rhel

A localização do pacote JDK a ser instalado nas instâncias do Amazon Linux e do RHEL (string). O valor padrão é `'http://aws.amazon.com/'`, que é simplesmente um valor de inicialização sem significado próprio. Se você deseja instalar um JDK não padrão, deve substituir este atributo e configurá-lo com o URL apropriado.

```
node['opsworks_java']['jvm_pkg']['custom_pkg_location_url_rhel']
```

java_home_basedir

O diretório para onde o pacote do JDK será extraído (string). O valor padrão é `/usr/local`. Você não precisa especificar esta configuração para pacotes RPM, pois eles possuem uma estrutura de diretórios completa.

```
node['opsworks_java']['jvm_pkg']['java_home_basedir']
```

jvm_options

As opções de linha de comando da JVM, que permitem a especificação de configurações como por exemplo, o tamanho da heap (string). Um conjunto comum de opções é `-Djava.awt.headless=true -Xmx128m -XX:+UseConcMarkSweepGC`. O valor padrão é sem opções.

```
node['opsworks_java']['jvm_options']
```

jvm_version

A versão do OpenJDK (número). O valor padrão é 7. Você pode substituir esse atributo e especificar o OpenJDK versão 6. Se você instalar um JDK não padrão, este atributo será ignorado.

```
node['opsworks_java']['jvm_version']
```

Atributos tomcat

Um conjunto de atributos que você pode substituir para instalar a configuração padrão do Tomcat.

ajp_port	apache_tomcat_bind_mod	apache_tomcat_bind_path
auto_deploy	connection_timeout	mysql_connector_jar
porta	secure_port	shutdown_port
threadpool_max_threads	threadpool_min_spare_threads	unpack_wars
uri_encoding	use_ssl_connector	use_threadpool
userdatabase_pathname		

ajp_port

A porta do AJP (número). O valor padrão é `8009`.

```
node['opsworks_java']['tomcat']['ajp_port']
```

apache_tomcat_bind_mod

O módulo de proxy (string). O valor padrão é `proxy_http`. Você pode substituir esse atributo para especificar o módulo de proxy do AJP `proxy_ajp`.

```
node['opsworks_java']['tomcat']['apache_tomcat_bind_mod']
```

apache_tomcat_bind_path

O caminho de associação do Apache Tomcat (string). O valor padrão é `/`. Você não deve substituir esse atributo, pois a alteração do caminho de associação pode fazer com que o aplicativo pare de funcionar.

```
node['opsworks_java']['tomcat']['apache_tomcat_bind_path']
```

auto_deploy

Determina se a implantação automática está ativada (booliano). O valor padrão é `true`.

```
node['opsworks_java']['tomcat']['auto_deploy']
```

connection_timeout

O tempo limite de conexão, em milissegundos (número). O valor padrão é `20000` (20 segundos).

```
node['opsworks_java']['tomcat']['connection_timeout']
```

mysql_connector_jar

O arquivo JAR da biblioteca de conectores do MySQL (string). O valor padrão é `mysql-connector-java.jar`.

```
node['opsworks_java']['tomcat']['mysql_connector_jar']
```

porta

A porta padrão (número). O valor padrão é `8080`.

```
node['opsworks_java']['tomcat']['port']
```

secure_port

A porta segura (número). O valor padrão é `8443`.

```
node['opsworks_java']['tomcat']['secure_port']
```

shutdown_port

A porta de desligamento (número). O valor padrão é `8005`.

```
node['opsworks_java']['tomcat']['shutdown_port']
```

threadpool_max_threads

O número máximo de threads no grupo de threads (número). O valor padrão é `150`.

```
node['opsworks_java']['tomcat']['threadpool_max_threads']
```

threadpool_min_spare_threads

O número mínimo de threads sobressalentes no grupo de threads (número). O valor padrão é `4`.

```
node['opsworks_java']['tomcat']['threadpool_min_spare_threads']
```

unpack_wars

Determina se vai descompactar os arquivos WAR (booliano). O valor padrão é `true`.

```
node['opsworks_java']['tomcat']['unpack_wars']
```

uri_encoding

A codificação do URI (string). O valor padrão é `UTF-8`.

```
node['opsworks_java']['tomcat']['uri_encoding']
```

use_ssl_connector

Determina se vai utilizar um conector SSL (booliano). O valor padrão é `false`.

```
node['opsworks_java']['tomcat']['use_ssl_connector']
```

use_threadpool

Determina se vai utilizar um grupo de threads (booliano). O valor padrão é `false`.

```
node['opsworks_java']['tomcat']['use_threadpool']
```

userdatabase_pathname

O nome do caminho de banco de dados do usuário (string). O valor padrão é `conf/tomcat-users.xml`.

```
node['opsworks_java']['tomcat']['userdatabase_pathname']
```

Atributos passenger_apache2

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esses atributos estão disponíveis apenas em pilhas do Linux.

Os atributos `passenger_apache2` especificam a configuração do Phusion Passenger. Para obter mais informações, consulte o guia do usuário do Phusion Passenger, versão Apache. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

friendly_error_pages	gem_bin	gems_path
high_performance_mode	root_path	max_instances_per_app
max_pool_size	max_requests	module_path
pool_idle_time	rails_app_spawner_idle_time	rails_framework_spawner_idle_time
rails_spawn_method	ruby_bin	ruby_wrapper_bin
stat_throttle_rate	version	

friendly_error_pages

Determina se vai exibir uma página de erro amigável caso um aplicativo não inicie (string). Este atributo pode ser definido como 'on' ou 'off'; o valor padrão é 'off'.

```
node[:passenger][:friendly_error_pages]
```

gem_bin

A localização dos arquivos binários do Gem (string). O valor padrão é `'/usr/local/bin/gem'`.

```
node[:passenger][:gem_bin]
```

gems_path

O caminho do gems (string). O valor padrão depende da versão do Ruby. Por exemplo:

- Ruby versão 1.8: `'/usr/local/lib/ruby/gems/1.8/gems'`
- Ruby versão 1.9: `'/usr/local/lib/ruby/gems/1.9.1/gems'`

```
node[:passenger][:gems_path]
```

high_performance_mode

Determina se vai utilizar o modo de alto desempenho do Passenger (string). Os valores possíveis são `'on'` e `'off'`. O valor padrão é `'off'`.

```
node[:passenger][:high_performance_mode ]
```

root_path

O diretório raiz do Passenger (string). O valor padrão depende das versões do Ruby e do Passenger. Na sintaxe do Chef, o valor é `"#{node[:passenger][:gems_path]}/passenger-#{passenger[:version]}"`.

```
node[:passenger][:root_path]
```

max_instances_per_app

O número máximo de processos de aplicativos por aplicativo (número). O valor padrão é `0`. Para obter mais informações, consulte PassengerMaxInstancesPerApp.

```
node[:passenger][:max_instances_per_app]
```

max_pool_size

O número máximo de processadores de aplicativos (número). O valor padrão é 8. Para obter mais informações, consulte PassengerMaxPoolSize.

```
node[:passenger][:max_pool_size]
```

max_requests

O número máximo de solicitações (número). O valor padrão é `0`.

```
node[:passenger][:max_requests]
```

module_path

O caminho do módulo (string). Os valores padrão são:

- Amazon Linux e RHEL: `"#{node['apache']['libexecdir']}/mod_passenger.so"`
- Ubuntu: `"#{passenger[:root_path]}/ext/apache2/mod_passenger.so"`

```
node[:passenger][:module_path]
```

pool_idle_time

O tempo máximo, em segundos, que o processo de um aplicativo pode ficar ocioso (número). O valor padrão é `14400` (4 horas). Para obter mais informações, consulte PassengerPoolIdleTime.

```
node[:passenger][:pool_idle_time]
```

rails_app_spawner_idle_time

O tempo máximo de ociosidade para o gerador de aplicativos do Rails (número). Se este atributo for definido como zero, o tempo limite para o gerador de aplicativos não será esgotado. O valor padrão é `0`. Para obter mais informações, consulte Informações sobre os métodos de geração.

```
node[:passenger][:rails_app_spawner_idle_time]
```

rails_framework_spawner_idle_time

O tempo máximo de ociosidade para o gerador de estruturas do Rails (número). Se este atributo for definido como zero, o tempo limite do gerador de estruturas não será esgotado. O valor padrão é `0`. Para obter mais informações, consulte Informações sobre os métodos de geração.

```
node[:passenger][:rails_framework_spawner_idle_time]
```

rails_spawn_method

O método de geração do Rails (string). O valor padrão é `'smart-lv2'`. Para obter mais informações, consulte Informações sobre os métodos de geração.

```
node[:passenger][:rails_spawn_method]
```

ruby_bin

A localização dos arquivos binários do Ruby (string). O valor padrão é `'/usr/local/bin/ruby'`.

```
node[:passenger][:ruby_bin]
```

ruby_wrapper_bin

A localização do script de wrapper do Ruby (string). O valor padrão é `'/usr/local/bin/ruby_gc_wrapper.sh'`.

```
node[:passenger][:ruby_wrapper_bin]
```

stat_throttle_rate

A frequência com que o Passenger executa as verificações do sistema de arquivos (número). O valor padrão é 5, o que significa que as verificações serão executadas no máximo uma vez a cada 5 segundos. Para obter mais informações, consulte PassengerStatThrottleRate .

```
node[:passenger][:stat_throttle_rate]
```

version

A versão (string). O valor padrão é `'3.0.9'`.

```
node[:passenger][:version]
```

Atributos ruby

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esses atributos estão disponíveis apenas em pilhas do Linux.

Os atributos `ruby` especificam a versão do Ruby usada pelos aplicativos. Observe que a utilização dos atributos foi alterada com a introdução do controle de versão semântico no Ruby 2.1. Para obter mais informações sobre como especificar uma versão, incluindo exemplos, consulte Versões do Ruby. Para obter detalhes completos sobre como o AWS OpsWorks Stacks determina a versão do Ruby, consulte o arquivo de atributos incorporado, ruby.rb. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

full_version

O número da versão completa (string). Você não deve substituir esse atributo. Em vez disso, utilize [:opsworks][:ruby_version] e o atributo de versão de patch adequado para especificar uma versão.

```
[:ruby][:full_version]
```

major_version

O número da versão principal (string). Você não deve substituir esse atributo. Em vez disso, utilize [:opsworks][:ruby_version] para especificar a versão principal.

```
[:ruby][:major_version]
```

minor_version

O número da versão secundária (string). Você não deve substituir esse atributo. Em vez disso, utilize [:opsworks][:ruby_version] para especificar a versão secundária.

```
[:ruby][:minor_version]
```

patch

O nível do patch (string). Esse atributo é válido para o Ruby versão 2.0.0 e anterior. Para versões posteriores do Ruby, use o atributo `patch_version`.

```
[:ruby][:patch]
```

O número do patch deve ser precedido por p. Por exemplo, você pode usar o seguinte JSON personalizado para especificar o nível de patch 484.

```
{
  "ruby":{"patch":"p484"}
}
```

patch_version

O número do patch (string). Este atributo é válido para o Ruby versão 2.1 e posterior. Para versões anteriores do Ruby, use o atributo `patch`.

```
[:ruby][:patch_version]
```

pkgrelease

O número da versão do pacote (string).

```
[:ruby][:pkgrelease]
```

Atributos unicorn

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esses atributos estão disponíveis apenas em pilhas do Linux.

Os atributos `unicorn` especificam a configuração do Unicorn. Para obter mais informações, consulte Unicorn::Configurator. Para obter mais informações sobre como substituir atributos integrados para especificar valores personalizados, consulte Sobrepor atributos.

accept_filter	backlog	delay
tcp_nodelay	tcp_nopush	preload_app
timeout	tries	version
worker_processes		

accept_filter

O filtro de aceitação `'httpready'` ou `'dataready'` (string). O valor padrão é `'httpready'`.

```
node[:unicorn][:accept_filter]
```

backlog

O número máximo de solicitações que a fila pode conter (número). O valor padrão é `1024`.

```
node[:unicorn][:backlog]
```

delay

O tempo de espera, em segundos, para tentar novamente a associação de um soquete (número). O valor padrão é `0.5`.

```
node[:unicorn][:delay]
```

preload_app

Determina se vai pré-carregar um aplicativo antes de bifurcar um processo de operador (booliano). O valor padrão é `true`.

```
node[:unicorn][:preload_app]
```

tcp_nodelay

Determina se vai desabilitar o algoritmo de Nagle para soquetes do TCP (booliano). O valor padrão é `true`.

```
node[:unicorn][:tcp_nodelay]
```

tcp_nopush

Determina se vai habilitar o TCP_CORK (booliano). O valor padrão é `false`.

```
node[:unicorn][:tcp_nopush]
```

timeout

O tempo máximo, em segundos, que um operador tem permissão para usar em cada solicitação (número). Os operadores que excederem o valor de tempo limite serão encerrados. O valor padrão é `60`.

```
node[:unicorn][:timeout]
```

tries

O número máximo de tentativas de associação a um soquete (número). O valor padrão é `5`.

```
node[:unicorn][:tries]
```

version

A versão do Unicorn (string). O valor padrão é `'4.7.0'`.

```
node[:unicorn][:version]
```

worker_processes

O número de processos de operadores (número). O valor padrão é `max_pool_size`, caso ele exista, e `4` caso contrário.

```
node[:unicorn][:worker_processes]
```

Solução de problemas do Chef 11.10 e versões anteriores para Linux

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Para obter mais informações sobre solução de problemas, consulte Guia de depuração e solução de problemas.

Logs do Chef para Chef 11.10 e versões anteriores para Linux

AWS OpsWorks O Stacks armazena os registros do Chef de cada instância em seu `/var/lib/aws/opsworks/chef` diretório. Você precisa de privilégios de sudo para acessar esse diretório. O log para cada execução está em um arquivo chamado *YYYY*-*MM*-*DD*-*HH*-*MM*-*SS*-*NN*.log.

Para obter mais informações, consulte:

- Exibir um Log do Chef com o Console
- Exibir um Log do Chef com CLI ou API
- Interpretar o log do Chef
- Erros comuns do log do Chef

Usando AWS OpsWorks Stacks com outros serviços da AWS

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode fazer com que os servidores de aplicativos em execução em uma pilha do AWS OpsWorks Stacks usem uma variedade de serviços da AWS que não estão diretamente integrados ao AWS OpsWorks Stacks. Por exemplo, você pode fazer com que seus servidores de aplicativos utilizem o Amazon RDS como um banco de dados de back-end. Para acessar esses serviços, use o seguinte padrão geral:

1. Crie e configure o serviço da AWS usando o console, API ou CLI da AWS e registre quaisquer dados de configuração necessários para que o aplicativo acesse o serviço como, por exemplo, o nome do host ou a porta.
2. Crie uma ou mais receitas personalizadas para configurar o aplicativo de forma que ele possa acessar o serviço.

   A receita obtém os dados de configuração dos atributos de configuração da pilha e implantação do JSON que você definir no JSON personalizado antes de executar as receitas.
3. Atribua a receita personalizada ao evento do ciclo de vida de implantação na camada do servidor de aplicativos.
4. Crie um objeto JSON personalizado que atribua os valores apropriados aos atributos de dados de configuração e inclua-o em seu objeto JSON de configuração de pilha e implantação.

5. Faça a implantação do aplicativo na pilha.

   A implantação executa as receitas personalizadas, que por sua vez usam os valores de configuração de dados definidos no JSON personalizado para configurar o aplicativo a fim de que ele possa acessar o serviço.

Esta seção descreve como fazer com que os servidores de aplicativos AWS OpsWorks Stacks acessem uma variedade de serviços da AWS. Pressupõe-se que você já está familiarizado com os livros de receitas do Chef e com a forma como as receitas usam os atributos JSON de pilha e de configuração para configurar aplicativos, normalmente por meio da criação de arquivos de configuração. Caso contrário, você deve primeiro ler Livros de receitas e receitas e Personalizando pilhas AWS OpsWorks.

Tópicos

- Uso de um armazenamento de dados de back-end
- Usando o ElastiCache Redis como um armazenamento de valores-chave na memória
- Usar um bucket do Amazon S3
- Usando AWS CodePipeline com AWS OpsWorks Stacks

Uso de um armazenamento de dados de back-end

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As pilhas de servidores de aplicativos geralmente incluem um servidor de banco de dados para fornecer um armazenamento de dados de back-end. AWS OpsWorks O Stacks fornece suporte integrado para servidores MySQL por meio da camada MySQL e para vários tipos de servidores de banco de dados por meio da camada Amazon Relational Database Service (Amazon RDS). No entanto, você pode personalizar uma pilha para que os servidores do aplicativo use outros servidores de banco de dados como Amazon DynamoDB ou MongoDB. Este tópico descreve o procedimento

básico para conectar um servidor de aplicativo a um servidor de banco de dados da AWS. Ele usa a pilha e o aplicativo de Conceitos básicos das pilhas Linux do Chef 11 para mostrar como conectar manualmente um servidor de aplicativo PHP a um banco de dados RDS. Embora o exemplo tenha por base uma pilha do Linux, os princípios básicos também se aplicam a pilhas do Windows. Para ver um exemplo de como incorporar um servidor de banco de dados MongoDB em uma pilha, consulte Implantando o MongoDB com. OpsWorks

> **Note**
>
> Este tópico usa o Amazon RDS como um exemplo conveniente. Entretanto, se você deseja usar um banco de dados do Amazon RDS com a pilha, é mais fácil usar uma camada do Amazon RDS.

Tópicos

- Como configurar uma Conexão de banco de dados
- Como conectar uma instância do servidor de aplicativos ao Amazon RDS

Como configurar uma Conexão de banco de dados

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode configurar a conexão entre um servidor de aplicativos e o banco de dados de backend usando uma receita personalizada. A receita configura o servidor de aplicativo conforme necessário, normalmente com a criação de um arquivo de configuração. A receita obtém os dados de conexão, como o nome do host e do banco de dados, de um conjunto de atributos na configuração da pilha e nos atributos de implantação que o AWS OpsWorks Stacks instala em cada instância.

Por exemplo, a Etapa 2 de Conceitos básicos das pilhas Linux do Chef 11 é baseada em uma pilha nomeada MyStack com duas camadas, PHP App Server e MySQL, cada uma com uma instância.

Você implanta um aplicativo chamado Simple PHPApp na instância do PHP App Server que usa o banco de dados na instância do MySQL como um armazenamento de dados de back-end. Ao implantar o aplicativo, o AWS OpsWorks Stacks instala a configuração da pilha e os atributos de implantação que contêm as informações de conexão do banco de dados. O exemplo a seguir mostra os atributos da conexão do banco de dados, representado como JSON:

```
{
  ...
  "deploy": {
    "simplephpapp": {
      ...
      "database": {
        "reconnect": true,
        "password": null,
        "username": "root",
        "host": null,
        "database": "simplephpapp"
        ...
      },
      ...
    }
  }
}
```

Os valores dos atributos são fornecidos pelo AWS OpsWorks Stacks e são gerados ou baseados nas informações fornecidas pelo usuário.

Para permitir que o Simple PHPApp acesse o armazenamento de dados, você deve configurar a conexão entre o servidor de aplicativos PHP e o banco de dados MySQL atribuindo uma receita personalizada chamada `appsetup.rb` ao evento de ciclo de vida Deploy da camada PHP App Server. Quando você implanta o SimplePHPApp, o AWS OpsWorks Stacks é executado`appsetup.rb`, o que cria um arquivo de configuração chamado `db-connect.`php que configura a conexão, conforme mostrado no trecho a seguir.

```
node[:deploy].each do |app_name, deploy|
  ...
  template "#{deploy[:deploy_to]}/current/db-connect.php" do
    source "db-connect.php.erb"
```

```
    mode 0660
    group deploy[:group]

    if platform?("ubuntu")
      owner "www-data"
    elsif platform?("amazon")
      owner "apache"
    end

    variables(
      :host =>     (deploy[:database][:host] rescue nil),
      :user =>     (deploy[:database][:username] rescue nil),
      :password => (deploy[:database][:password] rescue nil),
      :db =>       (deploy[:database][:database] rescue nil),
      :table =>    (node[:phpapp][:dbtable] rescue nil)
    )
    ...
  end
end
```

As variáveis que caracterizam a conexão, `host`, `user` e assim por diante, são definidas de acordo com os valores correspondentes dos atributos `[:deploy][:app_name][:database]` de implantação JSON. Para simplificar, o exemplo considera que você já criou uma tabela chamada `urler`, portanto, ela é representada por `[:phpapp][:dbtable]` no arquivo de atributos do livro de receitas.

Essa receita pode conectar o servidor de aplicativos PHP a qualquer servidor de banco de dados MySQL, não apenas membros de uma camada MySQL. Para usar um servidor MySQL diferente, basta definir os `[:database]` atributos para valores apropriados ao seu servidor, o que pode ser feito usando JSON personalizado. AWS OpsWorks Em seguida, as pilhas incorporam esses atributos e valores na configuração e nos atributos de implantação da pilha e os `appsetup.rb` usam para criar o modelo que configura a conexão. Para obter mais informações sobre como substituir as configurações da pilha e a implantação de JSON, consulte Sobrepor atributos.

Como conectar uma instância do servidor de aplicativos ao Amazon RDS

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve como personalizar Conceitos básicos das pilhas Linux do Chef 11 para que o servidor MyStack de aplicativos PHP se conecte a uma instância do RDS.

Tópicos

- Criar um banco de dados MySQL Amazon RDS
- Personalizar a pilha para se conectar ao banco de dados do RDS

Criar um banco de dados MySQL Amazon RDS

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Agora você está pronto para criar um banco de dados do RDS para o exemplo usando o Assistente de instância de banco de dados de inicialização do console Amazon RDS. O procedimento a seguir é um resumo dos detalhes essenciais. Para obter uma descrição detalhada de como criar um banco de dados, consulte Conceitos básicos do Amazon RDS.

Para criar o banco de dados do Amazon RDS

1. Se esta for sua primeira vez criando um banco de dados RDS, clique em Get Started Now. Caso contrário, clique em RDS Dashboard no painel de navegação e, em seguida, em Launch a DB Instance.
2. Selecione MySQL Community Edition como a instância de banco de dados.
3. Para Do you plan to use this database for production purposes? (Você planeja usar esse banco de dados para fins de produção?), selecione No, this instance... (Não, esta instância...), que é suficiente para o exemplo. Para uso de produção, você pode selecionar Yes, use Multi-AZ Deployment.... Clique em Próxima etapa.

4. Na página Specify DB Details, especifique as seguintes configurações:

   - DB Instance Class: db.t2.micro
   - Multi-AZ Deployment: No
   - Allocated Storage (Armazenamento alocado): **5** GB
   - DB Instance Identifier: **`rdsexample`**
   - Master Username (Nome do usuário mestre): **`opsworksuser`**
   - Master Password: especifique uma senha adequada e registre-a para uso posterior.

   Aceite as configurações padrão para as outras opções e, clique em Next Step.

5. Na página Configure Advanced Settings, especifique as seguintes configurações:

   - Na seção Network & Security, para VPC Security Group(s), selecione phpsecgroup (VPC)
   - Na seção Database options (Opções do banco de dados), em Database name (Nome do banco de dados), digite **`rdsexampledb`**
   - Na seção Backup, configure Backup Retention Period como 0 para os fins deste passo a passo.

   Aceite as configurações padrão para as outras opções e, clique em Launch DB Instance.

6. Selecione View Your DB Instances e veja a lista de instâncias de banco de dados.
7. Selecione a instância rdsexample na lista e clique na seta para revelar o endpoint da instância e outros detalhes. Registre o endpoint para uso posterior. Ela será parecida com `rdsexample.c6c8mntzhgv0.us-west-2.rds.amazonaws.com:3306`. Basta registrar o nome DNS; você não precisa do número da porta.
8. Use uma ferramenta como o MySQL Workbench para criar uma tabela chamada `urler` no banco de dados `rdsexampledb` ao usar o seguinte comando SQL:

```
CREATE TABLE urler(id INT UNSIGNED NOT NULL AUTO_INCREMENT,author VARCHAR(63) NOT
 NULL,message TEXT,PRIMARY KEY (id))
```

Personalizar a pilha para se conectar ao banco de dados do RDS

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Depois de criar uma instância do RDS para usar como banco de dados de back-end para o servidor de aplicativos PHP, você pode personalizar a partir de. MyStack Conceitos básicos das pilhas Linux do Chef 11

Para conectar o Servidor de aplicativos PHP a um banco de dados RDS

1. Abra o console AWS OpsWorks Stacks e crie uma pilha com uma camada de PHP App Server que contenha uma instância e implante o SimplePHPApp, conforme descrito em. Conceitos básicos das pilhas Linux do Chef 11 Essa pilha usa a versão 1 do SimplePHPApp, que não usa uma conexão de banco de dados.
2. Atualize a configuração da pilha para usar os livros de receitas personalizados que incluem a receita `appsetup.rb`, além do modelo relacionado e os arquivos de atributo.
   1. Defina Use custom Chef cookbooks como Yes.
   2. Defina Repository type como Git e Repository URL como `git://github.com/amazonwebservices/opsworks-example-cookbooks.git`.
3. Adicione a caixa Custom Chef JSON da pilha para atribuir os dados de conexão do RDS aos atributos `[:database]` que `appsetup.rb` utiliza para criar o arquivo de configuração.

```
{
  "deploy": {
    "simplephpapp": {
      "database": {
        "username": "opsworksuser",
        "password": "your_password",
        "database": "rdsexampledb",
        "host": "rds_endpoint",
```

```
        "adapter": "mysql"
      }
    }
  }
}
```

Use os seguintes valores de atributo:

- username: o nome de usuário principal que você especificou ao criar a instância do RDS.

  Este exemplo usa `opsworksuser`.
- password: a senha mestre que você especificou quando criou a instância do RDS.

  Preencha com a senha que você especificou.
- database: o banco de dados que você criou ao criar a instância de RDS.

  Este exemplo usa `rdsexampledb`.
- host: o endpoint RDS da instância, obtido no console RDS ao criar a instância na seção anterior. Não inclua o número da porta.
- adapter: o adaptador.

  A instância RDS deste exemplo usa MySQL, portanto adapter está definido como `mysql`. Diferentemente de outros atributos, adapter não é usado por `appsetup.rb`. Em vez disso, é usado pela receita Configurar interna da camada PHP App Server para criar um arquivo de configuração diferente.

4. Edite a PHPApp configuração Simple para especificar uma versão do Simple PHPApp que usa um banco de dados back-end, da seguinte forma:

   - Document root: defina essa opção como `web`.
   - Branch/Revision: defina essa opção como `version2`.

   Deixe as opções restantes inalteradas.
5. Edita a camada PHP App Server para configurar a conexão do banco de dados ao adicionar `phpapp::appsetup` para as receitas de implantação da camada.
6. Implante a nova PHPApp versão Simple.

7. Quando o Simple PHPApp for implantado, execute o aplicativo acessando a página Instâncias e clicando no endereço IP público da instância php-app1. Você deve visualizar a seguinte página no navegador, que permite inserir texto e armazená-lo no banco de dados.

> **Note**
>
> Se sua pilha tiver uma camada MySQL AWS OpsWorks , o Stacks atribuirá automaticamente os dados de conexão correspondentes aos atributos. `[:database]` No entanto, se você atribuir o JSON personalizado à pilha que define valores diferentes de `[:database]`, eles substituem os valores padrão. Como os atributos `[:deploy]` são instalados em cada instância, qualquer receita que depende dos atributos `[:database]` usará os dados de conexão personalizados e não os dados da camada MySQL. Se você deseja uma camada de servidor de aplicativo específico para usar os dados de conexão personalizados, atribua o JSON personalizado ao evento Implantar da camada e restrinja a implantação à camada. Para obter mais informações sobre como usar os atributos de implantação, consulte Implementação de aplicativos. Para obter mais informações sobre substituir os atributos internos do AWS OpsWorks Stacks, consulte Sobrepor atributos.

Usando o ElastiCache Redis como um armazenamento de valores-chave na memória

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Este tópico é baseado em uma pilha Linux, mas as pilhas do Windows também podem usar Amazon ElastiCache ()ElastiCache. Para obter um exemplo de como usar ElastiCache com uma instância do Windows, consulte ElastiCache como um repositório de sessões do ASP.NET.

Você pode, muitas vezes, melhorar o desempenho do servidor de aplicativos usando um servidor de armazenamento em cache para fornecer um armazenamento de chave-valor para pequenos itens de dados, como strings. ElastiCache A Amazon é um serviço da AWS que facilita o fornecimento de suporte de cache para seu servidor de aplicativos, usando os mecanismos de cache Memcached ou Redis. AWS OpsWorks O Stacks fornece suporte integrado para o Memcached. No entanto, se o Redis atender melhor às suas necessidades, você poderá personalizar sua pilha para que seus servidores de aplicativos ElastiCache usem o Redis.

Este tópico explica o processo básico de fornecer suporte de cache do ElastiCache Redis para pilhas Linux, usando um servidor de aplicativos Rails como exemplo. Ele pressupõe que você já tenha um aplicativo Ruby on Rails apropriado. Para obter mais informações sobre ElastiCache, consulte O que é a Amazon ElastiCache? .

Tópicos

- Etapa 1: criar um cluster ElastiCache Redis
- Etapa 2: Configurar uma pilha do Rails
- Etapa 3: criar e implantar um livro de receitas personalizado

- Etapa 4: atribuir a receita a um LifeCycle evento
- Etapa 5: Adicionar informações de acesso ao JSON de configuração da pilha
- Etapa 6: Implantar e executar o aplicativo

Etapa 1: criar um cluster ElastiCache Redis

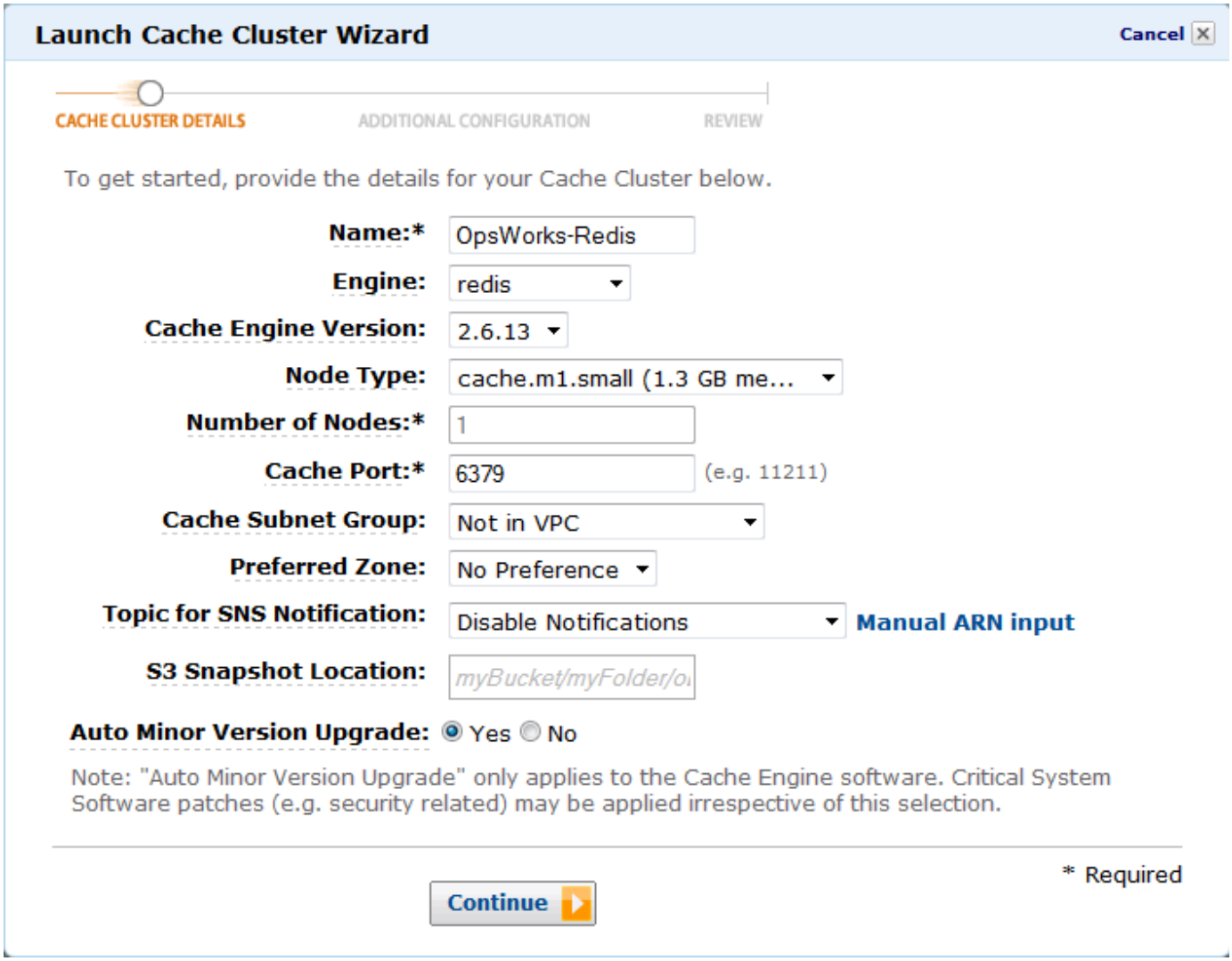
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Primeiro, você deve criar um cluster Amazon ElastiCache Redis usando o ElastiCache console, a API ou a CLI. Veja a seguir uma descrição de como usar o console para criar um cluster.

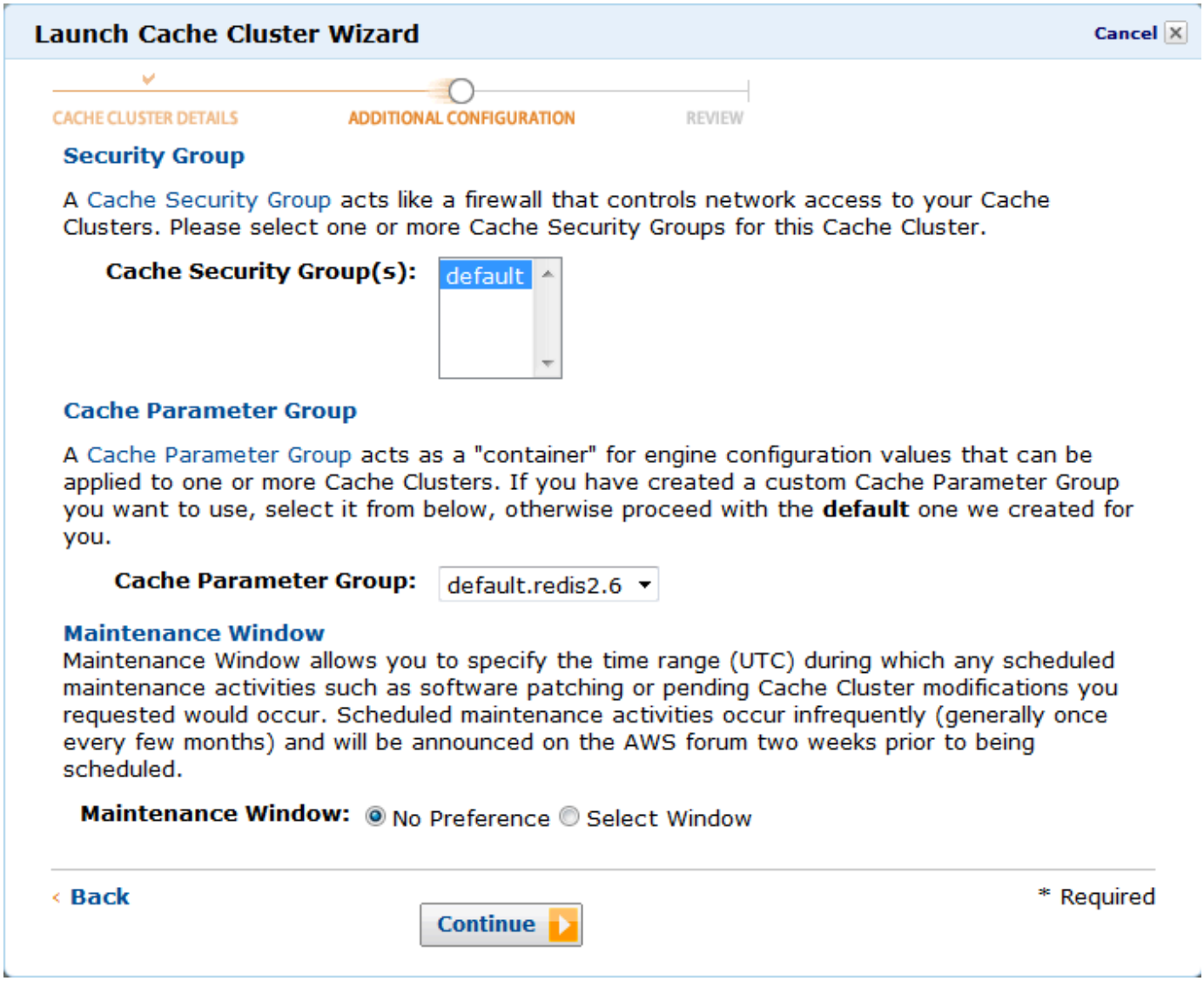
Para criar um cluster ElastiCache Redis

1. Acesse o console do ElastiCache e clique em Launch Cache Cluster (Executar cluster de cache) para iniciar o assistente Cache Cluster (Cluster de cache).
2. Na página Cache Cluster Details, faça o seguinte:
   - Defina Name para o nome do servidor de cache.

     Este exemplo usa OpsWorks -Redis.
   - Defina Engine como redis.
   - Defina Topic for SNS Notification como Disable Notifications.
   - Aceite os valores padrão para as outras configurações e clique em Continue.



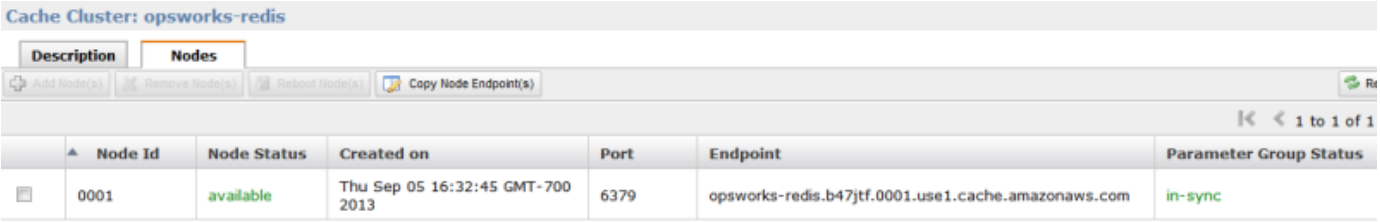
3. Na página Additional Configuration, aceite os valores padrão e clique em Continue.



4. Clique em Launch Cache Cluster para criar o cluster.

> **Important**
>
> O grupo de segurança de cache padrão é suficiente para este exemplo, mas, para uso em produção, você deve criar um que seja apropriado para seu ambiente. Para obter mais informações, consulte Gerenciamento de grupos de segurança de cache.

5. Após o cluster ter sido iniciado, clique no nome para abrir a página de detalhes e clique na guia Nodes. Anote os valores de Port e Endpoint do cluster para uso posterior.



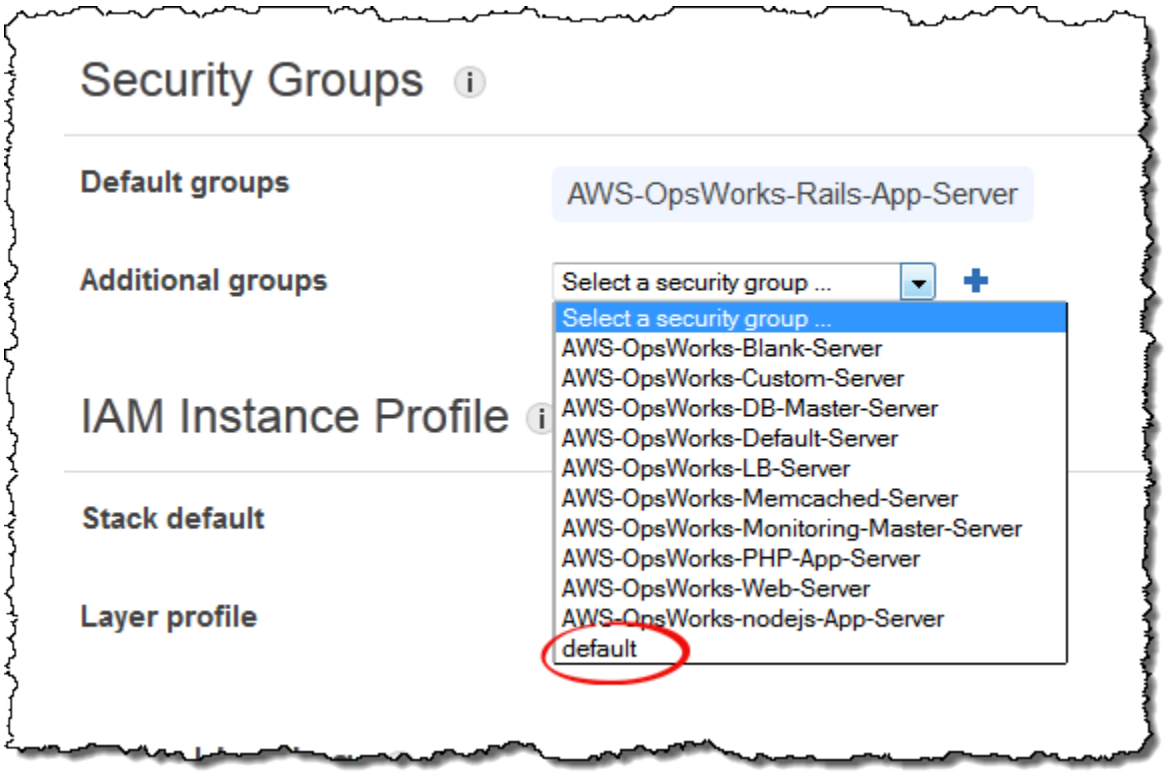
Etapa 2: Configurar uma pilha do Rails

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Além de criar uma pilha que oferece suporte a uma camada Rail App Server, também é necessário configurar grupos de segurança da camada, de modo que o servidor Rails possa se comunicar corretamente com o servidor Redis.

Para configurar uma pilha

1. Crie uma nova pilha, nomeada **`RedisStack`** para este exemplo, e adicione uma camada do Rails App Server. Você pode usar as configurações padrão para ambos. Para ter mais informações, consulte Criar uma nova pilha e Criando uma OpsWorks camada .
2. Na página Camadas, para Rails App Server, clique em Segurança e, em seguida, clique em Editar.
3. Vá para a seção Security Groups e adicione o grupo de segurança do cluster ElastiCache a Additional groups (Grupos adicionais). Para este exemplo, selecione o grupo de segurança default, clique em + para adicioná-lo à camada e clique em Save para salvar a nova configuração.



4. Adicione uma instância à camada Rails App Server e inicialize-a. Para obter mais informações sobre como adicionar e inicializar instâncias, consulte Adicionar uma instância a uma camada.

Etapa 3: criar e implantar um livro de receitas personalizado

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Do jeito que está, a pilha não é muito funcional ainda. É necessário habilitar seu aplicativo para acessar o servidor Redis. A abordagem mais flexível é colocar um arquivo YAML com as informações de acesso na subpasta `config` do aplicativo. Então, o aplicativo pode obter as informações do arquivo. Usando essa abordagem, você pode alterar as informações de conexão sem reescrever e reimplantar o aplicativo. Para este exemplo, o arquivo deve ser chamado `redis.yml`e conter o nome do host e a porta do cluster do ElastiCache , da seguinte forma:

```
host: cache-cluster-hostname
port: cache-cluster-port
```

Você pode copiar manualmente esse arquivo para seus servidores, mas uma abordagem melhor é implementar uma receita do Chef para gerar o arquivo e fazer com que o AWS OpsWorks Stacks execute a receita em cada servidor. As receitas do Chef são aplicativos Ruby especializados que o AWS OpsWorks Stacks usa para realizar tarefas em instâncias, como instalar pacotes ou criar arquivos de configuração. Receitas são incluídos em um livro de receitas, que pode conter várias receitas e arquivos relacionados, como modelos para arquivos de configuração. O livro de receitas é colocado em um repositório, como GitHub, e deve ter uma estrutura de diretórios padrão. Se você ainda não tem um repositório personalizado de livros de receitas, consulte Repositórios de livro de receitas para informações sobre como configurar um.

Para este exemplo, adicione um livro de receitas chamado `redis-config` em seu repositório de livros de receitas com o seguinte conteúdo:

```
my_cookbook_repository
  redis-config
    recipes
      generate.rb
    templates
      default
        redis.yml.erb
```

A pasta `recipes` contém uma receita chamada `generate.rb`, que gera o arquivo de configuração do aplicativo a partir de `redis.yml.erb`, da seguinte forma:

```
node[:deploy].each do |app_name, deploy_config|
  # determine root folder of new app deployment
  app_root = "#{deploy_config[:deploy_to]}/current"

  # use template 'redis.yml.erb' to generate 'config/redis.yml'
  template "#{app_root}/config/redis.yml" do
    source "redis.yml.erb"
    cookbook "redis-config"

    # set mode, group and owner of generated file
```

```
    mode "0660"
    group deploy_config[:group]
    owner deploy_config[:user]

    # define variable “@redis” to be used in the ERB template
    variables(
      :redis => deploy_config[:redis] || {}
    )

    # only generate a file if there is Redis configuration
    not_if do
      deploy_config[:redis].blank?
    end
  end
end
```

A receita depende dos dados do objeto JSON de configuração e implantação da pilha AWS OpsWorks Stacks, que é instalado em cada instância e contém informações detalhadas sobre a pilha e quaisquer aplicativos implantados. O nó `deploy` do objeto tem a seguinte estrutura:

```
{
   ...
  "deploy": {
    "app1": {
      "application" : "short_name",
      ...
    }
    "app2": {
      ...
    }
    ...
  }
}
```

O nó de implantação contém um conjunto de objetos JSON incorporados, um para cada aplicativo implementado, que é chamado pelo nome curto do aplicativo. Cada objeto do aplicativo contém um conjunto de atributos que definem a configuração do aplicativo, como a raiz do documento e o tipo de aplicativo. Para uma lista de atributos de implantação, consulte Atributos deploy. As receitas podem usar a sintaxe de atributos do Chef para representar valores JSON de configuração e implantação da

pilha. Por exemplo,`[:deploy][:app1][:application]` representa o nome curto do aplicativo app1.

Para cada aplicativo em `[:deploy]`, a receita executa o bloco de código associado, em que `deploy_config` representa o atributo do aplicativo. A receita define primeiro `app_root` para o diretório raiz do aplicativo, `[:deploy][:app_name][:deploy_to]/current`. Em seguida, usa um modelo de recurso do Chef para gerar um arquivo de configuração a partir de `redis.yml.erb` e colocá-lo em `app_root/config`.

Os arquivos de configuração normalmente são criados a partir de modelos, com muitas, se não a maioria, das configurações definidas pelos atributos do Chef. Com atributos personalizados, você pode alterar as configurações usando JSON personalizado, conforme descrito posteriormente, em vez de reescrever o arquivo de modelo. O modelo `redis.yml.erb` contém o seguinte:

```
host: <%= @redis[:host] %>
port: <%= @redis[:port] || 6379 %>
```

Os elementos **<%... %>** são espaços reservados que representam um valor de atributo.

- `<%= @redis[:host] %>` representa o valor de `redis[:host]`, que é o nome de host do cluster de cache.
- `<%= @redis[:port] || 6379 %>` representa o valor de `redis[:port]` ou, se esse atributo não estiver definido, o valor da porta padrão, 6379.

O recurso `template` funciona da seguinte forma:

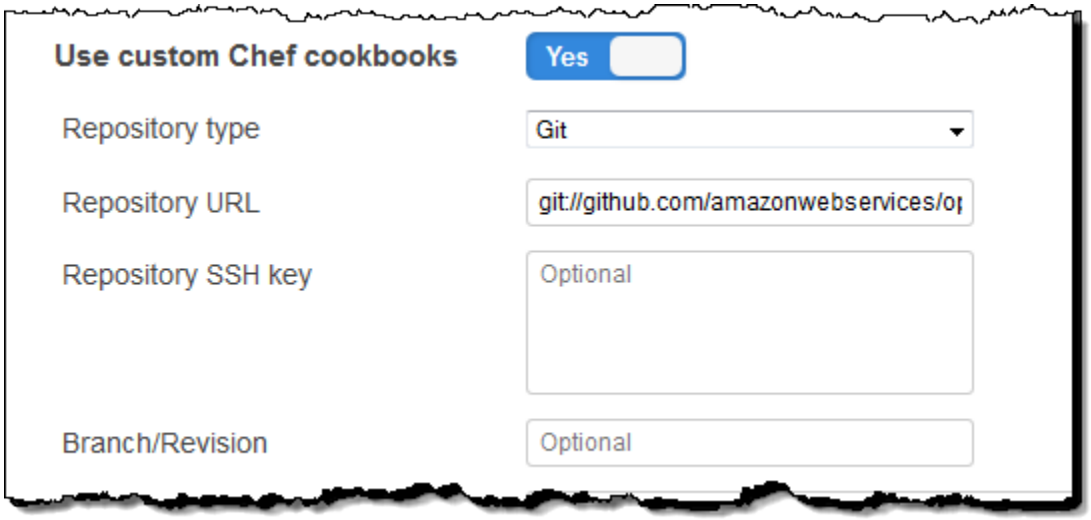
- `source` e `cookbook` especificam o modelo e os nomes de livros de receitas, respectivamente.
- `mode`, `groupe` `owner` fornecem o arquivo de configuração os mesmos direitos de acesso do aplicativo.
- A seção `variables` define a variável `@redis` usada no modelo, para o valor do atributo `[:redis]` do aplicativo.

  Os valores do atributo `[:redis]` são definidos usando um JSON personalizado, conforme descrito posteriormente. Ele não é um dos atributos de aplicativo padrão.
- A diretiva `not_if` garante que a receita não gere um arquivo de configuração se já existir outro.

Depois de criar o livro de receitas, você deve implantá-lo no cache do livro de receitas de cada instância. Essa operação não executa a receita; ela simplesmente instala o novo livro de receitas nas instâncias da pilha. Normalmente, você executa uma receita atribuindo-a a um evento de ciclo de vida da camada , conforme descrito posteriormente.

Para implantar seu livro de receitas personalizado

1. Na página AWS OpsWorks Stacks Stack, clique em Configurações da pilha e depois em Editar.
2. Na seção Configuration Management, defina Use custom Chef cookbooks como Yes, insira as informações do repositório do livro de receitas e clique em Save para atualizar as configurações da pilha.



3. Na página Stack, clique em Run Command, selecione o comando de pilha Update Custom Cookbooks e clique em Update Custom Cookbooks para instalar o novo livro de receitas nos caches de livros de receitas das instâncias.

Se você modificar seu livro de receitas, basta executar Update Custom Cookbooks novamente para instalar a versão atualizada. Para obter mais informações sobre esse procedimento, consulte Instalação de livros de receitas personalizados.

Etapa 4: atribuir a receita a um LifeCycle evento

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

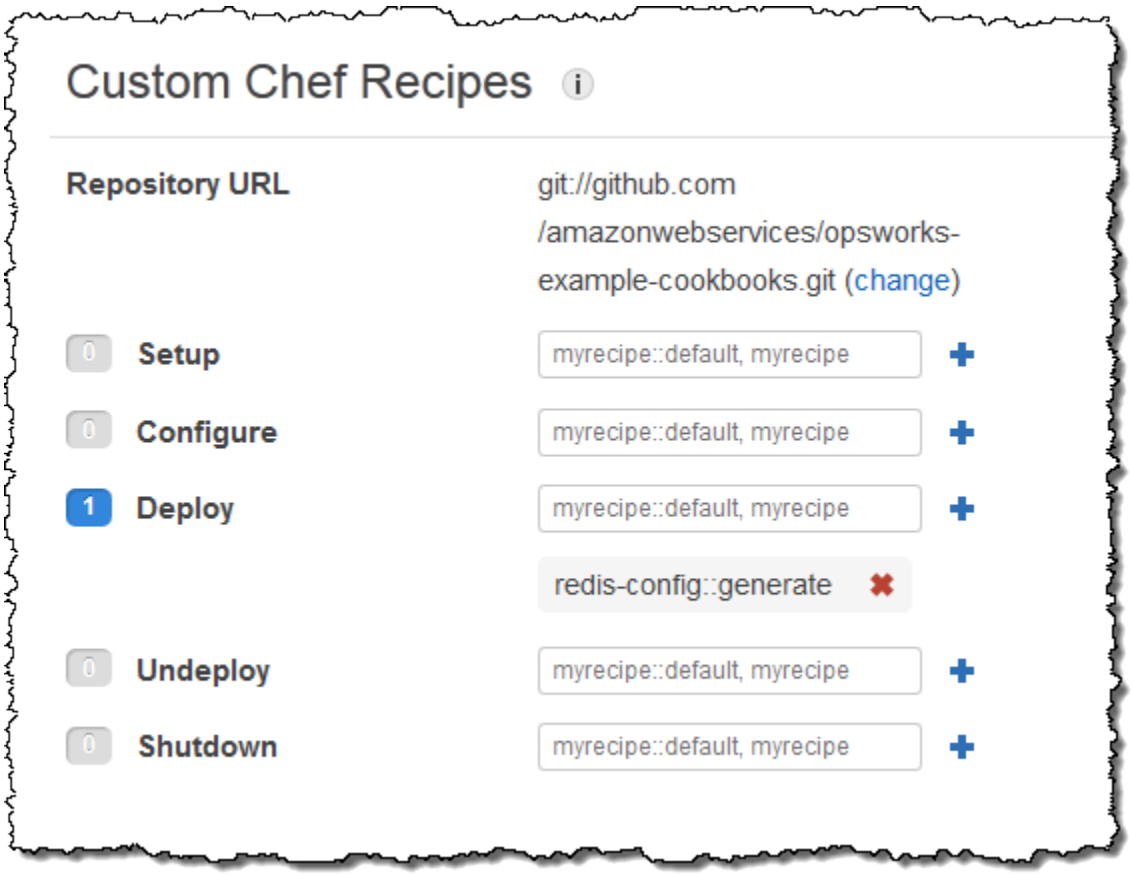
Você pode executar receitas personalizadas manualmente, mas a melhor abordagem geralmente é fazer com que AWS OpsWorks as pilhas as executem automaticamente. Cada camada tem um conjunto de receitas incorporadas atribuído a cada um dos cinco eventos de ciclo de vida: Setup, Configure, Deploy, Undeploy e Shutdown. Cada vez que ocorre um evento para uma instância, o AWS OpsWorks Stacks executa as receitas associadas a cada uma das camadas da instância, que lidam com suas respectivas tarefas. Por exemplo, quando a inicialização de uma instância é

concluída, o AWS OpsWorks Stacks aciona um evento de configuração. Esse evento executa as receitas de configuração da camada associada, que normalmente gerencia tarefas como instalação e configuração de pacotes.

Você pode fazer com que o AWS OpsWorks Stacks execute uma receita personalizada nas instâncias de uma camada atribuindo a receita ao evento de ciclo de vida apropriado. Neste exemplo, você deve atribuir a `generate.rb` receita ao evento Deploy da camada Rails App Server. AWS OpsWorks Em seguida, as pilhas serão executadas nas instâncias da camada durante a inicialização, após a conclusão das receitas de configuração e toda vez que você implantar um aplicativo. Para obter mais informações, consulte Execução automática de receitas.

Para associar uma receita ao evento Implantar da camada do Rails App Server

1. Na página AWS OpsWorks Stacks Layers, para Rails App Server, clique em Receitas e, em seguida, clique em Editar.
2. Em Custom Chef Recipes, adicione o nome da receita totalmente qualificado ao evento de implantação e clique em +. Um nome de receita totalmente qualificado usa o formato *`cookbookname`*`::`*`recipename`* , em que *`recipename`* não inclui a extensão `.rb`. Neste exemplo, o nome totalmente qualificado é `redis-config::generate`. Em seguida, clique em Save para atualizar a configuração da layer.



Etapa 5: Adicionar informações de acesso ao JSON de configuração da pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A receita `generate.rb` depende de um par de atributos JSON de configuração e implantação da pilha JSON que representam o nome de host e a porta do servidor Redis. Embora esses atributos façam parte do `[:deploy]` namespace padrão, eles não são definidos automaticamente pelo AWS OpsWorks Stacks. Em vez disso, você define os atributos e seus valores adicionando um objeto JSON personalizado à pilha. O exemplo a seguir mostra o JSON personalizado para este exemplo.

Para adicionar informações de acesso ao JSON de configuração e implantação da pilha

1. Na página AWS OpsWorks Stacks Stack, clique em Configurações da pilha e depois em Editar.
2. Na seção Configuration Management, adicione as informações de acesso à caixa Custom Chef JSON. Ela deve ser parecida com o exemplo a seguir, com estas modificações:
   - Substitua `elasticache_redis_example` pelo nome curto de seu aplicativo.
   - Substitua `port` os valores `host` e pelos valores da instância do servidor ElastiCache Redis que você criou emEtapa 1: criar um cluster ElastiCache Redis.

```
{
  "deploy": {
     "elasticache_redis_example": {
       "redis": {
         "host": "mycluster.XXXXXXXXX.amazonaws.com",
         "port": "6379"
       }
     }
  }
}
```

A vantagem dessa abordagem é que você pode alterar o valor da porta ou do host a qualquer momento sem tocar em seu livro de receitas personalizado. AWS OpsWorks O Stacks mescla o JSON personalizado com o JSON integrado e o instala nas instâncias da pilha para todos os

eventos subsequentes do ciclo de vida. Os aplicativos podem, então, acessar os valores de atributos usando a sintaxe de nó do Chef, conforme descrito em Etapa 3: criar e implantar um livro de receitas personalizado. Na próxima vez que você implementar um aplicativo, o AWS OpsWorks Stacks instalará um JSON de configuração e implantação de pilha que contém as novas definições e `generate.rb` criará um arquivo de configuração com os valores de host e porta atualizados..

> **Note**
>
> `[:deploy]` inclui automaticamente um atributo para cada aplicativo implantado, de modo que `[:deploy][elasticache_redis_example]` já está na pilha e no JSON de configuração JSON. No entanto, `[:deploy][elasticache_redis_example]` não inclui um `[:redis]` atributo, defini-los com JSON personalizado direciona as AWS OpsWorks pilhas às quais adicionar esses atributos. `[:deploy][elasticache_redis_example]` Você também pode usar o JSON personalizado para substituir atributos existentes. Para obter mais informações, consulte Sobrepor atributos.

Etapa 6: Implantar e executar o aplicativo

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Este exemplo pressupõe que você tenha o aplicativo Ruby on Rails que usa o Redis. Para acessar o arquivo de configuração, você pode adicionar o gem `redis` ao seu Gemfile e criar um inicializador Rails em `config/initializers/redis.rb` da seguinte forma:

```
REDIS_CONFIG = YAML::load_file(Rails.root.join('config', 'redis.yml'))
$redis = Redis.new(:host => REDIS_CONFIG['host'], :port => REDIS_CONFIG['port'])
```

Em seguida, crie um aplicativo para representar seu aplicativo e implementá-lo nas instâncias da camada do Rails App Server, que atualiza o código do aplicativo e executa `generate.rb` para gerar

o arquivo de configuração. Quando você executa o aplicativo, ele usa a instância do ElastiCache Redis como seu armazenamento de valores-chave na memória.

Usar um bucket do Amazon S3

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Os aplicativos, muitas vezes, utilizam um bucket do Amazon Simple Storage Service (Amazon S3) para armazenar grandes itens, como imagens ou outros arquivos de mídia. Embora o AWS OpsWorks Stacks não forneça suporte integrado para o Amazon S3, você pode personalizar facilmente uma pilha para permitir que a aplicação utilize o armazenamento do Amazon S3. Este tópico oferece orientação sobre como o processo básico de fornecer acesso a aplicativos ao Amazon S3, usando uma pilha do Linux com um servidor de aplicativos PHP como um exemplo. Os princípios básicos também se aplicam a pilhas do Windows.

O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Tópicos

- Etapa 1: crie um bucket do Amazon S3
- Etapa 2: criar uma PHP App Server Stack
- Etapa 3: criar e implantar um livro de receitas personalizado
- Etapa 4: Atribuir receitas aos LifeCycle eventos
- Etapa 5: adicionar informações de acesso aos atributos de configuração e implantação da pilha
- Etapa 6: implantar e executar PhotoApp

Etapa 1: crie um bucket do Amazon S3

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Primeiro, é necessário criar um bucket do Amazon S3. Você pode fazer isso diretamente usando a API, a CLI ou o console do Amazon S3. Porém, uma maneira mais simples de criar recursos normalmente é usar um modelo do AWS CloudFormation . O modelo a seguir cria um bucket do Amazon S3 para este exemplo e configura o perfil de instância com um perfil do IAM que concede acesso irrestrito ao bucket. Em seguida, você pode usar uma camada para anexar o perfil da instância às instâncias do servidor de aplicativos da pilha, o que permite que o aplicativo acesse o bucket, conforme descrito mais adiante. A utilidade de perfis de instância não fica limitada ao Amazon S3. Eles são valiosos para integrar uma variedade de serviços da AWS.

```
{
   "AWSTemplateFormatVersion" : "2010-09-09",
   "Resources" : {
      "AppServerRootRole": {
         "Type": "AWS::IAM::Role",
         "Properties": {
            "AssumeRolePolicyDocument": {
               "Statement": [ {
                  "Effect": "Allow",
                  "Principal": {
                     "Service": [ "ec2.amazonaws.com" ]
                  },
                  "Action": [ "sts:AssumeRole" ]
               } ]
            },
            "Path": "/"
         }
      },
      "AppServerRolePolicies": {
```

```
         "Type": "AWS::IAM::Policy",
         "Properties": {
            "PolicyName": "AppServerS3Perms",
            "PolicyDocument": {
               "Statement": [ {
                  "Effect": "Allow",
                  "Action": "s3:*",
                  "Resource": { "Fn::Join" : ["", [ "arn:aws:s3:::", { "Ref" :
 "AppBucket" } , "/*" ]
                  ] }
               } ]
            },
            "Roles": [ { "Ref": "AppServerRootRole" } ]
         }
      },
      "AppServerInstanceProfile": {
         "Type": "AWS::IAM::InstanceProfile",
         "Properties": {
            "Path": "/",
            "Roles": [ { "Ref": "AppServerRootRole" } ]
         }
      },
     "AppBucket" : {
      "Type" : "AWS::S3::Bucket"
      }
   },
   "Outputs" : {
       "BucketName" : {
           "Value" : { "Ref" : "AppBucket" }
       },
       "InstanceProfileName" : {
           "Value" : { "Ref" : "AppServerInstanceProfile" }
       }
   }
}
```

Várias coisas acontecem quando você inicia o modelo:

- O recurso `AWS::S3::Bucket` cria um bucket do Amazon S3.
- O recurso `AWS::IAM::InstanceProfile` cria um perfil de instância que será atribuído à instâncias do servidor de aplicativos.
- O recurso `AWS::IAM::Role` cria a função do perfil de instância.

- O recurso `AWS::IAM::Policy` define as permissões do perfil para conceder acesso irrestrito aos buckets do Amazon S3.
- A seção `Outputs` exibirá os nomes do bucket e do perfil de instância no console do AWS CloudFormation depois que você iniciar o modelo.

  Você precisará desses valores para configurar sua pilha e aplicativo.

Para obter mais informações sobre como criar AWS CloudFormation modelos, consulte Learn Template Basics.

Criar o bucket do Amazon S3

1. Copie o modelo de exemplo para um arquivo de texto no sistema.

   Este exemplo pressupõe que o arquivo é chamado `appserver.template`.
2. Abra o console do AWS CloudFormation e selecione Create Stack (Criar pilha).
3. Na caixa Stack Name, insira o nome da pilha.

   Este exemplo pressupõe que o nome é **`AppServer`**.
4. Selecione Upload template file (Carregar arquivo de modelo), selecione Browse (Navegar), selecione o arquivo `appserver.template` que você criou na etapa 1 e, em seguida, selecione Next Step (Próxima etapa).
5. Na página Specify Parameters (Especificar parâmetros), selecione I acknowledge that this template may create IAM resources (Estou ciente de que esse modelo pode criar recursos do IAM) e, em seguida, selecione Next Step (Próxima etapa) em todas as páginas do assistente até chegar à página final. Escolha Criar.
6. Depois que a AppServerpilha atingir o status CREATE_COMPLETE, selecione-a e escolha a guia Saídas.

   Talvez você precise atualizar algumas vezes para atualizar o status.
7. Na guia Saídas, BucketNameregistre os InstanceProfileNamevalores e para uso posterior.

> **Note**
>
> AWS CloudFormation usa o termo pilha para se referir à coleção de recursos que são criados a partir de um modelo; não é o mesmo que uma pilha de AWS OpsWorks pilhas.

Etapa 2: criar uma PHP App Server Stack

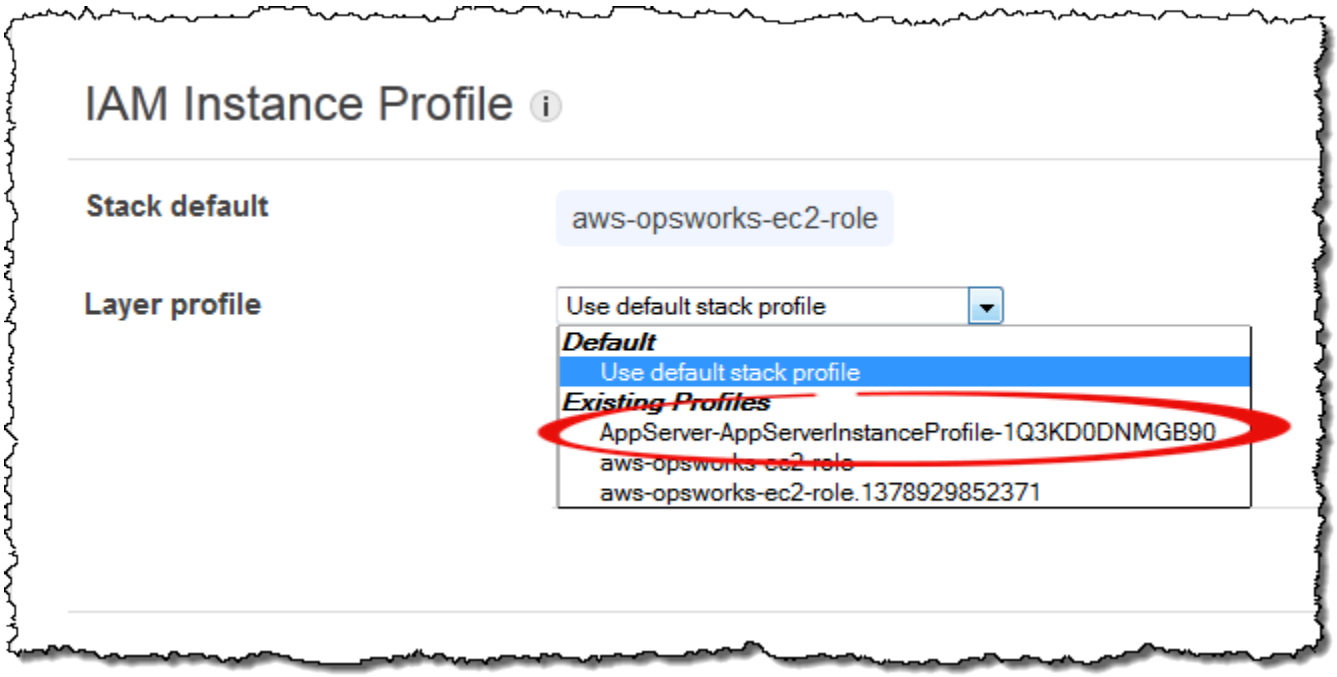
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A pilha consiste em duas camadas, PHP App Server e MySQL, cada uma com uma instância. O aplicativo armazena as fotos em um bucket do Amazon S3, mas usa a instância do MySQL como um armazenamento de dados de back-end para manter metadados para cada foto.

O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Para criar a pilha

1. Crie uma nova pilha, nomeada **`PhotoSite`** para este exemplo, e adicione uma camada PHP App Server. Você pode usar as configurações padrão para ambos. Para obter mais informações, consulte Criar uma nova pilha e Criando uma OpsWorks camada .
2. Na página Camadas, para PHP App Server selecione Segurança e, em seguida, selecione Editar.
3. Na seção Perfil de camada, selecione o nome do perfil da instância que você gravou anteriormente, depois de iniciar a AppServer AWS CloudFormation pilha. Será mais ou menos assim`AppServer-AppServerInstanceProfile-1Q3KD0DNMGB90`. AWS OpsWorks O Stacks atribui esse perfil a todas as EC2 instâncias da Amazon da camada, o que concede permissão para acessar seu bucket do Amazon S3 aos aplicativos em execução nas instâncias da camada.



4. Adicione uma instância à camada PHP App Server e inicialize-a. Para obter mais informações sobre como adicionar e inicializar instâncias, consulte Adicionar uma instância a uma camada.
5. Adicione uma camada MySQL à pilha, adicione uma instância e inicie-a. Você pode usar as configurações padrão para a camada e a instância. Especificamente, a instância do MySQL não precisa acessar o bucket do Amazon S3, assim ele pode usar o perfil de instância AWS OpsWorks padrão do Stacks, que é selecionado por padrão.

Etapa 3: criar e implantar um livro de receitas personalizado

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A pilha não está exatamente pronta ainda:

- O aplicativo precisa de algumas informações para acessar o servidor de banco de dados MySQL e o bucket do Amazon S3, como o nome de host do banco de dados e o nome do bucket do Amazon S3.
- Você precisa configurar um banco de dados no servidor de banco de dados MySQL e criar uma tabela para armazenar os metadados das fotos.

Você poderia lidar com essas tarefas manualmente, mas uma abordagem melhor é implementar a receita do Chef e fazer com que o AWS OpsWorks Stacks execute a receita automaticamente nas instâncias apropriadas. As receitas do Chef são aplicativos Ruby especializados que o AWS OpsWorks Stacks usa para realizar tarefas em instâncias, como instalar pacotes ou criar arquivos de configuração. Elas estão incluídas em um livro de receitas, que pode conter várias receitas e arquivos relacionados, como modelos para arquivos de configuração. O livro de receitas é colocado em um repositório como GitHub, e deve ter uma estrutura de diretórios padrão. Se você ainda não tem um repositório personalizado de livros de receitas, consulte Repositórios de livro de receitas para informações sobre como configurar um.

Neste exemplo, o livro de receitas foi implementado para você e está armazenado em um GitHub repositório público. O livro de receitas contém duas receitas, `appsetup.rb` e `dbsetup.rb`, além de um arquivo de modelo, `db-connect.php.erb`.

A receita `appsetup.rb` cria um arquivo de configuração que contém as informações que o aplicativo precisa para acessar o banco de dados e o bucket do Amazon S3. Ele é basicamente uma versão um pouco modificada da `appsetup.rb` receita descrita em Conexão do aplicativo ao banco de dados. A principal diferença são as variáveis que são passadas para o modelo, que representam as informações de acesso.

Os primeiros quatro atributos definem configurações de conexão de banco de dados, e são definidos automaticamente pelo AWS OpsWorks Stacks quando você cria a instância do MySQL.

Há duas diferenças entre essas variáveis e as contidas na receita original:

- Assim como a receita original, a variável `table` representa o nome da tabela do banco de dados que é criada por `dbsetup.rb` e está ajustada para o valor de um atributo definido no arquivo de atributos do livro de receitas.

  No entanto, o atributo tem um nome diferente: `[:photoapp][:dbtable]`.
- A variável `s3bucket` é específica deste exemplo e está ajustada para o valor de um atributo que representa o nome do bucket do Amazon S3, `[:photobucket]`.

`[:photobucket]` é definido usando JSON personalizado, conforme descrito mais adiante. Para obter mais informações sobre atributos, consulte Atributos.

Para obter mais informações sobre atributos, consulte Atributos.

A receita `dbsetup.rb` configura uma tabela de banco de dados para armazenar todos os metadados de fotos. Ele é basicamente uma versão um pouco modificada da receita `dbsetup.rb` descrita em Configuração do banco de dados. Consulte esse tópico para obter uma descrição detalhada.

A única diferença entre este exemplo e a receita original é o esquema do banco de dados, que tem três colunas contendo o ID, o URL e a legenda de cada foto armazenada no bucket do Amazon S3.

As receitas já estão implementados, portanto, tudo o que você precisa fazer é implantar o livro de receitas do photoapp para cada cache de livro de receitas da instância. AWS OpsWorks Em seguida, o Stacks executará as receitas em cache quando o evento de ciclo de vida adequado ocorrer, conforme descrito posteriormente.

Para implementar o livro de receitas do photoapp

1. Na página AWS OpsWorks Stacks Stack, escolha Stack Settings e, em seguida, escolha Editar.
2. Na seção Configuration Management:
   - Defina Use custom Chef cookbooks como Yes.
   - Defina Repository type como "Git".
   - Defina Repository URL (URL do repositório). como **`git://github.com/amazonwebservices/opsworks-example-cookbooks.git`**.
3. Na página Stack (Pilha), selecione Run Command, selecione o comando de pilha Update Custom Cookbooks (Atualizar livros de receitas personalizados) e, em seguida, selecione Update Custom Cookbooks (Atualizar livros de receitas personalizados) para instalar o novo livro de receitas nos caches de livros de receitas de instância.

Etapa 4: Atribuir receitas aos LifeCycle eventos

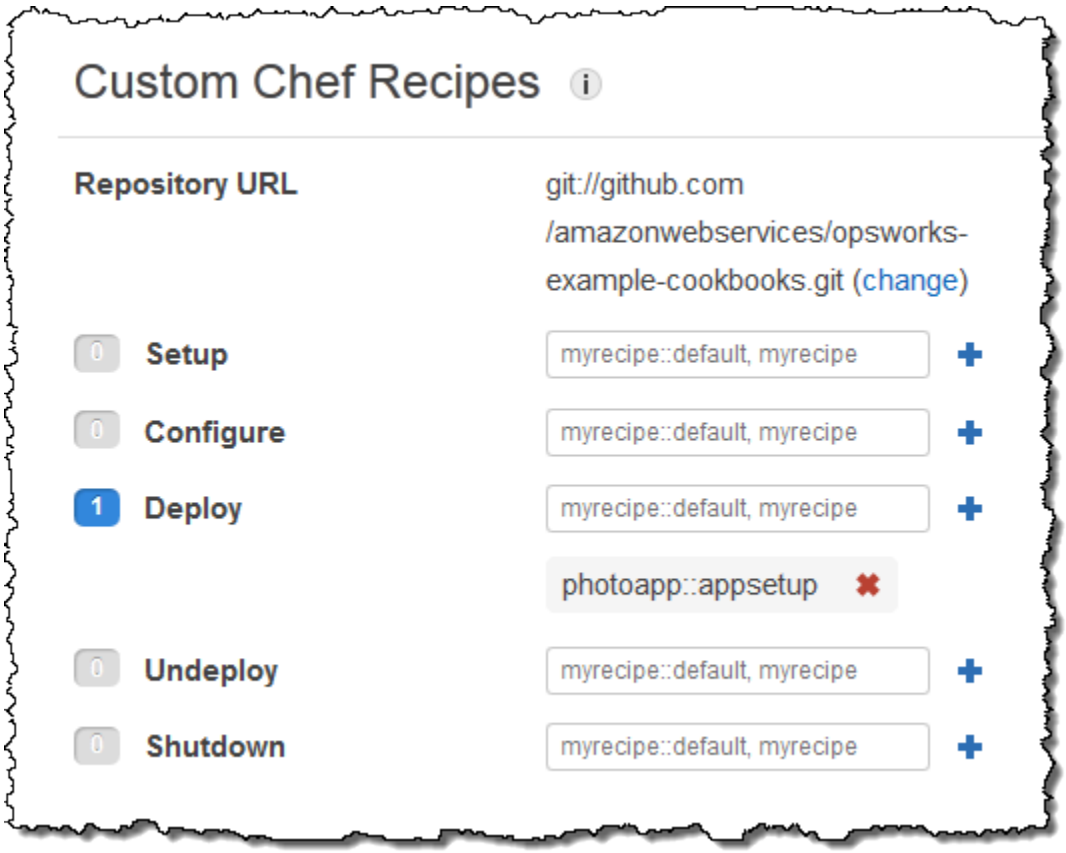
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Você pode executar receitas personalizadas manualmente, mas a melhor abordagem geralmente é fazer com que AWS OpsWorks as pilhas as executem automaticamente. Cada camada tem um conjunto de receitas incorporadas atribuído a cada um dos cinco eventos de ciclo de vida: Setup, Configure, Deploy, Undeploy e Shutdown. Sempre que um evento ocorre em uma instância, o AWS OpsWorks Stacks executa as receitas associadas para cada uma das camadas da instância, que processam as tarefas necessárias. Por exemplo, quando a inicialização de uma instância é concluída, o AWS OpsWorks Stacks aciona um evento de configuração para executar as receitas de configuração, que normalmente lidam com tarefas como instalação e configuração de pacotes.

O AWS OpsWorks Stacks pode executar receitas personalizadas nas instâncias de uma camada, associando cada receita ao evento de ciclo de vida apropriado. AWS OpsWorks O Stacks executará todas as receitas personalizadas após a conclusão das receitas integradas da camada. Para este exemplo, `appsetup.rb` atribua ao evento de implantação da camada PHP App Server e `dbsetup.rb` ao evento de implantação da camada MySQL. AWS OpsWorks Em seguida, o Stacks executará as receitas nas instâncias da camada associada durante a inicialização, após a conclusão das receitas de configuração integradas e sempre que você implantar um aplicativo, após a conclusão das receitas de Implantação integradas. Para obter mais informações, consulte Execução automática de receitas.

Para associar receitas personalizadas ao evento Implantar da camada

1. Na página AWS OpsWorks Stacks Layers, para o PHP App Server, escolha Receitas e, em seguida, escolha Editar.
2. Em Custom Chef Recipes (Receitas do Chef personalizadas), adicione o nome da receita ao evento de implantação e selecione +. O nome deve estar no formato *`cookbookname`*`::`*`recipename`* do Chef, em que *`recipename`* não inclui a extensão `.rb`. Para este exemplo, insira `photoapp::appsetup`. Em seguida, selecione Save (Salvar) para atualizar a configuração da camada.



3. Na página Camadas, selecione Editar na coluna Ações da camada MySQL.
4. Adicione `photoapp::dbsetup` ao evento Implantar da camada e salve a nova configuração.

Etapa 5: adicionar informações de acesso aos atributos de configuração e implantação da pilha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A `appsetup.rb` receita depende dos dados da configuração e dos atributos de implantação da pilha AWS OpsWorks Stacks, que são instalados em cada instância e contêm informações detalhadas sobre a pilha e quaisquer aplicativos implantados. Os atributos `deploy` do objeto têm a seguinte estrutura, que é exibida para conveniência como JSON:

```
{
   ...
  "deploy": {
    "app1": {
      "application" : "short_name",
      ...
    }
    "app2": {
      ...
    }
    ...
  }
}
```

O nó de implantação contém um atributo para cada aplicativo implantado com o nome curto do aplicativo. Cada atributo do aplicativo contém um conjunto de atributos que definem a configuração do aplicativo, como a raiz do documento e o tipo de aplicativo. Para uma lista de

atributos `deploy`, consulte Atributos deploy. Você pode representar os valores de atributos de configuração e implantação da pilha em suas receitas usando a sintaxe de atributos do Chef. Por exemplo,`[:deploy][:app1][:application]` representa o nome curto do aplicativo app1.

As receitas personalizadas dependem de vários atributos de configuração e implantação da pilha que representam informações de acesso ao banco de dados e ao Amazon S3:

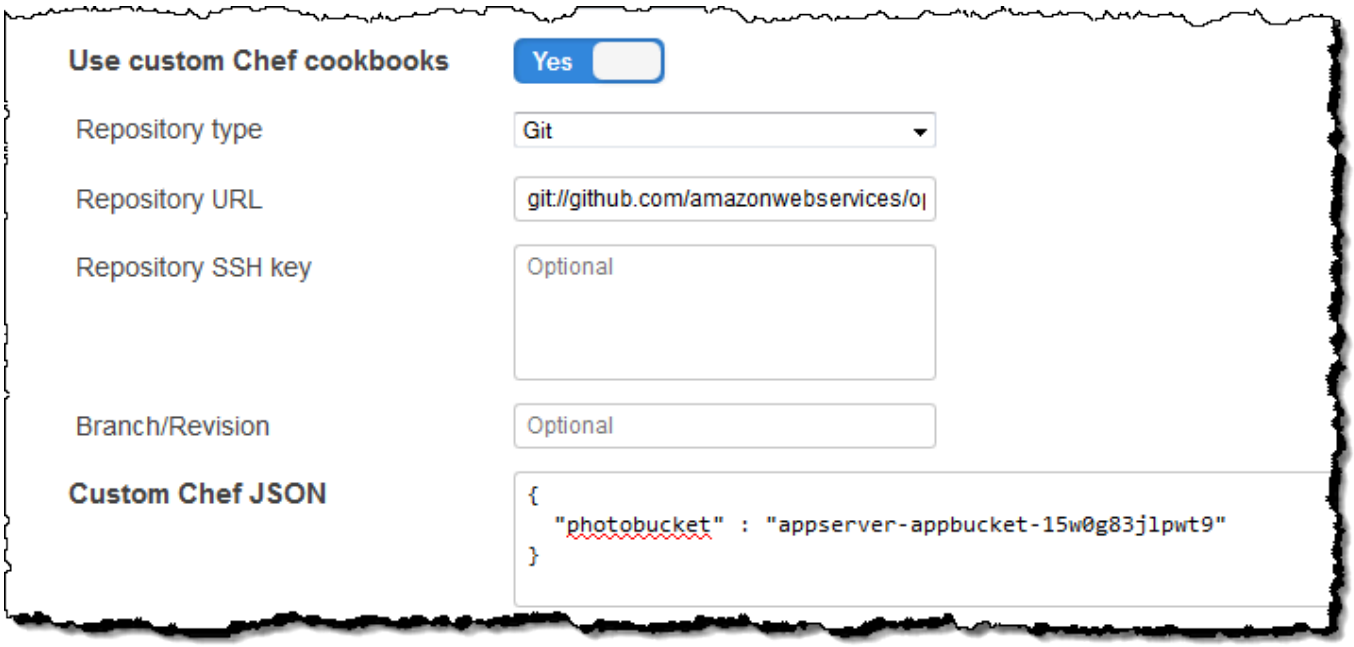
- Os atributos de conexão de banco de dados`[:deploy][:database][:host]`, como, são definidos pelo AWS OpsWorks Stacks quando ele cria a camada MySQL.
- O atributo de nome da tabela, `[:photoapp][:dbtable]`, é definido no arquivo de atributos do livro de receitas personalizado e é definido como `foto`.
- É necessário definir o atributo de nome do bucket `[:photobucket]`, usando um JSON personalizado para adicioná-lo aos atributos de configuração e implantação da pilha.

Para definir o atributo de nome do bucket do Amazon S3

1. Na página AWS OpsWorks Stacks Stack, escolha Stack Settings e depois Editar.
2. Na seção Configuration Management, adicione as informações de acesso à caixa Custom Chef JSON. Ela deve ser parecida com a seguinte:

```
{
  "photobucket" : "yourbucketname"
}
```

   *yourbucketname*Substitua pelo nome do bucket em que você gravouEtapa 1: crie um bucket do Amazon S3.



AWS OpsWorks O Stacks mescla o JSON personalizado nos atributos de configuração e implantação da pilha antes de instalá-los nas instâncias da pilha; `appsetup.rb` pode então obter o nome do bucket a partir do atributo. `[:photobucket]` Se você deseja alterar o bucket, não é necessário modificar a receita. Basta substituir o atributo para fornecer um novo nome do bucket.

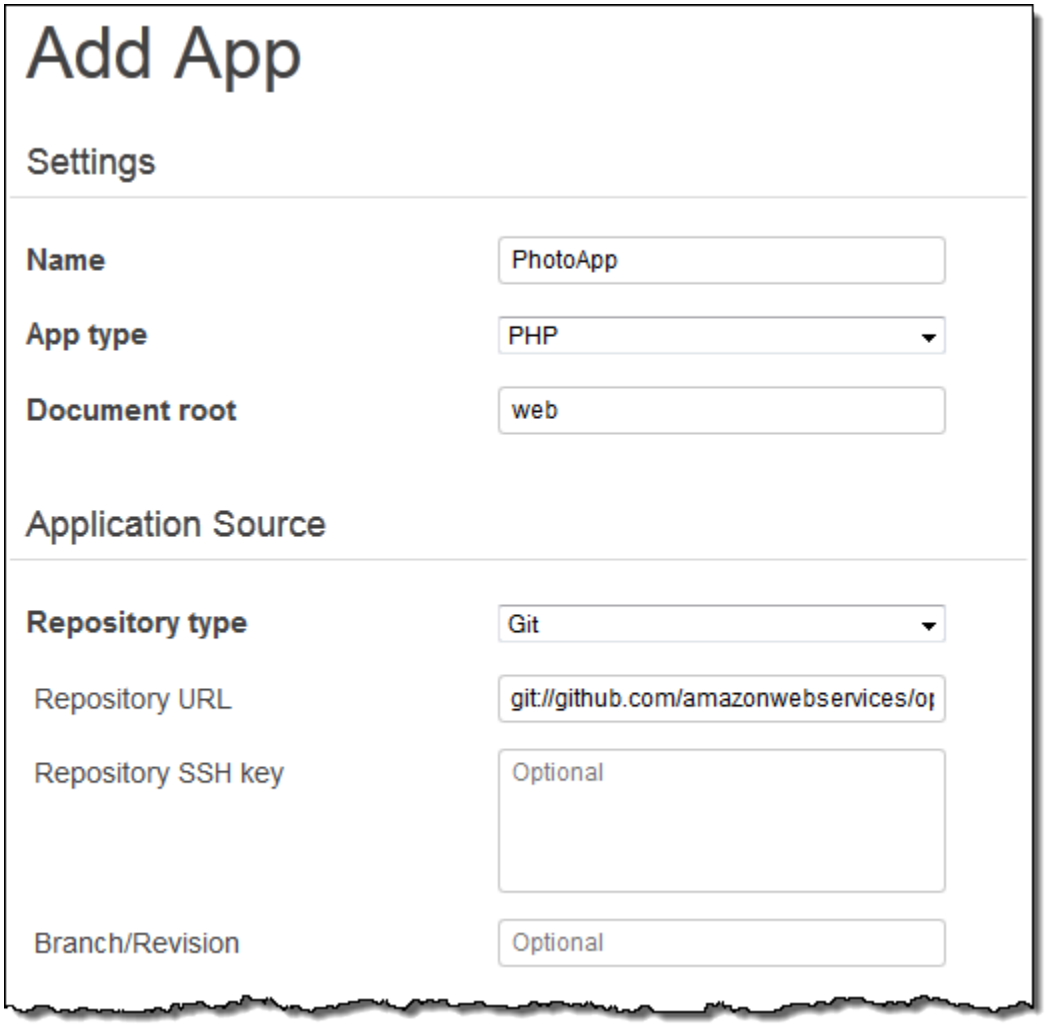
Etapa 6: implantar e executar PhotoApp

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos fortemente que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

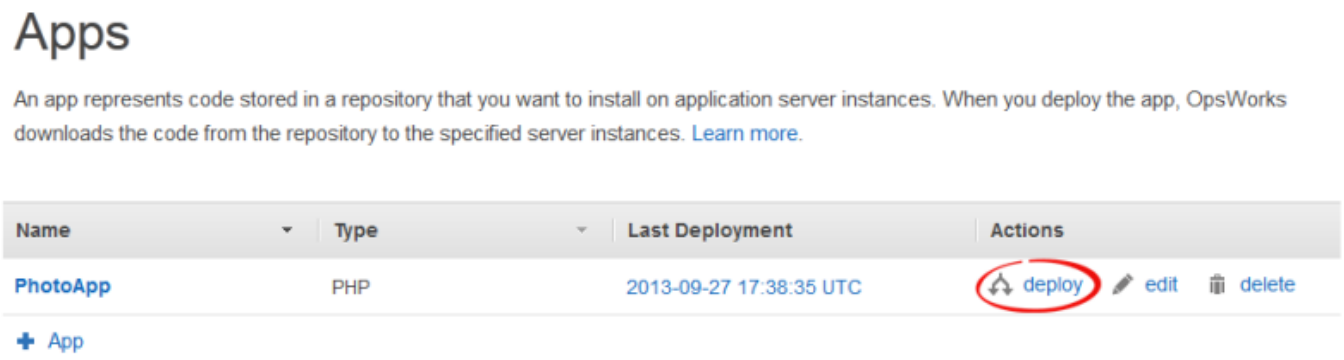
Neste exemplo, o aplicativo também foi implementado para você e está armazenado em um GitHub repositório público. Basta adicionar o aplicativo à pilha , implantá-lo nos servidores de aplicativos e executá-lo.

Para adicionar o aplicativo à pilha e implantá-lo nos servidores de aplicativos

1. Abra a página Apps (Aplicativos) e selecione Add an app (Adicionar um aplicativo).
2. Na página Add App, faça o seguinte:
   - Defina Name (Nome) como **`PhotoApp`**.
   - Defina App type como PHP.
   - Defina Document root (Raiz do documento:) como **`web`**.
   - Defina Repository type como Git.
   - Defina Repository URL (URL do repositório). como **`git://github.com/awslabs/opsworks-demo-php-photo-share-app.git`**.
   - Selecione Add App (Adicionar aplicativo) para aceitar os valores para as outras configurações.

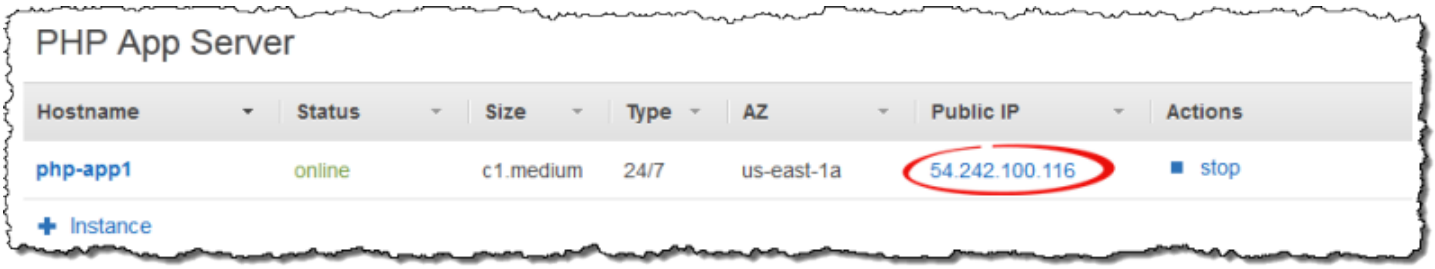


3. Na página Aplicativos, escolha implantar na coluna Ações do PhotoApp aplicativo.



4. Aceite os valores padrão e selecione Deploy (Implantar) para implantar o aplicativo no servidor.

Para executar PhotoApp, acesse a página Instâncias e selecione o endereço IP público da instância do PHP App Server.



Você deve ver a interface do usuário a seguir. Selecione Adicionar uma foto para armazenar uma foto no bucket do Amazon S3 e os metadados no datastore de back-end.

Usando AWS CodePipeline com AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS CodePipelinepermite criar pipelines de entrega contínua que rastreiam alterações de código de fontes como CodeCommit Amazon Simple Storage Service (Amazon S3) ou. GitHub Você pode usar CodePipeline para automatizar o lançamento de seus livros de receitas e código de aplicativo do Chef para AWS OpsWorks Stacks, nas pilhas Chef 11.10, Chef 12 e Chef 12.2. Os exemplos nesta seção descrevem como criar e usar um pipeline simples CodePipeline como uma ferramenta de implantação para o código que você executa nas camadas do AWS OpsWorks Stacks.

> **Note**
>
> CodePipeline e a integração de AWS OpsWorks pilhas não é suportada para implantação no Chef 11.4 e em pilhas mais antigas.

Tópicos

- AWS CodePipeline com AWS OpsWorks pilhas - Chef 12 pilhas
- AWS CodePipeline com AWS OpsWorks pilhas - Chef 11 pilhas

AWS CodePipeline com AWS OpsWorks pilhas - Chef 12 pilhas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS CodePipelineO permite que você crie pipelines de entrega contínua que rastreiam alterações de código nas origens CodeCommit, como Amazon Simple Storage Service (Amazon S3) ou. GitHub O exemplo neste tópico descreve como criar e usar um pipeline simples CodePipeline como uma ferramenta de implantação para o código executado nas camadas do AWS OpsWorks Stacks. Neste

exemplo, você cria um pipeline para um aplicativo Node.js simples e, em seguida, instrui o AWS OpsWorks Stacks a executar o aplicativo em todas as instâncias em uma camada em uma pilha do Chef 12 (nesse caso, uma única instância).

> **Note**
>
> Este tópico descreve como usar um pipeline para executar e atualizar um aplicativo em uma pilha do Chef 12. Para obter informações sobre como usar um pipeline para executar e atualizar um aplicativo em uma stack do Chef 11.10, consulte AWS CodePipeline com AWS OpsWorks pilhas - Chef 11 pilhas. O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Tópicos

- Pré-requisitos
- Outros cenários compatíveis
- Etapa 1: criar uma pilha, uma camada e uma instância em AWS OpsWorks Stacks
- Etapa 2: Configurar seu stack e sua layer para usar livros de receitas personalizados
- Etapa 3: fazer upload do código do aplicativo em um bucket do Amazon S3
- Etapa 4: adicionar aplicativo ao AWS OpsWorks Stacks
- Etapa 5: criar um pipeline no CodePipeline
- Etapa 6: Verificar a implantação do aplicativo no AWS OpsWorks Stacks
- Etapa 7 (opcional): Atualizar o código do aplicativo para ver o CodePipeline reimplantar o aplicativo automaticamente
- Etapa 8 (opcional): Limpar os recursos

Pré-requisitos

Antes de iniciar esta demonstração, verifique se você tem permissões de administrador para executar todas as tarefas a seguir. Você pode ser membro de um grupo que tenha a AdministratorAccesspolítica aplicada ou pode ser membro de um grupo que tenha as permissões e políticas mostradas na tabela a seguir. Como prática recomendada de segurança, você deve

pertencer a um grupo com permissões para executar as tarefas a seguir, em vez de atribuir permissões necessárias a usuários individuais.

Para obter mais informações sobre a criação de um grupo de segurança no IAM e a atribuição de permissões ao grupo, consulte Criar grupos de usuários do IAM. Para obter mais informações sobre como gerenciar permissões de AWS OpsWorks pilhas, consulte Melhores práticas: gerenciamento de permissões.

Permissões	Política recomendada para anexação a grupo
Crie e edite pilhas, camadas e instâncias em AWS OpsWorks Pilhas.	AWSOpsWorks_FullAccess
Crie, edite e execute modelos no AWS CloudFormation.	AmazonCloudFormationFullAccess
Crie, edite e acesse os buckets do Amazon S3.	Amazon S3 FullAccess
Crie, edite e execute pipelines em CodePipeline, especialmente os que usam o AWS OpsWorks Stacks como provedor.	AWSCodePipeline_FullAccess

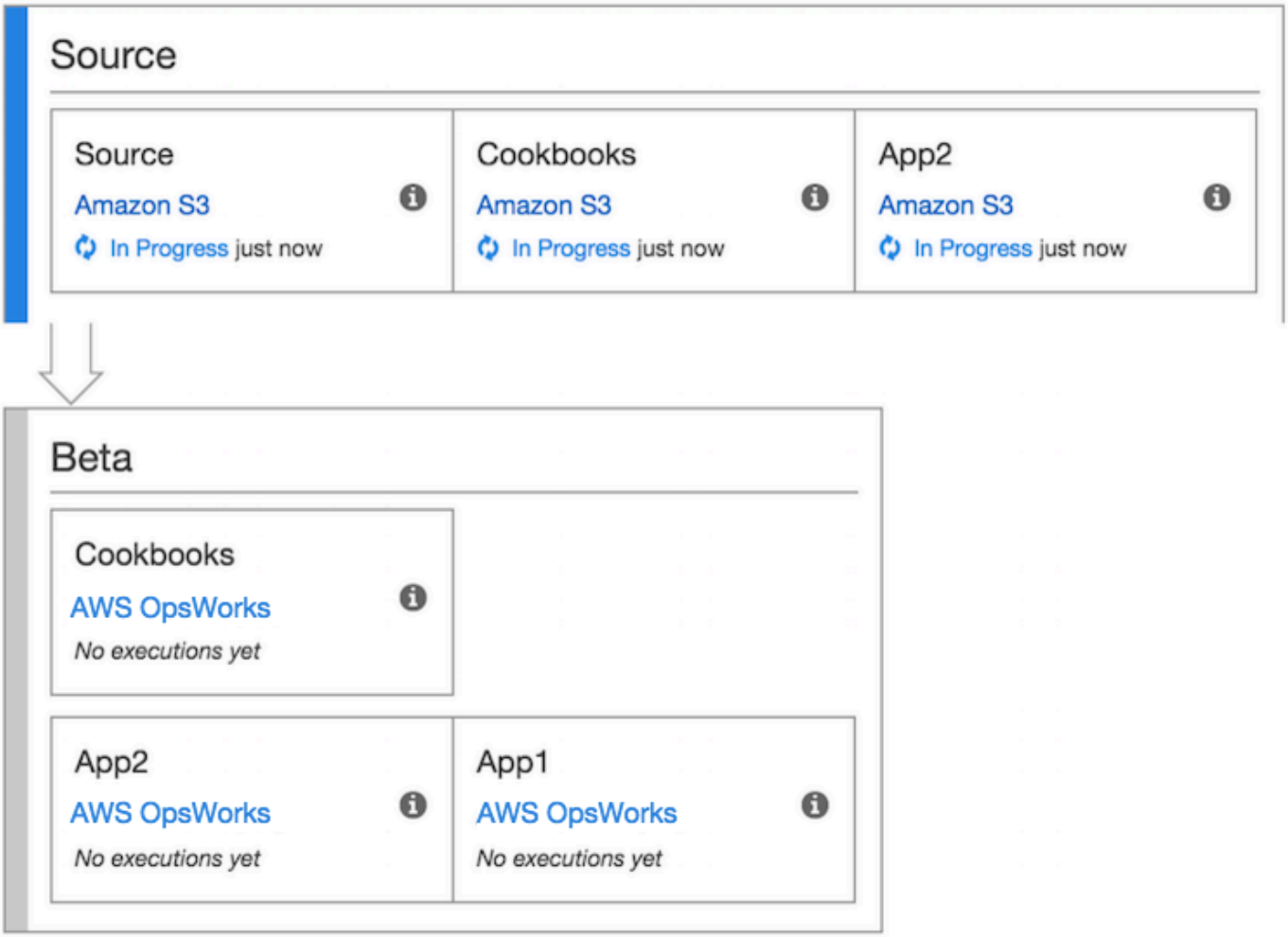
Você também deve ter um par de EC2 chaves da Amazon. Você será solicitado a fornecer o nome desse key pair ao executar o AWS CloudFormation modelo que cria a pilha, a camada e a instância de amostra neste passo a passo. Para obter mais informações sobre a obtenção de um par de chaves no EC2 console da Amazon, consulte Criar um par de chaves na EC2 documentação da Amazon. O par de chaves deve estar na região Leste dos EUA (Norte da Virgínia). Você pode usar um par de chaves existente se já houver um nessa região.

Outros cenários compatíveis

Esta demonstração cria um pipeline simples que inclui um estágio Source e outro Deploy. No entanto, você pode criar pipelines mais complexos que usam o AWS OpsWorks Stacks como provedor. Veja a seguir exemplos de pipelines e cenários compatíveis:

- Você pode editar um pipeline para adicionar um livro de receitas do Chef ao estágio Source e um destino associado de livros de receitas atualizados ao estágio Deploy. Nesse caso, você adiciona uma ação Deploy que aciona a atualização dos seus livros de receitas quando são feitas alterações na origem. O livro de receitas atualizado é implantado antes do seu aplicativo.

- Você pode criar um pipeline complexo, com livros de receitas personalizados e vários aplicativos, e implantar em uma AWS OpsWorks pilha Stacks. O pipeline rastreia as alterações nas origens de aplicativos e livros de receitas, e reimplanta quando você faz alterações. A tabela a seguir mostra um exemplo de um pipeline similar e complexo:



Para obter mais informações sobre como trabalhar com CodePipeline, consulte o Guia CodePipeline do usuário.

Etapa 1: criar uma pilha, uma camada e uma instância em AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas

cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para usar o AWS OpsWorks Stacks como provedor de implantação para um pipeline, você deve primeiro ter uma pilha, uma camada e pelo menos uma instância na camada. Embora você possa criar uma pilha no AWS OpsWorks Stacks seguindo as instruções em Getting Started with Linux Stacks ou Getting Started with Windows Stacks, para economizar seu tempo, este exemplo usa um AWS CloudFormation modelo para criar uma pilha, camada e instância do Chef 12 com base em Linux. A instância criada por este modelo executa o Amazon Linux 2016.03 e tem um tipo de instância `c3.large`. Embora o modelo não configure seu stack para usar livros de receitas personalizados, você fará isso mais tarde, na demonstração.

> **Important**
>
> O AWS CloudFormation modelo do deve ser armazenado e executado na mesma região que o bucket do Amazon S3 no qual você fará o upload do aplicativo posteriormente e na mesma região na qual criará o pipeline mais tarde. CodePipeline No momento, CodePipeline oferece suporte ao provedor do AWS OpsWorks Stacks apenas na região Leste dos EUA (us-east-1). Todos os recursos desta demonstração devem ser criados na região Leste dos EUA (N. da Virgínia).
>
> Se houver falha na criação da pilha, você pode estar se aproximando do número máximo permitido de funções do IAM na sua conta. A criação da pilha também pode falhar se sua conta não conseguir executar instâncias com o tipo de instância `c3.large`. Por exemplo, se você estiver usando o nível gratuito da AWS , talvez receba um erro como `Root device type: must be included in EBS`. Se sua conta tem limitações quanto aos tipos de instância que você pode criar, como limitações impostas pelo nível AWS gratuito, tente alterar o valor do `InstanceType` parâmetro no bloco de instâncias do modelo para um tipo de instância que sua conta pode usar.

Para criar uma pilha, camada e instância usando AWS CloudFormation

1. Copie o AWS CloudFormation modelo a seguir em um novo documento de texto sem formatação. Salve o arquivo em um local conveniente no computador local e nomeie-o como NewOpsWorksStack.template ou outro nome que seja conveniente para você.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Mappings": {
    "Region2Principal": {
      "us-east-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "us-west-2": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "us-west-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "eu-west-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "ap-southeast-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "ap-northeast-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "ap-northeast-2": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "ap-southeast-2": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "sa-east-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "cn-north-1": {
        "EC2Principal": "ec2.amazonaws.com.cn",
        "OpsWorksPrincipal": "opsworks.amazonaws.com.cn"
```

```
      },
      "eu-central-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      }
    }
  },
  "Parameters": {
    "EC2KeyPairName": {
   "Type": "String",
   "Description": "The name of an existing EC2 key pair that lets you use SSH to
connect to the OpsWorks instance."
  }
  },
  "Resources": {
"CPOpsDeploySecGroup": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {
    "GroupDescription" : "Lets you manage OpsWorks instances to which you deploy
apps with CodePipeline"
  }
},
"CPOpsDeploySecGroupIngressHTTP": {
  "Type": "AWS::EC2::SecurityGroupIngress",
  "Properties" : {
    "IpProtocol" : "tcp",
       "FromPort" : "80",
       "ToPort" : "80",
       "CidrIp" : "0.0.0.0/0",
 "GroupId": {
   "Fn::GetAtt": [
     "CPOpsDeploySecGroup", "GroupId"
   ]
 }
     }
},
"CPOpsDeploySecGroupIngressSSH": {
  "Type": "AWS::EC2::SecurityGroupIngress",
  "Properties" : {
    "IpProtocol" : "tcp",
       "FromPort" : "22",
       "ToPort" : "22",
       "CidrIp" : "0.0.0.0/0",
 "GroupId": {
```

```
    "Fn::GetAtt": [
      "CPOpsDeploySecGroup", "GroupId"
    ]
  }
   }
 },
 "MyStack": {
      "Type": "AWS::OpsWorks::Stack",
      "Properties": {
        "Name": {
          "Ref": "AWS::StackName"
        },
        "ServiceRoleArn": {
          "Fn::GetAtt": [
            "OpsWorksServiceRole",
            "Arn"
          ]
        },
  "ConfigurationManager" : { "Name": "Chef","Version": "12" },
  "DefaultOs": "Amazon Linux 2016.03",
        "DefaultInstanceProfileArn": {
          "Fn::GetAtt": [
            "OpsWorksInstanceProfile",
            "Arn"
          ]
        },
  "UseCustomCookbooks": "false"
      }
    },
    "MyLayer": {
      "Type": "AWS::OpsWorks::Layer",
      "Properties": {
        "StackId": {
          "Ref": "MyStack"
        },
        "Name": "Node.js App Server",
  "Type": "custom",
        "Shortname": "app1",
  "EnableAutoHealing": "true",
        "AutoAssignElasticIps": "false",
        "AutoAssignPublicIps": "true",
  "CustomSecurityGroupIds": [
    {
      "Fn::GetAtt": [
```

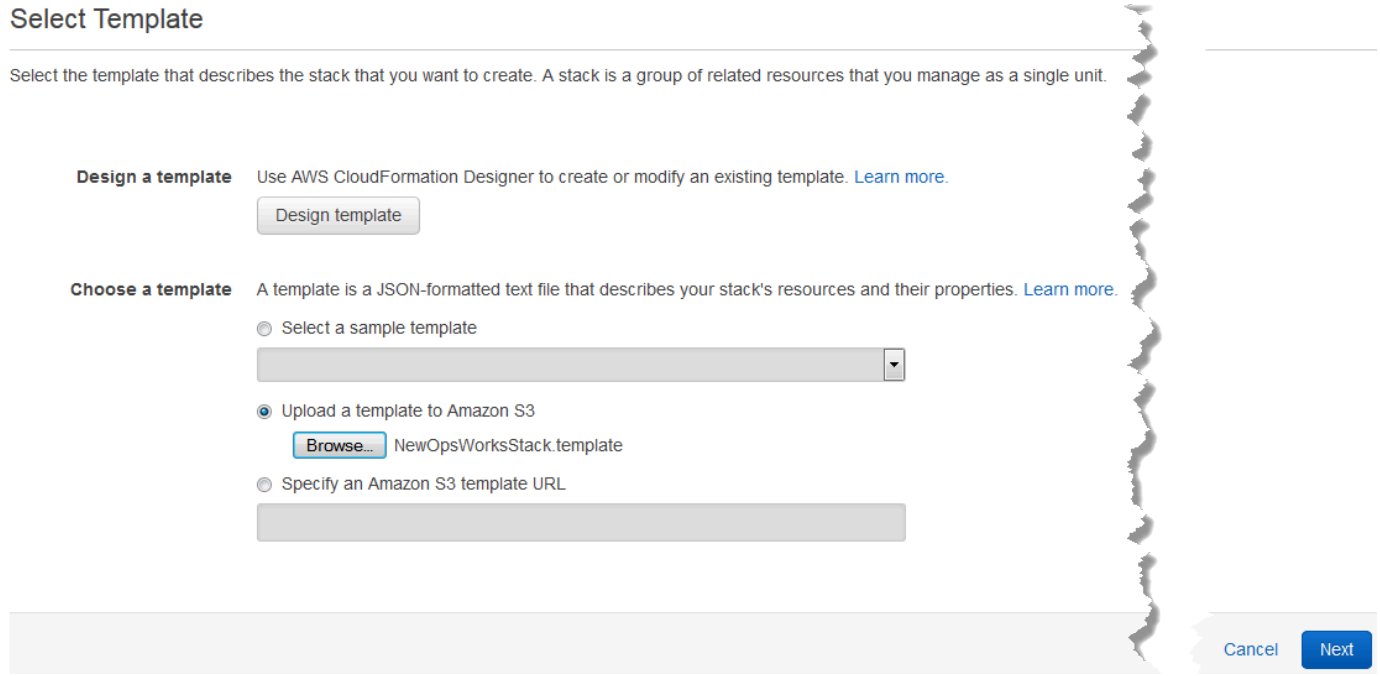
```
              "CPOpsDeploySecGroup", "GroupId"
      ]
    }
   ]
      },
      "DependsOn": [
        "MyStack",
        "CPOpsDeploySecGroup"
      ]
    },
    "OpsWorksServiceRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  {
                    "Fn::FindInMap": [
                      "Region2Principal",
                      {
                        "Ref": "AWS::Region"
                      },
                      "OpsWorksPrincipal"
                    ]
                  }
                ]
              },
              "Action": [
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path": "/",
        "Policies": [
          {
            "PolicyName": "opsworks-service",
            "PolicyDocument": {
              "Statement": [
                {
                  "Effect": "Allow",
```

```
                  "Action": [
                    "ec2:*",
                    "iam:PassRole",
                    "cloudwatch:GetMetricStatistics",
                    "elasticloadbalancing:*"
                  ],
                  "Resource": "*"
                }
              ]
            }
          }
        ]
      }
    },
    "OpsWorksInstanceProfile": {
      "Type": "AWS::IAM::InstanceProfile",
      "Properties": {
        "Path": "/",
        "Roles": [
          {
            "Ref": "OpsWorksInstanceRole"
          }
        ]
      }
    },
    "OpsWorksInstanceRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  {
                    "Fn::FindInMap": [
                      "Region2Principal",
                      {
                        "Ref": "AWS::Region"
                      },
                      "EC2Principal"
                    ]
                  }
                ]
```

```
              },
              "Action": [
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path": "/",
  "Policies": [
          {
            "PolicyName": "s3-get",
            "PolicyDocument": {
              "Version": "2012-10-17",
              "Statement": [
                {
                  "Effect": "Allow",
                  "Action": [
                    "s3:GetObject"
                  ],
                  "Resource": "*"
                }
              ]
            }
          }
        ]
      }
    },
    "myinstance": {
      "Type": "AWS::OpsWorks::Instance",
      "Properties": {
        "LayerIds": [
          {
            "Ref": "MyLayer"
          }
        ],
        "StackId": {
          "Ref": "MyStack"
        },
        "InstanceType": "c3.large",
        "SshKeyName": {
    "Ref": "EC2KeyPairName"
  }
      }
    }
```

```
  },
  "Outputs": {
    "StackId": {
      "Description": "Stack ID for the newly created AWS OpsWorks stack",
      "Value": {
        "Ref": "MyStack"
      }
    }
  }
}
```

2. Faça login no AWS Management Console e abra o AWS CloudFormation console em https://console.aws.amazon.com/cloudformation.
3. Na página AWS CloudFormation inicial, escolha Criar pilha.
4. Na página Select Template, na área Choose a template, selecione a opção Upload a template to Amazon S3 e, em seguida, Browse.
5. Navegue até o AWS CloudFormation modelo que você salvou na etapa 1 e escolha Abrir. Na página Select Template, escolha Next.

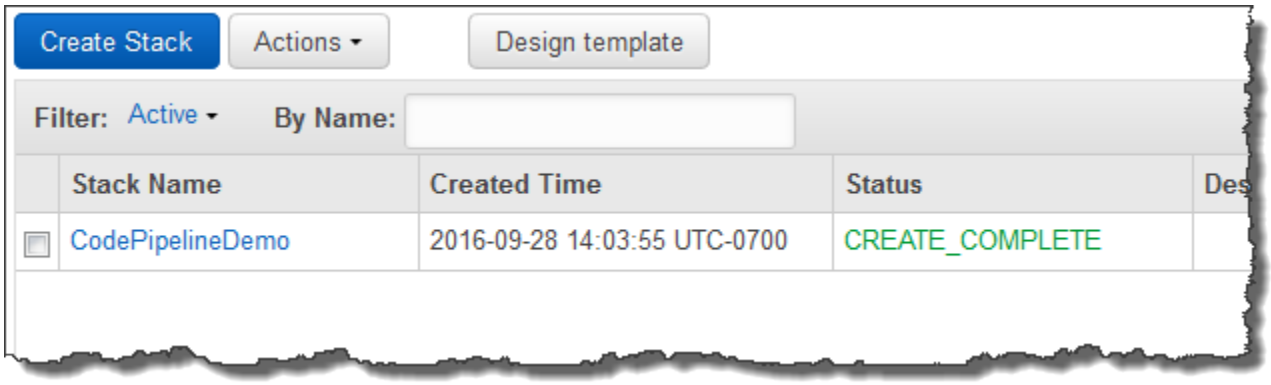


6. Na página Especificar detalhes, nomeie a pilha CodePipelineDemoou qualquer nome de pilha exclusivo da sua conta. Se você escolher um nome diferente para a pilha, altere o nome da pilha durante esta demonstração.
7. Na área Parâmetros, forneça o nome de um EC2 key pair que você deseja usar para acessar sua instância do AWS OpsWorks Stacks depois que ela for criada. Escolha Próximo.

8. Na página Options (Opções), escolha Next (Avançar). (As configurações desta página não são necessárias nesta demonstração.)
9. O AWS CloudFormation modelo que você usa neste passo a passo cria funções do IAM, um perfil de instância e uma instância.

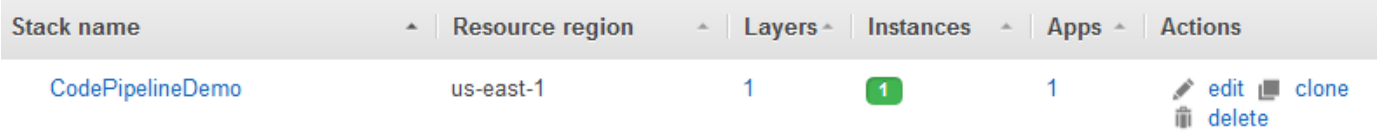
> **Important**
>
> Antes de escolher Criar, escolha Custo para estimar os encargos que podem ser gerados AWS pela criação de recursos com esse modelo.

   Se a criação de recursos do IAM for aceitável, marque a caixa de seleção Reconheço que esse modelo pode causar AWS CloudFormation a criação de recursos do IAM e, em seguida, escolha a opção Criar. Se a criação de recursos do IAM não for aceitável, você não poderá continuar esse procedimento.
10. No AWS CloudFormation painel, você pode visualizar o andamento da criação da pilha. Antes de passar para a próxima etapa, aguarde até que CREATE_COMPLETE seja exibido na coluna Status.



Para verificar a criação da pilha em AWS OpsWorks Pilhas

1. Abra o AWS OpsWorks console em https://console.aws.amazon.com/opsworks/.
2. No painel de AWS OpsWorks pilhas, veja a pilha que você criou.



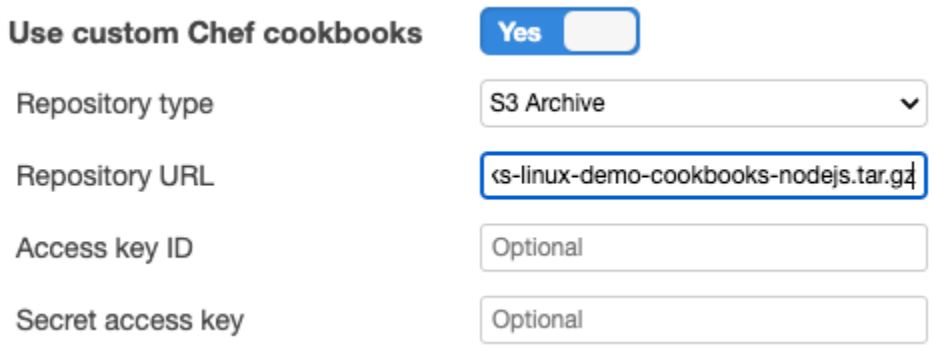
3. Abra a pilha e exiba a camada e a instância. Observe que a camada e a instância foram criadas com os nomes e outros metadados fornecidos no AWS CloudFormation modelo. Você está pronto para configurar sua pilha e sua camada para usar livros de receitas e receitas personalizados do Chef.

Etapa 2: Configurar seu stack e sua layer para usar livros de receitas personalizados

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

As pilhas do Chef 12 em AWS OpsWorks Stacks exigem livros de receitas próprios ou criados pela comunidade para criar camadas de aplicativos personalizadas. Para esta demonstração, você pode apontar para um repositório que contenha um conjunto de livros de receitas do Chef e de receitas do Chef. Essas receitas instalam o pacote do Node.js e as dependências em sua instância. Você usará outras receitas do Chef para implantar o aplicativo Node.js que você preparará em Etapa 4: adicionar aplicativo ao AWS OpsWorks Stacks. A receita do Chef especificada nesta etapa é executada sempre que uma nova versão do aplicativo é implantada pelo CodePipeline.

1. No console AWS OpsWorks Stacks, abra a pilha que você criou em. Etapa 1: criar uma pilha, uma camada e uma instância em AWS OpsWorks Stacks Selecione Stack Settings e, em seguida, Edit.
2. Defina Use custom Chef cookbooks como Yes. Isso mostra as configurações de livro de receitas relacionadas.
3. Na lista suspensa Repository type, escolha a opção S3 Archive. Para trabalhar com os dois CodePipeline e AWS OpsWorks, a origem do seu livro de receitas deve ser S3.
4. Para Repository URL, especifique **`https://s3.amazonaws.com/opsworks-demo-assets/opsworks-linux-demo-cookbooks-nodejs.tar.gz`**. Suas configurações devem se parecer com o seguinte.



5. Escolha Salvar.
6. No painel de navegação, escolha Camadas.
7. Escolha a opção Settings para a layer que você criou em Etapa 1: criar uma pilha, uma camada e uma instância em AWS OpsWorks Stacks.
8. Na guia General Settings, verifique se o nome da layer é Node.js App Server e se ele está abreviado como app1. Selecione Recipes.
9. Na guia Recipes (Receitas), especifique **`nodejs_demo`** como a receita que você deseja executar durante o evento de ciclo de vida Deploy (Implantar). Escolha Salvar.
10. Na guia Segurança, na lista suspensa Grupos de segurança, escolha o grupo de segurança AWS- OpsWorks -Webapp.
11. Escolha Salvar.

Etapa 3: fazer upload do código do aplicativo em um bucket do Amazon S3

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

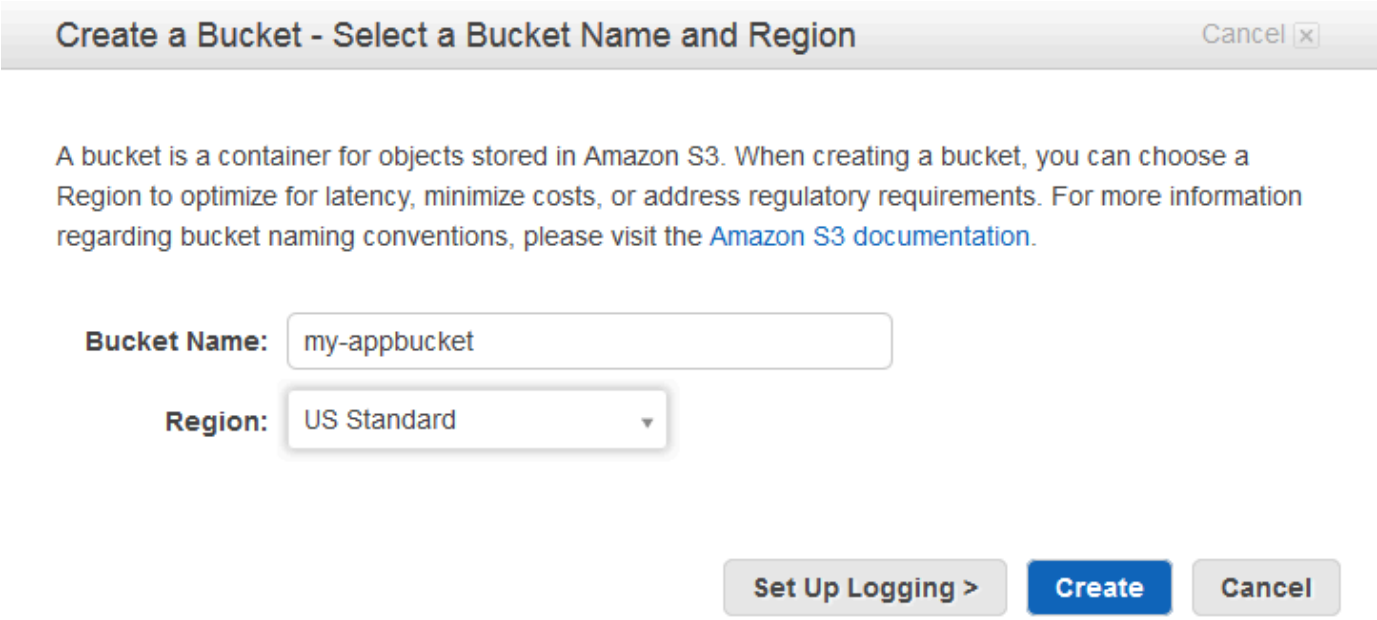
Como você deve fornecer um link para o repositório de código como parte da configuração do pipeline, tenha o código do repositório a mão antes de criar o pipeline. Nesta demonstração, você carrega um aplicativo Node.js em um bucket do Amazon S3.

Embora CodePipeline possa usar o código diretamente de GitHub ou CodeCommit como origens, este passo a passo demonstra como usar um bucket do Amazon S3. Nesta demonstração, você carrega o aplicativo Node.js app de exemplo para seu próprio bucket do Amazon S3, para que você possa fazer alterações no aplicativo. O bucket do Amazon S3 criado nesta etapa permite CodePipeline detectar alterações no código do aplicativo e implantar o aplicativo alterado automaticamente. Se desejar, use um bucket existente. Verifique se o bucket atende aos critérios descritos em Demonstração simples do pipeline (Bucket do Amazon S3) na documentação. CodePipeline

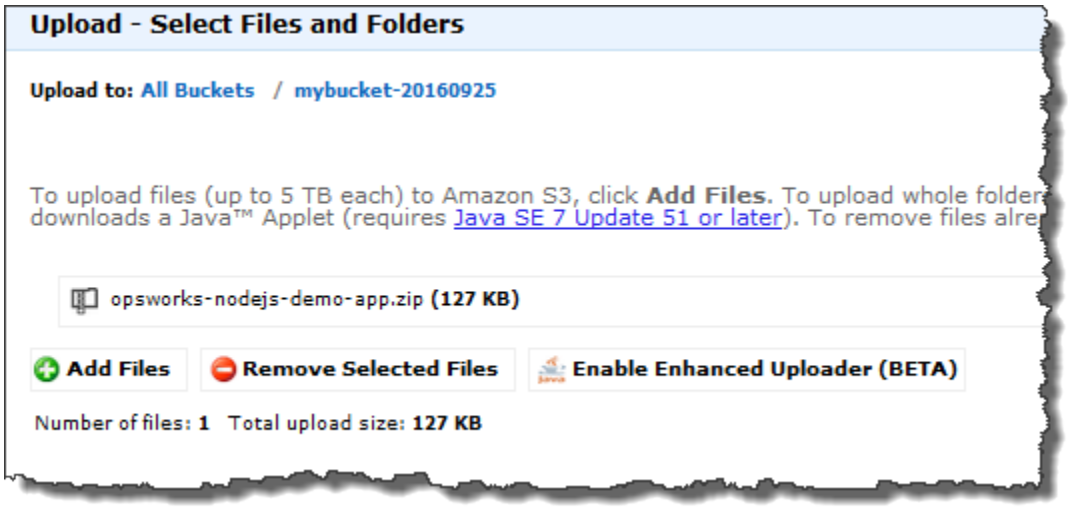
> **Important**
>
> O bucket do Amazon S3 deve estar na mesma região em que o pipeline será criado posteriormente. No momento, CodePipeline oferece suporte ao provedor do AWS OpsWorks Stacks apenas na região Leste dos EUA (us-east-1). Todos os recursos desta demonstração devem ser criados na região Leste dos EUA (N. da Virgínia). O bucket também deve ser versionado porque CodePipeline requer uma origem versionada. Para obter mais informações, consulte Usar versionamento.

Para carregar seu aplicativo para o bucket do Amazon S3

1. Baixe o arquivo ZIP do aplicativo AWS OpsWorks Stacks sample, Node.js, e salve-o em um local conveniente em seu computador local.
2. Abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
3. Escolha Criar bucket.
4. Na página Create a Bucket - Select a Bucket Name and Region, digite um nome exclusivo para o seu bucket em Bucket Name. Os nomes de bucket devem ser exclusivos em todas as AWS contas, não apenas na conta da. Esta demonstração usa o nome **`my-appbucket`**, mas é possível usar `my-appbucket-`*`yearmonthday`* para fazer com que o nome do bucket seja exclusivo. Na lista suspensa Region, escolha a opção US Standard e, em seguida, Create. US Standard equivale a `us-east-1`.



5. Na lista All Buckets (Todos os buckets), selecione o bucket que você criou.
6. Na página do bucket, escolha Upload (Carregar).
7. Na página Upload - Select Files and Folders, selecione Add files. Procure o arquivo ZIP salvo na etapa 1, clique em Open e, em seguida, Start Upload.



8. Após a conclusão do upload, selecione o arquivo ZIP na lista de arquivos do bucket e escolha a opção Properties.
9. No painel Properties, copie o link para o arquivo ZIP e anote o link. Você precisará da parte do nome do bucket e do nome do arquivo ZIP deste link para criar o pipeline.

Etapa 4: adicionar aplicativo ao AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Antes de criar um pipeline no CodePipeline, adicione o aplicativo de teste Node.js ao AWS OpsWorks Stacks. Ao criar o pipeline, você precisará selecionar o aplicativo que você adicionou ao AWS OpsWorks Stacks.

Prepare o link do bucket do Amazon S3 da etapa 9 do procedimento anterior. Você precisará do link para o bucket em que armazenou o aplicativo de teste para concluir esse procedimento.

Para adicionar um aplicativo ao AWS OpsWorks Stacks

1. No console AWS OpsWorks Stacks, abra e CodePipelineDemo, no painel de navegação, escolha Apps.
2. Escolha Add app (Adicionar aplicativo).
3. Na página Add App, forneça as seguintes informações:
   a. Especifique um nome para o aplicativo. Esta demonstração usa o nome `Node.js Demo` App.
   b. Para Data source type (Tipo de origem de dados), escolha a opção None (Nenhum). Este aplicativo não requer um banco de dados externo ou fonte de dados.
   c. Na lista suspensa Repository type, escolha a opção S3 Archive.
   d. Na caixa de strings Repository URL (URL do repositório), cole a URL copiada na etapa 9 de Etapa 3: fazer upload do código do aplicativo em um bucket do Amazon S3. O formulário deve ser semelhante ao seguinte:

Add App

All app attributes are stored in Chef data bags. Learn more.

Settings

Name Node.js Demo App

Document root opsworks-nodejs-demo-app

Data Sources

Data source type RDS None

Application Source

Repository type S3 Archive

Repository URL I60925/opsworks-nodejs-demo-app.zip

Access key ID Optional

Secret access key Optional

Environment Variables

KEY VALUE Protected value

Add Domains

Domain name Optional

SSL Settings

Enable SSL No

Cancel Add App

4. Você não precisa alterar outras configurações neste formulário. Escolha Add App.
5. Quando o aplicativo Node.js Demo App aparecer na lista da página Apps, prossiga para Etapa 5: criar um pipeline no CodePipeline.

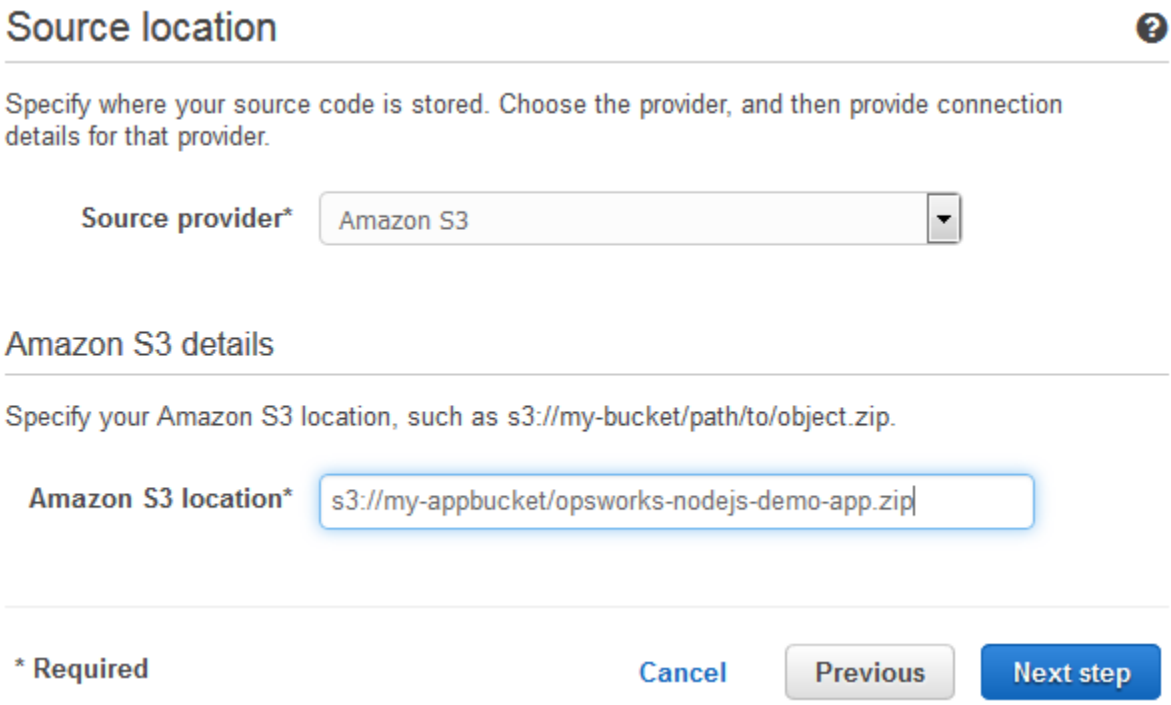
Etapa 5: criar um pipeline no CodePipeline

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

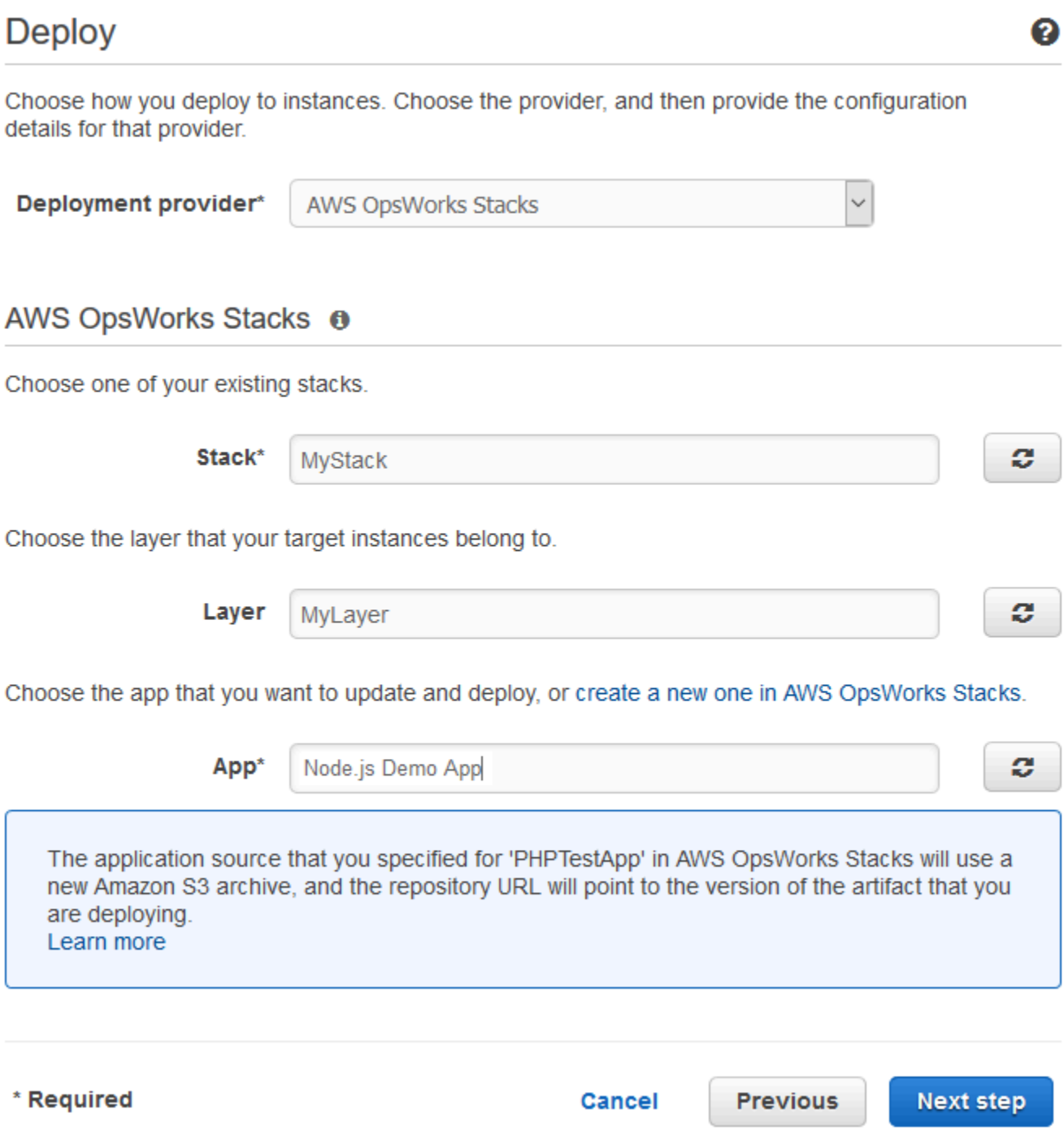
Quando você tiver uma pilha com uma camada e, pelo menos, uma instância configurado no AWS OpsWorks Stacks, crie um pipeline CodePipeline com o AWS OpsWorks Stacks como provedor para implantar aplicativos ou livros de receitas do Chef nos recursos do Stacks. AWS OpsWorks

Para criar um pipeline

1. Abra o CodePipeline console em https://console.aws.amazon.com/codepipeline/.
2. Selecione Criar pipeline.
3. Na CodePipeline página Introdução**`MyOpsWorksPipeline`**, digite ou qualquer outro nome de funil exclusivo da sua conta e escolha Próxima etapa.
4. Na página Source Location, selecione Amazon S3 na lista suspensa Source provider.
5. Na área Detalhes do Amazon S3, digite o caminho do bucket do Amazon S3 no formato **`s3://`***`bucket-name`***`/`***`file name`**. Consulte o link anotado na etapa 9 de Etapa 3: fazer upload do código do aplicativo em um bucket do Amazon S3. Nesta demonstração, o caminho é `s3://my-appbucket/opsworks-nodejs-demo-app.zip`. Escolha Próxima etapa.



6. Na página Build, escolha No Build na lista suspensa e escolha Next step.
7. Na página Deploy (Implantar), escolha AWS OpsWorks Stacks como provedor de implantação.



8. No campo Stack, digite `CodePipelineDemo` ou o nome da pilha criada em Etapa 1: criar uma pilha, uma camada e uma instância em AWS OpsWorks Stacks.
9. No campo Layer, digite `Node.js App Server` ou o nome da layer criada em Etapa 1: criar uma pilha, uma camada e uma instância em AWS OpsWorks Stacks.

10. No campo Aplicativo, selecione o aplicativo carregado no Amazon S3 em Etapa 3: fazer upload do código do aplicativo em um bucket do Amazon S3 e escolha Próxima etapa.
11. Na página Perfil de serviço da AWS , selecione Criar perfil.

    Uma nova janela é aberta com uma página do console do IAM que descreve o perfil que será criado para você, `AWS-CodePipeline-Service`. Na lista suspensa Policy name, escolha a opção Create new policy. Certifique-se de que o documento de política tem o conteúdo a seguir. Escolha Edit para alterar o documento da política, se necessário.

```
{
    "Statement": [
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetBucketVersioning"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": "opsworks:*",
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

    Assim que você terminar de fazer alterações no documento da política, escolha Allow (Permitir). Suas alterações serão exibidas no console do IAM.

Hide Details

Role Summary

Role Description	Provides read and write access to AWS services and resources.
IAM Role	AWS-CodePipeline-Service
Policy Name	Create a new Role Policy

Hide Policy Document

Edit

```
{
  "Statement": [
    {
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetBucketVersioning"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
```

> **Note**
>
> Se a criação da função falhar, talvez já haja uma função do IAM chamada AWS-CodePipeline -Service. Se você estava usando a função AWS- CodePipeline -Service antes de maio de 2016, a função pode não ter permissões para usar o AWS OpsWorks Stacks como provedor de implantação. Neste caso, você deve atualizar a declaração de política conforme mostrado nesta etapa. Caso uma mensagem de erro seja exibida, volte para o início da etapa e escolha Use existing role (Usar função existente) em vez de Create role (Criar função). Se você usar uma função existente, esta deve ter uma política anexada que inclua as permissões mostrados nesta etapa. Para obter mais informações sobre a função de serviço e sua declaração de política, consulte Editar uma política para uma função de serviço do IAM.

12. Se o processo de criação de perfil for bem-sucedido, a página do IAM será fechada e você retornará à página Perfil de serviço da AWS . Escolha Próxima etapa.
13. Na página Review your pipeline, verifique suas configurações e selecione Create pipeline.
14. Quando o pipeline estiver pronto, ele começará a localizar o código-fonte e implantar o aplicativo na pilha automaticamente. Esse processo pode levar alguns minutos.

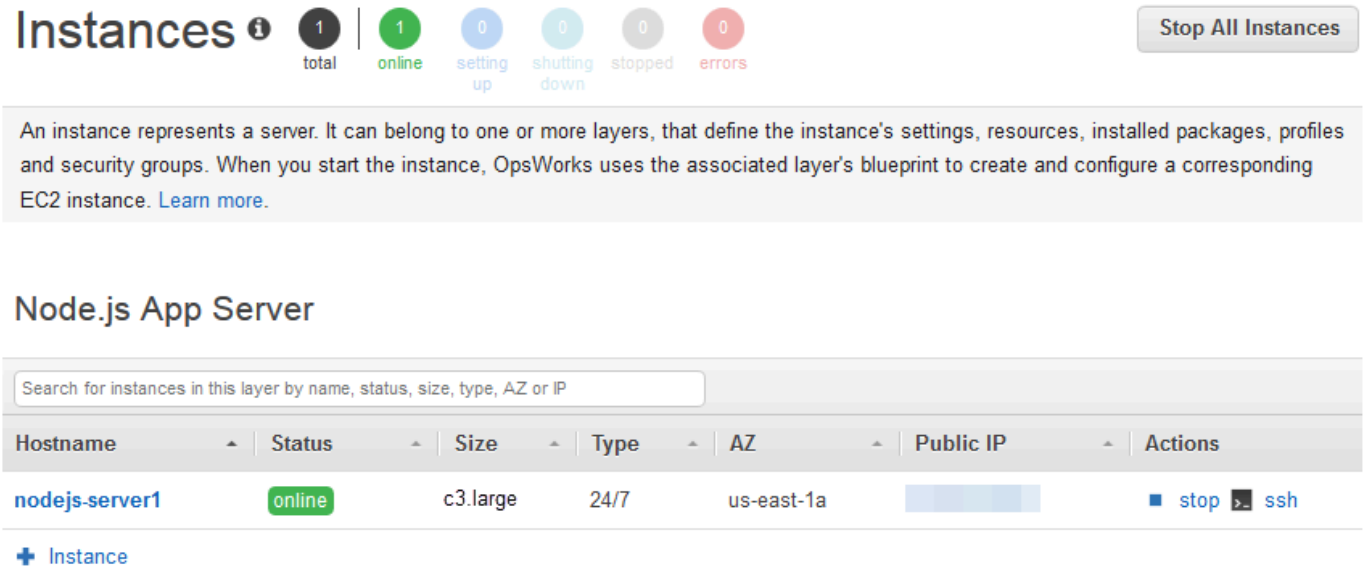
Etapa 6: Verificar a implantação do aplicativo no AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

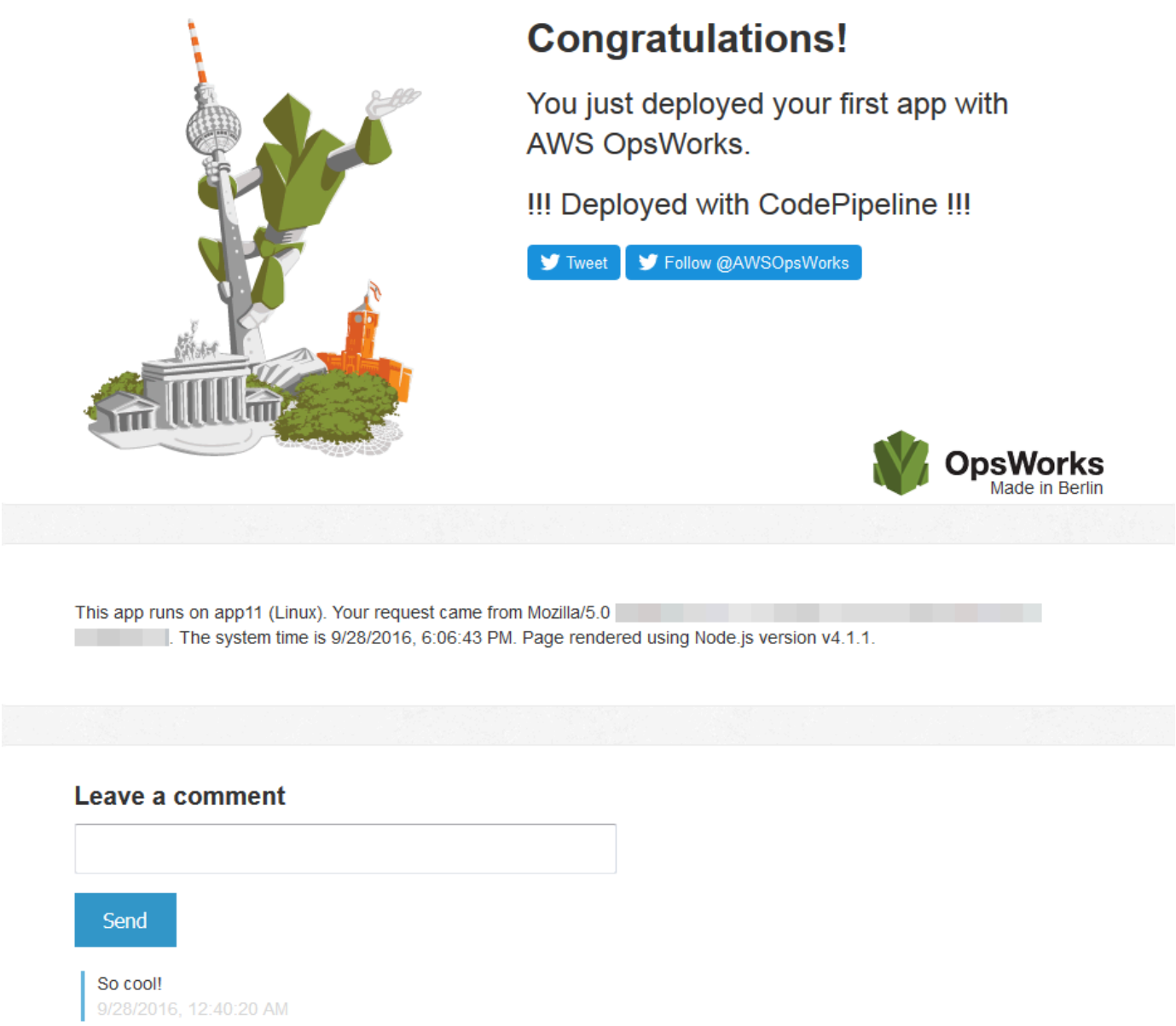
Para verificar se CodePipeline implantou o aplicativo Node.js na pilha, entre na instância criada em. Etapa 1: criar uma pilha, uma camada e uma instância em AWS OpsWorks Stacks Deve ser possível ver e usar o Node.js no aplicativo da Web.

Para verificar a implantação do aplicativo na sua instância do AWS OpsWorks Stacks

1. Abra o AWS OpsWorks console em https://console.aws.amazon.com/opsworks/.
2. No painel AWS OpsWorks Stacks, escolha e CodePipelineDemo, em seguida, escolha Node.js App Server.
3. No painel de navegação, escolha Instances e, em seguida, escolha o endereço IP público da instância criada para exibir o aplicativo web.



O aplicativo será exibido em uma nova guia de navegador.



Etapa 7 (opcional): Atualizar o código do aplicativo para ver o CodePipeline reimplantar o aplicativo automaticamente

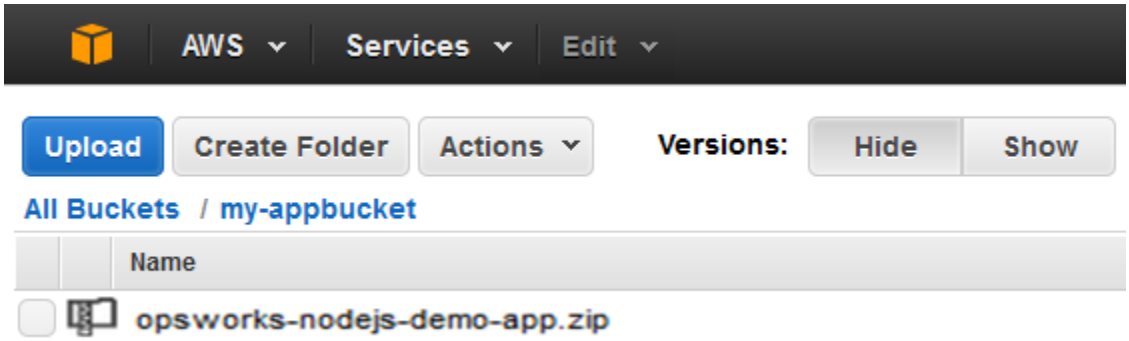
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre

migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Quando você fizer alterações de código nos aplicativos ou nos livros de receitas implantados por meio do CodePipeline, os artefatos atualizados serão implantados CodePipeline automaticamente nas instâncias de destino (nesse caso, em uma pilha de destino AWS OpsWorks do Stacks). Esta seção mostra a reimplantação automática quando o código é atualizado no aplicativo Node.js de exemplo. Se o código do aplicativo desta demonstração ainda estiver armazenado localmente, e se ninguém tiver feito alterações no código desde o início da demonstração, você poderá ignorar as etapas de 1 a 4 deste procedimento.

Para editar o código no aplicativo de exemplo

1. Faça login em AWS Management Console e abra o console do Amazon S3 em. https://console.aws.amazon.com/s3/
2. Abra o bucket no qual você está armazenando o aplicativo Node.js de exemplo.

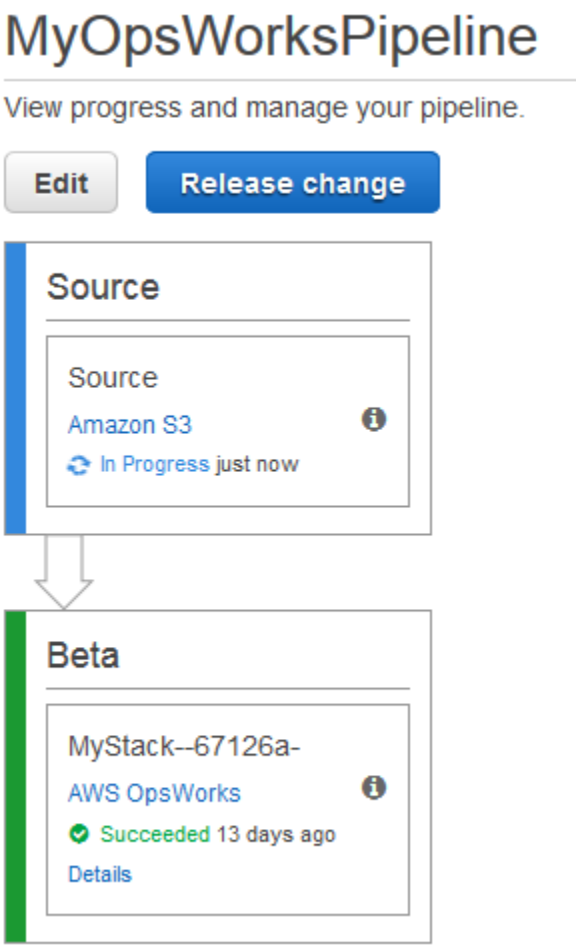


3. Selecione o arquivo ZIP que contém o aplicativo. No menu Actions, escolha Download.
4. Na caixa de diálogo, abra o menu de contexto (clique com o botão direito do mouse), escolha Download e salve o arquivo ZIP em um local conveniente. Escolha OK.
5. Extraia o conteúdo do arquivo ZIP para um local conveniente. Talvez você precise alterar as permissões na pasta extraída e em suas subpastas e conteúdo para permitir a edição. Na pasta `opsworks-nodejs-demo-app\views`, abra o arquivo `header.html` para edição.
6. Procure a frase `You just deployed your first app with`. Substitua a palavra `deployed` por `updated`. Na próxima linha, altere `AWS OpsWorks.` para `AWS OpsWorks and AWS CodePipeline.` Edite apenas o texto, nada mais.

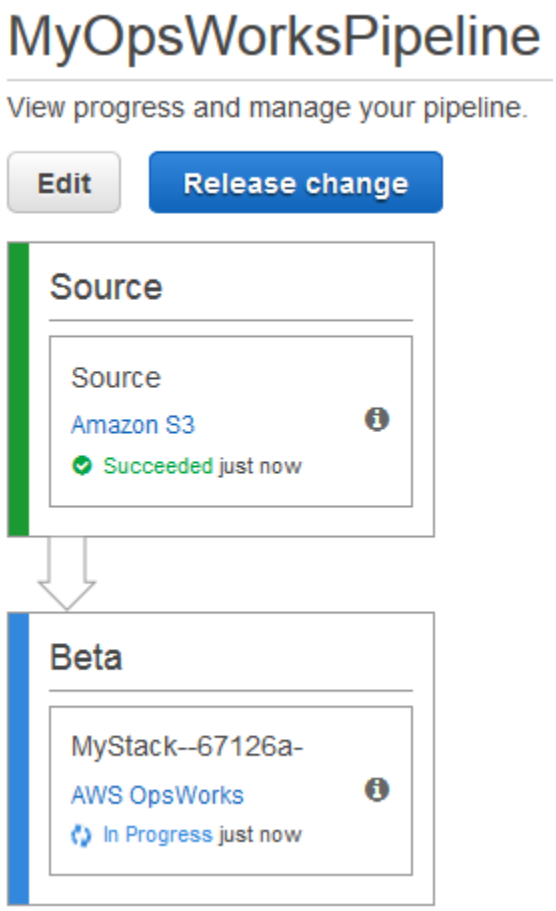
```
<div id="main" role="main">LF
  <div class="container">LF
    <div class="hero-unit">LF
      <div class="robot">LF
        <h1>Congratulations!</h1>LF
        <h2>LF
          You just updated your first app with<br/>LF
          AWS OpsWorks and AWS CodePipeline.LF
        </h2>LF
```

7. Salve e feche o arquivo `header.html`.
8. Compacte a pasta `opsworks-nodejs-demo-app` e salve o arquivo ZIP em um local conveniente. Não altere o nome do arquivo ZIP.
9. Carregue o novo arquivo ZIP em seu bucket no Amazon S3. Nesta demonstração, o nome do bucket é `my-appbucket`.
10. Abra o CodePipeline console do e o pipeline do AWS OpsWorks Stacks (MyOpsWorksPipeline). Selecione Release Change.

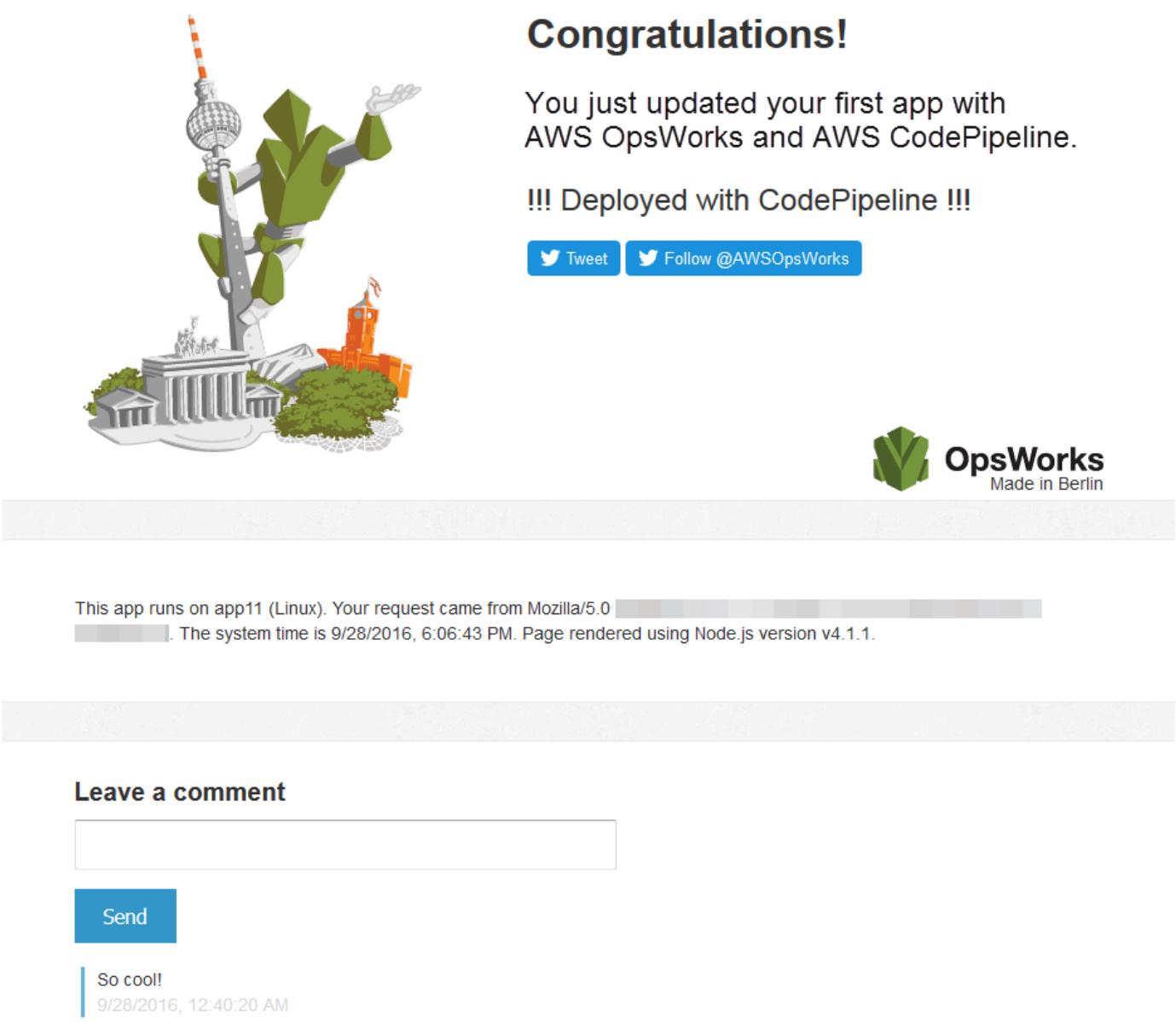
    (Você pode esperar CodePipeline para detectar a alteração de código na versão atualizada do aplicativo em seu bucket do Amazon S3. Para economizar seu tempo, este passo a passo instrui você a simplesmente escolher Release Change.)
11. Observe como CodePipeline são os estágios da tubulação. Primeiro, CodePipeline detecta as alterações no artefato de origem.



CodePipeline envia o código atualizado à pilha no AWS OpsWorks Stacks.



12. Quando ambos os estágios do pipeline tiverem sido concluídos com êxito, abra sua pilha no AWS OpsWorks Stacks.
13. Na página de propriedades da pilha, escolha Instances.
14. Na coluna Public IP (IP público), escolha o endereço IP público da instância para exibir o texto do aplicativo atualizado.



Etapa 8 (opcional): Limpar os recursos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre

migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para evitar cobranças indesejadas na AWS conta da, você pode excluir os AWS recursos da usados nesta demonstração. Esses AWS recursos da incluem a pilha do AWS OpsWorks Stacks, o perfil do IAM, o perfil de instância e o pipeline que você criou. CodePipeline No entanto, talvez seja necessário continuar usando os AWS recursos da para aprender mais sobre o AWS OpsWorks Stacks e. CodePipeline Se você quiser manter esses recursos, esta demonstração está concluída.

Para excluir o aplicativo da pilha

Como você não criou nem aplicou o aplicativo como parte do seu AWS CloudFormation modelo, exclua o aplicativo de teste Node.js antes de excluir a pilha. AWS CloudFormation

1. No console do AWS OpsWorks Stacks, no painel de navegação do serviço, escolha Aplicativos.
2. Na página Apps, selecione Node.js Demo App. Em Actions selecione delete. Quando for solicitada sua confirmação, escolha Excluir. AWS OpsWorks O Stacks excluirá o aplicativo.

Para excluir a pilha

Como você criou a pilha executando um AWS CloudFormation modelo, você pode excluir a pilha, incluindo a camada, a instância, o perfil da instância e o grupo de segurança que o modelo criou, no AWS CloudFormation console.

1. Abra o AWS CloudFormation console do.
2. No AWS CloudFormation painel do, selecione a pilha que você criou. No menu Actions (Ações), escolha Delete Stack (Excluir pilha). Quando for solicitada sua confirmação, escolha Yes, Delete (Sim, excluir).
3. Aguarde até DELETE_COMPLETE aparecer na coluna Status da pilha.

Para excluir o pipeline

1. Abra o CodePipeline console do.
2. No CodePipeline painel, escolha o pipeline que você criou para esta demonstração.
3. Na página do pipeline, selecione Edit.
4. Na página Editar, selecione Excluir. Quando for solicitada sua confirmação, escolha Delete.

AWS CodePipeline com AWS OpsWorks pilhas - Chef 11 pilhas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS CodePipelineO permite que você crie pipelines de entrega contínua que rastreiam alterações de código nas origens CodeCommit, como Amazon Simple Storage Service (Amazon S3) ou. GitHub O exemplo neste tópico descreve como criar e usar um pipeline simples CodePipeline como uma ferramenta de implantação para o código executado nas camadas do AWS OpsWorks Stacks. Neste exemplo, você cria um pipeline para um aplicativo PHP simples e, em seguida, instrui o AWS OpsWorks Stacks a executar o aplicativo em todas as instâncias em uma camada em uma pilha do Chef 11.10 (nesse caso, uma única instância).

> **Note**
>
> Este tópico descreve como usar um pipeline para executar e atualizar um aplicativo em uma pilha do Chef 11.10. Para obter informações sobre como usar um pipeline para executar e atualizar um aplicativo em uma pilha do Chef 12, consulte AWS CodePipeline com AWS OpsWorks pilhas - Chef 12 pilhas. O conteúdo entregue aos buckets do Amazon S3 pode conter conteúdo do cliente. Para obter mais informações sobre a remoção de dados confidenciais, consulte Como faço para esvaziar um bucket do S3? ou Como faço para excluir um bucket do S3?.

Tópicos

- Pré-requisitos
- Outros cenários compatíveis
- Etapa 1: Criar uma pilha, uma camada e uma instância no AWS OpsWorks Stacks
- Etapa 2: fazer upload do código do aplicativo em um bucket do Amazon S3
- Etapa 3: adicionar o aplicativo ao AWS OpsWorks Stacks

- Etapa 4: Criar um pipeline no CodePipeline
- Etapa 5: Verificar a implantação do aplicativo no Stacks AWS OpsWorks
- Etapa 6 (opcional): Atualizar o código do aplicativo para ver o CodePipeline reimplantar o aplicativo automaticamente
- Etapa 7 (opcional): Limpar os recursos

Pré-requisitos

Antes de iniciar esta demonstração, verifique se você tem permissões de administrador para executar todas as tarefas a seguir. Você pode ser membro de um grupo que tenha a AdministratorAccesspolítica aplicada ou pode ser membro de um grupo que tenha as permissões e políticas mostradas na tabela a seguir. Como prática recomendada de segurança, você deve pertencer a um grupo com permissões para executar as tarefas a seguir, em vez de atribuir permissões necessárias a usuários individuais.

Para obter mais informações sobre a criação de um grupo de segurança no IAM e a atribuição de permissões ao grupo, consulte Criar grupos de usuários do IAM. Para obter mais informações sobre como gerenciar permissões de AWS OpsWorks pilhas, consulte Melhores práticas: gerenciamento de permissões.

Permissões	Política recomendada para anexação a grupo
Crie e edite pilhas, camadas e instâncias em AWS OpsWorks Pilhas.	AWSOpsWorks_FullAccess
Crie, edite e execute modelos no AWS CloudFormation.	AmazonCloudFormationFullAccess
Crie, edite e acesse os buckets do Amazon S3.	Amazon S3 FullAccess
Crie, edite e execute pipelines em CodePipeline, especialmente os que usam o AWS OpsWorks Stacks como provedor.	AWSCodePipeline_FullAccess

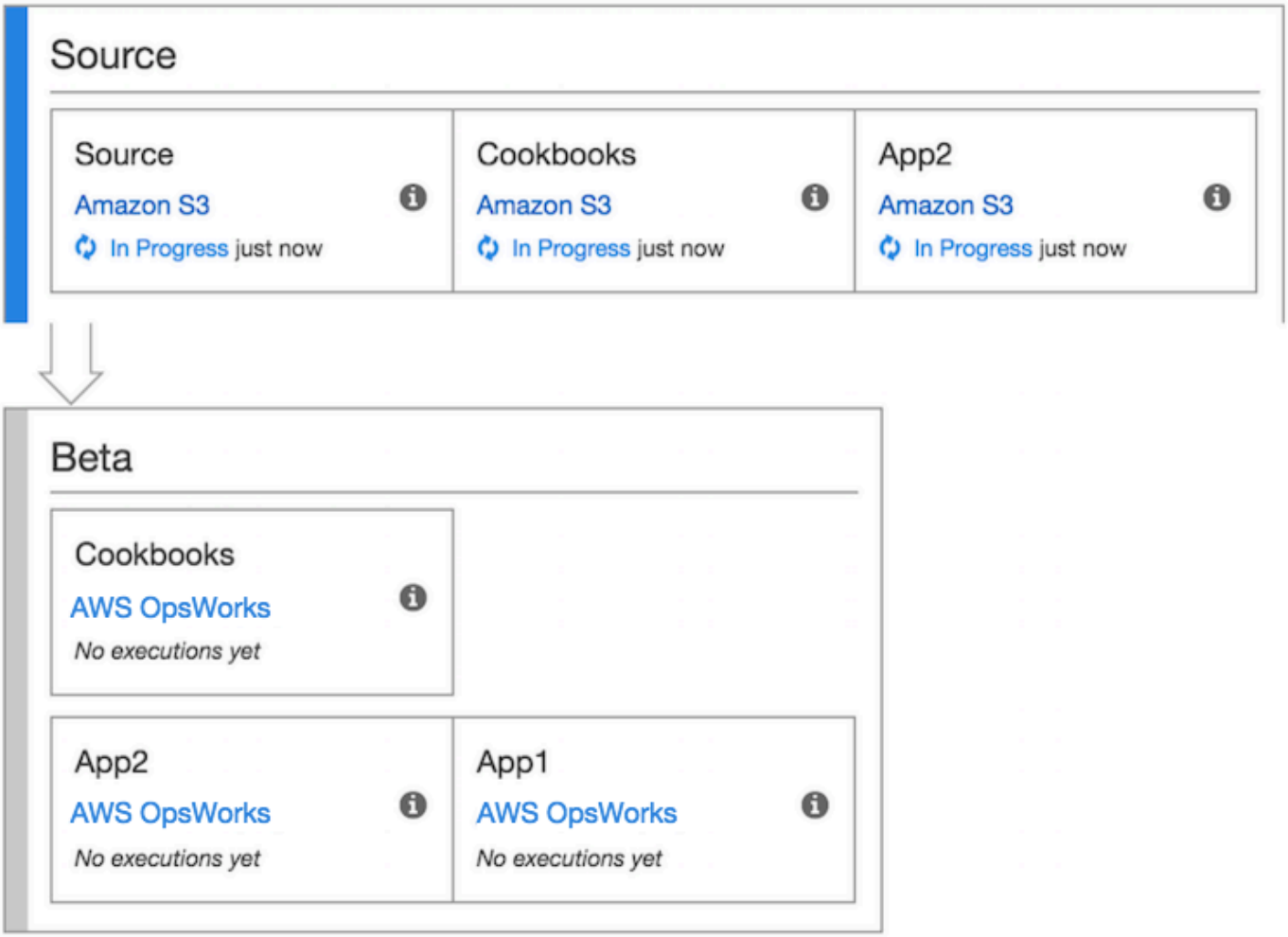
Você também deve ter um par de EC2 chaves da Amazon. Você será solicitado a fornecer o nome desse key pair ao executar o AWS CloudFormation modelo que cria a pilha, a camada e a instância de amostra neste passo a passo. Para obter mais informações sobre a obtenção de um par de

chaves no EC2 console da Amazon, consulte Criar um par de chaves na EC2 documentação da Amazon. O par de chaves deve estar na região Leste dos EUA (Norte da Virgínia). Você pode usar um par de chaves existente se já houver um nessa região.

Outros cenários compatíveis

Esta demonstração cria um pipeline simples que inclui um estágio Source e outro Deploy. No entanto, você pode criar pipelines mais complexos que usam o AWS OpsWorks Stacks como provedor. Veja a seguir exemplos de pipelines e cenários compatíveis:

- Você pode editar um pipeline para adicionar um livro de receitas do Chef ao estágio Source e um destino associado de livros de receitas atualizados ao estágio Deploy. Nesse caso, você adiciona uma ação Deploy que aciona a atualização dos seus livros de receitas quando são feitas alterações na origem. O livro de receitas atualizado é implantado antes do seu aplicativo.
- Você pode criar um pipeline complexo, com livros de receitas personalizados e vários aplicativos, e implantar em uma AWS OpsWorks pilha Stacks. O pipeline rastreia as alterações nas origens de aplicativos e livros de receitas, e reimplanta quando você faz alterações. A tabela a seguir mostra um exemplo de um pipeline similar e complexo:



Para obter mais informações sobre como trabalhar com CodePipeline, consulte a CodePipelinedocumentação.

Etapa 1: Criar uma pilha, uma camada e uma instância no AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para usar o AWS OpsWorks Stacks como provedor de implantação para um pipeline, você deve primeiro ter uma pilha, uma camada e pelo menos uma instância na camada. Embora você possa criar uma pilha no AWS OpsWorks Stacks seguindo as instruções em Getting Started with Linux Stacks ou Getting Started with Windows Stacks, para economizar seu tempo, este exemplo usa um AWS CloudFormation modelo para criar uma pilha, camada e instância do Chef 11.10 com base em Linux. A instância criada por este modelo executa o Amazon Linux 2016.03 e tem um tipo de instância `c3.large`.

> **Important**
>
> O AWS CloudFormation modelo do deve ser armazenado e executado na mesma região que o bucket do Amazon S3 no qual você fará o upload do aplicativo posteriormente e na mesma região na qual criará o pipeline mais tarde. CodePipeline No momento, CodePipeline oferece suporte ao provedor do AWS OpsWorks Stacks apenas na região Leste dos EUA (us-east-1). Todos os recursos desta demonstração devem ser criados na região Leste dos EUA (N. da Virgínia).
>
> Se houver falha na criação da pilha, você pode estar se aproximando do número máximo permitido de funções do IAM na sua conta. A criação da pilha também pode falhar se sua conta não conseguir executar instâncias com o tipo de instância `c3.large`. Por exemplo, se você estiver usando o nível gratuito da AWS , talvez receba um erro como `Root device type: must be included in EBS`. Se sua conta tem limitações quanto aos tipos de instância que você pode criar, como limitações impostas pelo nível AWS gratuito da, tente alterar o valor do `InstanceType` parâmetro no bloco de instâncias do modelo para um tipo de instância que sua conta pode usar.

Para criar uma pilha, camada e instância usando AWS CloudFormation

1. Copie o AWS CloudFormation modelo a seguir em um novo documento de texto sem formatação. Salve o arquivo em um local conveniente no computador local e nomeie-o como NewOpsWorksStack.template ou outro nome que seja conveniente para você.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Mappings": {
    "Region2Principal": {
      "us-east-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
```

```
      },
      "us-west-2": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "us-west-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "eu-west-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "ap-southeast-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "ap-northeast-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "ap-northeast-2": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "ap-southeast-2": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "sa-east-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      },
      "cn-north-1": {
        "EC2Principal": "ec2.amazonaws.com.cn",
        "OpsWorksPrincipal": "opsworks.amazonaws.com.cn"
      },
      "eu-central-1": {
        "EC2Principal": "ec2.amazonaws.com",
        "OpsWorksPrincipal": "opsworks.amazonaws.com"
      }
    }
  },
  "Parameters": {
```

```
    "EC2KeyPairName": {
   "Type": "String",
   "Description": "The name of an existing EC2 key pair that allows you to use SSH
to connect to the OpsWorks instance."
 }
 },
 "Resources": {
"CPOpsDeploySecGroup": {
  "Type": "AWS::EC2::SecurityGroup",
  "Properties": {
    "GroupDescription" : "Lets you manage OpsWorks instances deployed to by
CodePipeline"
  }
},
"CPOpsDeploySecGroupIngressHTTP": {
  "Type": "AWS::EC2::SecurityGroupIngress",
  "Properties" : {
    "IpProtocol" : "tcp",
       "FromPort" : "80",
       "ToPort" : "80",
       "CidrIp" : "0.0.0.0/0",
 "GroupId": {
   "Fn::GetAtt": [
     "CPOpsDeploySecGroup", "GroupId"
   ]
 }
     }
},
"CPOpsDeploySecGroupIngressSSH": {
  "Type": "AWS::EC2::SecurityGroupIngress",
  "Properties" : {
    "IpProtocol" : "tcp",
       "FromPort" : "22",
       "ToPort" : "22",
       "CidrIp" : "0.0.0.0/0",
 "GroupId": {
   "Fn::GetAtt": [
     "CPOpsDeploySecGroup", "GroupId"
   ]
 }
  }
},
"MyStack": {
     "Type": "AWS::OpsWorks::Stack",
```

```
      "Properties": {
        "Name": {
          "Ref": "AWS::StackName"
        },
        "ServiceRoleArn": {
          "Fn::GetAtt": [
            "OpsWorksServiceRole",
            "Arn"
          ]
        },
  "ConfigurationManager" : { "Name": "Chef","Version": "11.10" },
  "DefaultOs": "Amazon Linux 2016.03",
        "DefaultInstanceProfileArn": {
          "Fn::GetAtt": [
            "OpsWorksInstanceProfile",
            "Arn"
          ]
        }
      }
    },
    "MyLayer": {
      "Type": "AWS::OpsWorks::Layer",
      "Properties": {
        "StackId": {
          "Ref": "MyStack"
        },
        "Name": "MyLayer",
        "Type": "php-app",
  "Shortname": "mylayer",
        "EnableAutoHealing": "true",
        "AutoAssignElasticIps": "false",
        "AutoAssignPublicIps": "true",
  "CustomSecurityGroupIds": [
    {
      "Fn::GetAtt": [
              "CPOpsDeploySecGroup", "GroupId"
      ]
          }
        ]
      },
      "DependsOn": [
        "MyStack",
        "CPOpsDeploySecGroup"
      ]
```

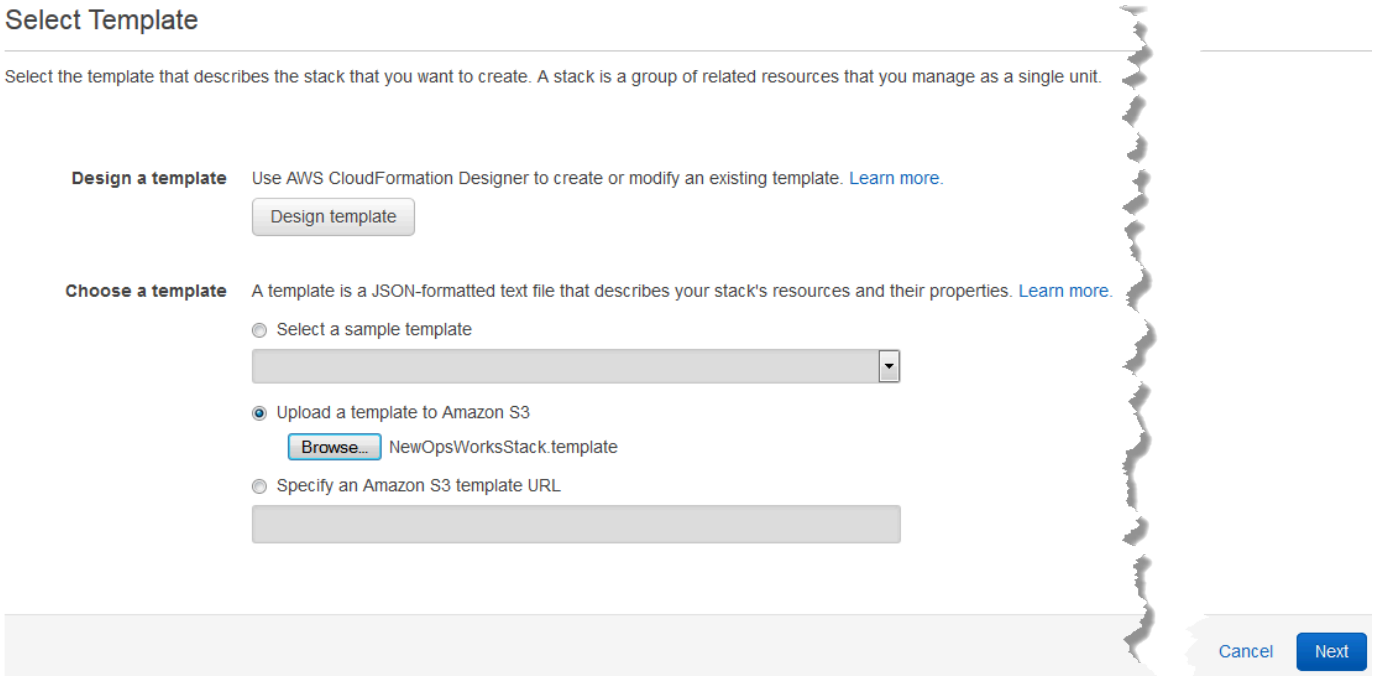
```
    },
    "OpsWorksServiceRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  {
                    "Fn::FindInMap": [
                      "Region2Principal",
                      {
                        "Ref": "AWS::Region"
                      },
                      "OpsWorksPrincipal"
                    ]
                  }
                ]
              },
              "Action": [
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path": "/",
        "Policies": [
          {
            "PolicyName": "opsworks-service",
            "PolicyDocument": {
              "Statement": [
                {
                  "Effect": "Allow",
                  "Action": [
                    "ec2:*",
                    "iam:PassRole",
                    "cloudwatch:GetMetricStatistics",
                    "elasticloadbalancing:*"
                  ],
                  "Resource": "*"
                }
              ]
```

```
            }
          }
        ]
      }
    },
    "OpsWorksInstanceProfile": {
      "Type": "AWS::IAM::InstanceProfile",
      "Properties": {
        "Path": "/",
        "Roles": [
          {
            "Ref": "OpsWorksInstanceRole"
          }
        ]
      }
    },
    "OpsWorksInstanceRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  {
                    "Fn::FindInMap": [
                      "Region2Principal",
                      {
                        "Ref": "AWS::Region"
                      },
                      "EC2Principal"
                    ]
                  }
                ]
              },
              "Action": [
                "sts:AssumeRole"
              ]
            }
          ]
        },
        "Path": "/",
  "Policies": [
```

```
          {
            "PolicyName": "s3-get",
            "PolicyDocument": {
              "Version": "2012-10-17",
              "Statement": [
                {
                  "Effect": "Allow",
                  "Action": [
                    "s3:GetObject"
                  ],
                  "Resource": "*"
                }
              ]
            }
          }
        ]
      }
    },
    "myinstance": {
      "Type": "AWS::OpsWorks::Instance",
      "Properties": {
        "LayerIds": [
          {
            "Ref": "MyLayer"
          }
        ],
        "StackId": {
          "Ref": "MyStack"
        },
        "InstanceType": "c3.large",
        "SshKeyName": {
    "Ref": "EC2KeyPairName"
  }
      }
    }
  },
  "Outputs": {
    "StackId": {
      "Description": "Stack ID for the newly created AWS OpsWorks stack",
      "Value": {
        "Ref": "MyStack"
      }
    }
  }
```

```
}
```

2. Faça login no AWS Management Console e abra o AWS CloudFormation console em https://console.aws.amazon.com/cloudformation.
3. Na página AWS CloudFormation inicial, escolha Criar pilha.
4. Na página Select Template, na área Choose a template, selecione a opção Upload a template to Amazon S3 e, em seguida, Browse.
5. Navegue até o AWS CloudFormation modelo que você salvou na etapa 1 e escolha Abrir. Na página Select Template, escolha Next.

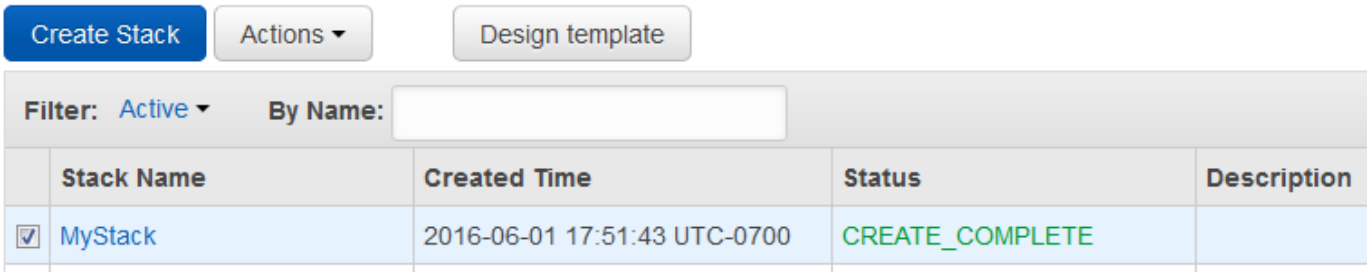


6. Na página Especificar detalhes, nomeie a pilha MyStackou qualquer nome de pilha exclusivo da sua conta. Se você escolher um nome diferente para a pilha, altere o nome da pilha durante esta demonstração.
7. Na área Parâmetros, forneça o nome de um EC2 key pair que você deseja usar para acessar sua instância do AWS OpsWorks Stacks depois que ela for criada. Escolha Próximo.
8. Na página Options (Opções), escolha Next (Avançar). (As configurações desta página não são necessárias nesta demonstração.)
9. O AWS CloudFormation modelo que você usa neste passo a passo cria funções do IAM, um perfil de instância e uma instância.

> **Important**
>
> Antes de escolher Criar, escolha Custo para estimar os encargos que podem ser gerados AWS pela criação de recursos com esse modelo.

Se a criação de recursos do IAM for aceitável, marque a caixa de seleção Eu reconheço que este modelo pode fazer com que CloudFormation a AWS crie recursos IAM e, em seguida, escolha a opção Criar. Se a criação de recursos do IAM não for aceitável, você não poderá continuar esse procedimento.

10. No AWS CloudFormation painel, você pode visualizar o andamento da criação da pilha. Antes de passar para a próxima etapa, aguarde até que CREATE_COMPLETE seja exibido na coluna Status.



Para verificar a criação da pilha em AWS OpsWorks Pilhas

1. Abra o AWS OpsWorks console em https://console.aws.amazon.com/opsworks/.
2. No painel de AWS OpsWorks pilhas, veja a pilha que você criou.

3. Abra a pilha e exiba a camada e a instância. Observe que a camada e a instância foram criadas com os nomes e outros metadados fornecidos no AWS CloudFormation modelo. Você está pronto para fazer upload do aplicativo no bucket do Amazon S3.

Etapa 2: fazer upload do código do aplicativo em um bucket do Amazon S3

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Como você deve fornecer um link para o repositório de código como parte da configuração do pipeline, tenha o código do repositório a mão antes de criar o pipeline. Nesta demonstração, você faz upload de um aplicativo PHP em um bucket do Amazon S3.

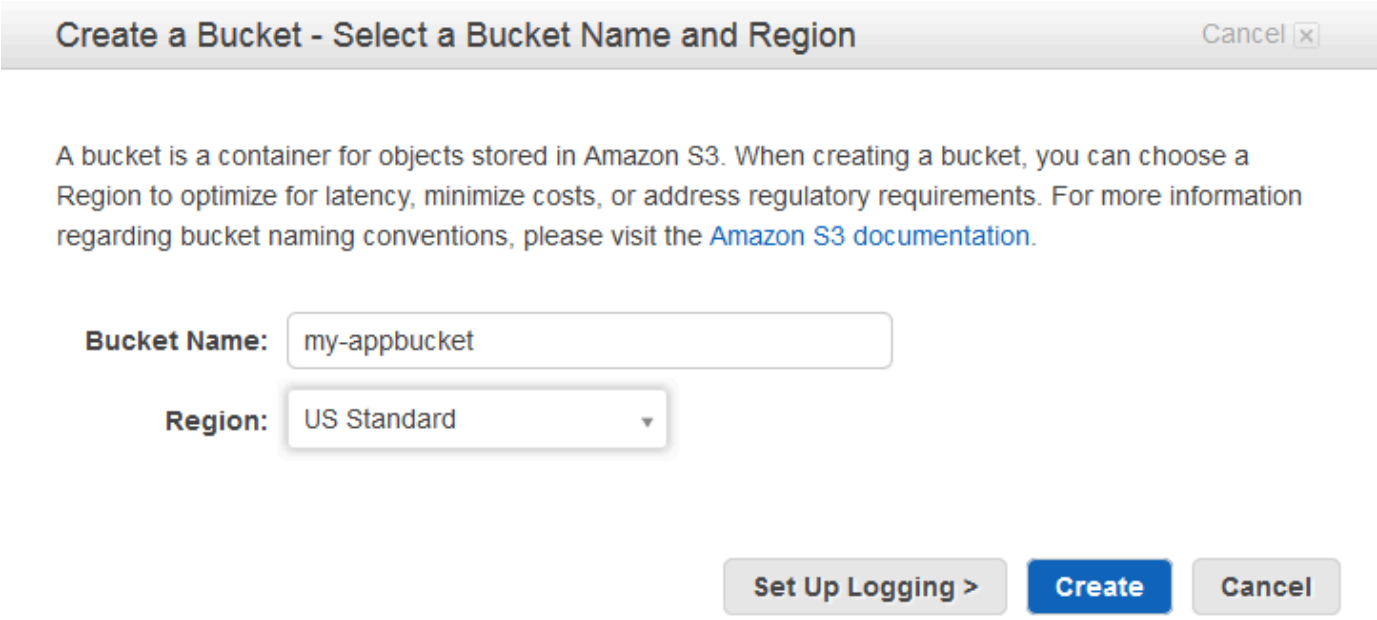
Embora CodePipeline possa usar o código diretamente de GitHub ou CodeCommit como origens, este passo a passo demonstra como usar um bucket do Amazon S3. O bucket do Amazon S3 permite CodePipeline detectar alterações no código do aplicativo e implantar o aplicativo alterado automaticamente. Se desejar, use um bucket existente. Verifique se o bucket atende aos critérios descritos em CodePipeline Demonstração do pipeline (Bucket do Amazon S3) na documentação. CodePipeline

> **Important**
>
> O bucket do Amazon S3 deve estar na mesma região em que você criará o pipeline posteriormente. No momento, CodePipeline oferece suporte ao provedor do AWS OpsWorks Stacks apenas na região Leste dos EUA (us-east-1). Todos os recursos desta demonstração devem ser criados na região Leste dos EUA (N. da Virgínia). O bucket também deve ser versionado porque CodePipeline requer uma origem versionada. Para obter mais informações, consulte Usar versionamento.

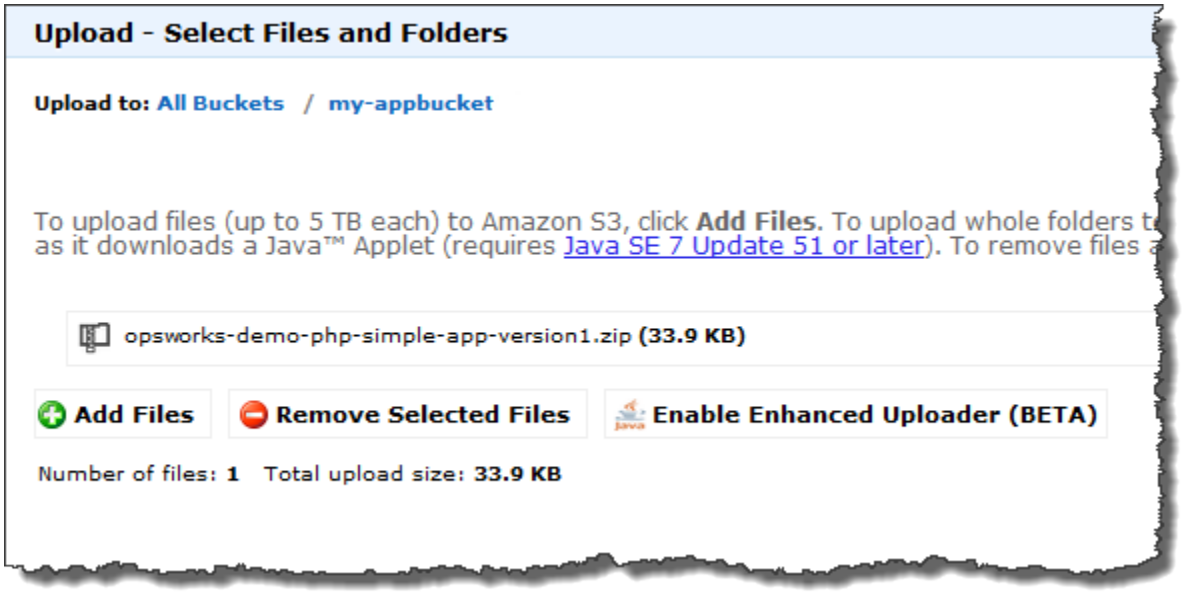
Para carregar seu aplicativo para o bucket do Amazon S3

1. No GitHub site, baixe um arquivo ZIP do aplicativo PHP de amostra AWS OpsWorks Stacks e salve-o em um local conveniente em seu computador local.

2. Certifique-se de que `index.php` e a pasta `ASSETS` estão no nível raiz do arquivo ZIP baixado. Se não estiverem, dezipe o arquivo e crie um novo arquivo ZIP que tenha esses arquivos no nível raiz.
3. Abra o console do Amazon S3 em https://console.aws.amazon.com/s3/.
4. Escolha Criar bucket.
5. Na página Create a Bucket - Select a Bucket Name and Region, digite um nome exclusivo para o seu bucket em Bucket Name. Os nomes de bucket devem ser exclusivos em todas as AWS contas, não apenas na conta da. Esta demonstração usa o nome **`my-appbucket`**, mas é possível usar `my-appbucket-`*`yearmonthday`* para fazer com que o nome do bucket seja exclusivo. Na lista suspensa Region, escolha a opção US Standard e, em seguida, Create. US Standard equivale a `us-east-1`.



6. Na lista All Buckets, selecione o bucket que você criou.
7. Na página do bucket, escolha Upload (Carregar).
8. Na página Upload - Select Files and Folders, selecione Add files. Procure o arquivo ZIP salvo na etapa 1, clique em Open e, em seguida, Start Upload.



9. Após a conclusão do upload, selecione o arquivo ZIP na lista de arquivos do bucket e escolha a opção Properties.
10. No painel Properties, copie o link para o arquivo ZIP e anote o link. Você precisará da parte do nome do bucket e do nome do arquivo ZIP deste link para criar o pipeline.

Etapa 3: adicionar o aplicativo ao AWS OpsWorks Stacks

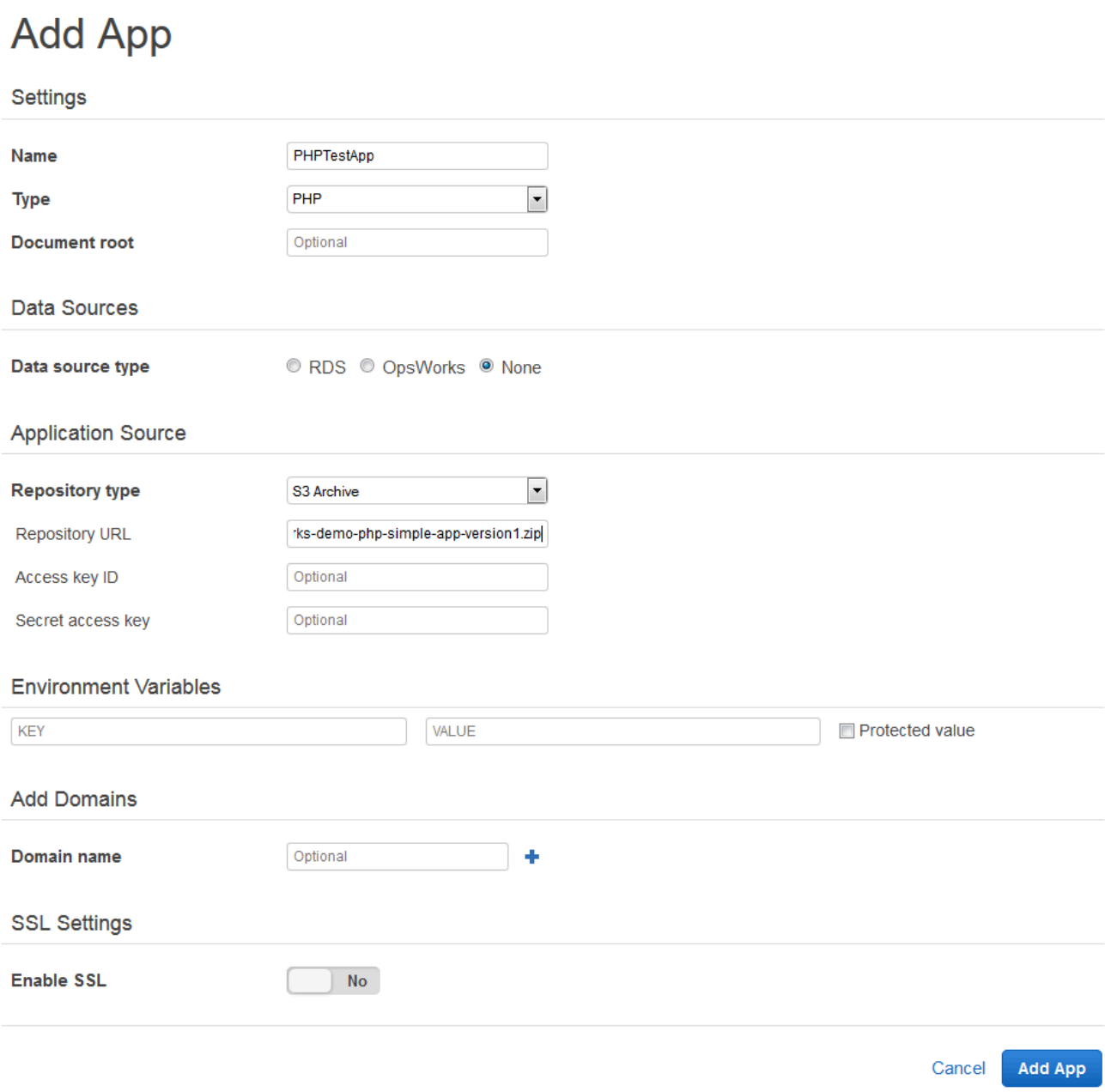
> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Antes de criar um pipeline no CodePipeline, adicione o aplicativo de teste PHP ao AWS OpsWorks Stacks. Ao criar o pipeline, você precisará selecionar o aplicativo que você adicionou ao AWS OpsWorks Stacks.

Prepare o link do bucket do Amazon S3 da etapa 10 do procedimento anterior. Você precisará do link para o bucket em que armazenou o aplicativo de teste para concluir esse procedimento.

Para adicionar um aplicativo ao AWS OpsWorks Stacks

1. No console AWS OpsWorks Stacks, abra e MyStack, no painel de navegação, escolha Apps.
2. Escolha Add app (Adicionar aplicativo).
3. Na página Add App, forneça as seguintes informações:
   a. Especifique um nome para o aplicativo. Esta demonstração usa o nome `PHPTestApp`.
   b. Na lista suspensa Type, escolha PHP.
   c. Para Data source type (Tipo de origem de dados), escolha a opção None (Nenhum). Este aplicativo não requer um banco de dados externo ou fonte de dados.
   d. Na lista suspensa Repository type, escolha a opção S3 Archive.
   e. Na caixa de strings Repository URL, cole a URL copiada na etapa 10 de Etapa 2: fazer upload do código do aplicativo em um bucket do Amazon S3. O formulário deve ser semelhante ao seguinte:



4. Você não precisa alterar outras configurações neste formulário. Escolha Add App.
5. Quando o PHPTestaplicativo App aparecer na lista na página Aplicativos, continue com o próximo procedimento,Etapa 4: Criar um pipeline no CodePipeline.

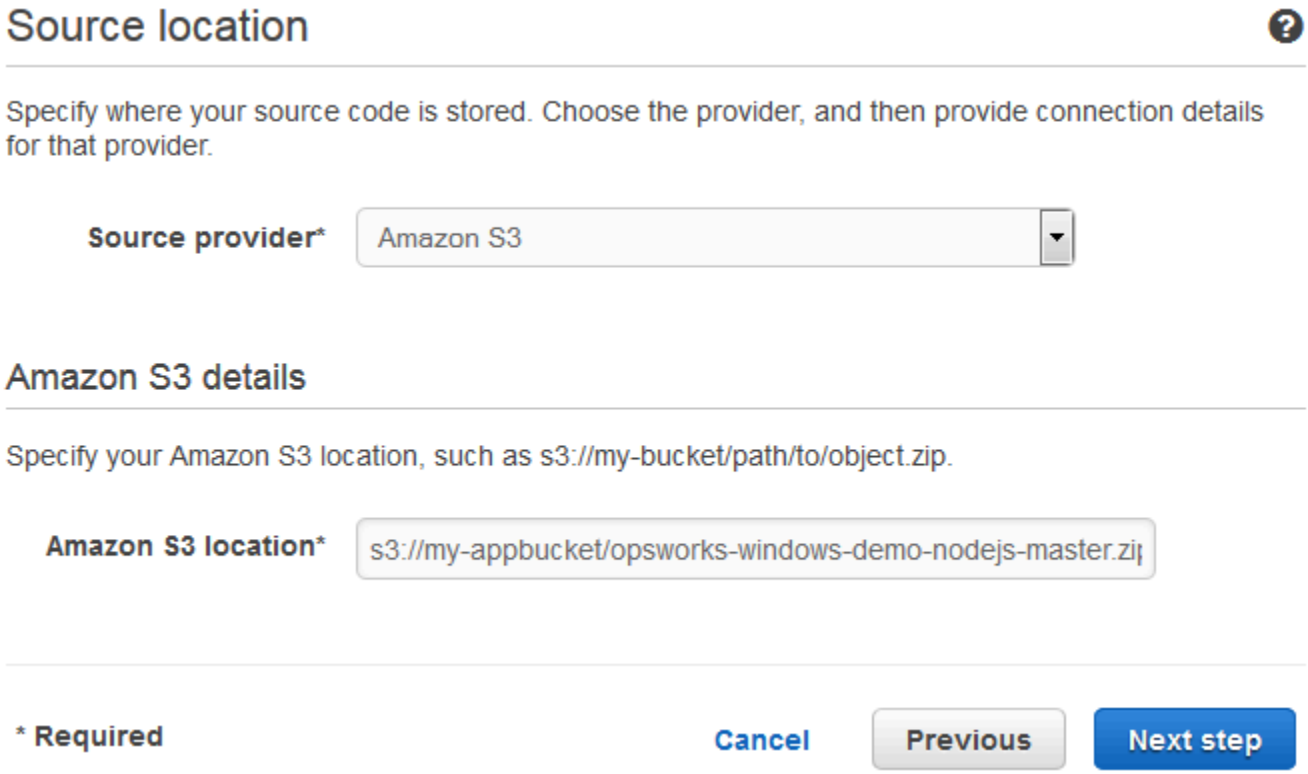
Etapa 4: Criar um pipeline no CodePipeline

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

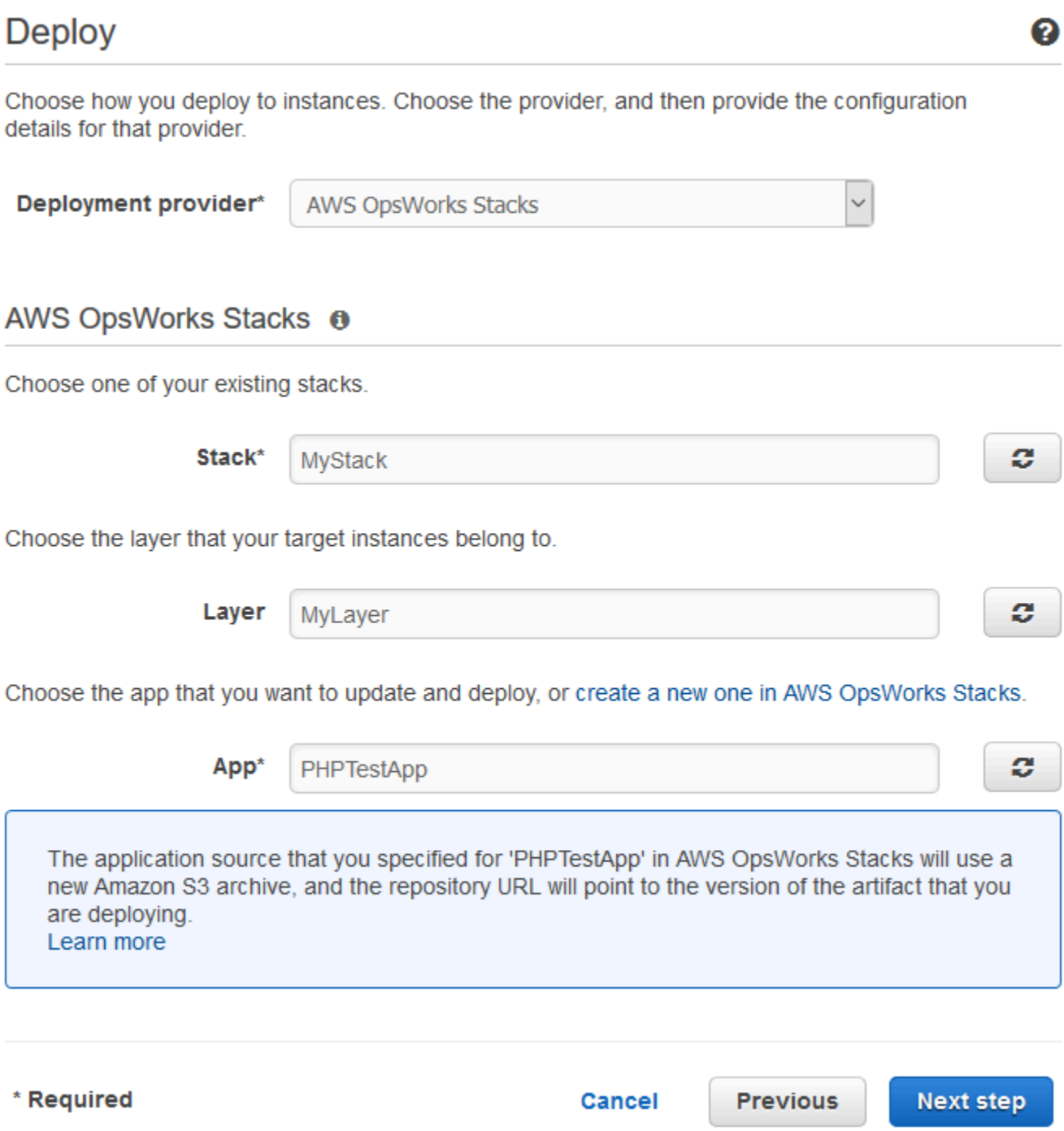
Quando você tiver uma pilha com uma camada e, pelo menos, uma instância configurado no AWS OpsWorks Stacks, crie um pipeline CodePipeline com o AWS OpsWorks Stacks como provedor para implantar aplicativos ou livros de receitas do Chef nos recursos do Stacks. AWS OpsWorks

Para criar um pipeline

1. Abra o CodePipeline console em https://console.aws.amazon.com/codepipeline/.
2. Selecione Criar pipeline.
3. Na CodePipeline página Introdução**`MyOpsWorksPipeline`**, digite ou qualquer outro nome de funil exclusivo da sua conta e escolha Próxima etapa.
4. Na página Source Location, selecione Amazon S3 na lista suspensa Source provider.
5. Na área Detalhes do Amazon S3, digite o caminho do bucket do Amazon S3 no formato **`s3://`*`bucket-name`*`/`*`file name`***. Consulte o link anotado na etapa 10 de Etapa 2: fazer upload do código do aplicativo em um bucket do Amazon S3. Nesta demonstração, o caminho é `s3://my-appbucket/opsworks-demo-php-simple-app-version1.zip`. Escolha Próxima etapa.



6. Na página Build, escolha No Build na lista suspensa e escolha Next step.
7. Na página Deploy (Implantar), escolha AWS OpsWorks Stacks como provedor de implantação.



8. No campo Stack, digite `MyStack` ou o nome da pilha criada em Etapa 1: Criar uma pilha, uma camada e uma instância no AWS OpsWorks Stacks.
9. No campo Layer, digite `MyLayer` ou o nome da layer criada em Etapa 1: Criar uma pilha, uma camada e uma instância no AWS OpsWorks Stacks.

10. No campo Aplicativo, selecione o aplicativo carregado no Amazon S3 em Etapa 2: fazer upload do código do aplicativo em um bucket do Amazon S3 e escolha Próxima etapa.
11. Na página AWS Service Role, selecione Create Role.

    Uma nova janela é aberta com uma página do console do IAM que descreve o perfil que será criado para você, `AWS-CodePipeline-Service`. Na lista suspensa Policy name, escolha a opção Create new policy. Certifique-se de que o documento de política tem o conteúdo a seguir. Escolha Edit para alterar o documento da política, se necessário.

```
{
    "Statement": [
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:GetBucketVersioning"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": "opsworks:*",
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

    Assim que você terminar de fazer alterações no documento da política, escolha Allow (Permitir). Suas alterações serão exibidas no console do IAM.

Hide Details

Role Summary

Role Description	Provides read and write access to AWS services and resources.
IAM Role	AWS-CodePipeline-Service
Policy Name	Create a new Role Policy

Hide Policy Document

Edit

```
{
  "Statement": [
    {
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion",
        "s3:GetBucketVersioning"
      ],
      "Resource": "*",
      "Effect": "Allow"
    },
```

> **Note**
>
> Se a criação da função falhar, talvez já haja uma função do IAM chamada AWS-CodePipeline -Service. Se você estava usando a função AWS- CodePipeline -Service antes de maio de 2016, a função pode não ter permissões para usar o AWS OpsWorks Stacks como provedor de implantação; nesse caso, você deve atualizar a declaração de política conforme mostrado nesta etapa. Caso uma mensagem de erro seja exibida, volte para o início da etapa e escolha Use existing role (Usar função existente) em vez de Create role (Criar função). Se você usar uma função existente, esta deve ter uma política anexada que inclua as permissões mostrados nesta etapa. Para obter mais informações sobre a função de serviço e sua declaração de política, consulte Editar uma política para uma função de serviço do IAM.

12. Se o processo de criação de perfil for bem-sucedido, a página do IAM será fechada e você retornará à página Perfil de serviço da AWS. Escolha Próxima etapa.
13. Na página Review your pipeline, verifique suas configurações e selecione Create pipeline.

We will create your pipeline with the following resources.

Source Stage

Source provider Amazon S3

Amazon S3 location s3://my-appbucket0/opsworks-demo-php-simple-app-version1.zip

Build Stage

Build provider No Build

Beta Stage

Deployment provider AWS OpsWorks

Stack MyStack

App PHPTestApp

Layer MyLayer

Pipeline settings

Pipeline name MyOpsWorksPipeline

Artifact location s3://codepipeline-us-east-
AWS CodePipeline will use this existing S3 bucket to store artifacts for this pipeline. Depending on the size of your artifacts, you might be charged for storage costs. For more information, see Amazon S3 storage pricing.

Role name AWS-CodePipeline-Service

To save this configuration with these resources, choose Create pipeline.

Would you like to create this pipeline?

Cancel Previous Create pipeline

14. Quando o pipeline estiver pronto, ele começará a localizar o código-fonte e implantar o aplicativo na pilha automaticamente. Esse processo pode levar alguns minutos.

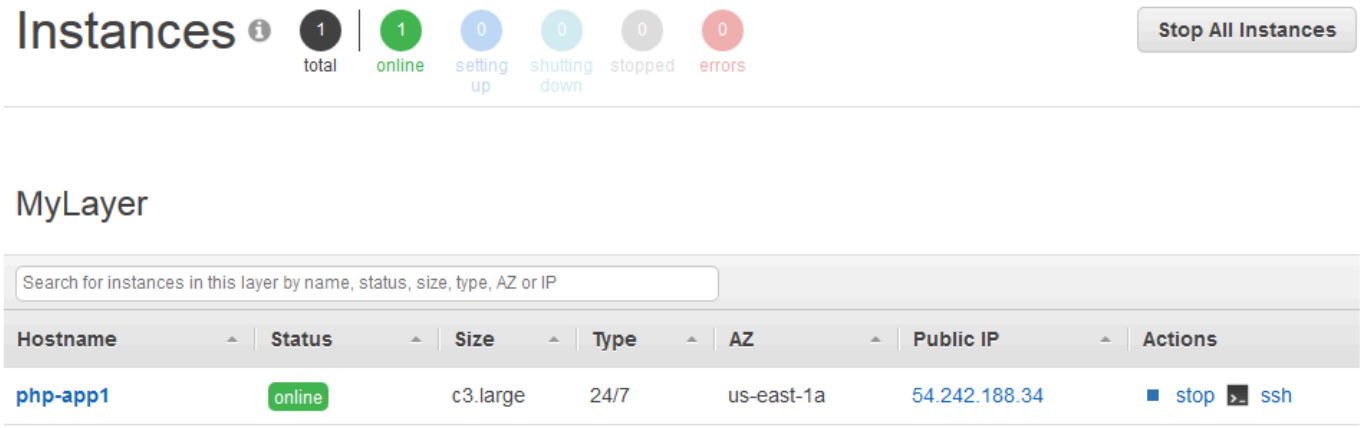
Etapa 5: Verificar a implantação do aplicativo no Stacks AWS OpsWorks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

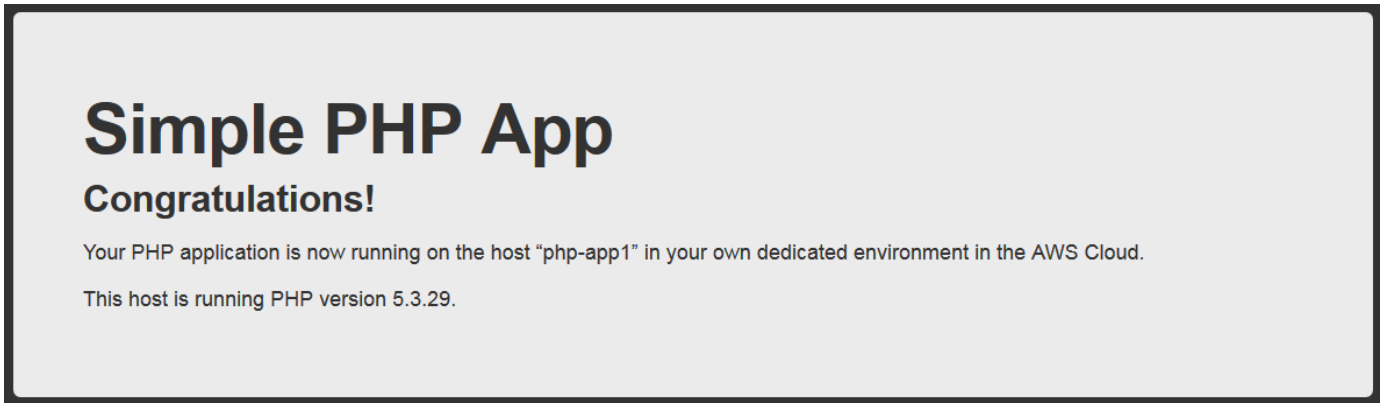
Para verificar se CodePipeline implantou o aplicativo PHP na pilha, entre na instância criada em. Etapa 1: Criar uma pilha, uma camada e uma instância no AWS OpsWorks Stacks Você conseguirá ver e usar a aplicação web PHP.

Para verificar a implantação do aplicativo na sua instância do AWS OpsWorks Stacks

1. Abra o AWS OpsWorks console em https://console.aws.amazon.com/opsworks/.
2. No painel AWS OpsWorks Stacks, escolha e MyStack, em seguida, escolha MyLayer.
3. No painel de navegação, escolha Instances e, em seguida, escolha o endereço IP público da instância criada para exibir o aplicativo web.



O aplicativo será exibido em uma nova guia de navegador.



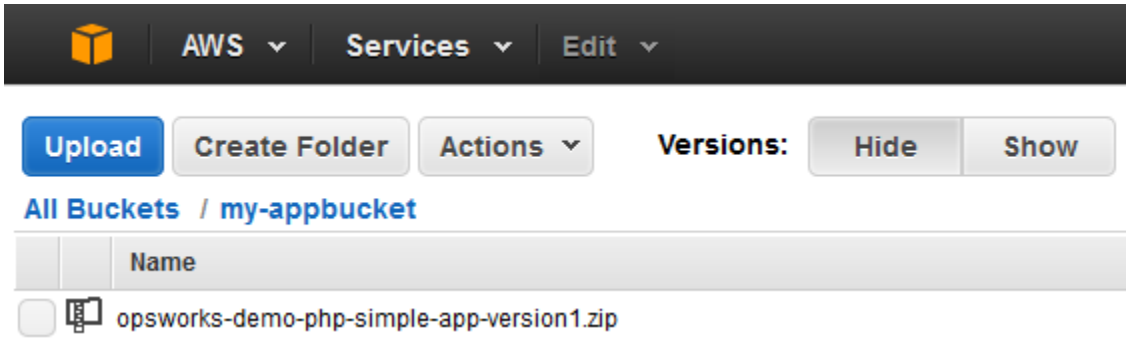
Etapa 6 (opcional): Atualizar o código do aplicativo para ver o CodePipeline reimplantar o aplicativo automaticamente

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Quando você fizer alterações de código nos aplicativos ou nos livros de receitas implantados por meio do CodePipeline, os artefatos atualizados serão implantados CodePipeline automaticamente nas instâncias de destino (nesse caso, em uma pilha de destino AWS OpsWorks do Stacks). Esta seção mostra a reimplantação automática quando o código é atualizado no aplicativo PHP de exemplo.

Para editar o código no aplicativo de exemplo

1. Faça login em AWS Management Console e abra o console da Amazon S3 em. https://console.aws.amazon.com/s3/
2. Abra o bucket no qual você está armazenando o aplicativo PHP de exemplo.



3. Selecione o arquivo ZIP que contém o aplicativo. No menu Actions, escolha Download.
4. Na caixa de diálogo, abra o menu de contexto (clique com o botão direito do mouse), escolha Download e salve o arquivo ZIP em um local conveniente. Escolha OK.
5. Extraia o conteúdo do arquivo ZIP para um local conveniente. Talvez você precise alterar as permissões na pasta extraída e em suas subpastas e conteúdo para permitir a edição. Na pasta `opsworks-demo-php-simple-app-version1`, abra o arquivo `index.php` para edição.
6. Procure a frase `Your PHP application is now running`. Substitua o texto `Your PHP application is now running` por `You've just deployed your first app to AWS OpsWorks with AWS CodePipeline,`. Não edite as variáveis.

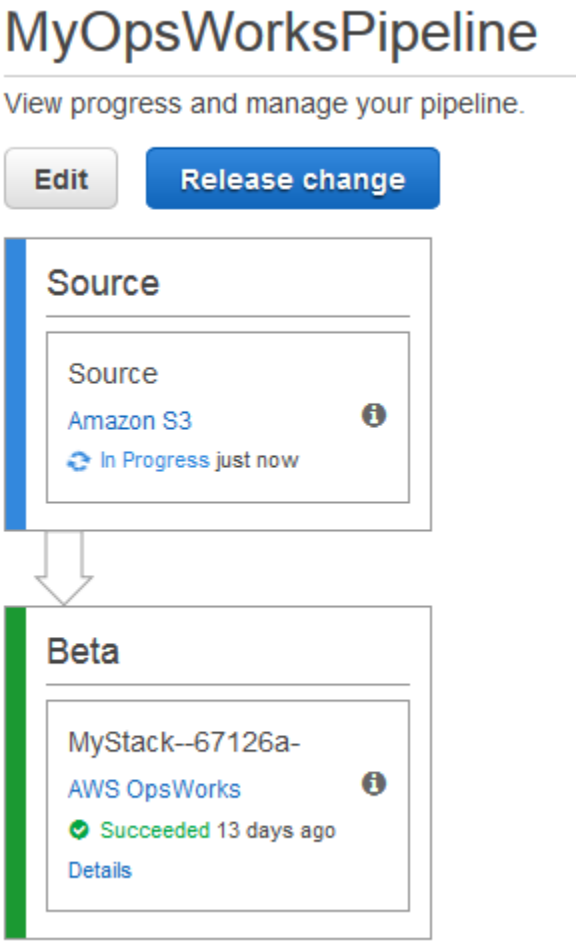
```
<body>
    <div class="container">
        <div class="hero-unit">
            <h1>Simple PHP App</h1>
            <h2>Congratulations!</h2>
            <p>You've just deployed your first app to AWS OpsWorks with AWS CodePipeline,</p>
            <p>on the host &ldquo;<?php echo gethostname(); ?>&rdquo; </p>
            <p>in your own dedicated environment in the AWS Cloud.</p>
            <p>This host is running PHP version <?php echo phpversion(); ?>.</p>
        </div>
    </div>

    <script src="//ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.min.js"></script>
    <script src="assets/js/bootstrap.min.js"></script>
</body>
```

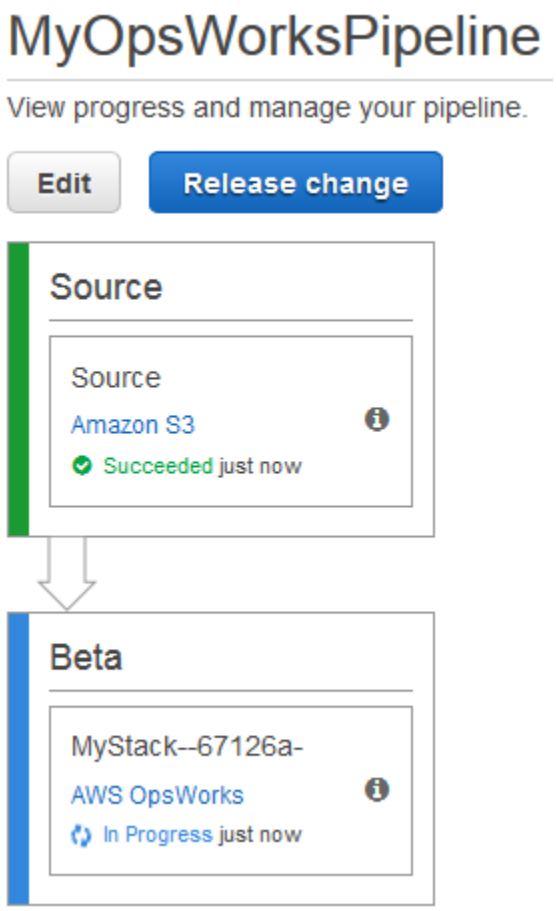
7. Salve e feche o arquivo `index.php`.
8. Compacte a pasta `opsworks-demo-php-simple-app-version1` e salve o arquivo ZIP em um local conveniente. Não altere o nome do arquivo ZIP.
9. Carregue o novo arquivo ZIP em seu bucket no Amazon S3. Nesta demonstração, o nome do bucket é `my-appbucket`.
10. Abra o CodePipeline console do e o pipeline do AWS OpsWorks Stacks (MyOpsWorksPipeline). Selecione Release Change.

(Você pode esperar CodePipeline para detectar a alteração no código da versão atualizada do aplicativo em seu bucket do Amazon S3. Para economizar seu tempo, este passo a passo instrui você a simplesmente escolher Release Change.)

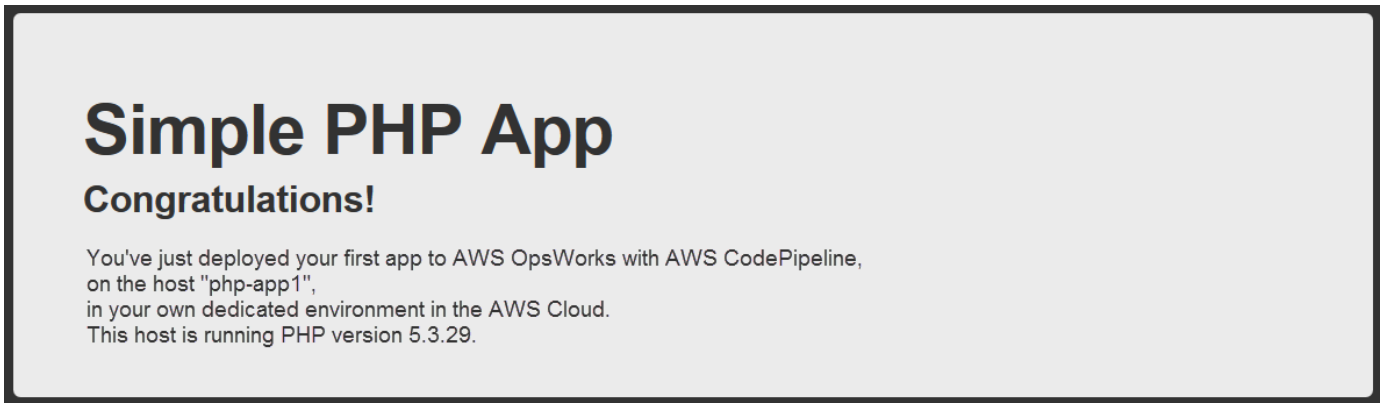
11. Observe como CodePipeline é executado nos estágios da tubulação. Primeiro, CodePipeline detecta as alterações no artefato de origem.



CodePipeline envia o código atualizado à pilha no AWS OpsWorks Stacks.



12. Quando ambos os estágios do pipeline forem concluídos com êxito, abra a pilha do AWS OpsWorks Stacks (MyStack).
13. Na página de MyStackpropriedades, escolha Instâncias.
14. Na coluna Public IP (IP público), escolha o endereço IP público da instância para exibir o texto do aplicativo atualizado.



Etapa 7 (opcional): Limpar os recursos

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. Recomendamos que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para evitar cobranças indesejadas na conta da AWS, você pode excluir os AWS recursos da usados nesta demonstração. Esses AWS recursos da incluem a pilha do AWS OpsWorks Stacks, o perfil do IAM, o perfil de instância e o pipeline que você criou em. CodePipeline No entanto, talvez seja necessário continuar usando os AWS recursos da para aprender mais sobre o AWS OpsWorks Stacks e. CodePipeline Se você quiser manter esses recursos, esta demonstração está concluída.

Para excluir o aplicativo da pilha

Como você não criou nem aplicou o aplicativo como parte do seu AWS CloudFormation modelo, exclua o aplicativo de teste de PHP antes de excluir a pilha. AWS CloudFormation

1. No console do AWS OpsWorks , no painel de navegação do serviço, escolha Apps.
2. Na página Aplicativos, selecione PHPTestAplicativo e, em Ações, escolha excluir. Quando for solicitada sua confirmação, escolha Excluir. AWS OpsWorks O Stacks excluirá o aplicativo.

Para excluir a pilha

Como você criou a pilha executando um AWS CloudFormation modelo, você pode excluir a pilha, incluindo a camada, a instância, o perfil da instância e o grupo de segurança que o modelo criou, no AWS CloudFormation console.

1. Abra o AWS CloudFormation console do.
2. No painel do AWS CloudFormation console, selecione a pilha que você criou (MyStack). No menu Actions (Ações), escolha Delete Stack (Excluir pilha). Quando for solicitada sua confirmação, escolha Yes, Delete (Sim, excluir).
3. Aguarde até DELETE_COMPLETE aparecer na coluna Status da pilha.

Para excluir o pipeline

1. Abra o CodePipeline console do.
2. No CodePipeline painel, escolha o pipeline que você criou para esta demonstração.
3. Na página do pipeline, selecione Edit.
4. Na página Editar, selecione Excluir. Quando for solicitada sua confirmação, escolha Delete.

Usando a CLI do AWS OpsWorks Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

A interface de linha de comando (CLI) do AWS OpsWorks Stacks fornece a mesma funcionalidade do console e pode ser usada para uma variedade de tarefas. O AWS OpsWorks Stacks CLI faz parte do. AWS CLI Para obter mais informações, incluindo como instalar e configurar o AWS CLI, acesse O que é a interface de linha de comando da AWS? . Para obter uma descrição completa de cada comando, acesse a referência do AWS OpsWorks Stacks.

> **Note**
>
> Se você estiver usando uma estação de trabalho baseada em Windows, também poderá executar AWS as Ferramentas do Windows PowerShell para realizar operações de AWS OpsWorks pilhas na linha de comando. Para obter mais informações, consulte AWS Tools for Windows PowerShell.

AWS OpsWorks Os comandos Stacks têm o seguinte formato geral:

```
aws opsworks --region us-west-1 opsworks command-name [--argument1 value] [...]
```

Se um valor de argumento for um objeto JSON, você deve escapar os caracteres `"` ou o comando pode retornar uma mensagem de erro informando que o JSON não é válido. Por exemplo, se o objeto JSON é `"{"somekey":"somevalue"}"`, você deve formatá-lo como `"{\"somekey\": \"somevalue\"}"`. Uma abordagem alternativa é colocar o objeto JSON em um arquivo e usar `file://` para incluí-lo na linha de comando. O exemplo a seguir cria um aplicativo usando um objeto fonte do aplicativo no appsource.json.

```
aws opsworks --region us-west-1 create-app --stack-id 8c428b08-a1a1-46ce-a5f8-
feddc43771b8 --name SimpleJSP --type java --app-source file://appsource.json
```

A maioria dos comandos retorna um ou mais valores, empacotados como um objeto JSON. As seções a seguir contêm alguns exemplos. Para obter uma descrição detalhada dos valores de retorno de cada comando, acesse a referência do AWS OpsWorks Stacks.

> **Note**
>
> AWS CLI os comandos devem especificar uma região, conforme mostrado nos exemplos. Os valores válidos para o parâmetro -region são mostrados na tabela a seguir. Para simplificar as cadeias de comando do AWS OpsWorks Stacks, configure a CLI para especificar sua região padrão, para que você possa omitir o parâmetro. `--region` Se você normalmente trabalha em vários endpoints regionais, não configure o AWS CLI para usar um endpoint regional padrão. O endpoint da região do Canadá (Central) está disponível somente na API AWS CLI ; ele não está disponível para pilhas que você cria no. AWS Management Console Para obter mais informações, consulte Configuração da região da AWS.

Nome da região	Código de comando
Região Leste dos EUA (Ohio)	us-east-2
Região Leste dos EUA (Norte da Virgínia)	us-east-1
Região Oeste dos EUA (Norte da Califórni a)	us-west-1
Região Oeste dos EUA (Oregon)	us-west-2
Região Canadá (Central)	ca-central-1

Nome da região	Código de comando
Região Europa (Irlanda)	eu-west-1
Região Europa (Londres)	eu-west-2
Região Europa (Paris)	eu-west-3
Região Europa (Frankfurt)	eu-central-1
Região Ásia-Pacífico (Tóquio)	ap-northeast-1
Região da Ásia-Pacífico (Seul)	ap-northeast-2
Região da Ásia-Pacífico (Mumbai)	ap-south-1
Região Ásia-Pacífico (Singapura)	ap-southeast-1
Região Ásia-Pacífico (Sydney)	ap-southeast-2
Região América do Sul (São Paulo)	sa-east-1

Para usar um comando da CLI, você precisa ter as permissões apropriadas. Para obter mais informações sobre os as permissões do AWS OpsWorks Stacks, consulte Gerenciamento de permissões de usuário. Para determinar as permissões necessárias para determinado comando, consulte a página de referência do comando na referência do AWS OpsWorks Stacks.

As seções a seguir descrevem como usar a CLI do AWS OpsWorks Stacks para realizar uma variedade de tarefas comuns.

Criar uma instância (create-instance)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando create-instance para criar uma instância em uma pilha especificada.

Tópicos

- Criar uma instância com um nome de host padrão
- Criar uma instância com um nome de host baseado em tema
- Criar uma instância com uma AMI personalizada

Criar uma instância com um nome de host padrão

```
C:\>aws opsworks --region us-west-1 create-instance --stack-id 935450cc-61e0-4b03-
a3e0-160ac817d2bb
    --layer-ids 5c8c272a-f2d5-42e3-8245-5bf3927cb65b --instance-type m1.large --os
 "Amazon Linux"
```

Os argumentos são os seguintes:

- `stack-id`— Você pode obter o ID da pilha na página de configurações da pilha no console (procure o OpsWorks ID) ou chamando describe-stacks.
- `layer-ids`— Você pode obter a camada na página IDs de detalhes da camada no console (procure por OpsWorks ID) ou chamando describe-layers. Neste exemplo, a instância pertence a apenas uma camada.
- `instance-type`: a especificação que define a memória, a CPU, a capacidade de armazenamento e o custo por hora da instância, `m1.large` neste exemplo.
- `os`: o sistema operacional da instância, Amazon Linux neste exemplo.

O comando retorna um objeto JSON que contém o ID da instância, da seguinte forma:

```
{
    "InstanceId": "5f9adeaa-c94c-42c6-aeef-28a5376002cd"
}
```

Este exemplo cria uma instância com um nome de host padrão, que é simplesmente um inteiro. A seção a seguir descreve como criar uma instância com um nome de host gerado a partir de um tema.

Criar uma instância com um nome de host baseado em tema

Também é possível criar uma instância com um nome de host baseado em tema. Você especifica o tema ao criar a pilha. Para obter mais informações, consulte Criar uma nova pilha .Para criar a instância, primeiro chame get-hostname-suggestionpara gerar um nome. Por exemplo:

```
C:\>aws opsworks get-hostname-suggestion --region us-west-1 --layer-id 5c8c272a-
f2d5-42e3-8245-5bf3927cb65b
```

Se você especificar o tema `Layer Dependent` padrão, `get-hostname-suggestion` simplesmente anexa um dígito ao nome abreviado da camada. Para obter mais informações, consulte Criar uma nova pilha.

O comando retorna o nome de host gerado.

```
{
    "Hostname": "php-app2",
    "LayerId": "5c8c272a-f2d5-42e3-8245-5bf3927cb65b"
}
```

Em seguida, você pode usar o argumento `hostname` para transmitir o nome gerado para `create-instance`, da seguinte forma:

```
c:\>aws --region us-west-1 opsworks create-instance --stack-id 935450cc-61e0-4b03-
a3e0-160ac817d2bb
   --layer-ids 5c8c272a-f2d5-42e3-8245-5bf3927cb65b --instance-type m1.large --os
 "Amazon Linux" --hostname "php-app2"
```

Criar uma instância com uma AMI personalizada

O comando create-instance a seguir cria uma instância com uma AMI personalizada, que precisa ser da região da pilha. Para obter mais informações sobre como criar uma AMI personalizada para AWS OpsWorks pilhas, consulteUsando o Custom AMIs.

```
C:\>aws opsworks create-instance --region us-west-1 --stack-id c5ef46ce-3ccd-472c-
a3de-9bec94c6028e
   --layer-ids 6ff8a2ac-c9cc-49cf-9c67-fc852539ade4 --instance-type c3.large --os
 Custom
   --ami-id ami-6c61f104
```

Os argumentos são os seguintes:

- `stack-id`— Você pode obter o ID da pilha na página de configurações da pilha no console (procure o OpsWorks ID) ou chamando describe-stacks.
- `layer-ids`— Você pode obter a camada na página IDs de detalhes da camada no console (procure por OpsWorks ID) ou chamando describe-layers. Neste exemplo, a instância pertence a apenas uma camada.
- `instance-type`: o valor define a memória, a CPU, a capacidade de armazenamento e o custo por hora da instância e deve ser compatível com a AMI (`c3.large` neste exemplo).
- `os`: o sistema operacional da instância, que deve ser definido como `Custom` para uma AMI personalizada.
- `ami-id`: o ID da AMI, que deve ser parecido com `ami-6c61f104`

> **Note**
>
> Quando você usa uma AMI personalizada, não há suporte para mapeamentos de dispositivos em bloco e os valores especificados para a opção `--block-device-mappings` são ignorados.

O comando retorna um objeto JSON que contém o ID da instância, da seguinte forma:

```
{
    "InstanceId": "5f9adeaa-c94c-42c6-aeef-28a5376002cd"
}
```

Implantar um aplicativo (create-deployment)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando create-deployment para implantar um aplicativo em uma pilha especificada.

Tópicos

- Implantar um aplicativo

Implantar um aplicativo

```
aws opsworks --region us-west-1 create-deployment --stack-id cfb7e082-ad1d-4599-8e81-
de1c39ab45bf
   --app-id 307be5c8-d55d-47b5-bd6e-7bd417c6c7eb --command "{\"Name\":\"deploy\"}"
```

Os argumentos são os seguintes:

- `stack-id`— Você pode obter o ID da pilha na página de configurações da pilha no console (procure o OpsWorks ID) ou ligando. `describe-stacks`
- `app-id`— Você pode obter o ID do aplicativo na página de detalhes do aplicativo (procure o OpsWorks ID) ou ligando para describe-apps.
- `command`: o argumento encontra um objeto JSON que define o nome do comando como `deploy`, o que implanta o aplicativo especificado na stack.

  Observe que todos os caracteres `"` no objeto JSON têm escape. Caso contrário, o comando pode retornar um erro de JSON inválido.

O comando retorna um objeto JSON que contém o ID da implantação, da seguinte forma:

```
{
    "DeploymentId": "5746c781-df7f-4c87-84a7-65a119880560"
}
```

> **Note**
>
> O exemplo anterior implanta em cada instância na stack. Para implantar em um subconjunto específico de instâncias, adicione um `instance-ids` argumento e liste a instância IDs.

Listar os aplicativos de uma pilha (describe-apps)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando describe-apps para listar os aplicativos de uma pilha ou obter detalhes sobre os aplicativos especificados.

```
aws opsworks --region us-west-1 describe-apps --stack-id 38ee91e2-abdc-4208-
a107-0b7168b3cc7a
```

O exemplo anterior retorna um objeto JSON que contém informações sobre cada aplicativo. Este exemplo tem apenas um aplicativo. Para uma descrição de cada parâmetro, consulte describe-apps.

```
{
  "Apps": [
    {
      "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",
      "AppSource": {
        "Url": "url",
        "Type": "archive"
```

```
      },
      "Name": "SimpleJSP",
      "EnableSsl": false,
      "SslConfiguration": {},
      "AppId": "da1decc1-0dff-43ea-ad7c-bb667cd87c8b",
      "Attributes": {
        "RailsEnv": null,
        "AutoBundleOnDeploy": "true",
        "DocumentRoot": "ROOT"
      },
      "Shortname": "simplejsp",
      "Type": "other",
      "CreatedAt": "2013-08-01T21:46:54+00:00"
    }
  ]
}
```

Listar os comandos de uma pilha (describe-commands)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando describe-commands para listar os comandos de uma pilha ou obter detalhes sobre os comandos especificados. O exemplo a seguir obtém informações sobre os comandos que foram executados em uma instância especificada.

```
aws opsworks --region us-west-1 describe-commands --instance-id
 8c2673b9-3fe5-420d-9cfa-78d875ee7687
```

O comando retorna um objeto JSON que contém detalhes sobre cada comando. O parâmetro `Type` identifica o nome do comando, deploy ou undeploy neste exemplo. Para obter uma descrição dos outros parâmetros, consulte describe-commands.

```
{
  "Commands": [
    {
      "Status": "successful",
      "CompletedAt": "2013-07-25T18:57:47+00:00",
      "InstanceId": "8c2673b9-3fe5-420d-9cfa-78d875ee7687",
      "DeploymentId": "6ed0df4c-9ef7-4812-8dac-d54a05be1029",
      "AcknowledgedAt": "2013-07-25T18:57:41+00:00",
      "LogUrl": "https://s3.amazonaws.com/prod_stage-log/logs/008c1a91-
ec59-4d51-971d-3adff54b00cc?AWSAccessKeyId=AIDACKCEVSQ6C2EXAMPLE
 &Expires=1375394373&Signature=HkXil6UuNfxTCC37EPQAa462E1E%3D&response-cache-
control=private&response-content-encoding=gzip&response-content- type=text%2Fplain",
      "Type": "undeploy",
      "CommandId": "008c1a91-ec59-4d51-971d-3adff54b00cc",
      "CreatedAt": "2013-07-25T18:57:34+00:00",
      "ExitCode": 0
    },
    {
      "Status": "successful",
      "CompletedAt": "2013-07-25T18:55:40+00:00",
      "InstanceId": "8c2673b9-3fe5-420d-9cfa-78d875ee7687",
      "DeploymentId": "19d3121e-d949-4ff2-9f9d-94eac087862a",
      "AcknowledgedAt": "2013-07-25T18:55:32+00:00",
      "LogUrl": "https://s3.amazonaws.com/prod_stage-log/
logs/899d3d64-0384-47b6-a586-33433aad117c?AWSAccessKeyId=AIDACKCEVSQ6C2EXAMPLE
 &Expires=1375394373&Signature=xMsJvtLuUqWmsr8s%2FAjVru0BtRs%3D&response-cache-
control=private&response-content-encoding=gzip&response-conten t-type=text%2Fplain",
      "Type": "deploy",
      "CommandId": "899d3d64-0384-47b6-a586-33433aad117c",
      "CreatedAt": "2013-07-25T18:55:29+00:00",
      "ExitCode": 0
    }
  ]
}
```


Listar as implantações de uma pilha (describe-deployments)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando describe-deployments para listar as implantações de uma pilha ou obter detalhes sobre as implantações especificadas.

```
aws opsworks --region us-west-1 describe-deployments --stack-id 38ee91e2-abdc-4208-
a107-0b7168b3cc7a
```

O comando anterior retorna um objeto JSON que contém detalhes sobre cada implantação da pilha. Para obter uma descrição de cada parâmetro, consulte describe-deployments.

```
{
  "Deployments": [
      {
          "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",
          "Status": "successful",
          "CompletedAt": "2013-07-25T18:57:49+00:00",
          "DeploymentId": "6ed0df4c-9ef7-4812-8dac-d54a05be1029",
          "Command": {
              "Args": {},
              "Name": "undeploy"
          },
          "CreatedAt": "2013-07-25T18:57:34+00:00",
          "Duration": 15,
          "InstanceIds": [
              "8c2673b9-3fe5-420d-9cfa-78d875ee7687",
              "9e588a25-35b2-4804-bd43-488f85ebe5b7"
          ]
      },
      {
          "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",
          "Status": "successful",
          "CompletedAt": "2013-07-25T18:56:41+00:00",
          "IamUserArn": "arn:aws:iam::444455556666:user/example-user",
          "DeploymentId": "19d3121e-d949-4ff2-9f9d-94eac087862a",
          "Command": {
              "Args": {},
              "Name": "deploy"
          },
          "InstanceIds": [
```

```
                "8c2673b9-3fe5-420d-9cfa-78d875ee7687",
                "9e588a25-35b2-4804-bd43-488f85ebe5b7"
            ],
            "Duration": 72,
            "CreatedAt": "2013-07-25T18:55:29+00:00"
        }
    ]
}
```

Listar os endereços IP elásticos de uma pilha () describe-elastic-ips

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando describe-elastic-ips para listar os endereços IP elásticos que foram registrados em uma pilha ou obter detalhes sobre os endereços IP elásticos especificados.

```
aws opsworks --region us-west-2 describe-elastic-ips --instance-id b62f3e04-e9eb-436c-
a91f-d9e9a396b7b0
```

O comando anterior retorna um objeto JSON que contém detalhes sobre cada endereço IP elástico (um neste exemplo) para uma instância especificada. Para uma descrição de cada parâmetro, consulte describe-elastic-ips.

```
{
  "ElasticIps": [
      {
          "Ip": "192.0.2.0",
          "Domain": "standard",
          "Region": "us-west-2"
      }
  ]
}
```

Listar as instâncias de uma pilha (describe-instances)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando describe-instances para listar as instâncias de uma pilha ou obter detalhes sobre as instâncias especificadas.

```
C:\>aws opsworks --region us-west-2 describe-instances --stack-id 38ee91e2-abdc-4208-
a107-0b7168b3cc7a
```

O comando anterior retorna um objeto JSON que contém detalhes sobre cada instância em uma pilha especificada. Para obter uma descrição de cada parâmetro, consulte describe-instances.

```
{
  "Instances": [
      {
          "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",
          "SshHostRsaKeyFingerprint":
 "f4:3b:8e:27:1b:73:98:80:5d:d7:33:e2:b8:c8:8f:de",
          "Status": "stopped",
          "AvailabilityZone": "us-west-2a",
          "SshHostDsaKeyFingerprint":
 "e8:9b:c7:02:18:2a:bd:ab:45:89:21:4e:af:0b:07:ac",
          "InstanceId": "8c2673b9-3fe5-420d-9cfa-78d875ee7687",
          "Os": "Amazon Linux",
          "Hostname": "db-master1",
          "SecurityGroupIds": [],
          "Architecture": "x86_64",
          "RootDeviceType": "instance-store",
          "LayerIds": [
              "41a20847-d594-4325-8447-171821916b73"
          ],
          "InstanceType": "c1.medium",
```

```
            "CreatedAt": "2013-07-25T18:11:27+00:00"
        },
        {
            "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",
            "SshHostRsaKeyFingerprint":
 "ae:3a:85:54:66:f3:ce:98:d9:83:39:1e:10:a9:38:12",
            "Status": "stopped",
            "AvailabilityZone": "us-west-2a",
            "SshHostDsaKeyFingerprint":
 "5b:b9:6f:5b:1c:ec:55:85:f3:45:f1:28:25:1f:de:e4",
            "InstanceId": "9e588a25-35b2-4804-bd43-488f85ebe5b7",
            "Os": "Amazon Linux",
            "Hostname": "tomcustom1",
            "SecurityGroupIds": [],
            "Architecture": "x86_64",
            "RootDeviceType": "instance-store",
            "LayerIds": [
                "e6cbcd29-d223-40fc-8243-2eb213377440"
            ],
            "InstanceType": "c1.medium",
            "CreatedAt": "2013-07-25T18:15:52+00:00"
        }
    ]
}
```

Listar as pilhas de uma conta (describe-stacks)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando describe-stacks para listar as pilhas de uma conta ou obter detalhes sobre as pilhas especificadas.

```
aws opsworks --region us-west-2 describe-stacks
```

O comando anterior retorna um objeto JSON que contém detalhes sobre cada pilha na conta (duas neste exemplo). Para obter uma descrição de cada parâmetro, consulte describe-stacks.

```
{
    "Stacks": [
        {
            "ServiceRoleArn": "arn:aws:iam::444455556666:role/aws-opsworks-service-
role",
            "StackId": "aeb7523e-7c8b-49d4-b866-03aae9d4fbcb",
            "DefaultRootDeviceType": "instance-store",
            "Name": "TomStack-sd",
            "ConfigurationManager": {
                "Version": "11.4",
                "Name": "Chef"
            },
            "UseCustomCookbooks": true,
            "CustomJson": "{\n  \"tomcat\": {\n    \"base_version\": 7,\n
 \"java_opts\": \"-Djava.awt.headless=true -Xmx256m\"\n  },\n  \
"datasources\": {\n    \"ROOT\": \"jdbc/mydb\"\n  }\n}",
            "Region": "us-west-2",
            "DefaultInstanceProfileArn": "arn:aws:iam::444455556666:instance-profile/
aws-opsworks-ec2-role",
            "CustomCookbooksSource": {
                "Url": "git://github.com/example-repo/tomcustom.git",
                "Type": "git"
            },
            "DefaultAvailabilityZone": "us-west-2a",
            "HostnameTheme": "Layer_Dependent",
            "Attributes": {
                "Color": "rgb(45, 114, 184)"
            },
            "DefaultOs": "Amazon Linux",
            "CreatedAt": "2013-08-01T22:53:42+00:00"
        },
        {
            "ServiceRoleArn": "arn:aws:iam::444455556666:role/aws-opsworks-service-
role",
            "StackId": "40738975-da59-4c5b-9789-3e422f2cf099",
            "DefaultRootDeviceType": "instance-store",
            "Name": "MyStack",
            "ConfigurationManager": {
                "Version": "11.4",
                "Name": "Chef"
```

```
            },
            "UseCustomCookbooks": false,
            "Region": "us-west-2",
            "DefaultInstanceProfileArn": "arn:aws:iam::444455556666:instance-profile/
aws-opsworks-ec2-role",
            "CustomCookbooksSource": {},
            "DefaultAvailabilityZone": "us-west-2a",
            "HostnameTheme": "Layer_Dependent",
            "Attributes": {
                "Color": "rgb(45, 114, 184)"
            },
            "DefaultOs": "Amazon Linux",
            "CreatedAt": "2013-10-25T19:24:30+00:00"
        }
    ]
}
```

Listar as camadas de uma pilha (describe-layers)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando describe-layers para listar as camadas de uma pilha ou obter detalhes sobre as camadas especificadas.

```
aws opsworks --region us-west-2 describe-layers --stack-id 38ee91e2-abdc-4208-
a107-0b7168b3cc7a
```

O comando anterior retorna um objeto JSON que contém detalhes sobre cada camada em uma pilha especificada. Neste exemplo, uma camada MySQL e uma camada personalizada. Para obter uma descrição de cada parâmetro, consulte describe-layers.

```
{
  "Layers": [
```

```
    {
        "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",
        "Type": "db-master",
        "DefaultSecurityGroupNames": [
            "AWS-OpsWorks-DB-Master-Server"
        ],
        "Name": "MySQL",
        "Packages": [],
        "DefaultRecipes": {
            "Undeploy": [],
            "Setup": [
                "opsworks_initial_setup",
                "ssh_host_keys",
                "ssh_users",
                "mysql::client",
                "dependencies",
                "ebs",
                "opsworks_ganglia::client",
                "mysql::server",
                "dependencies",
                "deploy::mysql"
            ],
            "Configure": [
                "opsworks_ganglia::configure-client",
                "ssh_users",
                "agent_version",
                "deploy::mysql"
            ],
            "Shutdown": [
                "opsworks_shutdown::default",
                "mysql::stop"
            ],
            "Deploy": [
                "deploy::default",
                "deploy::mysql"
            ]
        },
        "CustomRecipes": {
            "Undeploy": [],
            "Setup": [],
            "Configure": [],
            "Shutdown": [],
            "Deploy": []
        },
```

```
            "EnableAutoHealing": false,
            "LayerId": "41a20847-d594-4325-8447-171821916b73",
            "Attributes": {
                "MysqlRootPasswordUbiquitous": "true",
                "RubygemsVersion": null,
                "RailsStack": null,
                "HaproxyHealthCheckMethod": null,
                "RubyVersion": null,
                "BundlerVersion": null,
                "HaproxyStatsPassword": null,
                "PassengerVersion": null,
                "MemcachedMemory": null,
                "EnableHaproxyStats": null,
                "ManageBundler": null,
                "NodejsVersion": null,
                "HaproxyHealthCheckUrl": null,
                "MysqlRootPassword": "*****FILTERED*****",
                "GangliaPassword": null,
                "GangliaUser": null,
                "HaproxyStatsUrl": null,
                "GangliaUrl": null,
                "HaproxyStatsUser": null
            },
            "Shortname": "db-master",
            "AutoAssignElasticIps": false,
            "CustomSecurityGroupIds": [],
            "CreatedAt": "2013-07-25T18:11:19+00:00",
            "VolumeConfigurations": [
                {
                    "MountPoint": "/vol/mysql",
                    "Size": 10,
                    "NumberOfDisks": 1
                }
            ]
        },
        {
            "StackId": "38ee91e2-abdc-4208-a107-0b7168b3cc7a",
            "Type": "custom",
            "DefaultSecurityGroupNames": [
                "AWS-OpsWorks-Custom-Server"
            ],
            "Name": "TomCustom",
            "Packages": [],
            "DefaultRecipes": {
```

```
            "Undeploy": [],
            "Setup": [
                "opsworks_initial_setup",
                "ssh_host_keys",
                "ssh_users",
                "mysql::client",
                "dependencies",
                "ebs",
                "opsworks_ganglia::client"
            ],
            "Configure": [
                "opsworks_ganglia::configure-client",
                "ssh_users",
                "agent_version"
            ],
            "Shutdown": [
                "opsworks_shutdown::default"
            ],
            "Deploy": [
                "deploy::default"
            ]
        },
        "CustomRecipes": {
            "Undeploy": [],
            "Setup": [
                "tomcat::setup"
            ],
            "Configure": [
                "tomcat::configure"
            ],
            "Shutdown": [],
            "Deploy": [
                "tomcat::deploy"
            ]
        },
        "EnableAutoHealing": true,
        "LayerId": "e6cbcd29-d223-40fc-8243-2eb213377440",
        "Attributes": {
            "MysqlRootPasswordUbiquitous": null,
            "RubygemsVersion": null,
            "RailsStack": null,
            "HaproxyHealthCheckMethod": null,
            "RubyVersion": null,
            "BundlerVersion": null,
```

```
                "HaproxyStatsPassword": null,
                "PassengerVersion": null,
                "MemcachedMemory": null,
                "EnableHaproxyStats": null,
                "ManageBundler": null,
                "NodejsVersion": null,
                "HaproxyHealthCheckUrl": null,
                "MysqlRootPassword": null,
                "GangliaPassword": null,
                "GangliaUser": null,
                "HaproxyStatsUrl": null,
                "GangliaUrl": null,
                "HaproxyStatsUser": null
            },
            "Shortname": "tomcustom",
            "AutoAssignElasticIps": false,
            "CustomSecurityGroupIds": [],
            "CreatedAt": "2013-07-25T18:12:53+00:00",
            "VolumeConfigurations": []
        }
    ]
}
```

Executar uma receita (create-deployment)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando create-deployment para executar comandos da pilha e comandos de implantação. O exemplo a seguir executa um comando da pilha que, por sua vez, executa uma receita personalizada em uma stack especificada.

```
aws opsworks --region us-west-1 create-deployment --stack-id 935450cc-61e0-4b03-
a3e0-160ac817d2bb
```

```
    --command "{\"Name\":\"execute_recipes\", \"Args\":{\"recipes\":[\"phpapp::appsetup
\"]}}"
```

O argumento `command` usa um objeto JSON formatado da seguinte forma:

- `Name`: especifica o nome do comando. O comando `execute_recipes` utilizado neste exemplo executa uma receita especificada nas instâncias da pilha.
- `Args`: especifica uma lista de argumentos e os respectivos valores. Este exemplo tem um argumento (`recipes`) definido como a receita a ser executada (`phpapp::appsetup`).

Observe que todos os caracteres `"` no objeto JSON têm escape. Caso contrário, o comando pode retornar um erro de JSON inválido.

O comando retorna o ID de uma implantação, que você pode usar para identificar o comando para outros comandos da CLI, como `describe-commands`.

```
{
    "DeploymentId": "5cbaa7b9-4e09-4e53-aa1b-314fbd106038"
}
```

Instalar dependências (create-deployment)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando create-deployment para executar comandos da pilha e comandos de implantação. O exemplo a seguir executa o comando da pilha `update_dependencies` para atualizar as dependências nas instâncias de uma pilha.

```
aws opsworks --region us-west-1 create-deployment --stack-id 935450cc-61e0-4b03-
a3e0-160ac817d2bb
--command "{\"Name\":\"install_dependencies\"}"
```

O argumento `command` encontra um objeto JSON com um parâmetro `Name`, cujo valor especifica o nome do comando (`install_dependencies` neste exemplo). Observe que todos os caracteres `"` no objeto JSON têm escape. Caso contrário, o comando pode retornar um erro de JSON inválido.

O comando retorna o ID de uma implantação, que você pode usar para identificar o comando para outros comandos da CLI, como `describe-commands`.

```
{
    "DeploymentId": "aef5b255-8604-4928-81b3-9b0187f962ff"
}
```

Atualizar a configuração da pilha (update-stack)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Use o comando update-stack para atualizar a configuração de uma pilha especificada. O exemplo a seguir atualiza uma pilha para adicionar um JSON personalizado aos atributos de configuração da pilha.

```
aws opsworks --region us-west-1 update-stack --stack-id 935450cc-61e0-4b03-
a3e0-160ac817d2bb
   --custom-json "{\"somekey\":\"somevalue\"}" --service-role-arn
 arn:aws:iam::444455556666:role/aws-opsworks-service-role
```

Observe que todos os caracteres `"` no objeto JSON têm escape. Caso contrário, o comando pode retornar um erro de JSON inválido.

> **Note**
>
> O exemplo também especifica uma função de serviço para a pilha. É necessário definir `service-role-arn` como um ARN de função de serviço válido ou a ação falhará. Não

há valor padrão. É possível especificar o ARN de função de serviço atual da pilha, se você preferir, mas é necessário fazê-lo explicitamente.

O comando `update-stack` não retorna um valor.

Guia de depuração e solução de problemas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Se você precisar depurar uma receita ou solucionar um problema de serviço, a melhor abordagem costuma ser acompanhar as seguintes etapas, em ordem:

1. Verifique em Depuração e solução de problemas comuns seu problema específico.
2. Pesquise no fórum do AWS OpsWorks Stacks para ver se o problema foi discutido lá.

   O fórum inclui muitos usuários experientes e é monitorado pela equipe do AWS OpsWorks Stacks.
3. Para problemas com receitas, consulte Depurar receitas.
4. Entre em contato com o suporte do AWS OpsWorks Stacks ou publique seu problema no AWS OpsWorks Stacks Forum.

A seção a seguir fornece orientação para depurar receitas. A última seção descreve problemas comuns de depuração e solução de problemas e suas resoluções.

> **Note**
>
> Cada execução do Chef produz um log, que fornece uma descrição detalhada da execução e é um recurso de solução de problemas valioso. Para especificar a quantidade de detalhes no log, adicione uma instrução `Chef::Log.level` para uma receita personalizada que

especifica o nível de log desejado. O valor padrão é `:info`. O exemplo a seguir mostra como definir o nível de log do Chef para `:debug`, que fornece a descrição mais detalhada da execução.

```
Chef::Log.level = :debug
```

Para obter mais informações sobre a visualização e a interpretação de logs do Chef, consulte Logs do Chef.

Tópicos

- Depurar receitas
- Depuração e solução de problemas comuns

Depurar receitas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Quando um evento de ciclo de vida ocorre ou ao executar o comando de pilha Execute Recipes (Executar receitas), o AWS OpsWorks Stacks emite um comando para o agente iniciar uma execução de Chef Solo nas instâncias especificadas para executar as receitas adequadas, incluindo as receitas personalizadas. Esta seção descreve algumas maneiras para você depurar receitas com falha.

Tópicos

- Efetuar login em uma Instância com falha
- Logs do Chef
- Usando a CLI do AWS OpsWorks Stacks Agent

Efetuar login em uma Instância com falha

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Se houver falha de receita, a instância terá o estado `setup_failed` em vez de online. Mesmo que a instância não esteja on-line no que diz respeito ao AWS OpsWorks EC2 Stacks, ela está em execução e geralmente é útil fazer login para solucionar o problema. Por exemplo, você pode verificar se um aplicativo ou um livro de receitas personalizado está instalado corretamente. O suporte integrado do AWS OpsWorks Stacks para login SSH e RDP está disponível somente para instâncias no estado online. No entanto, se você tiver atribuído um par de chaves SSH à instância, ainda é possível fazer login da seguinte maneira:

- Instâncias Linux: use a chave privada do par de chaves SSH para fazer login com um cliente SSH de terceiros, como OpenSSH ou PuTTY.

  Você pode usar um par de EC2 chaves ou seu par de chaves SSH pessoal para essa finalidade.
- Instâncias do Windows — use a EC2 chave privada do par de chaves para recuperar a senha de administrador da instância.

  Use essa senha para se conectar com seu cliente RDP preferencial. Para obter mais informações, consulte Login como administrador. Você não pode usar um par de chaves SSH pessoais para recuperar uma senha de Administrador.

Logs do Chef

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Os registros do Chef são um dos seus principais recursos de solução de problemas, especialmente para receitas de depuração. AWS OpsWorks O Stacks captura o registro do Chef para cada comando e retém os registros dos 30 comandos mais recentes de uma instância. Como a execução está no modo de depuração, o log contém uma descrição detalhada da execução do Chef, incluindo o texto que é enviado para `stdout` e `stderror`. Se uma receita falhar, o log do Chef inclui o rastreamento de pilha.

AWS OpsWorks O Stacks oferece várias maneiras de visualizar os registros do Chef. Com as informações do log, você pode usá-las para depurar receitas com falha.

> **Note**
>
> Também é possível visualizar a parte final do log ao usar SSH para se conectar à instância e executar o comando `show_log` do agente de CLI. Para obter mais informações, consulte Exibição de logs do Chef.

Tópicos

- Exibir um Log do Chef com o Console
- Exibir um Log do Chef com CLI ou API
- Exibir um Log do Chef em uma instância
- Interpretar o log do Chef
- Erros comuns do log do Chef

Exibir um Log do Chef com o Console

A maneira mais simples de visualizar um log do Chef é acessar a página de detalhes da instância. A seção Logs inclui uma entrada para cada evento e o comando Executar receitas. As informações a seguir mostram a seção Logs de uma instância com os comandos configure e setup, que correspondem aos eventos de ciclo de vida Configurar e Configuração.

Clique em show na coluna Log do comando apropriado para visualizar o log do Chef correspondente. Se ocorrer um erro, o AWS OpsWorks Stacks abrirá automaticamente o registro do erro, que geralmente está no final do arquivo.

Exibir um Log do Chef com CLI ou API

Você pode usar o comando AWS OpsWorks Stacks `describe-commands`CLI ou `DescribeCommands`a ação da API para visualizar os registros, que são armazenados em um bucket do Amazon S3. As informações a seguir mostram como usar a CLI para visualizar qualquer arquivo do conjunto atual de arquivos de logs de uma instância especificada. O procedimento para usar `DescribeCommands` é essencialmente semelhante.

Para usar as AWS OpsWorks pilhas para visualizar os registros do Chef de uma instância

1. Abra a página de detalhes da instância e copie seu valor de OpsWorksID.
2. Use o valor da ID para executar o comando de CLI `describe-commands` da seguinte forma:

   ```
   aws opsworks describe-commands --instance-id 67bf0da2-29ed-4217-990c-d895d51812b9
   ```

   O comando retorna um objeto JSON com um objeto incorporado para cada comando que o AWS OpsWorks Stacks executou na instância, com o mais recente primeiro. O parâmetro `Type` contém o tipo de comando de cada objeto integrado, além dos comandos `configure` e `setup` neste exemplo.

   ```
   {
       "Commands": [
           {
               "Status": "successful",
               "CompletedAt": "2013-10-25T19:38:36+00:00",
               "InstanceId": "67bf0da2-29ed-4217-990c-d895d51812b9",
               "AcknowledgedAt": "2013-10-25T19:38:24+00:00",
   ```

```
            "LogUrl": "https://s3.amazonaws.com/prod_stage-log/logs/
b6c402df-5c23-45b2-a707-ad20b9c5ae40?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE
&Expires=1382731518&Signature=YkqS5IZN2P4wixjHwoC3aCMbn5s%3D&response-cache-
control=private&response-content-encoding=gzip&response-content-
type=text%2Fplain",
            "Type": "configure",
            "CommandId": "b6c402df-5c23-45b2-a707-ad20b9c5ae40",
            "CreatedAt": "2013-10-25T19:38:11+00:00",
            "ExitCode": 0
        },
        {
            "Status": "successful",
            "CompletedAt": "2013-10-25T19:31:08+00:00",
            "InstanceId": "67bf0da2-29ed-4217-990c-d895d51812b9",
            "AcknowledgedAt": "2013-10-25T19:29:01+00:00",
            "LogUrl": "https://s3.amazonaws.com/prod_stage-log/logs/2a90e862-
f974-42a6-9342-9a4f03468358?AWSAccessKeyId=AKIAIOSFODNN7EXAMPLE
&Expires=1382731518&Signature=cxKYHO8mCCd4MvOyFb6ywebeQtA%3D&response-cache-
control=private&response-content-encoding=gzip&response-content-
type=text%2Fplain",
            "Type": "setup",
            "CommandId": "2a90e862-f974-42a6-9342-9a4f03468358",
            "CreatedAt": "2013-10-25T19:26:01+00:00",
            "ExitCode": 0
        }
    ]
}
```

3. Copie o valor de `LogUrl` para o navegador a fim de exibir o log.

Se a instância tiver mais de alguns comandos, você pode adicionar parâmetros para `describe-commands` a fim de filtrar quais comandos estão incluídos no objeto da resposta. Para obter mais informações, consulte describe-commands.

Exibir um Log do Chef em uma instância

> **Note**
>
> Os tópicos desta seção se aplicam ao Chef 12. Para obter mais informações sobre a localização dos logs do Chef para Chef 11.10 e versões anteriores, consulte Solução de problemas do Chef 11.10 e versões anteriores para Linux.

Instâncias do Linux

AWS OpsWorks O Stacks armazena os registros do Chef de cada instância em seu `/var/chef/runs` diretório. (para instâncias do Linux, este diretório também inclui os recipientes de dados associados, armazenados como arquivos formatados para JSON.) Você precisa de privilégios de sudo para acessar esse diretório. O log de cada execução está em um arquivo chamado `chef.log` no subdiretório da execução individual.

AWS OpsWorks O Stacks armazena seus registros internos na `/var/log/aws/opsworks` pasta da instância. Em geral, as informações não são muito úteis para fins de solução de problemas. No entanto, esses registros são úteis para o suporte do AWS OpsWorks Stacks, e você pode ser solicitado a fornecê-los se encontrar algum problema com o serviço. Às vezes, os logs do Linux também podem fornecer dados de solução de problemas úteis.

Instâncias do Windows

Logs de agente

Nas instâncias do Windows, os OpsWorks registros são armazenados em um `ProgramData` caminho como o seguinte. O número inclui um carimbo de data e hora.

```
C:\ProgramData\OpsWorksAgent\var\logs\number
```

> **Note**
>
> Por padrão, `ProgramData` é uma pasta oculta. Para exibi-la, vá até Folder Options. Em View, selecione a opção de mostrar arquivos ocultos.

O exemplo a seguir mostra os logs de agente em uma instância do Windows.

```
Mode                LastWriteTime     Length Name
----                -------------     ------ ----
-a---         5/24/2015  11:59 PM     127277 command.20150524.txt
-a---         5/25/2015  11:59 PM     546772 command.20150525.txt
-a---         5/26/2015  11:59 PM     551514 command.20150526.txt
-a---         5/27/2015   9:43 PM     495181 command.20150527.txt
-a---         5/24/2015  11:59 PM      24353 keepalive.20150524.txt
-a---         5/25/2015  11:59 PM     106232 keepalive.20150525.txt
-a---         5/26/2015  11:59 PM     106208 keepalive.20150526.txt
```

```
-a---         5/27/2015   8:54 PM      92593 keepalive.20150527.txt
-a---         5/24/2015   7:19 PM       3891 service.20150524.txt
-a---         5/27/2015   8:54 PM       1493 service.20150527.txt
-a---         5/24/2015  11:59 PM     112549 wire.20150524.txt
-a---         5/25/2015  11:59 PM     501501 wire.20150525.txt
-a---         5/26/2015  11:59 PM     499640 wire.20150526.txt
-a---         5/27/2015   8:54 PM     436870 wire.20150527.txt
```

Logs do Chef

Em instâncias do Windows, os logs do Chef são armazenados em um caminho `ProgramData`, como o seguinte. O número inclui um carimbo de data e hora.

```
C:\ProgramData\OpsWorksAgent\var\commands\number
```

> **Note**
>
> Este diretório contém apenas a saída da primeira execução (de OpsWorks propriedade) do Chef.

O exemplo a seguir mostra registros OpsWorks próprios do Chef em uma instância do Windows.

```
 Mode                LastWriteTime            Name
 ----                -------------            ----
 d----         5/24/2015   7:23 PM
 configure-7ecb5f47-7626-439b-877f-5e7cb40ab8be
 d----         5/26/2015   8:30 PM            configure-8e74223b-d15d-4372-aeea-
a87b428ffc2b
 d----         5/24/2015   6:34 PM            configure-
c3980a1c-3d08-46eb-9bae-63514cee194b
 d----         5/26/2015   8:32 PM            grant_remote_access-70dbf834-1bfa-4fce-
b195-e50e85402f4c
 d----         5/26/2015  10:30 PM            revoke_remote_access-1111fce9-843a-4b27-
b93f-ecc7c5e9e05b
 d----         5/24/2015   7:21 PM            setup-754ec063-8b60-4cd4-
b6d7-0e89d7b7aa78
 d----         5/26/2015   8:27 PM            setup-af5bed36-5afd-4115-
af35-5766f88bc039
 d----         5/24/2015   6:32 PM            setup-d8abeffa-24d4-414b-
bfb1-4ad07319f358
```

```
 d----         5/24/2015   7:13 PM            shutdown-c7130435-9b5c-4a95-
be17-6b988fc6cf9a
 d----         5/26/2015   8:25 PM
 sync_remote_users-64c79bdc-1f6f-4517-865b-23d2def4180c
 d----         5/26/2015   8:48 PM
 update_custom_cookbooks-2cc59a94-315b-414d-85eb-2bdea6d76c6a
```

Logs do Chef do usuário

Os logs de execuções do Chef podem ser encontradas em arquivos chamados `logfile.txt`, em uma pasta como mesmo nome do comando numerado do Chef, como mostrado no diagrama a seguir.

C:/chef └── `runs` └── `command-12345` ├── `attribs.json` ├── `client.rb` └── `logfile.txt`

Interpretar o log do Chef

Em grande parte, o início do log contém registros internos do Chef.

```
# Logfile created on Thu Oct 17 17:25:12 +0000 2013 by logger.rb/1.2.6
[2013-10-17T17:25:12+00:00] INFO: *** Chef 11.4.4 ***
[2013-10-17T17:25:13+00:00] DEBUG: Building node object for php-app1.localdomain
[2013-10-17T17:25:13+00:00] DEBUG: Extracting run list from JSON attributes provided on
 command line
[2013-10-17T17:25:13+00:00] INFO: Setting the run_list to
 ["opsworks_custom_cookbooks::load", "opsworks_custom_cookbooks::execute"] from JSON
[2013-10-17T17:25:13+00:00] DEBUG: Applying attributes from json file
[2013-10-17T17:25:13+00:00] DEBUG: Platform is amazon version 2013.03
[2013-10-17T17:25:13+00:00] INFO: Run List is [recipe[opsworks_custom_cookbooks::load],
 recipe[opsworks_custom_cookbooks::execute]]
[2013-10-17T17:25:13+00:00] INFO: Run List expands to [opsworks_custom_cookbooks::load,
 opsworks_custom_cookbooks::execute]
[2013-10-17T17:25:13+00:00] INFO: Starting Chef Run for php-app1.localdomain
[2013-10-17T17:25:13+00:00] INFO: Running start handlers
[2013-10-17T17:25:13+00:00] INFO: Start handlers complete.
[2013-10-17T17:25:13+00:00] DEBUG: No chefignore file found at /opt/aws/opsworks/
releases/20131015111601_209/cookbooks/chefignore no files will be ignored
[2013-10-17T17:25:13+00:00] DEBUG: Cookbooks to compile: ["gem_support", "packages",
 "opsworks_bundler", "opsworks_rubygems", "ruby", "ruby_enterprise", "dependencies",
 "opsworks_commons", "scm_helper", :opsworks_custom_cookbooks]
[2013-10-17T17:25:13+00:00] DEBUG: Loading cookbook gem_support's library file: /
opt/aws/opsworks/releases/20131015111601_209/cookbooks/gem_support/libraries/
current_gem_version.rb
```

```
[2013-10-17T17:25:13+00:00] DEBUG: Loading cookbook packages's library file: /opt/aws/
opsworks/releases/20131015111601_209/cookbooks/packages/libraries/packages.rb
[2013-10-17T17:25:13+00:00] DEBUG: Loading cookbook dependencies's library file: /
opt/aws/opsworks/releases/20131015111601_209/cookbooks/dependencies/libraries/
current_gem_version.rb
[2013-10-17T17:25:13+00:00] DEBUG: Loading cookbook opsworks_commons's library file: /
opt/aws/opsworks/releases/20131015111601_209/cookbooks/opsworks_commons/libraries/
activesupport_blank.rb
[2013-10-17T17:25:13+00:00] DEBUG: Loading cookbook opsworks_commons's library file: /
opt/aws/opsworks/releases/20131015111601_209/cookbooks/opsworks_commons/libraries/
monkey_patch_chefgem_resource.rb
...
```

Esta parte do arquivo é útil principalmente para especialistas do Chef. Observe que a lista de execução inclui apenas duas receitas, mesmo quando a maioria dos comandos envolvem mais recursos. Essas duas receitas processam a tarefa de carregar e executar todas as outras receitas internas e personalizadas.

A parte mais interessante do arquivo é geralmente no final. Se uma execução termina com sucesso, você deve visualizar algo como o seguinte:

```
...
[Tue, 11 Jun 2013 16:00:50 +0000] DEBUG: STDERR:
[Tue, 11 Jun 2013 16:00:50 +0000] DEBUG: ---- End output of /sbin/service mysqld
 restart ----
[Tue, 11 Jun 2013 16:00:50 +0000] DEBUG: Ran /sbin/service mysqld restart returned 0
[Tue, 11 Jun 2013 16:00:50 +0000] INFO: service[mysql]: restarted successfully
[Tue, 11 Jun 2013 16:00:50 +0000] INFO: Chef Run complete in 84.07096 seconds
[Tue, 11 Jun 2013 16:00:50 +0000] INFO: cleaning the checksum cache
[Tue, 11 Jun 2013 16:00:50 +0000] DEBUG: removing unused checksum cache file /var/chef/
cache/checksums/chef-file--tmp-chef-rendered-template20130611-4899-8wef7e-0
[Tue, 11 Jun 2013 16:00:50 +0000] DEBUG: removing unused checksum cache file /var/chef/
cache/checksums/chef-file--tmp-chef-rendered-template20130611-4899-1xpwyb6-0
[Tue, 11 Jun 2013 16:00:50 +0000] DEBUG: removing unused checksum cache file /var/chef/
cache/checksums/chef-file--etc-monit-conf
[Tue, 11 Jun 2013 16:00:50 +0000] INFO: Running report handlers
[Tue, 11 Jun 2013 16:00:50 +0000] INFO: Report handlers complete
[Tue, 11 Jun 2013 16:00:50 +0000] DEBUG: Exiting
```

> **Note**
>
> Você pode usar a CLI do agente para exibir a parte final do log durante ou após a execução. Para obter mais informações, consulte Exibição de logs do Chef.

Se uma receita falhar, você deve procurar uma saída de nível de ERRO, que conterá uma exceção seguida por um rastreamento de pilha do Chef, como o seguinte:

```
...
Please report any problems with the /usr/scripts/mysqlbug script!

[  OK  ]
MySQL Daemon failed to start.
Starting mysqld:  [FAILED]STDERR: 130611 15:07:55 [Warning] The syntax '--log-slow-
queries' is deprecated and will be removed in a future release. Please use '--slow-
query-log'/'--slow-query-log-file' instead.
130611 15:07:56 [Warning] The syntax '--log-slow-queries' is deprecated and will be
 removed in a future release. Please use '--slow-query-log'/'--slow-query-log-file'
 instead.
---- End output of /sbin/service mysqld start ----

/opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/mixin/command.rb:184:in `handle_command_failures'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/mixin/command.rb:131:in `run_command'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/provider/service/init.rb:37:in `start_service'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/provider/service.rb:60:in `action_start'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/resource.rb:406:in `send'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/resource.rb:406:in `run_action'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/runner.rb:53:in `run_action'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/runner.rb:89:in `converge'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/runner.rb:89:in `each'
```

```
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/runner.rb:89:in `converge'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/resource_collection.rb:94:in `execute_each_resource'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/resource_collection/stepable_iterator.rb:116:in `call'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/resource_collection/stepable_iterator.rb:116:in
 `call_iterator_block'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/resource_collection/stepable_iterator.rb:85:in `step'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/resource_collection/stepable_iterator.rb:104:in `iterate'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/resource_collection/stepable_iterator.rb:55:in
 `each_with_index'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/resource_collection.rb:92:in `execute_each_resource'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/runner.rb:84:in `converge'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/client.rb:268:in `converge'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/client.rb:158:in `run'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/application/solo.rb:190:in `run_application'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/application/solo.rb:181:in `loop'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/application/solo.rb:181:in `run_application'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/../lib/chef/application.rb:62:in `run'
  /opt/aws/opsworks/releases/20130605160141_122/vendor/bundle/ruby/1.8/gems/
chef-0.9.15.5/bin/chef-solo:25
  /opt/aws/opsworks/current/bin/chef-solo:16:in `load'
  /opt/aws/opsworks/current/bin/chef-solo:16
```

O final do arquivo é o rastreamento de pilha do Chef. Você também deve examinar a saída antes da exceção, o que, muitas vezes, contém um erro do sistema, como `package not available`, que também pode ser útil para determinar a causa. Neste caso, o daemon MySQL falhou ao iniciar.

Erros comuns do log do Chef

Veja a seguir alguns dos erros comuns de log do Chef e como solucioná-los.

Não foi possível encontrar o log

No início de uma execução do Chef, as instâncias recebem um URL Amazon S3 pré-designado que permite visualizar o log em uma página da Web quando a execução do Chef for concluída. Como esse URL expira após duas horas, não há log carregados no site Amazon S3 se uma execução do Chef demora mais de duas horas, mesmo se não houver nenhum problema durante a execução do Chef. O comando para criar um log foi bem-sucedido, mas o log pode ser visualizado apenas na instância, não no URL pré-designado.

Log encerrado repentinamente

Se um log de Chef for encerrado repentinamente sem indicar sucesso ou exibir informações de erro, você provavelmente encontrou um estado de memória baixa que impediu o Chef de concluir o log. Sua melhor opção é tentar novamente com uma instância maior.

Livro de receitas ou receita ausente

Se a execução do Chef encontrar um livro de receitas ou receita que não está no cache do livro de receitas, você verá algo como o seguinte:

```
DEBUG: Loading Recipe mycookbook::myrecipe via include_recipe
ERROR: Caught exception during execution of custom recipe: mycookbook::myrecipe:
   Cannot find a cookbook named mycookbook; did you forget to add metadata to a
 cookbook?
```

Esta entrada indica que o livro de receitas `mycookbook` não está no cache de receitas. Com o Chef 11.4, você também pode encontrar esse erro se você não declarar dependências corretamente em `metadata.rb`.

AWS OpsWorks O Stacks executa receitas do cache do livro de receitas da instância. Ele baixa livros de receitas do repositório para o cache quando a instância é iniciada. No entanto, o AWS OpsWorks Stacks não atualizará automaticamente o cache em uma instância on-line se você modificar posteriormente os livros de receitas em seu repositório. Se você tiver modificado os livros de receitas ou adicionado novos livros de receitas desde o início da instância, siga estas etapas:

1. Certifique-se de confirmar suas alterações no repositório.

2. Execute o comando da pilha Atualizar livros de receitas para atualizar o cache do livro de receitas com a versão mais recente do repositório.

Falha de comando local

Se um recurso do Chef `execute` falhar ao executar o comando especificado, você visualizará algo como:

```
DEBUG: ---- End output of ./configure --with-config-file-path=/ returned 2
ERROR: execute[PHP: ./configure] (/root/opsworks-agent/site-cookbooks/php-fpm/
recipes/install.rb line 48) had an error:
   ./configure --with-config-file-path=/
```

Role para cima no logo para visualizar a saída `stderr` e `stdout` do comando, que devem ajudar você a determinar o motivo da falha do comando.

Falha de pacote

Em caso de falha de pacote de instalação, você verá algo como:

```
ERROR: package[zend-server-ce-php-5.3] (/root/opsworks-agent/site-cookbooks/
zend_server/recipes/install.rb line 20)
   had an error: apt-get -q -y --force-yes install zend-server-ce-php-5.3=5.0.4+b17
 returned 100, expected 0
```

Role para cima no log e você deve visualizar as saídas STDOUT e STDERROR do comando, que devem ajudar a determinar o motivo da falha de instalação do pacote.

Usando a CLI do AWS OpsWorks Stacks Agent

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> A CLI do agente está disponível apenas em instâncias do Linux.

Em cada instância on-line, o AWS OpsWorks Stacks instala um agente, que se comunica com o serviço. O serviço AWS OpsWorks Stacks, por sua vez, envia comandos ao agente para realizar tarefas como iniciar execuções do Chef na instância quando ocorre um evento de ciclo de vida. Em instâncias do Linux, o agente expõe uma CLI (interface de linha de comando), o que é muito útil para a solução de problemas. Para executar os comandos de CLI, usar o SSH para se conectar a uma instância. Em seguida, você pode executar os comandos de CLI do agente para executar diversas tarefas, incluindo o seguinte:

- Executar receitas.
- Exibir logs do Chef.
- Exibir a configuração de pilha e a implantação de JSON.

Para obter mais informações sobre como configurar uma conexão SSH com uma instância, consulte Login com SSH. Você também deve ter permissões de SSH e sudo para a pilha.

Esta seção descreve como usar a CLI de agente para a solução de problemas. Para obter mais informações e uma referência de comando completa, consulte AWS OpsWorks CLI do Stacks Agent.

Tópicos

- Execução de receitas
- Exibição de logs do Chef
- Exibição da configuração de pilha e implantação de JSON

Execução de receitas

O comando `run_command` da CLI de agente direciona o agente para executar novamente um comando executado anteriormente. Os comandos mais úteis para a solução de problemas, `setup`, `configure`, `deploy` e `undeploy`, correspondem a um evento de ciclo de vida. Eles direcionam o agente para iniciar uma execução do Chefe a fim de executar as receitas associadas.

> **Note**
>
> O comando `run_command` é limitado a executar o grupo de receitas associado a um comando especificado. Em geral, as receitas que são associadas a um evento de ciclo de vida. Não é possível usá-lo para executar uma receita específica. Para executar uma ou mais receitas específicas, use o comando de pilha Executar receitas ou as ações da API ou CLI equivalentes (`create-deployment` e `CreateDeployment`).

O comando `run_command` é útil para depurar receitas personalizadas, especificamente as atribuídas a eventos de ciclo de vida de Configuração e Instalação, que não podem ser acionados diretamente do console. Ao usar `run_command`, você pode executar as receitas de um evento específico com a frequência necessária sem precisas iniciar ou parar instâncias.

> **Note**
>
> AWS OpsWorks O Stacks executa receitas do cache do livro de receitas da instância, não do repositório do livro de receitas. AWS OpsWorks O Stacks baixa livros de receitas para esse cache quando a instância é iniciada, mas não atualiza automaticamente o cache nas instâncias on-line se você modificar seus livros de receitas posteriormente. Se você modificou os livros de receitas desde o início da instância, certifique-se de executar o comando de pilha Atualizar livros de receitas para atualizar o cache do livro de receitas com a versão mais recente do repositório.

O agente armazena em cache somente os comandos mais recentes. É possível listá-los ao executar `list_commands`, que retorna uma lista de comandos em cache e o horário em que foram executados.

```
sudo opsworks-agent-cli list_commands
2013-02-26T19:08:26        setup
2013-02-26T19:12:01        configure
2013-02-26T19:12:05        configure
2013-02-26T19:22:12        deploy
```

Para executar novamente o comando mais recente, execute:

```
sudo opsworks-agent-cli run_command
```

Para executar a instância mais recente de um comando especificado, execute:

```
sudo opsworks-agent-cli run_command command
```

Por exemplo, para executar novamente Configurar receitas, é possível executar o seguinte comando:

```
sudo opsworks-agent-cli run_command setup
```

Cada comando tem uma configuração de pilha e implantação JSON que representa o estado do pilha e da implantação quando o comando foi executado. Como os dados podem mudar de um comando para o outro, uma instância mais antiga de um comando pode usar dados diferentes da mais recente. Para executar novamente uma instância específica de um comando, copie o horário da saída `list_commands` e execute o seguinte:

```
sudo opsworks-agent-cli run_command time
```

Os exemplos anteriores todos executam novamente o comando utilizando o JSON padrão, que é o JSON instalado para o comando. Você pode executar novamente um comando em um arquivo JSON arbitrário da seguinte forma:

```
sudo opsworks-agent-cli run_command -f /path/to/valid/json.file
```

Exibição de logs do Chef

O comando `show_log` da CLI do agente exibe um log especificado. Depois que o comando é concluído, você visualiza o fim do arquivo. Portanto, o comando `show_log` oferece uma maneira conveniente de analisar o fim do log, que é geralmente onde as informações de erro são encontradas. Você pode rolar para cima a fim de visualizar as partes anteriores do log.

Para exibir o log dos comandos atuais, execute isto:

```
sudo opsworks-agent-cli show_log
```

Você também pode exibir logs de um comando específico, mas saiba que o agente armazena os logs em cache somente nos trinta últimos comandos. É possível listar os comandos de uma instância

ao executar `list_commands`, que retorna uma lista de comandos em cache e o horário em que foram executados. Para obter um exemplo, consulte Execução de receitas.

Para mostrar o log da execução mais recente de um comando específico, execute o seguinte:

```
sudo opsworks-agent-cli show_log command
```

O parâmetro do comando pode ser definido como `setup`, `configure`, `deploy`, `undeploy`, `start`, `stop` ou `restart`. A maioria desses comandos corresponde a eventos de ciclo de vida e direciona o agente para executar as receitas associadas.

Para exibir o log de uma execução de comando específica, copie a data da saída `list_commands` e execute:

```
sudo opsworks-agent-cli show_log date
```

Se um comando ainda está em execução, `show_log` exibe o estado atual do log.

> **Note**
>
> Uma forma de usar `show_log` para solucionar erros e out-of-memory problemas é rastrear um log durante a execução, da seguinte maneira:
>
> 1. Use `run_command` para acionar o evento de ciclo de vida adequado. Para obter mais informações, consulte Execução de receitas.
> 2. Execute repetidamente `show_log` para visualizar a parte final do log como é escrita.
>
> Se o Chef ficar sem memória ou sair inesperadamente, o log encerra abruptamente. Caso de falha de receita, o log termina com uma exceção e um rastreamento de pilha.

Exibição da configuração de pilha e implantação de JSON

A maior parte dos dados usados por receitas vem da configuração de pilha e implantação JSON, que define um conjunto de atributos do Chef e fornece uma descrição detalhada da configuração de pilha, implantações e atributos personalizados opcionais que os usuários podem adicionar. Para cada comando, o AWS OpsWorks Stacks instala um JSON que representa a pilha e o estado de implantação no momento do comando. Para obter mais informações, consulte Configuração de pilha e atributos de implantação.

Se as receitas personalizadas obtêm dados da configuração de pilha e da implantação JSON, é possível verificar os dados ao examinar o JSON. A maneira mais fácil de exibir a configuração de pilha e a implantação JSON é executar o comando `get_json` da CLI do agente, que exibe uma versão formatada do objeto JSON. Veja a seguir as primeiras linhas de uma saída típica:

```
{
  "opsworks": {
    "layers": {
      "php-app": {
        "id": "4a2a56c8-f909-4b39-81f8-556536d20648",
        "instances": {
          "php-app2": {
            "elastic_ip": null,
            "region": "us-west-2",
            "booted_at": "2013-02-26T20:41:10+00:00",
            "ip": "10.112.235.192",
            "aws_instance_id": "i-34037f06",
            "availability_zone": "us-west-2a",
            "instance_type": "c1.medium",
            "private_dns_name": "ip-10-252-0-203.us-west-2.compute.internal",
            "private_ip": "10.252.0.203",
            "created_at": "2013-02-26T20:39:39+00:00",
            "status": "online",
            "backends": 8,
            "public_dns_name": "ec2-10-112-235-192.us-west-2.compute.amazonaws.com"
...
```

Você pode exibir a configuração de pilha mais recente e a implantação JSON da seguinte forma:

```
sudo opsworks-agent-cli get_json
```

Você pode exibir a configuração de pilha e a implantação JSON mais recente de um comando especificado ao executar o seguinte:

```
sudo opsworks-agent-cli get_json command
```

O parâmetro do comando pode ser definido como `setup`, `configure`, `deploy`, `undeploy`, `start`, `stop` ou `restart`. A maioria desses comandos corresponde a eventos de ciclo de vida e direciona o agente para executar as receitas associadas.

Você pode exibir a configuração de pilha e a implantação JSON de determinado comando ao especificar a data do comando como:

```
sudo opsworks-agent-cli get_json date
```

A maneira mais simples de usar este comando é a seguinte:

1. Execute `list_commands`, que retorna uma lista de comandos que foram executados na instância e a data em que cada comando foi executado.
2. Copie a data do comando apropriado e use-a como `get_json` *`date`* argumento.

Depuração e solução de problemas comuns

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção descreve alguns problemas normalmente encontrados na depuração e na solução de problemas e suas soluções.

Tópicos

- Pilhas de solução AWS OpsWorks de problemas
- Solução de problemas do registro da instância

Pilhas de solução AWS OpsWorks de problemas

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Esta seção contém alguns problemas comuns do AWS OpsWorks Stacks e suas soluções.

Tópicos

- Não é possível gerenciar instâncias
- Depois de uma execução do Chef, as instâncias não serão inicializadas
- Todas as instâncias de uma camada falham na verificação de integridade do Elastic Load Balancing
- Não consigo me comunicar com um balanceador de carga do Elastic Load Balancing
- Uma instância no local importada deixa de concluir a configuração do volume após uma reinicialização
- Um volume EBS não é anexado novamente após uma reinicialização
- Não é possível excluir security groups do AWS OpsWorks Stacks
- Excluiu acidentalmente um grupo de segurança AWS OpsWorks do Stacks
- Log do Chef encerra abruptamente
- O livro de receitas não é atualizado
- As instâncias permanecem presas no status de inicialização
- Instâncias reiniciam inesperadamente
- Processos opsworks-agent em execução em instâncias
- Comandos execute_recipes inesperados

Não é possível gerenciar instâncias

Problema: você não é mais capaz de gerenciar uma instância que foi gerenciável no passado. Em alguns casos, os logs podem mostrar um erro semelhante ao seguinte.

```
Aws::CharlieInstanceService::Errors::UnrecognizedClientException - The security token
 included in the request is invalid.
```

Causa: isso poderá ocorrer se um recurso externo ao AWS OpsWorks do qual a instância dependa tiver sido editado ou excluído. Estes são exemplos de alterações feitas no recurso que podem parar a comunicação com uma instância.

- Um usuário ou função do IAM associado à instância foi excluído acidentalmente, fora do AWS OpsWorks Stacks. Isso causa uma falha de comunicação entre o AWS OpsWorks agente que está instalado na instância e o serviço AWS OpsWorks Stacks. O usuário do associado a uma instância é obrigatório durante todo o ciclo de vida da instância.
- Editar as configurações de volume ou armazenamento enquanto uma instância permanece off-line pode deixar uma instância ingerenciável.
- Adicionar EC2 instâncias a um ELB manualmente. AWS OpsWorks reconfigura um load balancer atribuído do Elastic Load Balancing sempre que uma instância entra ou sai do estado on-line. AWS OpsWorks considera apenas as instâncias que conhece como membros válidos; as instâncias que são adicionadas fora ou por algum outro processo são removidas. AWS OpsWorks Todas as demais instâncias são removidas.

Solução: não exclua usuários do IAM ou perfis de que as instâncias dependam. Se possível, só edite configurações de volume ou armazenamento enquanto as instâncias dependentes estiverem em execução. Use AWS OpsWorks para gerenciar o balanceador de carga ou as associações EIP das instâncias. AWS OpsWorks Quando você estiver registrando uma instância, para ajudar a evitar problemas no gerenciamento de instâncias registradas caso o usuário seja excluído acidentalmente, em vez disso, adicione o parâmetro `--use-instance-profile` ao comando `register` para usar o perfil de instância interno da instância.

Depois de uma execução do Chef, as instâncias não serão inicializadas

Problema: No Chef 11.10 ou anterior, as pilhas configuradas para usar livros de receitas personalizados, depois de uma execução do Chef que usou livros de receitas da comunidade, as instâncias não serão inicializadas. As mensagens de log podem informar que as receitas deixaram de ser compiladas ("Erro na compilação da receita") ou não podem ser carregadas porque não conseguem encontrar uma dependência.

Causa: A causa mais provável é que o livro de receitas personalizado ou da comunidade não dá suporte à versão do Chef usada pela pilha. Alguns livros de receitas populares da comunidade, como `apt` e `build-essential`, têm problemas de compatibilidade conhecidos com o Chef 11.10.

Solução: Em AWS OpsWorks pilhas com a configuração Usar livros de receitas personalizados do Chef ativada, os livros de receitas personalizados ou comunitários devem sempre oferecer suporte à versão do Chef que sua pilha usa. Fixe os livros de receitas da comunidade em uma versão (ou seja, defina o número da versão do livro de receitas como uma versão específica), que seja compatível com a versão do Chef configurada nas configurações da pilha. Para encontrar uma versão do livro

de receitas da comunidade compatível, visualize o log de alterações de um livro de receitas não compilado e só use a versão mais recente do livro de receitas para a qual a pilha dará suporte. Para fixar uma versão do livro de receitas, especifique um número de versão exato no Berksfile do repositório do livro de receitas personalizado. Por exemplo, `cookbook 'build-essential', '= 3.2.0'`.

Todas as instâncias de uma camada falham na verificação de integridade do Elastic Load Balancing

Problema: você anexa um balanceador de carga do Elastic Load Balancing a uma camada do servidor de aplicações, mas todas as instâncias falham na verificação de integridade.

Causa: ao criar um balanceador de carga do Elastic Load Balancing, você deve especificar o caminho de ping que o balanceador de carga chama para determinar se a instância é íntegra. Não se esqueça de especificar um caminho de ping que seja apropriado para a aplicação; o valor padrão é /index.html. Caso a aplicação não inclua um `index.html`, você deve especificar um caminho apropriado, ou a verificação de integridade falhará. Por exemplo, o PHPApp aplicativo Simple usado em Conceitos básicos das pilhas Linux do Chef 11 não usa index.html; o caminho de ping apropriado para esses servidores é /.

Solução: Edite o caminho de ping do balanceador de carga. Para obter mais informações, consulte Elastic Load Balancing

Não consigo me comunicar com um balanceador de carga do Elastic Load Balancing

Problema: você cria um balanceador de carga do Elastic Load Balancing e o anexa a uma camada do servidor de aplicações, mas ao clicar no nome DNS ou no endereço IP do balanceador de carga para executar a aplicação, você recebe um erro informando que o servidor remoto não está respondendo.

Causa: caso a pilha esteja em execução em uma VPC padrão, ao criar um balanceador de carga do na região, você deve especificar um grupo de segurança. O security group deve ter regras de entrada que permitem o tráfego de entrada pelo endereço IP. Caso você especifique default VPC security group, a regra de entrada padrão não aceitará nenhum tráfego de entrada.

Solução: Edite as regras de entrada do security group para aceitar tráfego de entrada por endereços IP apropriados.

1. Clique em Security Groups no painel de navegação do EC2 console Amazon.
2. Selecione o security group do balanceador de carga.

3. Clique em Edit na guia Inbound.
4. Adicione uma regra de entrada com Source definido como um CIDR apropriado.

   Por exemplo, a especificação de Anywhere define o CIDR como 0.0.0.0/0, que leva o balanceador de carga a aceitar o tráfego de entrada de qualquer endereço IP.

Uma instância no local importada deixa de concluir a configuração do volume após uma reinicialização

Problema: você reinicia uma EC2 instância que importou para o AWS OpsWorks Stacks e as exibições do console AWS OpsWorks Stacks falharam como status da instância. Isso pode ocorrer em instâncias do Chef 11 ou do Chef 12.

Causa: O AWS OpsWorks Stacks talvez não consiga anexar um volume à instância durante o processo de configuração. Uma causa possível é que o AWS OpsWorks Stacks substitui a configuração do volume na instância quando você executa o comando `setup`.

Solução: abra a página Details da instância e verifique a configuração do volume na área Volumes. Você só pode alterar a configuração do volume quando a instância está no estado stopped. Certifique-se de que todo volume tenha um ponto de montagem e um nome especificados. Confirme se você forneceu o ponto de montagem correto em sua configuração no AWS OpsWorks Stacks antes de reiniciar a instância.

Um volume EBS não é anexado novamente após uma reinicialização

Problema: Você usa o EC2 console da Amazon para conectar um volume do Amazon EBS a uma instância, mas quando você reinicializa a instância, o volume não está mais conectado.

Causa: AWS OpsWorks As pilhas podem reconectar somente os volumes do Amazon EBS que ela conhece, que estão limitados ao seguinte:

- Volumes que foram criados pelo AWS OpsWorks Stacks.
- Os volumes da conta que você registrou explicitamente com uma pilha usando a página Resources.

Solução: gerencie seus volumes do Amazon EBS somente usando o console, a API ou a CLI do AWS OpsWorks Stacks. Se você quiser usar um dos volumes do Amazon EBS da conta com uma pilha, use a página Recursos da pilha para registrar o volume e anexá-lo a uma instância. Para obter mais informações, consulte Gerenciamento de recursos.

Não é possível excluir security groups do AWS OpsWorks Stacks

Problema: depois de excluir uma pilha, restam vários grupos de segurança de AWS OpsWorks pilhas que não podem ser excluídos.

Causa: Os security groups devem ser excluídos em uma ordem específica.

Solução: Primeiro, certifique-se de que nenhuma instância esteja usando os security groups. Em seguida, exclua qualquer um dos seguintes security groups, caso eles existam, na seguinte ordem:

1. AWS- OpsWorks -Servidor em branco
2. AWS- OpsWorks -Servidor-mestre de monitoramento
3. Servidor mestre de banco de dados OpsWorks da AWS
4. AWS- OpsWorks -Memcached-Server
5. AWS- OpsWorks -Servidor personalizado
6. AWS- OpsWorks -NodeJS-App-Server
7. Servidor de aplicativos AWS- OpsWorks -PHP
8. Servidor de aplicativos AWS- OpsWorks -Rails
9. AWS- OpsWorks -Servidor web
10. AWS- OpsWorks -Servidor padrão
11. Servidor AWS- OpsWorks -LB

Excluiu acidentalmente um grupo de segurança AWS OpsWorks do Stacks

Problema: você excluiu um dos grupos de segurança do AWS OpsWorks Stacks e precisa recriá-lo.

Causa: Esses security groups normalmente são excluídos por acidente.

Solução: O grupo recriado deve ser uma cópia exata do original, inclusive a mesma capitalização para o nome do grupo. Em vez de recriar manualmente o grupo, a abordagem preferida é para o AWS OpsWorks Stacks realizar a tarefa para você. Basta criar uma nova pilha na mesma região da AWS — e VPC, se AWS OpsWorks presente — e o Stacks recriará automaticamente todos os grupos de segurança integrados, incluindo aquele que você excluiu. Você pode então excluir a pilha se não tiver mais uso para ela; os security groups permanecerão.

Log do Chef encerra abruptamente

Problema: Um log do Chef é encerrado abruptamente; o final do log não indica uma execução bem-sucedida ou exibe uma exceção e um rastreamento da pilha.

Causa: Esse comportamento costuma ser causado por memória inadequada.

Solução: Crie uma instância maior e use o comando `run_command` da CLI do agente para reexecutar as receitas. Para obter mais informações, consulte Execução de receitas.

O livro de receitas não é atualizado

Problema: você atualizou seus livros de receitas, mas as instâncias da pilha continuam executando as receitas antigas.

Causa: o AWS OpsWorks Stacks armazena livros de receitas em cada instância e executa receitas do cache, não do repositório. Quando você inicia uma nova instância, o AWS OpsWorks Stacks baixa seus livros de receitas do repositório para o cache da instância. No entanto, caso você modifique depois os livros de receitas personalizados, o AWS OpsWorks Stacks não atualiza automaticamente os caches das instâncias online.

Solução: execute o comando Atualizar pilha de livros de receitas para direcionar explicitamente as AWS OpsWorks pilhas a atualizar os caches de livros de receitas de suas instâncias on-line.

As instâncias permanecem presas no status de inicialização

Problema: Quando você reinicia uma instância, ou a correção automática é reiniciada automaticamente, a operação de inicialização para no status `booting`.

Causa: Uma causa possível desse problema é a configuração da VPC, inclusive uma VPC padrão. As instâncias devem sempre ser capazes de se comunicar com o serviço AWS OpsWorks Stacks, o Amazon S3 e os repositórios de pacotes, livros de receitas e aplicativos. Se, por exemplo, você remover um gateway padrão de uma VPC padrão, as instâncias perderão a conexão com o serviço AWS OpsWorks Stacks. Como o AWS OpsWorks Stacks não consegue mais se comunicar com o agente da instância, ele trata a instância como falhada e a cura automaticamente. No entanto, sem uma conexão, o AWS OpsWorks Stacks não pode instalar um agente de instância na instância recuperada. Sem um agente, o AWS OpsWorks Stacks não pode executar as receitas de configuração na instância, portanto, a operação de inicialização não pode progredir além do status de “inicialização”.

Solução: Modifique a configuração da VPC, de maneira que as instâncias tenham a conectividade necessária.

Instâncias reiniciam inesperadamente

Problema: Uma instância interrompida reinicia inesperadamente.

Causa 1: Se você ativou a recuperação automática para suas instâncias, o AWS OpsWorks Stacks realiza periodicamente uma verificação de saúde nas EC2 instâncias associadas da Amazon e reinicia as que não estão íntegras. Se você interromper ou encerrar uma instância AWS OpsWorks gerenciada pelo Stacks usando o EC2 console, a API ou a CLI da Amazon, o Stacks não será notificado. AWS OpsWorks Em vez disso, ele irá considerar a instância parada como não íntegra e iniciá-la automaticamente.

Solução: Gerencie apenas as instâncias do usando a API ou a CLI do console do AWS OpsWorks Stacks. Se você usar o AWS OpsWorks Stacks para interromper ou excluir uma instância, ela não será reiniciada. Para ter mais informações, consulte Descreve como iniciar, parar e reiniciar instâncias 24/7 e Excluindo instâncias do AWS OpsWorks Stacks.

Causa 2: As instâncias podem falhar por vários motivos. Se você tiver a recuperação automática ativada, o AWS OpsWorks Stacks reinicia automaticamente as instâncias com falha.

Solução: Essa é uma operação normal; não há necessidade de fazer nada, a menos que você não queira que o AWS OpsWorks Stacks reinicie as instâncias com falha. Neste caso, você deve desativar a correção automática.

Processos **`opsworks-agent`** em execução em instâncias

Problema: Diversos processos `opsworks-agent` são executados nas instâncias. Por exemplo:

```
aws 24543 0.0 1.3 172360 53332 ? S Feb24 0:29 opsworks-agent: master 24543
aws 24545 0.1 2.0 208932 79224 ? S Feb24 22:02 opsworks-agent: keep_alive of master
 24543
aws 24557 0.0 2.0 209012 79412 ? S Feb24 8:04 opsworks-agent: statistics of master
 24543
aws 24559 0.0 2.2 216604 86992 ? S Feb24 4:14 opsworks-agent: process_command of master
 24
```

Causa: Estes são os processos legítimos necessários à operação normal do agente. Eles realizam tarefas como o processamento de implantações e o reenvio de mensagens keep-alive para o serviço.

Solução: Este é o comportamento normal. Não pare esses processos, pois isso comprometerá a operação do agente.

Comandos execute_recipes inesperados

Problema: a seção Logs na página de detalhes de uma instância inclui comandos `execute_recipes` inesperados. Comandos `execute_recipes` inesperado também podem ser exibidos nas páginas Stack e Deployments.

Causa: Este problema normalmente é causado por alterações feitas na permissão. Quando você altera as permissões SSH ou sudo de um usuário ou grupo, o AWS OpsWorks Stacks é executado `execute_recipes` para atualizar as instâncias da pilha e também aciona um evento Configure. Outra fonte do comando `execute_recipes` é a atualização do agente da instância pelo AWS OpsWorks Stacks.

Solução: Esta é uma operação normal. Não há necessidade de fazer nada.

Para ver quais ações um comando `execute_recipes` realizou, vá até a página Deployments e clique no time stamp do comando. Isso abre a página de detalhes do comando, que lista as principais receitas que foram executadas. Por exemplo, a página de detalhes a seguir é de um comando `execute_recipes` que executou `ssh_users` para atualizar as permissões SSH.

Para ver todos os detalhes, clique em show na coluna Log do comando para exibir o log do Chef associado. Pesquise no registro por**`Run List`**. AWS OpsWorks As receitas de manutenção de pilhas estarão abaixo`OpsWorks Custom Run List`. Por exemplo, esta é a lista de execuções do comando `execute_recipes` mostrado na captura de tela anterior e que mostra todas as receitas associadas ao comando.

```
[2014-02-21T17:16:30+00:00] INFO: OpsWorks Custom Run List:
 ["opsworks_stack_state_sync",
  "ssh_users", "test_suite", "opsworks_cleanup"]
```

Solução de problemas do registro da instância

Esta seção contém alguns problemas normalmente encontrados no registro da instância e as soluções.

> **Note**
>
> Se você estiver enfrentando problemas no registro, execute `register` com o argumento `--debug`, que apresenta informações adicionais sobre depuração.

Tópicos

- EC2O usuário não está autorizado a realizar:...
- A credencial deve ter como escopo uma região válida

EC2O usuário não está autorizado a realizar:...

Problema: Um comando `register` retorna algo semelhante ao seguinte:

```
A client error (AccessDenied) occurred when calling the CreateGroup operation:
User: arn:aws:iam::123456789012:user/ImportEC2User is not authorized to
perform: iam:CreateGroup on resource:
arn:aws:iam::123456789012:group/AWS/OpsWorks/OpsWorks-b583ce55-1d01-4695-b3e5-
ee19257d1911
```

Causa: o comando `register` permanece em execução com as credenciais que não concedem as permissões necessárias. A política do usuário deve permitir a ação `iam:CreateGroup`, dentre outras.

Solução Dê a `register` credenciais de usuário do IAM que tenham as permissões necessárias. Para obter mais informações, consulte Instalar e configurar a AWS CLI.

A credencial deve ter como escopo uma região válida

Problema: Um comando `register` retorna o seguinte:

```
A client error (InvalidSignatureException) occurred when calling the
DescribeStacks operation: Credential should be scoped to a valid region, not 'cn-
north-1'.
```

Causa: A região do comando deve ser uma região do AWS OpsWorks Stacks válida. Para ver uma lista de regiões compatíveis, consulte Suporte regional. Este erro normalmente ocorre por um dos seguintes motivos:

- A pilha está em uma região diferente, e você atribuiu uma pilha da região ao argumento `--region` do comando.

  Você não precisa especificar uma região da pilha; as AWS OpsWorks pilhas a determinam automaticamente a partir do ID da pilha.
- Você omitiu o argumento `--region`, o que implicitamente especifica a região padrão, mas a região padrão não é compatível com o AWS OpsWorks Stacks.

Solução: `--region` defina explicitamente como uma região de AWS OpsWorks pilhas compatível ou edite seu AWS CLI `config` arquivo para alterar a região padrão para uma região de AWS OpsWorks pilhas compatível. Para obter mais informações, consulte Configurar a interface de linha de comando da AWS.

AWS OpsWorks CLI do Stacks Agent

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

> **Note**
>
> Esse recurso está disponível apenas em instâncias do Linux.

O agente que o AWS OpsWorks Stacks instala em cada instância expõe uma interface de linha de comando (CLI). Se usar SSH para entrar na instância, você pode usar a ILC para o seguinte:

- Acessar arquivos de log para execuções do Chef.

- Comandos do Access AWS OpsWorks Stacks.
- Executar receitas do Chef manualmente.
- Ver relatórios de instância.
- Ver relatórios de agente.
- Ver um conjunto limitado de atributos de configuração e implantação da pilha.

> **Important**
>
> Você pode executar comandos ILC de agente apenas como raiz ou usando `sudo`.

A sintaxe de comando básica é:

```
sudo opsworks-agent-cli [--help] [command [activity] [date]]
```

Os quatro argumentos são os seguintes:

help

(Opcional) Exibe uma breve sinopse dos comandos disponíveis quando usado por ele mesmo. Quando usado com um comando, `help` exibe uma descrição do comando.

command

(Opcional) O comando de ILC do agente, que deve ser definido para um dos seguintes:

- agent_report
- get_json
- instance_report
- list_commands
- run_command
- show_log
- stack_state

atividade

(Opcional) Usado como um argumento com alguns comandos para especificar uma determinada atividade do AWS OpsWorks Stacks: `setup`, `configure`, `deploy`, `undeploy`, `start`, `stop` ou `restart`.

data

(Opcional) Usado como um argumento com alguns comandos para especificar uma execução de comando do AWS OpsWorks Stacks. Especifique a execução do comando definindo a data do carimbo de data/hora em que o comando foi executado no *`yyyy-mm-ddThh:mm:ss`* formato, incluindo as aspas simples. Por exemplo, para às 10:31:55 de 5 de fevereiro na terça-feira de 2013, use: `'2013-02-05T10:31:55'`. Para determinar quando um determinado comando do AWS OpsWorks Stacks foi executado, execute list_commands.

> **Note**
>
> Se o agente executou a mesma atividade do AWS OpsWorks Stacks várias vezes, você pode escolher uma execução específica especificando a atividade e a hora em que ela foi executada. Você especifica uma atividade e omite o tempo, o comando da ILC do agente age na execução mais recente da atividade. Você omite os argumentos, o comando da ILC do agente age na atividade mais recente.

As seguintes seções descrevem os comandos e seus argumentos associados. Para ser sucinto, as seções de sintaxe omitem a opção de `--help` opcional, que pode ser usada com qualquer comando.

Tópicos

- agent_report
- get_json
- instance_report
- list_commands
- run_command
- show_log
- stack_state

agent_report

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Devolve um relatório de agente.

```
sudo opsworks-agent-cli agent_report
```

O exemplo de saída a seguir é de uma instância que, mais recentemente, executou uma atividade configurada.

```
$ sudo opsworks-agent-cli agent_report

AWS OpsWorks Instance Agent State Report:

  Last activity was a "configure" on 2015-12-01 18:19:23 UTC
  Agent Status: The AWS OpsWorks agent is running as PID 30998
  Agent Version: 4004-20151201152533, up to date
```

get_json

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Devolve informações sobre uma execução do Chef com um objeto JSON.

```
sudo opsworks-agent-cli get_json [activity] [date] [-i | --internal | --no-i | --no-
internal]
```

Por padrão, `get_json` exibe informações fornecidas por cliente para a execução do Chef mais recente. Use as seguintes opções para especificar um conjunto particular de informações.

atividade

Exibe informações para a execução do Chef associadas com a atividade especificada mais recente. Para obter uma lista de atividades válida, execute list_commands.

data

Exibe informações para a execução do Chef associadas com a atividade executada para o timestamp especificado. Para obter uma lista de timestamps válidos, execute list_commands.

-i, --internal

Exibe informações que o AWS OpsWorks Stacks usa internamente para a execução do Chef.

--no-i, --no-internal

Exibe explicitamente informações fornecidas por cliente para a execução do Chef. Esse é o padrão se outro não for especificado.

> **Note**
>
> Para instâncias do Chef 12 Linux, executar esse comando retornará informações válidas como a configuração da pilha da instância e implantação de atributos. No entanto, para obter informações mais completas, consulte os pacotes de dados do Chef que o AWS OpsWorks Stacks cria na instância. Para obter mais informações, consulte AWS OpsWorks Referência do Stacks Data Bag.

O exemplo de saída a seguir mostra as informações fornecidas pelo cliente para a execução do Chef mais recente da atividade de configuração mais recente.

```
$ sudo opsworks-agent-cli get_json configure

{
  "run_list": [
```

```
    "recipe[opsworks_cookbook_demo::configure]"
  ]
}
```

O exemplo de saída a seguir mostra as informações que o AWS OpsWorks Stacks usa internamente para a execução do Chef executada no timestamp especificado.

```
$ sudo opsworks-agent-cli get_json 2015-12-01T18:20:24 -i

{
  "aws_opsworks_agent": {
    "version": "4004-20151201152533",
    "valid_client_activities": [
      "reboot",
      "stop",
      "deploy",
      "grant_remote_access",
      "revoke_remote_access",
      "update_agent",
      "setup",
      "configure",
      "update_dependencies",
      "install_dependencies",
      "update_custom_cookbooks",
      "execute_recipes",
      "sync_remote_users"
    ],
    "command": {
      "type": "configure",
      "args": {
        "app_ids": [

        ]
      },
      "sent_at": "2015-12-01T18:19:23+00:00",
      "command_id": "5c2113f3-c6d5-40eb-bcfa-77da2885eeEX",
      "iam_user_arn": null,
      "instance_id": "cfdaa716-42fe-4e3b-9762-fef184ddd8EX"
    },
    "resources": {
      "apps": [

      ],
```

```
    "layers": [
      {
        "layer_id": "93f50d83-1e73-45c4-840a-0d4f07cda1EX",
        "name": "MyCookbooksDemoLayer",
        "packages": [

        ],
        "shortname": "cookbooks-demo",
        "type": "custom",
        "volume_configurations": [

        ]
      }
    ],
    "instances": [
      {
        "ami_id": "ami-d93622EX",
        "architecture": "x86_64",
        "auto_scaling_type": null,
        "availability_zone": "us-west-2a",
        "created_at": "2015-11-18T00:21:05+00:00",
        "ebs_optimized": false,
        "ec2_instance_id": "i-a480e960",
        "elastic_ip": null,
        "hostname": "cookbooks-demo1",
        "instance_id": "cfdaa716-42fe-4e3b-9762-fef184ddd8EX",
        "instance_type": "c3.large",
        "layer_ids": [
          "93f50d83-1e73-45c4-840a-0d4f07cda1EX"
        ],
        "os": "Amazon Linux 2015.09",
        "private_dns": "ip-192-0-2-0.us-west-2.compute.internal",
        "private_ip": "10.122.69.33",
        "public_dns": "ec2-203-0-113-0.us-west-2.compute.amazonaws.com",
        "public_ip": "192.0.2.0",
        "root_device_type": "ebs",
        "root_device_volume_id": "vol-f6f7e8EX",
        "ssh_host_dsa_key_fingerprint": "f2:...:15",
        "ssh_host_dsa_key_public": "ssh-dss AAAAB3Nz...a8vMbqA=",
        "ssh_host_rsa_key_fingerprint": "0a:...:96",
        "ssh_host_rsa_key_public": "ssh-rsa AAAAB3Nz...yhPanvo7",
        "status": "online",
        "subnet_id": null,
        "virtualization_type": "paravirtual",
```

```
          "infrastructure_class": "ec2",
          "ssh_host_dsa_key_private": "-----BEGIN DSA PRIVATE KEY-----
\nMIIDVwIB...g5OtgQ==\n-----END DSA PRIVATE KEY-----\n",
          "ssh_host_rsa_key_private": "-----BEGIN RSA PRIVATE KEY-----
\nMIIEowIB...78kprtIw\n-----END RSA PRIVATE KEY-----\n"
        }
      ],
      "users": [

      ],
      "elastic_load_balancers": [

      ],
      "rds_db_instances": [

      ],
      "stack": {
        "arn": "arn:aws:opsworks:us-west-2:80398EXAMPLE:stack/040c3def-b2b4-4489-bb1b-
e08425886fEX/",
        "custom_cookbooks_source": {
          "type": "s3",
          "url": "https://s3.amazonaws.com/amzn-s3-demo-bucket/opsworks-cookbook-
demo.tar.gz",
          "username": "AKIAJUQN...WG644EXA",
          "password": "O5v+4Zz+...rcKbFTJu",
          "ssh_key": null,
          "revision": null
        },
        "name": "MyCookbooksDemoStack",
        "region": "us-west-2",
        "stack_id": "040c3def-b2b4-4489-bb1b-e08425886fEX",
        "use_custom_cookbooks": true,
        "vpc_id": null
      },
      "ecs_clusters": [

      ],
      "volumes": [

      ]
    },
    "chef": {
      "customer_recipes": [
        "opsworks_cookbook_demo::configure"
```

```
      ],
      "customer_json": "e30=\n",
      "customer_data_bags": "e30=\n"
    }
  }
}
```

instance_report

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Devolve um relatório de instância estendido.

```
sudo opsworks-agent-cli instance_report
```

O seguinte exemplo de saída e de uma instância.

```
$ sudo opsworks-agent-cli instance_report

AWS OpsWorks Instance Agent State Report:

  Last activity was a "configure" on 2015-12-01 18:19:23 UTC
  Agent Status: The AWS OpsWorks agent is running as PID 30998
  Agent Version: 4004-20151201152533, up to date
  OpsWorks Stack: MyCookbooksDemoStack
  OpsWorks Layers: MyCookbooksDemoLayer
  OpsWorks Instance: cookbooks-demo1
  EC2 Instance ID: i-a480e9EX
  EC2 Instance Type: c3.large
  Architecture: x86_64
  Total Memory: 3.84 Gb
  CPU: 2x  Intel(R) Xeon(R) CPU E5-2680 v2 @ 2.80GHz
```

```
Location:

  EC2 Region: us-west-2
  EC2 Availability Zone: us-west-2a

Networking:

  Public IP: 192.0.2.0
  Private IP: 198.51.100.0
```

list_commands

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Lista o tempo em que cada atividade foi executada nessa instância. Você pode usar esses momentos para outros comandos de ILC de agente especificarem uma execução em particular.

```
sudo opsworks-agent-cli list_commands  [activity] [date]
```

O exemplo de saída a seguir é de uma instância que executou atividades dos livros de receitas personalizadas de definição, configuração e atualização.

```
$ sudo opsworks-agent-cli list_commands

2015-11-24T21:00:28        update_custom_cookbooks
2015-12-01T18:19:09        setup
2015-12-01T18:20:24        configure
```

run_command

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Executa um comando AWS OpsWorks Stacks, que é um arquivo JSON contendo uma lista de execução do Chef que contém as informações necessárias para executar uma atividade do AWS OpsWorks Stacks (instalação, configuração, implantação e assim por diante). O comando `run_command` gera uma entrada de log que você pode ver executando show_log . Essa opção é destinada apenas para fins de desenvolvimento, portanto, o AWS OpsWorks Stacks não rastreia as alterações.

```
sudo opsworks-agent-cli run_command [activity] [date] [/path/to/valid/json.file]
```

Por padrão, `run_command` executa o comando AWS OpsWorks Stacks mais recente. Use as seguintes opções para especificar um comando particular.

atividade

Execute um comando AWS OpsWorks Stacks especificado:`setup`,`configure`,`deploy`,`undeploy`, `startstop`, `ourestart`.

data

Execute o OpsWorks comando da AWS executado no timestamp especificado. Para obter uma lista de timestamps válidos, execute list_commands.

file

Executa o arquivo JSON do comando especificado. Para obter um caminho de arquivo de comando, execute get_json.

O exemplo de saída a seguir é de uma instância e executa o comando de configuração.

```
$ sudo opsworks-agent-cli run_command configure

[2015-12-02 16:52:53]  INFO [opsworks-agent(21970)]: About to re-run 'configure' from
 2015-12-01T18:20:24
...
[2015-12-02 16:53:02]  INFO [opsworks-agent(21970)]: Finished Chef run with exitcode 0
```

show_log

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Devolve um arquivo de log de comando.

```
sudo opsworks-agent-cli show_log [activity] [date]
```

Por padrão, `show_log` segue o arquivo de log mais recente. Use as seguintes opções para especificar um comando particular.

atividade

Exibe o arquivo de log da atividade especificada.

data

Exibe o arquivo de log da atividade executada no timestamp especificado. Para obter uma lista de timestamps válidos, execute list_commands.

O exemplo de saída a seguir exibe o log mais recentes.

```
$ sudo opsworks-agent-cli show_log

[2015-12-02T16:52:59+00:00] INFO: Storing updated cookbooks/opsworks_cookbook_demo/
opsworks-cookbook-demo.tar.gz in the cache.
```

```
...
[2015-12-02T16:52:59+00:00] INFO: Report handlers complete
```

stack_state

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Exibe informações que o AWS OpsWorks Stacks usa internamente para a execução mais recente do Chef.

```
opsworks-agent-cli stack_state
```

> **Note**
>
> Para instâncias do Chef 12 Linux, executar esse comando retornará informações válidas como a configuração da pilha da instância e implantação de atributos. No entanto, para obter informações mais completas, consulte os pacotes de dados do Chef que o AWS OpsWorks Stacks cria na instância. Para obter mais informações, consulte AWS OpsWorks Referência do Stacks Data Bag.

O seguinte exemplo de saída e de uma instância.

```
$ sudo opsworks-agent-cli stack_state

{
  "last_command": {
    "sent_at": "2015-12-01T18:19:23+00:00",
    "activity": "configure"
  },
  "instance": {
    "ami_id": "ami-d93622EX",
```

```
    "architecture": "x86_64",
    "auto_scaling_type": null,
    "availability_zone": "us-west-2a",
    "created_at": "2015-11-18T00:21:05+00:00",
    "ebs_optimized": false,
    "ec2_instance_id": "i-a480e9EX",
    "elastic_ip": null,
    "hostname": "cookbooks-demo1",
    "instance_id": "cfdaa716-42fe-4e3b-9762-fef184ddd8EX",
    "instance_type": "c3.large",
    "layer_ids": [
      "93f50d83-1e73-45c4-840a-0d4f07cda1EX"
    ],
    "os": "Amazon Linux 2015.09",
    "private_dns": "ip-192-0-2-0.us-west-2.compute.internal",
    "private_ip": "10.122.69.33",
    "public_dns": "ec2-203-0-113-0.us-west-2.compute.amazonaws.com",
    "public_ip": "192.0.2.0",
    "root_device_type": "ebs",
    "root_device_volume_id": "vol-f6f7e8EX",
    "ssh_host_dsa_key_fingerprint": "f2:...:15",
    "ssh_host_dsa_key_public": "ssh-dss AAAAB3Nz...a8vMbqA=",
    "ssh_host_rsa_key_fingerprint": "0a:...:96",
    "ssh_host_rsa_key_public": "ssh-rsa AAAAB3Nz...yhPanvo7",
    "status": "online",
    "subnet_id": null,
    "virtualization_type": "paravirtual",
    "infrastructure_class": "ec2",
    "ssh_host_dsa_key_private": "-----BEGIN DSA PRIVATE KEY-----\nMIIDVwIB...g5OtgQ==
\n-----END DSA PRIVATE KEY-----\n",
    "ssh_host_rsa_key_private": "-----BEGIN RSA PRIVATE KEY-----\nMIIEowIB...78kprtIw
\n-----END RSA PRIVATE KEY-----\n"
  },
  "layers": [
    {
      "layer_id": "93f50d83-1e73-45c4-840a-0d4f07cda1EX",
      "name": "MyCookbooksDemoLayer",
      "packages": [

      ],
      "shortname": "cookbooks-demo",
      "type": "custom",
      "volume_configurations": [
```

```
      ]
    }
  ],
  "applications": null,
  "stack": {
    "arn": "arn:aws:opsworks:us-west-2:80398EXAMPLE:stack/040c3def-b2b4-4489-bb1b-
e08425886fEX/",
    "custom_cookbooks_source": {
      "type": "s3",
      "url": "https://s3.amazonaws.com/amzn-s3-demo-bucket/opsworks-cookbook-
demo.tar.gz",
      "username": "AKIAJUQN...WG644EXA",
      "password": "O5v+4Zz+...rcKbFTJu",
      "ssh_key": null,
      "revision": null
    },
    "name": "MyCookbooksDemoStack",
    "region": "us-west-2",
    "stack_id": "040c3def-b2b4-4489-bb1b-e08425886fEX",
    "use_custom_cookbooks": true,
    "vpc_id": null
  },
  "agent": {
    "valid_activities": [
      "reboot",
      "stop",
      "deploy",
      "grant_remote_access",
      "revoke_remote_access",
      "update_agent",
      "setup",
      "configure",
      "update_dependencies",
      "install_dependencies",
      "update_custom_cookbooks",
      "execute_recipes",
      "sync_remote_users"
    ]
  }
}
```

AWS OpsWorks Referência do Stacks Data Bag

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

AWS OpsWorks Stacks expõe uma grande variedade de configurações de receitas como conteúdo do pacote de dados do Chef. Essa referência lista esse conteúdo de data bag.

Um data bag é um conceito do Chef. O data bag é uma variável global armazenada como dados JSON em uma instância; os dados JSON podem ser acessados no Chef. Por exemplo, um pacote de dados pode armazenar variáveis globais, como o URL de origem de um aplicativo, o nome do host da instância e o identificador VPC da pilha associada. AWS OpsWorks O Stacks armazena seus pacotes de dados nas instâncias de cada pilha. Em instâncias Linux, o AWS OpsWorks Stacks armazena pacotes de dados no `/var/chef/runs/`*`run-ID`*`/data_bags` diretório. Nas instâncias do Linux, ele armazena recipientes de dados no diretório *`drive`*`:\chef\runs\`*`run-id`*`\data_bags`. Em ambos os casos, *`run-ID`* é um ID exclusivo que o AWS OpsWorks Stacks atribui a cada execução do Chef em uma instância. Esses diretórios incluem um conjunto de data bags (subdiretórios). Cada data bag contém zero ou mais itens de data bag, que são arquivos formatados pelo JSON que contêm conjuntos de conteúdo de data bag.

> **Note**
>
> AWS OpsWorks O Stacks não oferece suporte a pacotes de dados criptografados. Para armazenar dados confidenciais de maneira criptografada, como senhas ou certificados, recomendamos armazená-los em um bucket do S3 privado. Você pode então criar uma receita personalizada que usa o Amazon SDK para Ruby para recuperar os dados. Para obter um exemplo, consulte Usar o SDK for Ruby.

O conteúdo de data bag pode incluir:

- Conteúdo String, que segue a sintaxe padrão do Ruby e pode usar aspas duplas ou simples, embora as strings com determinados caracteres especiais devam ter aspas duplas. Para obter mais informações, consulte o site de documentação do Ruby.
- Conteúdo Booliano, que é `true` ou `false` (sem aspas).
- Conteúdo Número, que é um número inteiro ou decimal; por exemplo, `4` ou `2.5` (sem aspas).
- Conteúdo Lista, que assume a forma de valores separados por vírgula dentro de colchetes (sem aspas); por exemplo, `[ '80', '443' ]`
- Objetos JSON, que possuem conteúdo de data bag adicional, como `"my-app": {"elastic_ip": null,...}`.

As receitas do Chef podem acessar data bags, itens de data bag e conteúdo de data bag por meio da pesquisa do Chef ou diretamente. Os dados a seguir descrevem como usar as duas abordagens de acesso (embora a pesquisa do Chef seja preferencial).

Para acessar uma bolsa de dados por meio da pesquisa do Chef, use o método de pesquisa, especificando o índice de pesquisa desejado. AWS OpsWorks O Stacks fornece os seguintes índices de pesquisa:

- aws_opsworks_app, que representa um conjunto de aplicativos implantados de uma pilha.
- aws_opsworks_command, que representa um conjunto de comandos que foram executados em uma pilha.
- aws_opsworks_ecs_cluster, que representa um conjunto de instâncias de cluster do Amazon Elastic Container Service (Amazon ECS) de uma pilha.
- aws_opsworks_elastic_load_balancer, que representa um conjunto de balanceadores de carga do Elastic Load Balancing de uma pilha.
- aws_opsworks_instance, que representa um conjunto de instâncias de uma pilha.
- aws_opsworks_layer, que representa um conjunto de camadas de uma pilha.
- aws_opsworks_rds_db_instance, que representa um conjunto de instâncias do Amazon Relational Database Service (Amazon RDS) de uma pilha.
- aws_opsworks_stack, que representa uma pilha.
- aws_opsworks_user, que representa um conjunto de usuários de uma pilha.

Assim que você souber o nome do índice de pesquisa, acesse o conteúdo do data bag desse índice de pesquisa. Por exemplo, o código de receita a seguir usa o índice de pesquisa

`aws_opsworks_app` para obter o conteúdo do primeiro item de data bag (o primeiro arquivo JSON) no data bag`aws_opsworks_app` (o diretório `aws_opsworks_app`). Em seguida, o código grava duas mensagens no log do Chef, uma com o conteúdo de data bag de nome curto do aplicativo (uma string no arquivo JSON) e outra com o conteúdo de data bag de URL de origem do aplicativo (outra string no arquivo JSON):

```
app = search("aws_opsworks_app").first
Chef::Log.info("********** The app's short name is '#{app['shortname']}' **********")
Chef::Log.info("********** The app's URL is '#{app['app_source']['url']}' **********")
```

Em que `['shortname']` e `['app_source']['url']` especificam o conteúdo de data bag a seguir no arquivo JSON correspondente:

```
{
  ...
  "shortname": "mylinuxdemoapp",
  ...
  "app_source": {
    ...
    "url": "https://s3.amazonaws.com/opsworks-demo-assets/opsworks-linux-demo-
nodejs.tar.gz",
  },
  ...
}
```

Para obter uma lista do conteúdo de data bag que você pode pesquisar, consulte os tópicos de referência nesta seção.

Você também pode percorrer um conjunto de itens de data bag em um data bag. Por exemplo, o código de receita a seguir é semelhante ao exemplo anterior; ele percorre cada um dos itens de data bag no data bag quando há mais de um item de data bag:

```
search("aws_opsworks_app").each do |app|
  Chef::Log.info("********** The app's short name is '#{app['shortname']}' **********")
  Chef::Log.info("********** The app's URL is '#{app['app_source']['url']}'
 **********")
end
```

Se você souber que existe um conteúdo de data bag específico, poderá encontrar o item de data bag correspondente com a seguinte sintaxe:

```
search("search_index", "key:value").first
```

Por exemplo, o código de receita a seguir usa o índice de pesquisa `aws_opsworks_app` para encontrar o item de data bag que contém o nome curto de aplicativo `mylinuxdemoapp`. Em seguida, ele usa o conteúdo do item de data bag para gravar uma mensagem no log do Chef com o nome curto e o URL de origem do aplicativo correspondente:

```
app = search("aws_opsworks_app", "shortname:mylinuxdemoapp").first
Chef::Log.info("********** For the app with the short name '#{app['shortname']}', the
 app's URL is '#{app['app_source']['url']}' **********")
```

Para o índice de pesquisa `aws_opsworks_instance` somente, você pode especificar `self:true` para representar a instância em que a receita está sendo executada. O código de receita a seguir usa o conteúdo do item do pacote de dados correspondente para gravar uma mensagem no registro do Chef com o ID e o sistema operacional AWS OpsWorks gerados pelo Stacks da instância correspondente:

```
instance = search("aws_opsworks_instance", "self:true").first
Chef::Log.info("********** For instance '#{instance['instance_id']}', the instance's
 operating system is '#{instance['os']}' **********")
```

Em vez de usar a pesquisa do Chef para acessar data bags, itens de data bag e conteúdo de data bag, você pode acessá-los diretamente. Para fazer isso, use os métodos data_bag e data_bag_item para acessar data bags e itens de data bags, respectivamente. Por exemplo, o código de receita a seguir executa a mesma função demonstrada nos exemplos anteriores; a diferença é que ele acessa diretamente um único item de data bag e, depois, vários itens de data bag quando há mais de um:

```
# Syntax: data_bag_item("the data bag name", "the file name in the data bag without the
 file extension")
app = data_bag_item("aws_opsworks_app", "mylinuxdemoapp")
Chef::Log.info("********** The app's short name is '#{app['shortname']}' **********")
Chef::Log.info("********** The app's URL is '#{app['app_source']['url']}' **********")

data_bag("aws_opsworks_app").each do |data_bag_item|
  app = data_bag_item("aws_opsworks_app", data_bag_item)
  Chef::Log.info("********** The app's short name is '#{app['shortname']}' **********")
  Chef::Log.info("********** The app's URL is '#{app['app_source']['url']}'
 **********")
end
```

Entre essas duas abordagens, recomendamos que você use a pesquisa do Chef. Todos os exemplos relacionados neste guia demonstram essa abordagem.

Tópicos

- Data bag do aplicativo (aws_opsworks_app)
- Data bag do comando (aws_opsworks_command)
- Data bag do cluster do Amazon ECS (aws_opsworks_ecs_cluster)
- Data Bag do Elastic Load Balancing (aws_opsworks_elastic_load_balancer)
- Data bag da instância (aws_opsworks_instance)
- Data bag de camada (aws_opsworks_layer)
- Data bag do Amazon RDS (aws_opsworks_rds_db_instance)
- Data bag da pilha (aws_opsworks_stack)
- Data bag do usuário (aws_opsworks_user)

Data bag do aplicativo (aws_opsworks_app)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Para um evento Deploy ou um comando de pilha Executar receitas, representa as configurações de um aplicativo.

O exemplo a seguir mostra como usar a pesquisa do Chef para pesquisar um único item do pacote de dados e, em seguida, vários itens do saco de dados para gravar mensagens no log do Chef com os nomes abreviados e a fonte URLs dos aplicativos:

```
app = search("aws_opsworks_app").first
Chef::Log.info("********** The app's short name is '#{app['shortname']}' **********")
Chef::Log.info("********** The app's URL is '#{app['app_source']['url']}' **********")
```

```
search("aws_opsworks_app").each do |app|
  Chef::Log.info("********** The app's short name is '#{app['shortname']}' **********")
  Chef::Log.info("********** The app's URL is '#{app['app_source']['url']}'
 **********")
end
```

app_id	app_source	data_sources
implantar	attributes	domínios
enable_ssl	ambiente	nome
shortname	ssl_configuration	type

app_id

O ID do aplicativo (string). Um GUID que identifica o aplicativo.

app_source

Um conjunto de conteúdo que especifica as informações que o AWS OpsWorks Stacks usa para implantar o aplicativo a partir de seu repositório de controle de origem. O conteúdo varia de acordo com o tipo de repositório.

password

A senha para repositórios privados e `"null"` para repositórios públicos (string). Para buckets do S3 privados, este conteúdo é definido como a chave secreta.

revisão

Se o repositório tiver várias ramificações, o conteúdo especificará a ramificação ou a versão do aplicativo; por exemplo, `"version1"` (string). Caso contrário, ele será definido como `"null"`.

ssh_key

Uma chave SSH de implantação para acesso a repositórios Git privados e `"null"` para repositórios públicos (string).

type

O local de origem do aplicativo (string). Os valores válidos são:

- `"archive"`
- `"git"`
- `"other"`
- `"s3"`

url

A origem em que o aplicativo está localizado (string).

usuário

O nome de usuário para repositórios privados e `"null"` para repositórios públicos (string). Para buckets S3 privados, o conteúdo é definido como a chave de acesso.

attributes

Um conjunto de conteúdo que descreve a estrutura de diretórios e o conteúdo do aplicativo.

document_root

O diretório raiz da árvore de documentos. Define o caminho para a raiz do documento (ou o local da página inicial do aplicativo, como `home_html`), que é relativa ao diretório de implantação. A não ser que este atributo seja especificado, o document_root adotará `public` como valor padrão. O valor de `document_root` pode começar apenas com os caracteres `a-z`, `A-Z`, `0-9`, _ (sublinhado) ou `-` (hífen).

data_sources

As informações necessárias para conexão ao banco de dados do aplicativo. Se o aplicativo tiver uma camada de banco de dados anexada, o AWS OpsWorks Stacks atribuirá automaticamente os valores apropriados a esse conteúdo.

O valor de data_sources é uma matriz, e as matrizes são acessadas por um offset integral, e não por chave. Por exemplo, para acessar a primeira fonte de dados do aplicativo, use `app[:data_sources][0][:type]`.

database_name

O nome do banco de dados, que geralmente é o nome curto do aplicativo (string).

type

O tipo da instância do banco de dados, normalmente `"RdsDbInstance"` (string).

arn

O Nome de recurso da Amazon (ARN) da instância do banco de dados (string).

implantar

Se o aplicativo deve ser implantado ou não (Booliano). `true` para aplicativos que devem ser implantados em um evento de ciclo de vida Deploy. Em um evento de ciclo de vida Setup, este conteúdo será `true` para todos os aplicativos. Para determinar quais aplicativos devem ser implantados em uma instância, verifique as camadas às quais a instância pertence.

domínios

Uma lista dos domínios do aplicativo (lista de strings).

enable_ssl

Se o suporte ao SSL está ativado ou não (Booliano).

ambiente

Uma coleção de variáveis de ambiente especificadas pelo usuário que foram definidas para o aplicativo. Para obter mais informações sobre como definir as variáveis do ambiente de uma aplicação, consulte Adição de aplicativos. Cada nome de conteúdo é definido como um nome de variável de ambiente, e o valor correspondente é definido como o valor da variável.

nome

O nome do aplicativo, que é usado para fins de exibição (string).

shortname

O nome curto do aplicativo, que é gerado pelo AWS OpsWorks Stacks a partir do nome (string). O nome curto é usado internamente e pelas receitas; ele é usado como o nome do diretório em que os arquivos de aplicativo são instalados.

ssl_configuration

certificado

Se você ativou o suporte a SSL, este será o certificado SSL do aplicativo; do contrário, ele será `"null"` (string).

chain

Se o SSL estiver ativado, este será o conteúdo para especificar uma chave de autoridade de certificado intermediária ou a autenticação do cliente (string).

private_key

Se você ativou o suporte a SSL, este será a chave privada SSL do aplicativo; do contrário, ele será `"null"` (string).

type

O tipo do aplicativo, que sempre é definido como `"other"` para as pilhas Chef 12 no Linux e Chef 12.2 no Windows (string).

Data bag do comando (aws_opsworks_command)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Representa as configurações de um comando que o AWS OpsWorks Stacks executa em uma ou mais instâncias.

O exemplo a seguir mostra como usar a pesquisa do Chef para pesquisar um único item de data bag e, depois, vários itens de data bag, a fim de gravar mensagens no log do Chef com os tipos dos comandos e quando eles foram enviados:

```
command = search("aws_opsworks_command").first
Chef::Log.info("********** The command's type is '#{command['type']}' **********")
Chef::Log.info("********** The command was sent at '#{command['sent_at']}' **********")

search("aws_opsworks_command").each do |command|
  Chef::Log.info("********** The command's type is '#{command['type']}' **********")
  Chef::Log.info("********** The command was sent at '#{command['sent_at']}'
 **********")
end
```

[args](#)	[command_id](#)	[iam_user_arn](#)

instance_id	sent_at	type

args

Argumentos do comando (string).

command_id

O identificador exclusivo aleatório do comando, atribuído por AWS OpsWorks Stacks (string).

iam_user_arn

Se o comando for criado pelo cliente, o Nome de recurso da Amazon (ARN) do usuário que criou o comando (string).

instance_id

O identificador da instância em que o comando foi executado (string).

sent_at

A data e hora de quando o AWS OpsWorks Stacks executou o comando (string).

type

O tipo do comando (string). Os valores válidos são:

- `"configure"`
- `"deploy"`
- `"deregister"`
- `"execute_recipes"`
- `"grant_remote_access"`
- `"install_dependencies"`
- `"restart"`
- `"revoke_remote_access"`
- `"rollback"`
- `"setup"`
- `"shutdown"`

- "start"
- "stop"
- "sync_remote_users"
- "undeploy"
- "update_agent"
- "update_custom_cookbooks"
- "update_dependencies"

Data bag do cluster do Amazon ECS (aws_opsworks_ecs_cluster)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Representa as configurações de um cluster do Amazon ECS.

O exemplo a seguir mostra como usar a pesquisa do Chef para pesquisar um único item do pacote de dados e, em seguida, vários itens do saco de dados para gravar mensagens no log do Chef com os nomes dos clusters do Amazon ECS e os nomes dos recursos da Amazon (): ARNs

```
ecs_cluster = search("aws_opsworks_ecs_cluster").first
Chef::Log.info("********** The ECS cluster's name is
 '#{ecs_cluster['ecs_cluster_name']}' **********")
Chef::Log.info("********** The ECS cluster's ARN is '#{ecs_cluster['ecs_cluster_arn']}'
 **********")

search("aws_opsworks_ecs_cluster").each do |ecs_cluster|
  Chef::Log.info("********** The ECS cluster's name is
 '#{ecs_cluster['ecs_cluster_name']}' **********")
  Chef::Log.info("********** The ECS cluster's ARN is
 '#{ecs_cluster['ecs_cluster_arn']}' **********")
end
```

ecs_cluster_arn	ecs_cluster_name

ecs_cluster_arn

O Nome de recurso da Amazon (ARN) do cluster (string).

ecs_cluster_name

O nome do cluster (string).

Data Bag do Elastic Load Balancing (aws_opsworks_elastic_load_balancer)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Representa as configurações do balanceador de carga do Elastic Load Balancing.

O exemplo a seguir mostra como usar a pesquisa do Chef para pesquisar um único item de data bag e, depois, vários itens de data bag, a fim de gravar mensagens no log do Chef com os nomes dos balanceadores de carga do Elastic Load Balancing e os nomes DNS:

```
elastic_load_balancer = search("aws_opsworks_elastic_load_balancer").first
Chef::Log.info("********** The ELB's name is
 '#{elastic_load_balancer['elastic_load_balancer_name']}' **********")
Chef::Log.info("********** The ELB's DNS name is '#{elastic_load_balancer['dns_name']}'
 **********")

search("aws_opsworks_elastic_load_balancer").each do |elastic_load_balancer|
  Chef::Log.info("********** The ELB's name is
 '#{elastic_load_balancer['elastic_load_balancer_name']}' **********")
  Chef::Log.info("********** The ELB's DNS name is
 '#{elastic_load_balancer['dns_name']}' **********")
```

```
end
```

elastic_load_balancer_name	dns_name	layer_id

elastic_load_balancer_name

O nome do load balancer (string).

dns_name

O nome DNS do load balancer (string).

layer_id

O ID das AWS OpsWorks pilhas da camada à qual o balanceador de carga está atribuído (string).

Data bag da instância (aws_opsworks_instance)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Representa as configurações de uma instância.

O exemplo a seguir mostra como usar a pesquisa do Chef para pesquisar um único item do pacote de dados e, em seguida, vários itens do saco de dados para gravar mensagens no log do Chef com os nomes de host das instâncias e: IDs

```
instance = search("aws_opsworks_instance").first
Chef::Log.info("********** The instance's hostname is '#{instance['hostname']}'
 **********")
Chef::Log.info("********** The instance's ID is '#{instance['instance_id']}'
 **********")
```

```
search("aws_opsworks_instance").each do |instance|
  Chef::Log.info("********** The instance's hostname is '#{instance['hostname']}'
 **********")
  Chef::Log.info("********** The instance's ID is '#{instance['instance_id']}'
 **********")
end
```

O exemplo a seguir mostra maneiras diferentes de usar a pesquisa do Chef para pesquisar vários itens do saco de dados para encontrar o item do saco de dados que contém o ID de EC2 instância da Amazon especificado. Em seguida, o exemplo usa o conteúdo do item de data bag para gravar uma mensagem no log do Chef com o endereço IP público da instância correspondente:

```
instance = search("aws_opsworks_instance", "ec2_instance_id:i-12345678").first
Chef::Log.info("********** For instance '#{instance['ec2_instance_id']}', the
 instance's public IP address is '#{instance['public_ip']}' **********")

search("aws_opsworks_instance").each do |instance|
  if instance['ec2_instance_id'] == 'i-12345678'
    Chef::Log.info("********** For instance '#{instance['ec2_instance_id']}', the
 instance's public IP address is '#{instance['public_ip']}' **********")
  end
end
```

O exemplo a seguir mostra como usar a pesquisa do Chef com `self:true` para encontrar o item de data bag que contém informações relacionadas à instância em que a receita está sendo executada. Em seguida, o exemplo usa o conteúdo do item do saco de dados para gravar uma mensagem no registro do Chef com o ID AWS OpsWorks gerado pelo Stacks da instância correspondente e o endereço IP público da instância:

```
instance = search("aws_opsworks_instance", "self:true").first
Chef::Log.info("********** For instance '#{instance['instance_id']}', the instance's
 public IP address is '#{instance['public_ip']}' **********")
```

ami_id	arquitetura	auto_scaling_type
availability_zone	created_at	ebs_optimized
ec2_instance_id	elastic_ip	hostname

instance_id	instance_type	layer_ids
os	private_dns	private_ip
public_dns	public_ip	root_device_type
root_device_volume_id	self	ssh_host_dsa_key_fingerprint
ssh_host_dsa_key_private	ssh_host_dsa_key_public	ssh_host_rsa_key_fingerprint
ssh_host_rsa_key_private	ssh_host_rsa_key_public	status
subnet_id	virtualization_type	

ami_id

O ID da AMI (Imagem de máquina da Amazon) da instância (string).

arquitetura

A arquitetura da instância, que é sempre definida como `"x86_64"` (string).

auto_scaling_type

O tipo de escalabilidade da instância: `null`, `timer` ou `load` (string).

availability_zone

A Zona de disponibilidade (AZ) da instância, como `"us-west-2a"` (string).

created_at

A hora em que a instância foi criada, usando o formato UTC `"yyyy-mm-dddThh:mm:ss+hh:mm"` (string). Por exemplo, `"2013-10-01T08:35:22+00:00"` corresponde a 08:35:22, em 1 de out. de 2013, sem diferença de fuso horário. Para obter mais informações, consulte ISO 8601.

ebs_optimized

Se a instância é otimizada pelo EBS (Booliano).

ec2_instance_id

O ID da EC2 instância (string).

elastic_ip

O endereço IP elástico; definido como `"null"` se a instância não tiver um endereço IP elástico (string).

hostname

O nome do host, como `"demo1"` (string).

instance_id

O ID da instância, que é um GUID AWS OpsWorks gerado pelo Stacks que identifica exclusivamente a instância (string).

instance_type

O tipo de instância, como `"c1.medium"` (string).

layer_ids

Uma lista das camadas da instância, identificadas por sua exclusividade IDs; por exemplo,`307ut64c-c7e4-40cc-52f0-67d5k1f9992c`.

os

O sistema operacional da instância (string). Os valores válidos são:

- `"Amazon Linux 2"`
- `"Amazon Linux 2018.03"`
- `"Amazon Linux 2017.09"`
- `"Amazon Linux 2017.03"`
- `"Amazon Linux 2016.09"`
- `"Custom"`
- `"Microsoft Windows Server 2022 Base"`
- `"Microsoft Windows Server 2022 with SQL Server Express"`
- `"Microsoft Windows Server 2022 with SQL Server Standard"`
- `"Microsoft Windows Server 2022 with SQL Server Web"`
- `"Microsoft Windows Server 2019 Base"`
- `"Microsoft Windows Server 2019 with SQL Server Express"`
- `"Microsoft Windows Server 2019 with SQL Server Standard"`

- `"Microsoft Windows Server 2019 with SQL Server Web"`
- `"CentOS 7"`
- `"Red Hat Enterprise Linux 7"`
- `"Ubuntu 20.04 LTS"`
- `"Ubuntu 18.04 LTS"`
- `"Ubuntu 16.04 LTS"`
- `"Ubuntu 14.04 LTS"`

private_dns

O nome DNS privado (string).

private_ip

O endereço IP privado (string).

public_dns

O nome DNS público (string).

public_ip

O endereço IP público (string).

root_device_type

O tipo de dispositivo raiz (string). Os valores válidos são:

- `"ebs`
- `"instance-store"`

root_device_volume_id

O ID de volume do dispositivo raiz (string).

self

`true` se este item de data bag contiver informações sobre a instância em que a receita está sendo executada; caso contrário, `false` (Booliano). Esse valor está disponível somente para receitas, não por meio da API AWS OpsWorks Stacks.

ssh_host_dsa_key_fingerprint

Uma sequência de bytes mais curta que identifica a chave pública DSA mais longa (string).

ssh_host_dsa_key_private

A chave privada gerada por DSA para autenticação SSH com a instância (string).

ssh_host_dsa_key_public

A chave pública gerada por DSA para autenticação SSH com a instância (string).

ssh_host_rsa_key_fingerprint

Uma sequência de bytes mais curta que identifica a chave pública RSA mais longa (string).

ssh_host_rsa_key_private

A chave privada gerada por RSA para autenticação SSH com a instância (string).

ssh_host_rsa_key_public

A chave pública gerada por RSA para autenticação SSH com a instância (string).

status

O status da instância (string). Os valores válidos são:

- `"requested"`
- `"booting"`
- `"running_setup"`
- `"online"`
- `"setup_failed"`
- `"start_failed"`
- `"terminating"`
- `"terminated"`
- `"stopped"`
- `"connection_lost"`

subnet_id

O ID de sub-rede da instância (string).

virtualization_type

O tipo de virtualização da instância (string).

Data bag de camada (aws_opsworks_layer)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Representa as configurações de uma camada.

O exemplo a seguir mostra como usar a pesquisa do Chef para pesquisar um único item de data bag e, depois, vários itens de data bag para gravar mensagens no log do Chef com os nomes e os nomes curtos das camadas:

```
layer = search("aws_opsworks_layer").first
Chef::Log.info("********** The layer's name is '#{layer['name']}' **********")
Chef::Log.info("********** The layer's shortname is '#{layer['shortname']}'
 **********")

search("aws_opsworks_layer").each do |layer|
  Chef::Log.info("********** The layer's name is '#{layer['name']}' **********")
  Chef::Log.info("********** The layer's shortname is '#{layer['shortname']}'
 **********")
end
```

ecs_cluster_arn	layer_id	nome
packages	shortname	type
volume_configurations		

ecs_cluster_arn

Se a camada tem um cluster do Amazon ECS atribuído, o Nome do recurso da Amazon (ARN) do cluster do Amazon ECS (string).

criptografado

`true` se o volume do EBS é criptografado; caso contrário, `false` (Booliano).

layer_id

O ID da camada, que é um GUID gerado pelo AWS OpsWorks Stacks e que identifica exclusivamente a camada (string).

nome

O nome da camada, que é usado para representar a camada do console (string). Ele pode ser definido pelo usuário e não precisa ser exclusivo.

packages

Uma lista de pacotes a ser instalado (lista de strings).

shortname

O nome curto da camada, que é definido pelo usuário (string).

type

O tipo da camada, que sempre é definido como `"custom"` para Chef 12 Linux e Chef 12.2 Windows (string).

volume_configurations

Uma lista de configurações de volume do Amazon EBS.

iops

O número de operações de E/S por segundo ao qual o volume oferece suporte.

mount_point

O diretório do ponto de montagem do volume.

number_of_disks

O número de discos do volume.

raid_level

O nível de configuração de RAID do volume.

size

O tamanho do volume em GiB.

volume_type

O tipo do volume: uso geral, magnético, IOPS provisionadas, disco rígido com throughput otimizado ou disco rígido frio.

Data bag do Amazon RDS (aws_opsworks_rds_db_instance)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Um conjunto de conteúdo de data bag que especifica a configuração de uma instância do Amazon Relational Database Service (Amazon RDS) da seguinte forma:

address	db_instance_identifier	db_password
db_user	engine	rds_db_instance_arn
região		

O exemplo a seguir mostra como usar a pesquisa do Chef para pesquisar um único item de data bag e, depois, vários itens de data bag, a fim de gravar mensagens no log do Chef com os endereços das instâncias do Amazon RDS e os tipos de mecanismo de banco de dados:

```
rds_db_instance = search("aws_opsworks_rds_db_instance").first
Chef::Log.info("********** The RDS instance's address is
 '#{rds_db_instance['address']}' **********")
Chef::Log.info("********** The RDS instance's database engine type is
 '#{rds_db_instance['engine']}' **********")

search("aws_opsworks_rds_db_instance").each do |rds_db_instance|
  Chef::Log.info("********** The RDS instance's address is
 '#{rds_db_instance['address']}' **********")
```

```
  Chef::Log.info("********** The RDS instance's database engine type is
 '#{rds_db_instance['engine']}' **********")
end
```

address

O nome DNS da instância.

porta

A porta da instância.

db_instance_identifier

O ID da instância.

db_password

A senha principal da senha.

db_user

O nome de usuário principal da instância.

engine

O mecanismo de banco de dados da instância, como `mysql`.

rds_db_instance_arn

O Nome de recurso da Amazon (ARN) da instância.

região

A região da AWS da instância, como `us-west-2`.

Data bag da pilha (aws_opsworks_stack)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Representa as configurações de uma pilha.

O exemplo a seguir mostra como usar o Chef de pesquisa para gravar mensagens no log do Chef com o nome da pilha e o URL de origem do livro de receitas:

```
stack = search("aws_opsworks_stack").first
Chef::Log.info("********** The stack's name is '#{stack['name']}' **********")
Chef::Log.info("********** The stack's cookbook URL is
 '#{stack['custom_cookbooks_source']['url']}' **********")
```

arn	custom_cookbooks_source	nome
região	stack_id	use_custom_cookbooks
vpc_id		

arn

O Nome de recurso da Amazon (ARN) da pilha (string).

custom_cookbooks_source

Um conjunto de conteúdo que especifica o repositório de origem do livro de receita personalizado.

type

O tipo de repositório (string). Os valores válidos são:

- `"archive"`
- `"git"`
- `"s3"`

url

O URL do repositório, como `"git://github.com/amazonwebservices/opsworks-demo-php-simple-app.git"` (string).

username

O nome de usuário para repositórios privados e `null` para repositórios públicos (string). Para buckets privados do Amazon Simple Storage Service (Amazon S3), o conteúdo é definido como a chave de acesso.

password

A senha para repositórios privados e `null` para repositórios públicos (string). Para buckets S3 privados, este conteúdo é definido como a chave secreta.

ssh_key

Uma chave SSH de implantação para acesso a repositórios Git privados e `null` para repositórios públicos (string).

revisão

Se o repositório tiver várias ramificações, o conteúdo especificará a ramificação ou a versão do aplicativo; por exemplo, `"version1"` (string). Caso contrário, ele será definido como `null`.

nome

O nome da pilha (string).

região

A região da AWS da pilha (string).

stack_id

Um GUID que identifica a pilha (string).

use_custom_cookbooks

Se os livros de receitas personalizados estão ativados (Booliano).

vpc_id

Se a pilha estiver em execução em uma VPC, o ID da VPC, se a pilha estiver em execução em uma VPC (string).

Data bag do usuário (aws_opsworks_user)

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver

dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Representa as configurações de um usuário.

O exemplo a seguir mostra como usar a pesquisa do Chef para pesquisar um único item do pacote de dados e, em seguida, vários itens do saco de dados para gravar mensagens no log do Chef com os nomes de usuário dos usuários e os nomes de recursos da Amazon (ARNs):

```
user = search("aws_opsworks_user").first
Chef::Log.info("********** The user's user name is '#{user['username']}' **********")
Chef::Log.info("********** The user's user ARN is '#{user['iam_user_arn']}'
 **********")

# Or...

search("aws_opsworks_user").each do |user|
  Chef::Log.info("********** The user's user name is '#{user['username']}' **********")
  Chef::Log.info("********** The user's user ARN is '#{user['iam_user_arn']}'
 **********")
end
```

administrator_privileges	iam_user_arn	remote_access
ssh_public_key	unix_user_id	username

administrator_privileges

Se o usuário tem privilégios de administrador (Booliano).

iam_user_arn

O Nome de recurso da Amazon (ARN) do usuário (string).

remote_access

Se o usuário pode usar o RDP para fazer login na instância (Booliano).

ssh_public_key

A chave pública do usuário, conforme fornecida pelo console do AWS OpsWorks Stacks ou pela API (string).

unix_user_id

O ID do Unix do usuário (número).

username

O nome do usuário (string).

OpsWorks Mudanças no agente

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Versões do agente do Chef 12

A tabela a seguir descreve mudanças importantes no agente Chef 12 que o AWS OpsWorks Stacks instala nas instâncias que ele gerencia.

Versão do agente	Descrição	Data de lançamento
4042	• Esta versão do agente contém apenas pequenas alterações sem novos recursos	7 de fevereiro de 2023
4041	• Esta versão do agente contém apenas pequenas alterações sem novos recursos • Atualizar certificados Amazon CA	27 de janeiro de 2023

Versão do agente	Descrição	Data de lançamento
4040	• Esta versão do agente contém apenas pequenas alterações sem novos recursos	22 de julho de 2022
4039	• Corrigir a integração do ECS para o Ubuntu AMIs	30 de abril de 2020
4038	• Corrigir erro ao enviar estatísticas de instância durante a alteração do DST • Respeitar a variável de ambiente `no_proxy` durante o download e a instalação do agente	5 de março de 2020
4037	• Adicione suporte para solicitações de assinatura no S3 URLs sem região usando SigV4 • Remover o suporte para assinar solicitações do S3 usando SigV2	4 de junho de 2019
4035	• Corrigir bug durante a configuração do ECS • Corrigir entradas `fstab` duplicadas após uma alteração no tipo de instância	8 de maio de 2019
4033	• Adicionar suporte para Ubuntu 18.04 • Corrigir bug de instalação do agente no Amazon Linux 2	26 de novembro de 2018
4032	• Suporte adicionado para o Amazon Linux 2	24 de outubro de 2018
4031	• Adicionar suporte para o Amazon Linux 2018.03 • Dar suporte a arquivos públicos do S3 hospedados em outra conta	15 de agosto de 2018
4030	• Corrigir o gerenciamento de volume para instâncias c5d	31 de maio de 2018

Versão do agente	Descrição	Data de lançamento
4029	• Instalar `nvme-cli` no Ubuntu 14.04 • Corrigir a montagem do volume em instâncias c5, m5 • Sempre preservar o hostname na reinicialização	2 de maio de 2018
4028	• Corrigir a configuração `monit` para CentOS	20 de março de 2018
4027	• Support para montagem de NVMe volumes no Ubuntu 14.04 (`nvme-cli`deve ser instalado manualmente) • Não exigem a propriedade `name` para volumes	17 de fevereiro de 2018
4026	• Monte volumes do EBS NVMe baseados usando o ID do volume do EBS • Corrigir a montagem do volume do EBS em instâncias i3 • Corrigir a ordem dos volumes do EBS montados em instâncias c5, m5	31 de janeiro de 2018
4025	• Corrigir o manuseio de NVMe dispositivos	13 de dezembro de 2017
4024	• Adicionar suporte para o Amazon Linux 2017.09	5 de dezembro de 2017
4023	• Adicionar suporte para integração CloudWatch de registros	2 de abril de 2017
4022	• Atualizar a versão do cliente do Chef para 12.18.31	1 de fevereiro de 2017
4021	• Melhorar tratamento de proxy	16 de dezembro de 2016

Versão do agente	Descrição	Data de lançamento
4020	• Atualizar a versão do cliente do Chef para 12.16.42	8 de dezembro de 2016
4019	• Encontrar variáveis de proxy durante a instalação do agente • O Red Hat Enterprise Linux 7 já usa `systemd`, em vez de `monit` • Não configurar EPEL no Red Hat Enterprise Linux 7 • Usar `flock`, em vez de `lockrun.c` para processar o bloqueio • Evitar a saída diferente de `ps -p1` durante a verificação de `systemd`	19 de outubro de 2016
4018	• Atualizar a versão do cliente do Chef para 12.13.37 • Adicionar suporte para o Amazon Linux 2016.09	25 de agosto de 2016
4017	• Atualizar a versão do cliente do Chef para 12.12.15	10 de agosto de 2016
4016	• Corrigir a desinstalação do agente em sistemas em que `monit` não seja usado	23 de junho de 2016
4015	• Corrigir a configuração do ECS para o Amazon Linux 2016.03	17 de junho de 2016
4011	• Atualizar a versão do cliente do Chef para 12.10.24 • Melhorar o processamento de upload do log	19 de maio de 2016
4008	• Adicionar suporte para o Amazon Linux 2016.03 • Adicionar tempo limite para a instalação do pacote • Adicione xfs to /etc/filesystems se existir	16 de março de 2016

Versão do agente	Descrição	Data de lançamento
4007	• Atualizar a versão do cliente do Chef para 12.7.2 • Melhorias feitas no tratamento de erros para instâncias no local (servidores hospedados fora da AWS) • Melhorar a compatibilidade com o segredo mais recente do chef • Repetir download do arquivo para implantação	4 de março de 2016
4006	• Atualizar a versão do cliente do Chef para 12.6.0 • Não instale pacotes libxml2- devel/libxml2-dev and libxslt-devel/libxslt -dev na instalação do agente	21 de janeiro de 2016
4005	• Corrigir a importação de ec2 permitindo sempre os dados de ec2 em ohai para a infraestrutura de ec2	17 de dezembro de 2015
4004	• AWS OpsWorks Suporte de pilhas para Chef 12 Linux- Chef Client 12.5.1	3 de dezembro de 2015

Versões do agente do Chef 11.10

A tabela a seguir descreve mudanças importantes no agente Chef 11.10 que o AWS OpsWorks Stacks instala nas instâncias que ele gerencia.

Versão do agente	Descrição	Data de lançamento
3456	• Esta versão do agente contém apenas pequenas alterações sem novos recursos • Atualizar certificados Amazon CA	27 de janeiro de 2023
3455	• Esta versão do agente contém apenas pequenas alterações sem novos recursos	1º de novembro de 2022

Versão do agente	Descrição	Data de lançamento
3454	• Corrigir a integração do ECS para o Ubuntu AMIs	28 de abril de 2020
3453	• Corrigir erro ao enviar estatísticas de instância durante a alteração do DST • Corrigir erro de pacotes ausentes na RHEL7 configuração • Respeitar a variável de ambiente `no_proxy` durante o download e a instalação do agente	5 de março de 2020
3452	• Não incluir a região no URL do caminho virtual do Amazon S3 se ela for `us-east-1` • Extrair e fazer upload de livros de receitas internos para buckets específicos da região de estágio • Corrigir as entradas `fstab` do Chef 11.10 • Remover o uso de SigV2 para S3 e obter a região do bucket na solicitação	13 de agosto de 2019
3451	• Adicionar suporte para o Ruby 2.6.1	20 de março de 2019
3450	• Corrigir atributos padrão do EBS • Instalação CloudWatchLogs do agente de correção para Amazon Linux 2 • Corrigir instalação do bundler para versões de rubygem mais recentes que 2.6.14 • Corrigir suporte a arquivos públicos do S3	3 de dezembro de 2018
3449	• Corrigir o gerenciamento de volume para instâncias c5d • Corrige o suporte à matriz RAID em instâncias NVMe -device	5 de junho de 2018

Versão do agente	Descrição	Data de lançamento
3448	• Atualizar a versão padrão 2.3 do Ruby para 2.3.7 • Corrige a montagem de volumes do EBS em instâncias NVMe baseadas em instâncias do Ubuntu 14.04 • Dar suporte a arquivos públicos do Amazon S3 hospedados em outra conta • Corrigir problemas de inicialização de `opsworks-agent` em instâncias do Red Hat Enterprise Linux	8 de maio de 2018
3447	• Monte volumes do EBS NVMe baseados usando o ID do volume do EBS • Corrigir a montagem do volume do EBS em instâncias i3 • Corrigir a ordem dos volumes do EBS montados em c5, m5 • Atualizar a versão padrão 2.3 do Ruby para 2.3.6	31 de janeiro de 2018
3446	• Corrigir o manuseio de NVMe dispositivos • Atualizar a versão padrão 2.3 do Ruby para 2.3.5	14 de dezembro de 2017
3445	• Adicionar suporte para o Amazon Linux 2017.09 • Atualizar a versão padrão 2.2 do Ruby para 2.2.8	31 de outubro de 2017
3444	• Adicionar suporte para CloudWatch registros	1 de abril de 2017
3443	• Melhorar tratamento de proxy	15 de dezembro de 2016
3442	• Atualizar a versão padrão 2.3 do Ruby para 2.3.3 • Atualizar a versão padrão 2.2 do Ruby para 2.2.6	6 de dezembro de 2016
3441	• Encontrar variáveis de proxy durante a instalação do agente	21 de outubro de 2016

Versão do agente	Descrição	Data de lançamento
3440	• Adicionar suporte para o Amazon Linux 2016.09	13 de setembro de 2016
3439	• Pequenas alterações; sem novos recursos	29 de julho de 2016
3438	• Adicionar suporte para o Ruby 2.3.1 • Melhorar o registro da instância com as credenciais de perfil da instância do IAM • Remover restos de `s3curl.pl` • Corrigir a configuração do ECS para o Amazon Linux 2016.03	17 de junho de 2016
3437	• Atualizar a versão padrão 2.2 do Ruby para 2.2.5	4 de maio de 2016
3436	• Atualize o URL do EPEL para Red Hat Enterpris e Linux. IMPORTANTE: sem essa alteração, as instâncias do Red Hat Enterprise Linux não podem inicializar.	18 de abril de 2016
3435	• Atualizar a versão padrão 2.1 do Ruby para 2.1.9 • Melhorar o gerenciamento do Amazon S3 e arquivar implantações	6 de abril de 2016
3434	• Adicionar suporte para o Amazon Linux 2016.03 • Tentar instalações de pacote novamente	16 de março de 2016
3433	• Algumas melhorias para instâncias locais (servidor es hospedados fora do AWS) • Melhorar a compatibilidade com o `chef-sugar` mais recente • Repetir download do arquivo para implantação • Corrigir o URL de instalação de gems Ruby	27 de fevereiro de 2016

Versão do agente	Descrição	Data de lançamento
3432	• Melhorar o gerenciamento de caracteres especiais em nomes de bucket • Atualizar `s3_file` para a versão 2.6.6 • Ignore a montagem de volumes sem ponto de montagem especificado • Sempre reiniciar `unicorn` em vez de um interromper e iniciar para evitar o tempo de inatividade durante a implantação • Sempre atualizar o livro de receitas personalizado para o comando `setup` • Após a criação de matrizes RAID, atualize `initramfs` para evitar problemas de mapeamento de dispositivo na reinicialização	20 de janeiro de 2016
3431	• Problema de instalação do gem de `passenger` e `unicorn` corrigido na camada Rails • Atualização das versões padrão 2.0, 2.1 e 2.2 do Ruby para 2.0.0p648, 2.1.8 e 2.2.4 • Permitir que nomes de pacote `postgres` sejam definidos no JSON personalizado • Atualizar a versão padrão do Node.js para 0.12.9	22 de dezembro de 2015
3430	• Pequenas alterações; sem novos recursos	25 de novembro de 2015
3429	• Melhore a daemonização do OpsWorks agente (feche stdout/stderr) • Melhore a robustez do recurso `s3_file` (novas tentativas, exceções detectadas)	18 de novembro de 2015

Versão do agente	Descrição	Data de lançamento
3428	• Adicionar detecção de `postgres` adaptador com base no Gemfile corrige https://github.com/aws/opsworks-cookbooks/issues/136	17 de junho de 2016
3427	• Corrigido o problema na recuperação de credencia is no agente • Atualização das versões padrão 2.0, 2.1 e 2.2 do Ruby para 2.0.0p647, 2.1.7 e 2.2.3	11 de setembro de 2015
3426	• `aws-sdk` atualizado para 1.65.0 • Aprimoramento do download do Amazon S3 substituindo `s3curl` por `s3_file cookbook` • Alterar a versão padrão do Node.js para 0.12.7 • Registro adicionado para aplicativos do Node.js. STDOUT e STDERR registrados e alternados no diretório `shared/log` • Tornar explícita a atualização da verificação do submódulo do livro de receitas personalizado • Solução alternativa adicionada para https://g ithub.com/aws/ opsworks-cookbooks/issues/213 que verificará se as montagens de vinculação foram feitas antes da criação do diretório `deploy`	27 de agosto de 2015
3425	• Suporte do ECS para Amazon Linux e Ubuntu	27 de julho de 2015
3424	• Pequenas alterações; sem novos recursos	9 de julho de 2015
3422	• Suporte total do Red Hat Enterprise Linux 7 • Tornar a geração `/etc/hosts` mais resiliente a erros	29 de junho de 2015

Versão do agente	Descrição	Data de lançamento
3421	• Opção para substituir o nome do pacote do banco de dados do Red Hat Enterprise Linux 7 • Atualizada a configuração `monit systemd` para evitar que `systemd` envie o sinal `kill` a processos monitorados por `monit`	11 de junho de 2015

AWS OpsWorks Recursos do Stacks

> **Important**
>
> O AWS OpsWorks Stacks serviço chegou ao fim da vida útil em 26 de maio de 2024 e foi desativado para clientes novos e existentes. É altamente recomendável que os clientes migrem suas cargas de trabalho para outras soluções o mais rápido possível. Se você tiver dúvidas sobre migração, entre em contato com a AWS Support equipe no AWS re:POST ou por meio do Premium AWS Support.

Os recursos relacionados a seguir podem ajudar você à medida que trabalha com este serviço.

Guias de referência, Ferramentas e Recursos de suporte

Vários guias úteis, fóruns, informações de contato e outros recursos estão disponíveis no AWS OpsWorks Stacks e na Amazon Web Services.

- AWS OpsWorks Referência da Stacks API — descrições, sintaxe e exemplos de uso sobre ações e tipos de dados do AWS OpsWorks Stacks, incluindo parâmetros comuns e códigos de erro.
- AWS OpsWorks Perguntas frequentes técnicas do Stacks — Principais perguntas que os desenvolvedores fizeram sobre esse produto.
- Notas de versão do AWS OpsWorks Stacks: uma visão geral de alto nível da versão atual. Este documento menciona especificamente quaisquer recursos novos, correções e problemas conhecidos.
- AWS Tools for PowerShell — Um conjunto de PowerShell cmdlets do Windows que expõe a funcionalidade do AWS SDK para .NET no ambiente. PowerShell
- AWS Command Line Interface: uma sintaxe de linha de comando uniforme para acessar serviços da AWS. A CLI da AWS usa um único processo de configuração para fornecer acesso a todos os serviços suportados.
- AWS OpsWorks Referência de linha de comando do Stacks — Comandos AWS OpsWorks específicos do Stacks para uso em um prompt de linha de comando.

- Aulas e workshops — Links para cursos especializados e baseados em funções, além de laboratórios individualizados para ajudar a aprimorar suas AWS habilidades e ganhar experiência prática.
- AWS Centro do desenvolvedor — explore tutoriais, baixe ferramentas e saiba mais sobre eventos para AWS desenvolvedores.
- AWS Ferramentas do desenvolvedor — Links para ferramentas do desenvolvedor SDKs, kits de ferramentas do IDE e ferramentas de linha de comando para desenvolver e gerenciar AWS aplicativos.
- Centro de recursos de introdução — Saiba como configurar seu aplicativo Conta da AWS, participar da AWS comunidade e lançar seu primeiro aplicativo.
- Tutoriais práticos — Siga os tutoriais para iniciar seu step-by-step primeiro aplicativo no. AWS
- AWS Whitepapers — Links para uma lista abrangente de AWS white papers técnicos, abrangendo tópicos como arquitetura, segurança e economia e criados por arquitetos de AWS soluções ou outros especialistas técnicos.
- AWS Support Center — O hub para criar e gerenciar seus AWS Support casos. Também inclui links para outros recursos úteis, como fóruns, informações técnicas FAQs, status de integridade do serviço AWS Trusted Advisor e.
- Suporte— A principal página da web com informações sobre Suporte um one-on-one canal de suporte de resposta rápida para ajudá-lo a criar e executar aplicativos na nuvem.
- Entrar em contato: um ponto central de contato para consultas relativas a faturas da AWS , contas, eventos, uso abusivo e outros problemas.
- AWS Termos do site — Informações detalhadas sobre nossos direitos autorais e nossa marca registrada; sua conta, licença e acesso ao site; e outros tópicos.

AWS Kits de desenvolvimento de software

A Amazon Web Services fornece kits de desenvolvimento de software para acessar AWS OpsWorks Stacks de várias linguagens de programação diferentes. As bibliotecas do SDK automatizam um número de tarefas comuns, incluindo assinatura criptográfica de suas solicitações de serviço, novas tentativas de solicitações ou lidar com respostas de erro.

- AWS SDK para Java: configuração e outra documentação
- AWS SDK para .NET: configuração e outra documentação.
- AWS SDK para PHP: documentação

- AWS SDK para Ruby: documentação
- outra documentação
- AWS SDK for Python (Boto): configuração e outra documentação

Software de código aberto

AWS OpsWorks O Stacks inclui uma variedade de pacotes de software de código aberto, que são regidos por suas respectivas licenças. Para obter mais informações, consulte:

- Para instâncias do Chef 12 Linux, abra o arquivo `THIRD_PARTY_LICENSES` no diretório `/opt/aws/opsworks/current` na instância.
- Para o Chef 11.10 e versões anteriores para Linux, baixe o PDF do Documento de Atribuições do Agente OpsWorks Linux.

AWS OpsWorks histórico do documento

Alteração	Descrição	Data
Atualizações para AWS OpsWorks Stacks	Agora você pode usar a ferramenta Detach in Place para separar suas OpsWorks instâncias do serviço OpsWorks Stacks. Consulte Como usar a ferramenta AWS OpsWorks Stacks Detach in Place neste guia.	11 de abril de 2024
Atualizações para AWS OpsWorks Stacks	Agora você pode migrar suas AWS OpsWorks pilhas para o AWS Systems Manager Application Manager usando um script de migração. Para obter mais informações, consulte Migrando seus AWS OpsWorks Stacks aplicativos para o AWS Systems Manager Application Manager neste guia.	22 de dezembro de 2022
Atualizações para AWS OpsWorks for Chef Automate e AWS OpsWorks for Puppet Enterprise	Agora está disponível um procedimento de solução de problemas que descreve o que você pode fazer se a manutenção do sistema falhar no seu servidor AWS OpsWorks for Chef Automate ou OpsWorks no do Puppet Enterprise. Para obter mais informações, consulte Falha	29 de setembro de 2022

	na manutenção do sistema para o servidor Chef Automate ou Falha na manutenção do sistema para o servidor Puppet Enterprise neste guia.	
Atualizações para AWS OpsWorks for Chef Automate e AWS OpsWorks for Puppet Enterprise	Um procedimento de solução de problemas agora está disponível se o seu servidor AWS OpsWorks for Chef Automate ou OpsWorks o do Puppet Enterprise entrar em um `Connection lost` estado. Para obter mais informações, consulte O servidor Chef Automate está em um estado `Connection lost` ou o servidor Puppet Enterprise está em um estado `Connection lost` neste guia.	23 de março de 2022

Atualizações para AWS OpsWorks Stacks	Como prática recomendada de segurança, agora você pode adicionar uma chave de `aws:SourceAccount` condição `aws:SourceArn` ou (ou ambas) para confiar nas políticas de relaciona mento que permitem que o AWS OpsWorks Stacks acesse a execução de tarefas em outros AWS serviços. Para obter mais informações, consulte Prevenção de confused deputy entre serviços para o AWS OpsWorks Stacks neste guia.	4 de março de 2022
Atualizações para AWS OpsWorks for Chef Automate e AWS OpsWorks for Puppet Enterprise	Como prática recomendada de segurança, agora você pode adicionar uma chave de `aws:SourceAccount` condição `aws:SourceArn` ou de condição (ou ambas) para confiar nas políticas de relacionamento que permitem AWS OpsWorks for Chef Automate que o Puppet Enterprise acesse a execução de tarefas em outros AWS serviços. OpsWorks Para obter mais informaçõ es, consulte Prevenção de confused deputy entre serviços neste guia.	10 de janeiro de 2022

Atualizações para AWS OpsWorks for Chef Automate e AWS OpsWorks for Puppet Enterprise	AWS OpsWorks for Chef Automate e OpsWorks para o Puppet Enterprise atualizaram as políticas gerenciadas `AWSOpsWorksCMServiceRole` e `AWSOpsWorksCMInstanceProfileRole`, agora, armazenam segredos em AWS Secrets Manager.	3 de maio de 2021
Atualizações para AWS OpsWorks for Puppet Enterprise	A versão do mecanismo de um servidor OpsWorks para Puppet Enterprise que você cria no console agora é 2019.8.5. Ao usar a API, você pode especificar qualquer versão `2019` ou `2017` quando você cria um servidor Puppet Enterprise. A API `DescribeServers` agora retorna um atributo chamado `PUPPET_API_CRL` em seus resultados. Esse atributo contém uma lista de revogação de certificados para uso interno.	28 de abril de 2021

AWS OpsWorks O Stacks usa uma nova política gerenciada	AWS OpsWorks O Stacks mudou a política gerenciad a que inclui permissões para realizar todas as ações no AWS OpsWorks Stacks. A nova política é AWSOpsWor ks_FullAccess. Para obter mais informações sobre exemplos nesta política de permissões, consulte Políticas de exemplo.	19 de fevereiro de 2021
Migre AWS OpsWorks Stacks pilhas de EC2 -Classic para uma VPC	Foi adicionada documenta ção descrevendo como migrar uma AWS OpsWorks Stacks pilha de EC2 -Classic para uma VPC.	29 de setembro de 2020
Regenere um kit inicial para e AWS OpsWorks for Chef AutomateAWS OpsWorks for Puppet Enterprise	Foi adicionada documentação descrevendo como regenerar o kit inicial para um AWS OpsWorks for Chef Automate ou um AWS OpsWorks for Puppet Enterprise servidor.	29 de julho de 2020
AWS OpsWorks for Puppet Enterprise permite criar um servidor que usa um domínio, certificado e chave privada personalizados	Agora você pode criar um servidor OpsWorks para o Puppet Enterprise que usa um domínio, certificado e chave privada personali zados. É possível atualizar um servidor Puppet Enterpris e existente para usar um domínio personalizado criando um servidor do backup de um servidor existente.	17 de abril de 2020

AWS OpsWorks for Chef Automate e AWS OpsWorks for Puppet Enterprise agora suporta marcação no console	Agora você pode adicionar tags a um AWS OpsWorks for Chef Automate servidor ou AWS OpsWorks for Puppet Enterprise mestre, ou aos backups do servidor, usando o AWS Management Console ou AWS CLI o. Para obter mais informações, consulte Trabalhar com tags (Chef) ou Trabalhar com tags (Puppet).	26 de fevereiro de 2020
AWS OpsWorks for Chef Automate simplifica a atualização dos servidores Chef Automate 1 existentes para o Chef Automate 2	Você pode atualizar AWS OpsWorks for Chef Automate os servidores qualificados que executam o Chef Automate 1 para o Chef Automate 2 escolhendo Iniciar atualização na página de detalhes do seu servidor no console ou executando a ação da `StartMaintenance` API. Para obter mais informações, consulte Atualizar um AWS OpsWorks for Chef Automate servidor para o Chef Automate 2.	24 de janeiro de 2020
AWS OpsWorks for Chef Automate and AWS OpsWorks for Puppet Enterprise	Um novo capítulo sobre Segurança no AWS OpsWorks CM (AWS OpsWorks for Chef Automate e AWS OpsWorks for Puppet Enterprise) foi adicionado ao guia.	23 de dezembro de 2019

AWS OpsWorks for Chef Automate e AWS OpsWorks for Puppet Enterprise suporte à marcação	Agora você pode adicionar tags a um AWS OpsWorks for Chef Automate servidor ou AWS OpsWorks for Puppet Enterprise mestre, ou aos backups do servidor, usando AWS CLI o. AWS OpsWorks O CM agora oferece suporte à autorização baseada em tags.	18 de dezembro de 2019
AWS OpsWorks for Chef Automate permite criar um servidor que usa um domínio, certificado e chave privada personalizados	Agora você pode criar um servidor AWS OpsWorks for Chef Automate 2.0 que usa um domínio, certificado e chave privada personali zados. É possível atualizar um servidor Chef Automate 2.0 existente para usar um domínio personalizado criando um servidor do backup de um servidor existente.	22 de outubro de 2019
AWS OpsWorks O Stacks agora é compatível com Ruby 2.6.1	AWS OpsWorks O Stacks suporta camadas do Ruby 2.6.1 on Rails App Server nas pilhas do Chef 11.10.	2 de maio de 2019
AWS OpsWorks for Chef Automate agora suporta Chef Automate 2.0	Novos AWS OpsWorks for Chef Automate servidores executarão o Chef Automate 2.0, que inclui atualizações para o Chef InSpec, novos recursos em verificação e relatórios de conformidade e o Chef Infra.	30 de abril de 2019

AWS OpsWorks for Chef Automate and AWS OpsWorks for Puppet Enterprise	Agora você pode usar AWS CloudFormation para criar um AWS OpsWorks for Chef Automate servidor ou um servidor AWS OpsWorks for Puppet Enterprise mestre.	24 de janeiro de 2019
AWS OpsWorks Pilhas	AWS OpsWorks O Stacks agora oferece suporte a instâncias que executam o Ubuntu 18.04 LTS nas pilhas do Chef 12.	18 de dezembro de 2018
AWS OpsWorks para Puppet Enterprise	Procedimento adicionado para configurar uma conexão baseada em SSH com um repositório de controle que usa. CodeCommit	3 de dezembro de 2018
AWS OpsWorks Pilhas	AWS OpsWorks O Stacks agora oferece suporte a instâncias que executam o Amazon Linux 2 nas pilhas do Chef 12.	15 de novembro de 2018
AWS OpsWorks Pilhas	AWS OpsWorks O Stacks agora oferece suporte a instâncias que executam o Amazon Linux 2018.03 nas pilhas do Chef 11.10.	23 de outubro de 2018
AWS OpsWorks Pilhas	AWS OpsWorks O Stacks agora oferece suporte a instâncias que executam o Amazon Linux 2018.03 nas pilhas do Chef 12.	23 de agosto de 2018

AWS OpsWorks for Chef Automate e OpsWorks para a Puppet Enterprise	OpsWorks for Puppet Enterprise foi atualizado para PE 2018.1.2. AWS OpsWorks for Chef Automate foi atualizado para o Chef Automate 1.8.68.	29 de junho de 2018

- AWS OpsWorks for Chef Automate e OpsWorks para a versão da API Puppet Enterprise: 2016-11-01
- AWS OpsWorks Versão da Stacks API: 08/03/2016
- Última atualização da documentação: 2024-04-11

Atualizações anteriores

A tabela a seguir descreve alterações importantes em cada versão do Guia do usuário do AWS OpsWorks antes de junho de 2018.

Descrição	Data
AWS OpsWorks A versão Stacks Chef para pilhas baseadas em Windows foi atualizada para 12.22; a versão Ruby agora é 2.3.6.	19 de abril de 2018
Novos procedimentos para criar um AWS OpsWorks for Chef Automate servidor ou um mestre OpsWorks para o Puppet Enterprise usando o. AWS CLI	23 de março de 2018
Chef Automate atualizado para a versão 1.8. Configuração do Chef Compliance simplificada com a adição do livro de receitas `opsworks-audit` .	5 de março de 2018
Foi adicionado suporte para eventos AWS OpsWorks Stacks na Amazon CloudWatch Events.	20 de fevereiro de 2018
Foi adicionado suporte para novos tipos de volume do EBS no AWS OpsWorks Stacks e uma nova API,. DescribeOperatingSystems	25 de janeiro de 2018

Descrição	Data
OpsWorks para Puppet Enterprise e AWS OpsWorks for Chef Automate agora suporta a seleção de vários grupos de segurança ao criar um servidor.	18 de janeiro de 2018
Foi adicionado suporte para AWS OpsWorks Stacks na região da Europa (Paris).	19 de dezembro de 2017
Foi adicionado suporte para AWS OpsWorks for Chef Automate e OpsWorks para o Puppet Enterprise em seis regiões adicionais e procedimentos adicionais para criar backups de AWS OpsWorks for Chef Automate e OpsWorks para os servidores do Puppet Enterprise no. AWS Management Console	18 de dezembro de 2017
Foi adicionado o novo OpsWorks serviço e a documentação do Puppet Enterprise.	16 de novembro de 2017
Foi adicionado suporte para Amazon Linux 2017.09 ao AWS OpsWorks Stacks.	7 de novembro de 2017
Foi adicionado suporte para Chef Compliance ao AWS OpsWorks for Chef Automate.	25 de outubro de 2017
Foi adicionado suporte para Amazon Linux 2017.09 ao. AWS OpsWorks for Chef Automate	9 de outubro de 2017
Tópico de manutenção do sistema adicionado ao AWS OpsWorks for Chef Automate capítulo.	28 de julho de 2017
Foi adicionado suporte para tags no AWS OpsWorks Stacks.	6 de junho de 2017
Foi adicionada integração com o CloudWatch Logs.	10 de abril de 2017
Adicionou o novo AWS OpsWorks for Chef Automate serviço e a nova documentação.	1° de dezembro de 2016
Adicionado o suporte à região de endpoint regional do Leste dos EUA (Ohio).	12 de outubro de 2016

Descrição	Data
Adicionado suporte para pilhas e instâncias que executam o sistema operacional Amazon Linux 2016.09.	30 de setembro de 2016
Adicionado suporte para a região Ásia Pacífico (Seul) e nove endpoints regionais adicionais.	15 de agosto de 2016
Adicionado suporte para Node.js 0.12.15 e Ruby 2.3 em camadas integradas.	6 de julho de 2016
Suporte adicionado à região da Ásia-Pacífico (Mumbai).	28 de junho de 2016
Adicionado suporte para pilhas e instâncias que executam o sistema operacional CentOS 7.	22 de junho de 2016
Foi adicionada uma descrição passo a passo CodePipeline e a integração com o AWS OpsWorks Stacks.	2 de junho de 2016
Adicionado suporte para pilhas e instâncias que executam o sistema operacional Ubuntu 16.04 LTS.	1 de junho de 2016
Adicionado suporte para Chef 12 Linux e a documentação relaciona da.	3 de dezembro de 2015
Adicionada demonstração do Node.js para Conceitos básicos.	14 de julho de 2015
Adicionados dois novos exemplos de livros de receitas para Introduçã o a livros de receitas.	14 de julho de 2015
Adicionado suporte para gerenciamento de versões de agente.	23 de junho de 2015
Adicionado suporte para gerenciamento de versões de agente.	24 de junho de 2015
Foi adicionado suporte para Windows personalizado AMIs.	22 de junho de 2015
Adicionados três novos tópicos de práticas recomendadas.	11 de junho de 2015
Adicionado suporte para pilhas do Windows.	18 de maio de 2015

Descrição	Data
Adicionado um capítulo Práticas recomendadas.	15 de dezembro de 2014
Adicionado suporte para drenagem da conexão do Elastic Load Balancing e tempos limites personalizados de Desligamento.	15 de dezembro de 2014
Suporte adicional para registrar instâncias criadas fora do AWS OpsWorks Stacks.	9 de dezembro de 2014
Adicionado suporte para a Amazon SWF.	4 de setembro de 2014
Adicionado suporte para associação de variáveis do ambiente com aplicativos e Introdução a livros de receitas estendida.	16 de julho de 2014
Adicionado Introdução a livros de receitas, um tutorial de introdução para a implementação de livros de receitas.	16 de julho de 2014
Suporte adicionado para CloudTrail.	4 de junho de 2014
Adicionado suporte para Amazon RDS.	14 de maio de 2014
Adicionado suporte para o Chef 11.10 e o Berkshelf.	27 de março de 2014
Adicionado suporte para os volumes PIOPS do Amazon EBS.	16 de dezembro de 2013
Adicionadas permissões com base em recursos.	5 de dezembro de 2013
Adicionado gerenciamento de recursos.	7 de outubro de 2013
Suporte adicionado para VPCs.	29 de agosto de 2013
Foi adicionado suporte para customização AMIs e Chef 11.4.	24 de julho de 2013
Adicionado suporte para console para várias camadas por instância.	1 de julho de 2013
Foi adicionado suporte para instâncias apoiadas pelo Amazon EBS, Elastic Load Balancing e monitoramento da Amazon. CloudWatch	14 de maio de 2013

Descrição	Data
Versão inicial do Guia do usuário do AWS OpsWorks Stacks.	18 de fevereiro de 2013

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.