Guida per l'utente

AWS Proton

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

AWS Proton: Guida per l'utente

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

I marchi e l'immagine commerciale di Amazon non possono essere utilizzati in relazione a prodotti o servizi che non siano di Amazon, in una qualsiasi modalità che possa causare confusione tra i clienti o in una qualsiasi modalità che denigri o discrediti Amazon. Tutti gli altri marchi non di proprietà di Amazon sono di proprietà delle rispettive aziende, che possono o meno essere associate, collegate o sponsorizzate da Amazon.

Table of Contents

Che cos'è AWS Proton? 1
team di piattaforma 1
Sviluppatori 2
Flusso di lavoro 2
Configurazione 4
Configurazione con IAM 4
Registrati per AWS 5
Crea un utente IAM 5
Ruoli di servizio 6
Configurazione con AWS Proton 7
Configurazione di un bucket Amazon S3 8
Configurazione di una AWS CodeStar connessione 8
Configurazione delle impostazioni della CI/CD pipeline dell'account 8
Configurazione del AWS CLI 11
Nozioni di base 12
Prerequisiti 12
Flusso di lavoro introduttivo 13
Nozioni di base sulla console 14
Passaggio 1: aprire la AWS Proton console 15
Passaggio 2: Preparati a utilizzare i modelli di esempio 15
Fase 3: Creare un modello di ambiente 15
Fase 4: Creare un modello di servizio 16
Fase 5: Creare un ambiente 17
Passaggio 6: Facoltativo: creazione di un servizio e distribuzione di un'applicazione 18
Fase 7: Pulizia. 20
Guida introduttiva alla CLI 21
1. Registrare un modello di ambiente 21
2. Registrare un modello di servizio 22
3. Implementa un ambiente 24
4. Implementare un servizio 24
5. Eliminazione 27
Libreria di modelli 27
Come AWS Proton funziona 29
Oggetti 30

Metodi di approvvigionamento 33
AWS-approvvigionamento gestito 35
CodeBuild approvvigionamento 37
Provisioning autogestito 39
AWS Proton terminologia 43
Creazione di modelli e pacchetti 45
Pacchetti di modelli 45
Parametri 47
Tipi di parametro 47
Utilizzo dei parametri 48
Parametri di ambiente CloudFormation IAc 52
Parametri del servizio CloudFormation IAc 57
Parametri del componente CloudFormation IAc 59
CloudFormation filtri parametrici 63
CodeBuild parametri di provisioning 70
Parametri Terraform IAc 71
Infrastruttura come file di codice 72
AWS CloudFormation file IAc 73
CodeBuild pacchetto 126
file Terraform iAC 132
File di schema 140
Requisiti dello schema ambientale 140
Requisiti dello schema di servizio 144
Manifesta e concludi 147
Riepilogo del pacchetto di modelli di ambiente 150
Riepilogo del pacchetto di modelli di servizio 151
Considerazioni sul pacchetto di modelli 152
Modelli 153
Versioni 154
Pubblicare 156
Pubblica modelli di ambiente 156
Pubblica modelli di servizio 163
Visualizza i modelli 172
Aggiornare un modello 176
Eliminare modelli 178
Configurazioni di sincronizzazione dei modelli 182

Inviare un commit 182
Modelli di servizio di sincronizzazione 183
Considerazioni sulla sincronizzazione dei modelli 183
Crea 184
Vista 191
Modificare 192
Eliminazione 194
Configurazioni di sincronizzazione dei servizi 194
AWS Proton File OPS 195
Crea 199
Vista 200
Modificare 201
Eliminazione 202
Ambienti 204
Ruoli IAM 204
AWS Proton ruolo di servizio 204
Crea 205
Crea ed esegui il provisioning nello stesso account 207
Crea un account ed esegui il provisioning in un altro 209
Provisioning autogestito 214
Vista 217
Aggiornamento 218
Aggiornare un ambiente di provisioning AWS gestito 220
Aggiornare un ambiente di provisioning autogestito 222
Annullare l'implementazione di un ambiente in corso 226
Eliminazione 228
Connessioni all'account 230
Crea un ambiente con connessioni di account ambientali 232
Gestisci le connessioni degli account di ambiente 233
Gestito dal cliente 240
Utilizzo di ambienti gestiti dal cliente 240
CodeBuild creazione di ruoli di approvvigionamento 242
Servizi 246
Crea 246
Cosa c'è in un servizio? 247
Modelli di servizio 247

Creazione di un servizio ... 248
Vista ... 252
Modificare ... 254
Modifica la descrizione del servizio ... 254
Aggiungere o rimuovere istanze di servizio ... 256
Eliminazione ... 263
Visualizza le istanze ... 264
Aggiorna istanza ... 266
Aggiorna pipeline ... 272
Componenti ... 280
Componenti e altre risorse ... 282
AWS Proton console ... 283
AWS Proton API e AWS CLI ... 284
Domande frequenti sui componenti ... 285
Stati dei componenti ... 286
File IAc dei componenti ... 287
Utilizzo dei parametri con i componenti ... 288
Creazione di file IAc robusti ... 288
AWS CloudFormation Esempio di componente ... 289
Passaggi per amministratori ... 290
Passaggi per gli sviluppatori ... 293
Repositories ... 296
Creazione di un collegamento di repository ... 297
Visualizza i dati del repository collegato ... 299
Eliminare il collegamento a un repository ... 301
Monitoraggio ... 303
Automatizza con AWS Proton EventBridge ... 303
Event types (Tipi di evento) ... 303
AWS Proton esempi di eventi ... 306
EventBridgeTutorial: invia avvisi di Amazon Simple Notification Service per le modifiche allo stato del AWS Proton servizio ... 308
Prerequisiti ... 308
Fase 1: Creazione e sottoscrizione a un argomento Amazon SNS ... 308
Fase 2: Registrazione di una regola di evento ... 309
Fase 3: Verifica la regola del tuo evento ... 310
Fase 4: pulizia ... 311

AWS Proton dashboard 312
AWS Proton console 312
Sicurezza 315
Identity and Access Management 316
Destinatari 316
Autenticazione con identità 317
Gestione dell'accesso con policy 320
Come AWS Proton funziona con IAM 323
Esempi di policy 330
AWS politiche gestite 344
Uso di ruoli collegati ai servizi 361
Risoluzione dei problemi 369
Analisi della configurazione e delle vulnerabilità 371
Protezione dei dati 371
Crittografia lato server inattiva 372
Crittografia in transito 372
AWS Proton gestione delle chiavi di crittografia 373
AWS Proton contesto di crittografia 373
Sicurezza dell'infrastruttura 374
Endpoint VPC (AWS PrivateLink) 375
Registrazione di log e monitoraggio 377
Resilienza 378
AWS Proton backup 378
Best practice di sicurezza 378
Utilizza IAM per controllare gli accessi 379
Non incorporate credenziali nei modelli e nei pacchetti di modelli 379
Usa la crittografia per proteggere i dati sensibili 380
Utilizzalo AWS CloudTrail per visualizzare e registrare le chiamate API 380
Prevenzione del confused deputy tra servizi 380
supporto personalizzato Codebuild 381
Aggiornamento del modello di ambiente 382
Assegnazione di tag 386
AWS etichettatura 386
AWS Proton etichettatura 387
AWS ProtonAWS tag gestiti 387
Propagazione dei tag alle risorse assegnate 388

Tag gestiti dal cliente 391
Crea tag utilizzando la console e la CLI 391
Crea tag usando il AWS ProtonAWS CLI 393
Risoluzione dei problemi 394
Errori di distribuzione che fanno riferimento AWS CloudFormation a parametri dinamici 394
AWS Proton quote 396
Cronologia dei documenti 397
AWS Glossario 402
..... cdiii

Che cos'è AWS Proton?

AWS Proton è:

- Infrastruttura automatizzata come fornitura di codice e implementazione di applicazioni serverless e basate su container

  Il AWS Proton servizio è un framework di automazione su due fronti. In qualità di amministratore, puoi creare modelli di servizio con versioni diverse che definiscono l'infrastruttura e gli strumenti di distribuzione standardizzati per applicazioni serverless e basate su container. In qualità di sviluppatore di applicazioni, puoi scegliere tra i modelli di servizio disponibili per automatizzare le distribuzioni di applicazioni o servizi.

  AWS Proton identifica per te tutte le istanze di servizio esistenti che utilizzano una versione di modello obsoleta. In qualità di amministratore, puoi richiedere di AWS Proton aggiornarle con un clic.
- Infrastruttura standardizzata

  I team della piattaforma possono utilizzare l' AWS Proton infrastruttura con versioni diverse come modelli di codice. Possono utilizzare questi modelli per definire e gestire stack di applicazioni standard che contengono l'architettura, le risorse dell'infrastruttura e la pipeline di distribuzione del software CI/CD.
- Implementazioni integrate con CI/CD

  Quando gli sviluppatori utilizzano l'interfaccia AWS Proton self-service per selezionare un modello di servizio, scelgono una definizione standardizzata dello stack di applicazioni per le distribuzioni di codice. AWS Proton effettua automaticamente il provisioning delle risorse, configura la pipeline CI/CD e distribuisce il codice nell'infrastruttura definita.

AWS Proton per i team di piattaforma

In qualità di amministratore, tu o i membri del team della piattaforma, create modelli di ambiente e modelli di servizio contenenti l'infrastruttura come codice. Il modello di ambiente definisce l'infrastruttura condivisa utilizzata da più applicazioni o risorse. Il modello di servizio definisce il tipo di infrastruttura necessaria per distribuire e gestire una singola applicazione o microservizio in un ambiente. Un AWS Proton servizio è un'istanza di un modello di servizio, che normalmente include diverse istanze di servizio e una pipeline. Un'istanza AWS Proton di servizio è un'istanza di un

modello di servizio in un ambiente specifico. Tu o altri membri del tuo team potete specificare quali modelli di ambiente sono compatibili con un determinato modello di servizio. Per ulteriori informazioni sui modelli, consultaAWS Proton modelli.

È possibile utilizzare la seguente infrastruttura come provider di codice con AWS Proton:

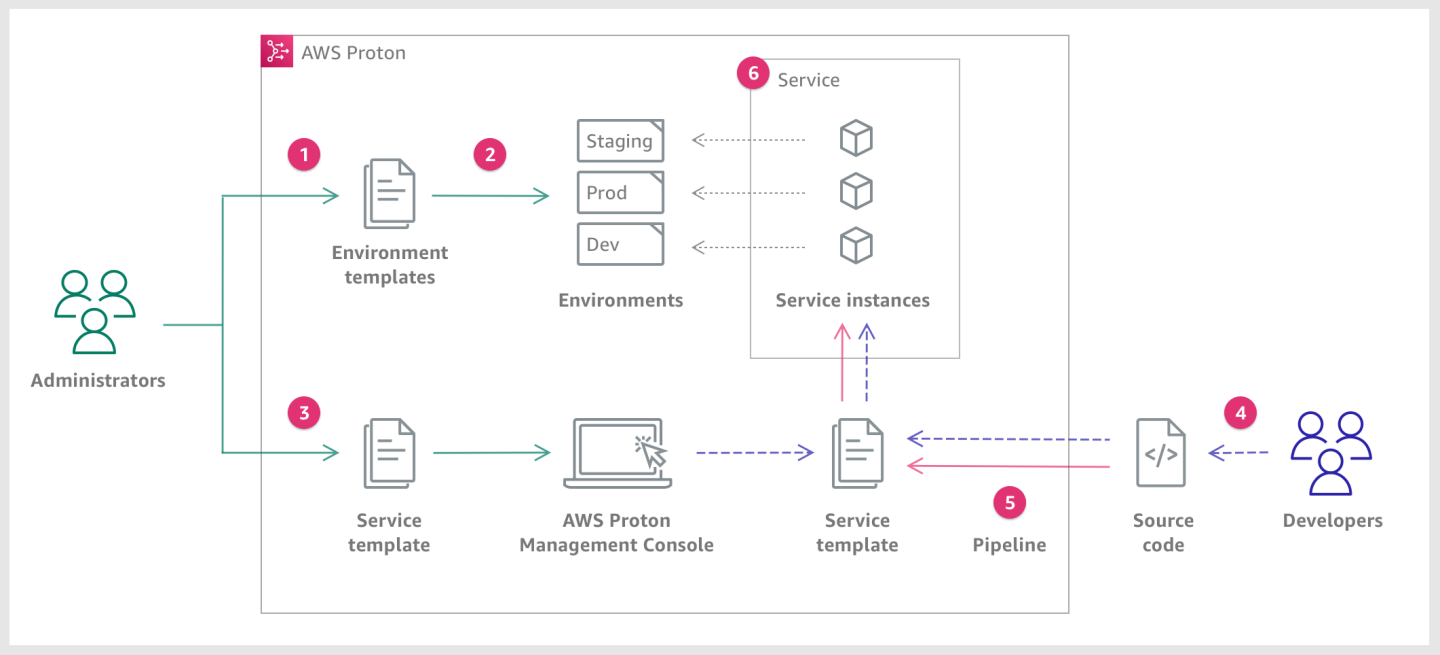
- AWS CloudFormation
- Terraform

AWS Proton per sviluppatori

In qualità di sviluppatore di applicazioni, devi selezionare un modello di servizio standardizzato da utilizzare per creare un servizio che distribuisce e gestisce l'applicazione in un'istanza di servizio. AWS Proton Un AWS Proton servizio è un'istanza di un modello di servizio, che normalmente include diverse istanze di servizio e una pipeline.

AWS Proton flusso di lavoro

Il diagramma seguente è una visualizzazione dei AWS Proton concetti principali discussi nel paragrafo precedente. Offre inoltre una panoramica di alto livello di ciò che costituisce un semplice flusso di lavoro. AWS Proton



1. In qualità di amministratore, crei e registri un modello di ambiente con AWS Proton, che definisce le risorse condivise.

2. AWS Proton distribuisce uno o più ambienti, sulla base di un modello di ambiente.

3. In qualità di amministratore, crei e registri un modello di servizio con AWS Proton, che definisce l'infrastruttura, il monitoraggio e le risorse CI/CD correlate, nonché i modelli di ambiente compatibili.

4. In qualità di sviluppatore, devi selezionare un modello di servizio registrato e fornire un collegamento al tuo repository di codice sorgente.

5. AWS Proton fornisce il Servizio con una pipeline CI/CD per le istanze del Servizio.

6. AWS Proton fornisce e gestisce il Servizio e le istanze di servizio che eseguono il codice sorgente come definito nel modello di servizio selezionato. Un'istanza di servizio è un'istanza del modello di servizio selezionato in un ambiente per una singola fase di una pipeline (ad esempio Prod).

Configurazione

Completa le attività in questa sezione in modo da poter creare e registrare modelli di servizi e ambienti. Ne hai bisogno per implementare ambienti e servizi con AWS Proton.

> **Note**
>
> Lo offriamo senza AWS Proton costi aggiuntivi. È possibile creare, registrare e gestire modelli di servizi e ambienti gratuitamente. Puoi anche contare sulla AWS Proton gestione autonoma delle proprie operazioni, come lo storage, la sicurezza e l'implementazione. Le uniche spese sostenute durante l'utilizzo AWS Proton sono le seguenti.
>
> - Costi di distribuzione e utilizzo Cloud AWS delle risorse che hai incaricato di distribuire e AWS Proton mantenere per te.
> - Costi per il mantenimento di una AWS CodeStar connessione al repository di codice.
> - Costi di manutenzione di un bucket Amazon S3, se utilizzi un bucket a cui fornire input. AWS Proton Puoi evitare questi costi se passi a the section called “Configurazioni di sincronizzazione dei modelli” utilizzare i repository Git per il tuothe section called “Pacchetti di modelli”.

Argomenti

- Configurazione con IAM
- Configurazione con AWS Proton

Configurazione con IAM

Quando ti iscrivi AWS, ti iscrivi Account AWS automaticamente a tutti i servizi in AWS, inclusi AWS Proton. Ti vengono addebitati solo i servizi e le risorse che utilizzi.

> **Note**
>
> Tu e il tuo team, compresi gli amministratori e gli sviluppatori, dovete avere tutti lo stesso account.

Registrati per AWS

Se non ne hai uno Account AWS, completa i seguenti passaggi per crearne uno.

Per iscriverti a un Account AWS

1. Apri la https://portal.aws.amazon.com/billing/registrazione.
2. Segui le istruzioni online.

   Parte della procedura di registrazione prevede la ricezione di una telefonata o di un messaggio di testo e l'immissione di un codice di verifica sulla tastiera del telefono.

   Quando ti iscrivi a un Account AWS, Utente root dell'account AWSviene creato un. L'utente root dispone dell'accesso a tutte le risorse e tutti i Servizi AWS nell'account. Come best practice di sicurezza, assegna l'accesso amministrativo a un utente e utilizza solo l'utente root per eseguire attività che richiedono l'accesso di un utente root.

Crea un utente IAM

Per creare un utente amministratore, scegli una delle seguenti opzioni.

Scelta di un modo per gestire il tuo amministratore	Per	Come	Puoi anche
In IAM Identity Center (Consigliato)	Usa credenziali a breve termine per accedere a AWS. Ciò è in linea con le best practice per la sicurezza. Per informazioni sulle best practice, consulta	Segui le istruzioni riportate in Nozioni di base nella Guida per l'utente di AWS IAM Identity Center .	Configura l'accesso programmatico configurando l'uso AWS IAM Identity Center nella Guida AWS CLI per l'AWS Command Line Interface utente.

Scelta di un modo per gestire il tuo amministratore	Per	Come	Puoi anche
	Best practice per la sicurezza in IAM nella Guida per l'utente di IAM.		
In IAM (Non consigliato)	Usa credenziali a lungo termine per accedere a AWS.	Seguendo le istruzioni contenute in Creare un utente IAM per l'accesso di emergenza nella Guida per l'utente IAM.	Configura l'accesso programmatico tramite Manage access keys for IAM users nella IAM User Guide.

Configurazione dei ruoli AWS Proton di servizio

Esistono alcuni ruoli IAM che potresti voler creare per diverse parti della tua AWS Proton soluzione. Puoi crearli in anticipo utilizzando la console IAM oppure puoi utilizzare la AWS Proton console per crearli per te.

Crea ruoli AWS Proton ambientali per consentire di AWS Proton effettuare chiamate API ad altri servizi di elaborazione e archiviazione Servizi AWS AWS CloudFormation AWS CodeBuild, simili e vari, per tuo conto, affinché forniscano risorse per te. Un ruolo AWS di provisioning gestito è necessario quando un ambiente o una qualsiasi delle istanze di servizio in esso eseguite utilizzano il provisioning gestito.AWS Un CodeBuildruolo è necessario quando un ambiente o una delle relative istanze di servizio utilizzano il provisioning. CodeBuild Per ulteriori informazioni sui ruoli AWS Proton ambientali, consulta. the section called “Ruoli IAM” Quando crei un ambiente, puoi utilizzare la AWS Proton console per scegliere un ruolo esistente per uno di questi due ruoli o per creare un ruolo con privilegi amministrativi per te.

Allo stesso modo, create ruoli di AWS Proton pipeline per consentire di AWS Proton effettuare chiamate API ad altri servizi per vostro conto, al fine di fornire una pipeline CI/CD per voi. Per ulteriori

informazioni sui ruoli della AWS Proton pipeline, consulta. the section called “ruoli del servizio di pipeline” Per ulteriori informazioni sulla configurazione delle CI/CD impostazioni, vedere. the section called “Configurazione delle impostazioni della CI/CD pipeline dell'account”

> **Note**
>
> Poiché non sappiamo quali risorse definirai nei tuoi AWS Proton modelli, i ruoli che crei utilizzando la console dispongono di ampie autorizzazioni e possono essere utilizzati sia come ruoli di servizio della AWS Proton pipeline che come ruoli di servizio. AWS Proton Per le implementazioni di produzione, ti consigliamo di limitare le autorizzazioni alle risorse specifiche che verranno distribuite creando policy personalizzate sia per i ruoli del servizio di AWS Proton pipeline che per i ruoli di servizio dell'ambiente. AWS Proton È possibile creare e personalizzare questi ruoli utilizzando o IAM. AWS CLI Per ulteriori informazioni, consultare Ruoli di servizio per AWS Proton e Creazione di un servizio.

Configurazione con AWS Proton

Se desideri utilizzare AWS CLI to run AWS Proton APIs, verifica di averlo installato. Se non l'hai installato, consultaConfigurazione del AWS CLI.

AWS Proton configurazione specifica:

- Per creare e gestire modelli:
  - Se utilizzi configurazioni di sincronizzazione dei modelli, configura una AWS CodeStar connessione.
  - Altrimenti, configura un bucket Amazon S3.
- Per effettuare il provisioning dell'infrastruttura:
  - Per il provisioning autogestito, è necessario configurare una AWS CodeStar connessione.
- (Facoltativo) Per effettuare il provisioning delle pipeline:
  - Per il AWS provisioning gestito e il provisioning CodeBuildbasato, impostate i ruoli della pipeline.
  - Per il provisioning autogestito, configura un repository di pipeline.

Per ulteriori informazioni sui metodi di provisioning, vedere. the section called “AWS-approvvigionamento gestito”

Configurazione di un bucket Amazon S3

Per configurare un bucket S3, segui le istruzioni in Crea il tuo primo bucket S3 per configurare un bucket S3. Inserisci i tuoi input nel bucket dove puoi recuperarli AWS Proton . AWS Proton Questi input sono noti come pacchetti di modelli. Puoi saperne di più su di essi in altre sezioni di questa guida.

Configurazione di una AWS CodeStar connessione

Per connettersi AWS Proton a un repository, si crea una AWS CodeStar connessione che attiva una pipeline quando viene effettuato un nuovo commit su un repository di codice sorgente di terze parti.

AWS Proton utilizza la connessione per:

- Attiva una pipeline di servizi quando viene effettuato un nuovo commit sul codice sorgente del repository.
- Effettua una pull request su un'infrastruttura come repository di codice.
- Crea una nuova versione secondaria o principale del modello ogni volta che un commit viene inviato a un repository di modelli che modifica uno dei tuoi modelli, se la versione non esiste già.

Puoi connetterti ai repository Bitbucket GitHub, GitHub Enterprise ed GitHub Enterprise Server con. CodeConnections Per ulteriori informazioni, consulta CodeConnections nella Guida per l'utente di AWS CodePipeline .

Per configurare una connessione. CodeStar

1. Apri la AWS Proton console.
2. Nel riquadro di navigazione, seleziona Impostazioni e quindi Connessioni al repository per accedere alla pagina Connessioni nelle Impostazioni degli strumenti per sviluppatori. La pagina mostra un elenco di connessioni.
3. Scegli Crea connessione e segui le istruzioni.

Configurazione delle impostazioni della CI/CD pipeline dell'account

AWS Proton può fornire CI/CD pipeline per la distribuzione del codice dell'applicazione nelle istanze di servizio. Le AWS Proton impostazioni necessarie per il provisioning della pipeline dipendono dal metodo di provisioning scelto per la pipeline.

AWS-provisioning gestito e basato: imposta i ruoli della pipeline CodeBuild

Con AWS-managed provisioning and provisioning, esegui il provisioning delle pipeline al posto tuo. CodeBuild AWS Proton Pertanto, AWS Proton necessita di un ruolo di servizio che fornisca le autorizzazioni per il provisioning delle pipeline. Ciascuno di questi due metodi di provisioning utilizza il proprio ruolo di servizio. Questi ruoli sono condivisi tra tutte le pipeline AWS Proton di servizi e possono essere configurati una sola volta nelle impostazioni dell'account.

Per creare ruoli di servizio di pipeline utilizzando la console

1. Apri la AWS Proton console.
2. Nel riquadro di navigazione, scegli Impostazioni, quindi scegli Impostazioni account.
3. Nella pagina CI/CD Impostazioni account, scegli Configura.
4. Esegui una di queste operazioni:
   - Avere AWS Proton creato un ruolo di servizio di pipeline per te

     [Per abilitare il provisioning AWS gestito delle pipeline] Nella pagina Configura le impostazioni dell'account, nella sezione AWS-managed provisioning pipeline role:

     a. Seleziona Nuovo ruolo di servizio.
     b. Inserisci un nome per il ruolo, ad esempio**`myProtonPipelineServiceRole`**.
     c. Seleziona la casella di controllo per accettare di creare un AWS Proton ruolo con privilegi amministrativi nel tuo account.

     [Per abilitare il provisioning CodeBuild basato sulle pipeline] Nella pagina Configura le impostazioni dell'account, nella sezione ruolo CodeBuild pipeline, scegli Ruolo di servizio esistente e scegli il ruolo di servizio che hai creato nella CloudFormation sezione ruolo pipeline. Oppure, se non hai assegnato un ruolo di CloudFormation pipeline, ripeti i tre passaggi precedenti per creare un nuovo ruolo di servizio.
   - Per scegliere i ruoli esistenti del servizio di pipeline

     [Per abilitare il provisioning AWS gestito delle pipeline] Nella pagina Configura impostazioni account, nella sezione AWS-managed provisioning pipeline role, scegli Ruolo di servizio esistente e scegli un ruolo di servizio nel tuo account. AWS

[Per abilitare il CodeBuild provisioning delle pipeline] Nella pagina Configura le impostazioni dell'account, nella sezione CodeBuildPipeline Provisioning role, scegli Existing service role e scegli un ruolo di servizio nel tuo account. AWS

5. Scegli Save changes (Salva modifiche).

   Il tuo nuovo ruolo di servizio di pipeline viene visualizzato nella pagina delle impostazioni dell'account.

Provisioning autogestito: configura un repository di pipeline

Con il provisioning autogestito, AWS Proton invia una pull request (PR) a un repository di provisioning che hai configurato e il tuo codice di automazione è responsabile del provisioning delle pipeline. Pertanto, AWS Proton non è necessario un ruolo di servizio per il provisioning delle pipeline. Ha invece bisogno di un repository di provisioning registrato. Il codice di automazione nel repository deve assumere un ruolo appropriato che fornisce le autorizzazioni per il provisioning delle pipeline.

Per registrare un repository di provisioning delle pipeline utilizzando la console

1. Crea un repository di provisioning CI/CD della pipeline se non ne hai ancora creato uno. Per ulteriori informazioni sulle pipeline nel provisioning autogestito, consulta. the section called “Provisioning autogestito”
2. Nel riquadro di navigazione, scegli Impostazioni, quindi scegli Impostazioni account.
3. Nella pagina CI/CD Impostazioni account, scegli Configura.
4. Nella pagina Configura le impostazioni dell'account, nella sezione CI/CD Pipeline repository:
   a. Seleziona Nuovo repository, quindi scegli uno dei provider di repository.
   b. Per la CodeStar connessione, scegli una delle tue connessioni.

> **Note**
>
> Se non disponi ancora di una connessione all'account del provider di repository pertinente, scegli Aggiungi una nuova CodeStar connessione, completa il processo di creazione della connessione, quindi scegli il pulsante di aggiornamento accanto al menu di CodeStarconnessione. Ora dovresti essere in grado di scegliere la tua nuova connessione nel menu.

c. Per il nome del repository, scegli il tuo repository di provisioning della pipeline. Il menu a discesa mostra l'elenco dei repository presenti nell'account del provider.

d. Per Nome del ramo, scegli uno dei rami del repository.

5. Scegli Save changes (Salva modifiche).

   Il tuo repository della pipeline viene visualizzato nella pagina delle impostazioni dell'account.

Configurazione del AWS CLI

Per utilizzare AWS CLI per effettuare chiamate AWS Proton API, verifica di aver installato la versione più recente di AWS CLI. Per ulteriori informazioni, consulta Nozioni di base su AWS CLI nella Guida per l'utente di AWS Command Line Interface . Quindi, per iniziare a usare AWS CLI with AWS Proton, consultathe section called “Guida introduttiva alla CLI”.

Guida introduttiva con AWS Proton

Prima di iniziare, configurati per utilizzare AWS Proton e verifica di aver soddisfatto i prerequisiti Getting started.

Inizia AWS Proton scegliendo uno o più dei seguenti percorsi:

- Segui una console di esempio guidata o un flusso di lavoro CLI tramite i link alla documentazione.
- Esegui un esempio guidato di flusso di lavoro della console.
- Esegui un esempio di AWS CLI flusso di lavoro guidato.

Argomenti

- Prerequisiti
- Flusso di lavoro introduttivo
- Guida introduttiva a AWS Management Console
- Iniziare con AWS CLI
- La libreria AWS Proton di modelli

Prerequisiti

Prima di iniziare a utilizzare AWS Proton, assicurati che siano soddisfatti i seguenti prerequisiti. Per ulteriori informazioni, consulta Configurazione.

- Hai un account IAM con autorizzazioni di amministratore. Per ulteriori informazioni, consulta Configurazione con IAM.
- Hai il ruolo AWS Proton di servizio e il ruolo di servizio AWS Proton pipeline è associato al tuo account. Per ulteriori informazioni, consultare Configurazione dei ruoli AWS Proton di servizio e Ruoli di servizio per AWS Proton.
- Hai una AWS CodeStar connessione. Per ulteriori informazioni, consulta Configurazione di una AWS CodeStar connessione.
- Hai familiarità con la creazione di AWS CloudFormation modelli e la parametrizzazione Jinja. Per ulteriori informazioni, consulta Cos'è? AWS CloudFormation nella Guida per l' AWS CloudFormation utente e nel sito web di Jinja.

- Hai una conoscenza pratica dei servizi di AWS infrastruttura.
- Hai effettuato l'accesso al tuo Account AWS.

Flusso di lavoro introduttivo

Impara a creare pacchetti di modelli, creare e registrare modelli e creare ambienti e servizi seguendo i passaggi e i link di esempio.

Prima di iniziare, verifica di aver creato un ruolo AWS Proton di servizio.

Se il modello di servizio include una pipeline di AWS Proton servizio, verifica di aver creato una AWS CodeStar connessione e un ruolo del servizio di AWS Proton pipeline.

Per ulteriori informazioni, consulta The AWS Proton service API Reference.

Esempio: flusso di lavoro introduttivo

1. Fate riferimento al diagramma riportato di seguito Come AWS Proton funziona per una visualizzazione di alto livello degli AWS Proton ingressi e delle uscite.
2. Crea un pacchetto di ambienti e un pacchetto di modelli di servizio.
    a. Identifica i parametri di input.
    b. Crea un file di schema.
    c. Crea file Infrastructure as Code (IaC).
    d. Per completare il pacchetto di modelli, crea un file manifest e organizza i file IaC, i file manifest e il file di schema in directory.
    e. Rendi il tuo pacchetto di modelli accessibile a. AWS Proton
3. Crea e registra una versione del modello di ambiente con AWS Proton.

    Quando si utilizza la console per creare e registrare un modello, viene creata automaticamente una versione del modello.

    Quando si utilizza il AWS CLI per creare e registrare un modello:

    a. Crea un modello di ambiente.
    b. Crea una versione del modello di ambiente.

Per ulteriori informazioni, consulta CreateEnvironmentTemplatee CreateEnvironmentTemplateVersionnel riferimento all'AWS Proton API.

4. Pubblica il modello di ambiente per renderlo disponibile all'uso.

   Per ulteriori informazioni, UpdateEnvironmentTemplateVersionconsulta il riferimento all'AWS Proton API.

5. Per creare un ambiente, seleziona una versione del modello di ambiente pubblicato e fornisci i valori per gli input richiesti.

   Per ulteriori informazioni, consulta il riferimento CreateEnvironmentall'AWS Proton API.

6. Crea e registra una versione del modello di servizio con AWS Proton.

   Quando si utilizza la console per creare e registrare un modello, viene creata automaticamente una versione del modello.

   Quando si utilizza il AWS CLI per creare e registrare un modello:

   a. Crea un modello di servizio.
   b. Crea una versione del modello di servizio.

   Per ulteriori informazioni, consulta CreateServiceTemplatee CreateServiceTemplateVersionnel riferimento all'AWS Proton API.

7. Pubblica il modello di servizio per renderlo disponibile all'uso.

   Per ulteriori informazioni, UpdateServiceTemplateVersionconsulta il riferimento all'AWS Proton API.

8. Per creare un servizio, seleziona una versione del modello di servizio pubblicata e fornisci i valori per gli input richiesti.

   Per ulteriori informazioni, consulta il riferimento CreateServiceall'AWS Proton API.

Guida introduttiva a AWS Management Console

Inizia con AWS Proton

- Crea e visualizza un modello di ambiente.

- Crea, visualizza e pubblica un modello di servizio che utilizza il modello di ambiente appena creato.
- Creare un ambiente e un servizio (opzionale).
- Eliminare il modello di servizio, il modello di ambiente, l'ambiente e il servizio, se creati.

Passaggio 1: aprire la AWS Proton console

- Apri la console AWS Proton

Passaggio 2: Preparati a utilizzare i modelli di esempio

1. Crea una connessione Codestar a Github e assegna un nome alla connessione. my-proton-connection
2. Accedere a https://github.com/aws-samples/aws-proton-cloudformation-sample-templates
3. Crea un fork del repository nel tuo account Github.

Fase 3: Creare un modello di ambiente

Nel riquadro di navigazione, scegli Modelli di ambiente.

1. Nella pagina Modelli di ambiente, scegli Crea modello di ambiente.
2. Nella pagina Crea modello di ambiente, nella sezione Opzioni modello, scegli Crea un modello per il provisioning di nuovi ambienti.
3. Nella sezione Sorgente del pacchetto di modelli, scegli Sincronizza un pacchetto di modelli da Git.
4. Nella sezione Archivio delle definizioni dei modelli, seleziona Scegli un repository Git collegato.
5. Seleziona my-proton-connectiondall'elenco dei repository.
6. Seleziona main dall'elenco Branch.
7. Nella sezione dei dettagli del modello di ambiente Proton.
    a. Inserisci il nome del modello come**`fargate-env`**.
    b. Immettete il nome visualizzato del modello di ambiente come**`My Fargate Environment`**.
    c. (Facoltativo) Inserite una descrizione per il modello di ambiente.
8. (Facoltativo) Nella sezione Tag, scegli Aggiungi nuovo tag e inserisci una chiave e un valore per creare un tag gestito dal cliente.

9. Scegli il modello Create Environment.

   Ora ti trovi su una nuova pagina che mostra lo stato e i dettagli del tuo nuovo modello di ambiente. Questi dettagli includono un elenco di AWS tag gestiti dai clienti. AWS Proton genera automaticamente tag AWS gestiti per te quando crei AWS Proton risorse. Per ulteriori informazioni, consulta AWS Proton risorse e etichettatura.
10. Lo stato di un nuovo modello di ambiente inizia nello stato Bozza. Tu e altre persone con `proton:CreateEnvironment` autorizzazioni potete visualizzarlo e accedervi. Segui il passaggio successivo per rendere il modello disponibile agli altri.
11. Nella sezione Versioni del modello, scegli il pulsante di opzione a sinistra della versione secondaria del modello che hai appena creato (1.0). In alternativa, puoi scegliere Pubblica nel banner di avviso informativo e saltare il passaggio successivo.
12. Nella sezione Versioni dei modelli, scegli Pubblica.
13. Lo stato del modello cambia in Pubblicato. Poiché è la versione più recente del modello, è la versione consigliata.
14. Nel riquadro di navigazione, seleziona Modelli di ambiente.

    Una nuova pagina mostra un elenco dei modelli di ambiente insieme ai dettagli del modello.

Fase 4: Creare un modello di servizio

Creare un modello di servizio.

1. Nel riquadro di navigazione, scegli Modelli di servizio.
2. Nella pagina Modelli di servizio, scegli Crea modello di servizio.
3. Nella pagina Crea modello di servizio, nella sezione Sorgente del pacchetto di modelli, scegli Sincronizza un pacchetto di modelli da Git.
4. Nella sezione Modello, seleziona Scegli un repository Git collegato.
5. Seleziona my-proton-connectiondall'elenco dei repository.
6. Seleziona main dall'elenco Branch.
7. Nella sezione dei dettagli del modello di servizio Proton.

   a. Inserisci il nome del modello di servizio come**`backend-fargate-svc`**.
   b. Immettere il nome visualizzato del modello di servizio come**`My Fargate Service`**.
   c. (Facoltativo) Inserire una descrizione per il modello di servizio.

8. Nella sezione Modelli di ambiente compatibili.

   - Seleziona la casella di controllo a sinistra del modello di ambiente My Fargate Environment per selezionare il modello di ambiente compatibile per il nuovo modello di servizio.

9. Per le impostazioni di crittografia, mantieni le impostazioni predefinite.
10. Nella sezione Definizione della pipeline.

    - Mantieni selezionato il pulsante This template include un pulsante CI/CD pipeline.

11. Scegli Crea modello di servizio.

    Ora ti trovi su una nuova pagina che mostra lo stato e i dettagli del tuo nuovo modello di servizio, incluso un elenco di AWS tag gestiti dai clienti.

12. Lo stato di un nuovo modello di servizio inizia nello stato Bozza. Solo gli amministratori possono visualizzarlo e accedervi. Per rendere il modello di servizio disponibile per l'uso da parte degli sviluppatori, segui il passaggio successivo.
13. Nella sezione Versioni del modello, scegli il pulsante di opzione a sinistra della versione secondaria del modello che hai appena creato (1.0). In alternativa, puoi scegliere Pubblica nel banner di avviso informativo e saltare il passaggio successivo.
14. Nella sezione Versioni dei modelli, scegli Pubblica.
15. Lo stato del modello cambia in Pubblicato.

    La prima versione secondaria del modello di servizio viene pubblicata e può essere utilizzata dagli sviluppatori. Poiché è la versione più recente del modello, è la versione consigliata.

16. Nel riquadro di navigazione, scegli Modelli di servizio.

    Una nuova pagina mostra un elenco dei modelli e dei dettagli del servizio.

Fase 5: Creare un ambiente

Nel riquadro di navigazione, selezionare Compute environments (Ambienti di calcolo).

1. Seleziona Create environment (Crea ambiente).
2. Nella pagina Scegli un modello di ambiente, seleziona il modello che hai appena creato. Si chiama My Fargate Environment. Quindi, scegli Configura.
3. Nella pagina Configura ambiente, nella sezione Provisioning, scegli Provisioning through AWS Proton.

4. Nella sezione Account di distribuzione, seleziona Questo Account AWS.
5. In Impostazioni dell'ambiente, inserisci il nome dell'ambiente come**`my-fargate-environment`**.
6. Nella sezione Ruoli di ambiente, seleziona Nuovo ruolo di servizio o, se hai già creato un ruolo di AWS Proton servizio, seleziona Ruolo di servizio esistente.

   a. Seleziona Nuovo ruolo di servizio per creare un nuovo ruolo.

      i. Inserisci il nome del ruolo Ambiente come**`MyProtonServiceRole`**.

      ii. Seleziona la casella di controllo per accettare la creazione di un ruolo AWS Proton di servizio con privilegi amministrativi per il tuo account.

   b. Seleziona Ruolo di servizio esistente per utilizzare un ruolo esistente.

      - Seleziona il tuo ruolo nel campo a discesa Environment role name.
7. Scegli Next (Successivo).
8. Nella pagina Configura impostazioni personalizzate, utilizza le impostazioni predefinite.
9. Scegli Avanti e controlla i dati inseriti.
10. Scegli Create (Crea).

    Visualizza i dettagli e lo stato dell'ambiente, nonché i tag AWS gestiti e i tag gestiti dal cliente per il tuo ambiente.
11. Nel riquadro di navigazione, selezionare Compute environments (Ambienti di calcolo).

    Una nuova pagina mostra un elenco degli ambienti insieme allo stato e ad altri dettagli dell'ambiente.

Passaggio 6: Facoltativo: creazione di un servizio e distribuzione di un'applicazione

1. Apri la AWS Proton console.
2. Nel riquadro di navigazione, scegli Servizi.
3. Nella pagina Servizi, scegli Crea servizio.
4. Nella pagina Scegli un modello di servizio, seleziona il modello My Fargate Service scegliendo il pulsante di opzione nell'angolo in alto a destra della scheda modello.
5. Scegli Configura nell'angolo inferiore destro della pagina.

6. Nella pagina Configura servizio, nella sezione Impostazioni del servizio, inserisci il nome del servizio**`my-service`**.
7. (Facoltativo) Inserisci una descrizione per il servizio.
8. Nella sezione Impostazioni del repository del servizio:
   a. Per la CodeStar connessione, scegli la tua connessione dall'elenco.
   b. Per Nome del repository, scegli il nome del tuo repository di codice sorgente dall'elenco.
   c. Per Nome del ramo, scegli il nome del ramo del tuo repository del codice sorgente dall'elenco.
9. (Facoltativo) Nella sezione Tag, scegli Aggiungi nuovo tag e inserisci una chiave e un valore per creare un tag gestito dal cliente. Quindi scegli Successivo.
10. Nella pagina Configura impostazioni personalizzate, nella sezione Istanze di servizio, nella sezione Nuova istanza, segui i passaggi successivi per fornire valori personalizzati per i parametri dell'istanza di servizio.
    a. Inserisci il nome **`my-app-service`** dell'istanza.
    b. Scegli l'ambiente **`my-fargate-environment`** per la tua istanza di servizio.
    c. Mantieni le impostazioni predefinite per i parametri rimanenti dell'istanza.
    d. Mantieni i valori predefiniti per gli input di Pipeline.
    e. Scegli Avanti e controlla i tuoi input.
    f. Scegli Crea e visualizza lo stato e i dettagli del servizio.
11. Nella pagina dei dettagli del servizio, visualizza lo stato dell'istanza del servizio e della pipeline scegliendo le schede Overview e Pipeline. In queste pagine puoi anche visualizzare AWS e gestire i tag dai clienti. AWS Proton crea automaticamente tag AWS gestiti per te. Scegli Gestisci tag per creare e modificare i tag gestiti dai clienti. Per ulteriori informazioni sull'assegnazione di tag, consulta AWS Proton risorse e etichettatura.
12. Dopo che il servizio è attivo, nella scheda Panoramica, nella sezione Istanze di servizio, scegli il nome dell'istanza del servizio, my-app-service.

    Ora ti trovi nella pagina dei dettagli dell'istanza di servizio.
13. Per visualizzare la tua applicazione, nella sezione Output, copia il ServiceEndpointlink nel tuo browser.

    Nella pagina web viene visualizzato un AWS Proton grafico.

14. Dopo aver creato il servizio, nel riquadro di navigazione, scegli Servizi per visualizzare un elenco dei tuoi servizi.

Fase 7: Pulizia.

1. Apri la AWS Proton console.
2. Elimina un servizio (se ne hai creato uno)
    a. Nel riquadro di navigazione, scegli Servizi.
    b. Nella pagina Servizi, scegli il nome del servizio my-service.

       Ora ti trovi nella pagina dei dettagli del servizio my-service.
    c. Nell'angolo in alto a destra della pagina, scegli Azioni, quindi Elimina.
    d. Una finestra modale richiede di confermare l'azione di eliminazione.
    e. Segui le istruzioni e scegli Sì, elimina.
3. Eliminare un ambiente
    a. Nel riquadro di navigazione, selezionare Compute environments (Ambienti di calcolo).
    b. Nella pagina Ambienti, seleziona il pulsante di opzione a sinistra dell'ambiente che hai appena creato.
    c. Scegli Operazioni, quindi Elimina.
    d. Una finestra modale richiede di confermare l'azione di eliminazione.
    e. Segui le istruzioni e scegli Sì, elimina.
4. Eliminare un modello di servizio
    a. Nel riquadro di navigazione, scegli Modelli di servizio.
    b. Nella pagina Modelli di servizio, seleziona il pulsante di opzione a sinistra del modello di servizio my-svc-template.
    c. Scegli Operazioni, quindi Elimina.
    d. Un modale richiede di confermare l'azione di eliminazione.
    e. Segui le istruzioni e scegli Sì, elimina. In questo modo verranno eliminati il modello di servizio e tutte le relative versioni.
5. Eliminare un modello di ambiente
    a. Nel riquadro di navigazione, scegli Modelli di ambiente.

b. Nella pagina Modelli di ambiente, seleziona il pulsante di opzione a sinistra di my-env-template.

c. Scegli Operazioni, quindi Elimina.

d. Una finestra modale richiede di confermare l'azione di eliminazione.

e. Segui le istruzioni e scegli Sì, elimina. Questo elimina il modello di ambiente e tutte le sue versioni.

6. Elimina la tua connessione Codestar

Iniziare con AWS CLI

Per iniziare a AWS Proton usare AWS CLI, segui questo tutorial. Il tutorial illustra un servizio di carico bilanciato rivolto al pubblico basato su AWS Proton . AWS Fargate Il tutorial fornisce anche una pipeline CI/CD che implementa un sito Web statico con un'immagine visualizzata.

Prima di iniziare, assicurati di essere configurato correttamente. Per informazioni dettagliate, consultare the section called “Prerequisiti”.

Fase 1: Registrare un modello di ambiente

In questa fase, in qualità di amministratore, registri un modello di ambiente di esempio, che contiene un cluster Amazon Elastic Container Service (Amazon ECS) e un Amazon Virtual Private Cloud (Amazon VPC) con due sottoreti pubbliche/private.

Per registrare un modello di ambiente

1. Inserisci il repository AWS Proton Sample CloudFormation Templates nel tuo GitHub account o nella tua organizzazione. Questo repository include i modelli di ambiente e servizio che utilizziamo in questo tutorial.

   Quindi, registra il tuo repository biforcato con. AWS Proton Per ulteriori informazioni, consulta the section called “Creazione di un collegamento di repository”.

2. Crea un modello di ambiente.

   La risorsa del modello di ambiente tiene traccia delle versioni del modello di ambiente.

```
$ aws proton create-environment-template \
  --name "fargate-env" \
  --display-name "Public VPC Fargate" \
```

```
  --description "VPC with public access and ECS cluster"
```

3. Crea una configurazione di sincronizzazione dei modelli.

   AWS Proton imposta una relazione di sincronizzazione tra il repository e il modello di ambiente. Quindi crea la versione 1.0 del modello nello DRAFT stato.

```
$ aws proton create-template-sync-config \
  --template-name "fargate-env" \
  --template-type "ENVIRONMENT" \
  --repository-name "your-forked-repo" \
  --repository-provider "GITHUB" \
  --branch "your-branch" \
  --subdirectory "environment-templates/fargate-env"
```

4. Attendi che la versione del modello di ambiente venga registrata correttamente.

   Quando questo comando ritorna con uno stato di uscita di0, la registrazione della versione è completa. Ciò è utile negli script per garantire la corretta esecuzione del comando nel passaggio successivo.

```
$ aws proton wait environment-template-version-registered \
  --template-name "fargate-env" \
  --major-version "1" \
  --minor-version "0"
```

5. Pubblicate la versione del modello di ambiente per renderla disponibile per la creazione dell'ambiente.

```
$ aws proton update-environment-template-version \
  --template-name "fargate-env" \
  --major-version "1" \
  --minor-version "0" \
  --status "PUBLISHED"
```

Fase 2: Registrazione di un modello di servizio

In questa fase, in qualità di amministratore, registri un modello di servizio di esempio, che contiene tutte le risorse necessarie per fornire un servizio Amazon ECS Fargate con un sistema di bilanciamento del carico e una pipeline CI/CD che utilizza. AWS CodePipeline

Per registrare un modello di servizio

1. Creare un modello di servizio.

   La risorsa del modello di servizio tiene traccia delle versioni del modello di servizio.

   ```
   $ aws proton create-service-template \
     --name "load-balanced-fargate-svc" \
     --display-name "Load balanced Fargate service" \
     --description "Fargate service with an application load balancer"
   ```

2. Crea una configurazione di sincronizzazione dei modelli.

   AWS Proton imposta una relazione di sincronizzazione tra il repository e il modello di servizio. Quindi crea la versione 1.0 del modello nello DRAFT stato.

   ```
   $ aws proton create-template-sync-config \
     --template-name "load-balanced-fargate-svc" \
     --template-type "SERVICE" \
     --repository-name "your-forked-repo" \
     --repository-provider "GITHUB" \
     --branch "your-branch" \
     --subdirectory "service-templates/load-balanced-fargate-svc"
   ```

3. Attendi che la versione del modello di servizio venga registrata correttamente.

   Quando questo comando ritorna con uno stato di uscita di0, la registrazione della versione è completa. Ciò è utile negli script per garantire la corretta esecuzione del comando nel passaggio successivo.

   ```
   $ aws proton wait service-template-version-registered \
     --template-name "load-balanced-fargate-svc" \
     --major-version "1" \
     --minor-version "0"
   ```

4. Pubblica la versione del modello di servizio per renderla disponibile per la creazione del servizio.

   ```
   $ aws proton update-service-template-version \
     --template-name "load-balanced-fargate-svc" \
     --major-version "1" \
     --minor-version "0" \
     --status "PUBLISHED"
   ```

Fase 3: Implementazione di un ambiente

In questo passaggio, in qualità di amministratore, si crea un'istanza di un AWS Proton ambiente dal modello di ambiente.

Per distribuire un ambiente

1. Ottieni un file di specifiche di esempio per il modello di ambiente che hai registrato.

   È possibile scaricare il file `environment-templates/fargate-env/spec/spec.yaml` dal repository di esempio del modello. In alternativa, è possibile recuperare l'intero repository localmente ed eseguire il create-environment comando dalla directory. `environment-templates/fargate-env`

2. Crea un ambiente.

   AWS Proton legge i valori di input dalle specifiche dell'ambiente, li combina con il modello di ambiente ed effettua il provisioning delle risorse ambientali nell' AWS account utilizzando il ruolo di servizio dell'utente. AWS Proton

   ```
   $ aws proton create-environment \
     --name "fargate-env-prod" \
     --template-name "fargate-env" \
     --template-major-version 1 \
     --proton-service-role-arn "arn:aws:iam::123456789012:role/AWS ProtonServiceRole"
    \
     --spec "file://spec/spec.yaml"
   ```

3. Attendi che l'ambiente venga distribuito correttamente.

   ```
   $ aws proton wait environment-deployed --name "fargate-env-prod"
   ```

Fase 4: Implementazione di un servizio [sviluppatore di applicazioni]

Nei passaggi precedenti, un amministratore ha registrato e pubblicato un modello di servizio e distribuito un ambiente. In qualità di sviluppatore di applicazioni, ora puoi creare un AWS Proton servizio e distribuirlo nell'ambiente AWS Proton

Per distribuire un servizio

1. Ottieni un file di specifiche di esempio per il modello di servizio registrato dall'amministratore.

   È possibile scaricare il file `service-templates/load-balanced-fargate-svc/spec/spec.yaml` dal repository di esempio del modello. In alternativa, è possibile recuperare l'intero repository localmente ed eseguire il create-service comando dalla directory. `service-templates/load-balanced-fargate-svc`
2. Inserisci il repository di AWS Proton Sample Services nel tuo account o nella tua GitHub organizzazione. Questo repository include il codice sorgente dell'applicazione che utilizziamo in questo tutorial.
3. Crea un servizio.

   AWS Proton legge i valori di input dalle specifiche del servizio, li combina con il modello di servizio e fornisce le risorse di servizio nell' AWS account nell'ambiente specificato nelle specifiche. Una AWS CodePipeline pipeline distribuisce il codice dell'applicazione dal repository specificato nel comando.

   ```
   $ aws proton create-service \
     --name "static-website" \
     --repository-connection-arn \
       "arn:aws:codestar-connections:us-east-1:123456789012:connection/your-codestar-connection-id" \
     --repository-id "your-GitHub-account/aws-proton-sample-services" \
     --branch-name "main" \
     --template-major-version 1 \
     --template-name "load-balanced-fargate-svc" \
     --spec "file://spec/spec.yaml"
   ```
4. Attendi che il servizio venga distribuito correttamente.

   ```
   $ aws proton wait service-created --name "static-website"
   ```
5. Recupera gli output e visualizza il tuo nuovo sito web.

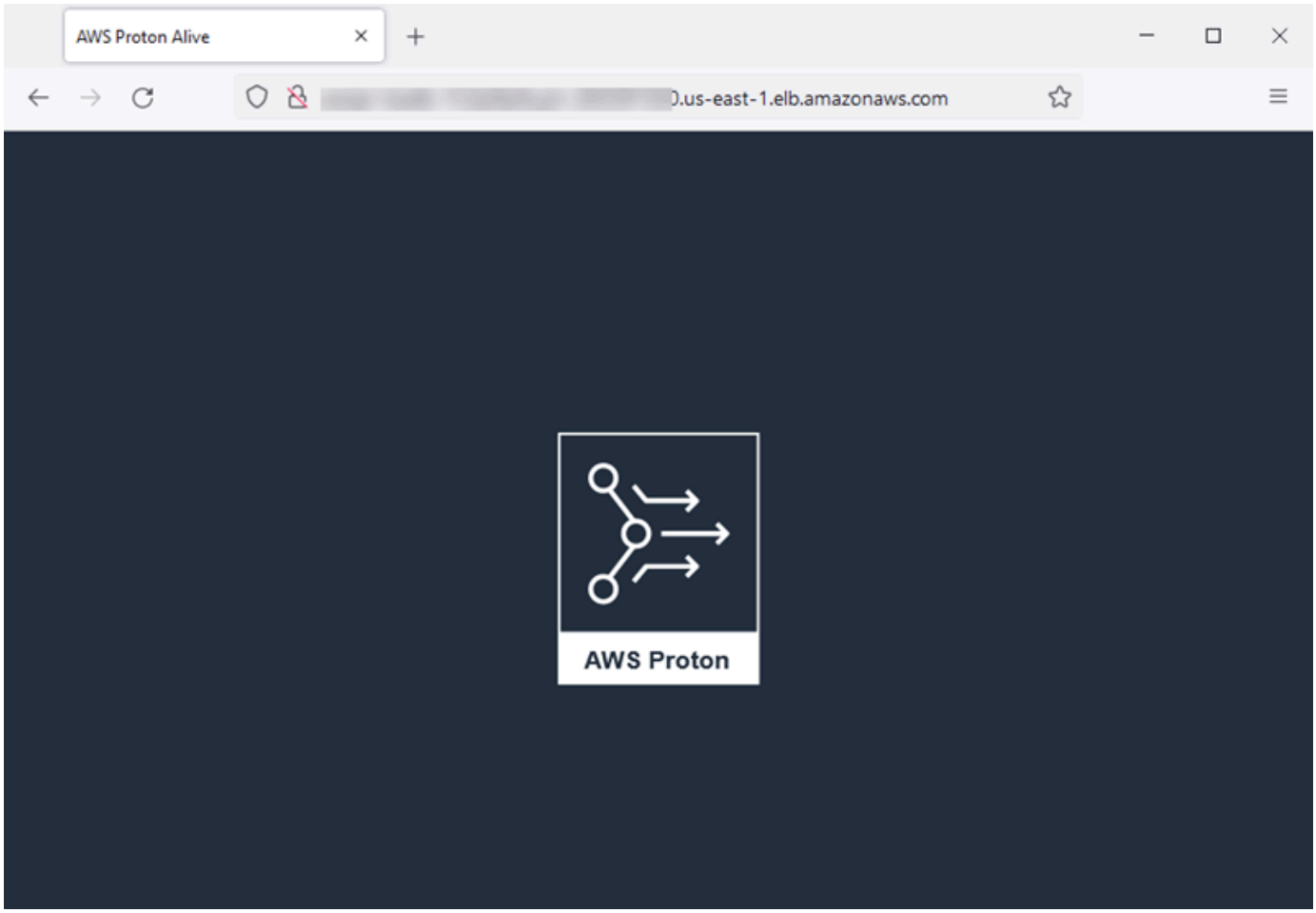
   Esegui il comando seguente:

   ```
   $ aws proton list-service-instance-outputs \
     --service-name "static-website" \
     --service-instance-name load-balanced-fargate-svc-prod
   ```

L'output del comando dovrebbe essere simile al seguente:

```
{
    "outputs": [
        {
            "key": "ServiceURL",
            "valueString": "http://your-service-endpoint.us-
east-1.elb.amazonaws.com"
        }
    ]
}
```

Il valore dell'output dell'`ServiceURL`istanza è l'endpoint del nuovo sito Web di servizio. Usa il tuo browser per accedervi. Dovresti vedere il seguente grafico su una pagina statica:



Fase 5: Pulizia (opzionale)

In questo passaggio, quando hai finito di esplorare le AWS risorse che hai creato come parte di questo tutorial e per risparmiare sui costi associati a tali risorse, le elimini.

Per eliminare le risorse del tutorial

1. Per eliminare il servizio, esegui il seguente comando:

```
$ aws proton delete-service --name "static-website"
```

2. Per eliminare l'ambiente, esegui il comando seguente:

```
$ aws proton delete-environment --name "fargate-env-prod"
```

3. Per eliminare il modello di servizio, esegui i seguenti comandi:

```
$ aws proton delete-template-sync-config \
  --template-name "load-balanced-fargate-svc" \
  --template-type "SERVICE"
$ aws proton delete-service-template --name "load-balanced-fargate-svc"
```

4. Per eliminare il modello di ambiente, esegui i seguenti comandi:

```
$ aws proton delete-template-sync-config \
  --template-name "fargate-env" \
  --template-type "ENVIRONMENT"
$ aws proton delete-environment-template --name "fargate-env"
```

La libreria AWS Proton di modelli

Il AWS Proton team gestisce una libreria di esempi di modelli su GitHub. La libreria include esempi di file Infrastructure as Code (IaC) per molti scenari comuni di ambiente e infrastruttura applicativa.

La libreria di modelli è archiviata in due GitHub repository:

- aws-proton-cloudformation-sample-templates — Esempi di pacchetti di modelli che utilizzano Jinja come AWS CloudFormationlinguaggio IAc. Puoi usare questi esempi per gli ambienti. AWS-approvvigionamento gestito

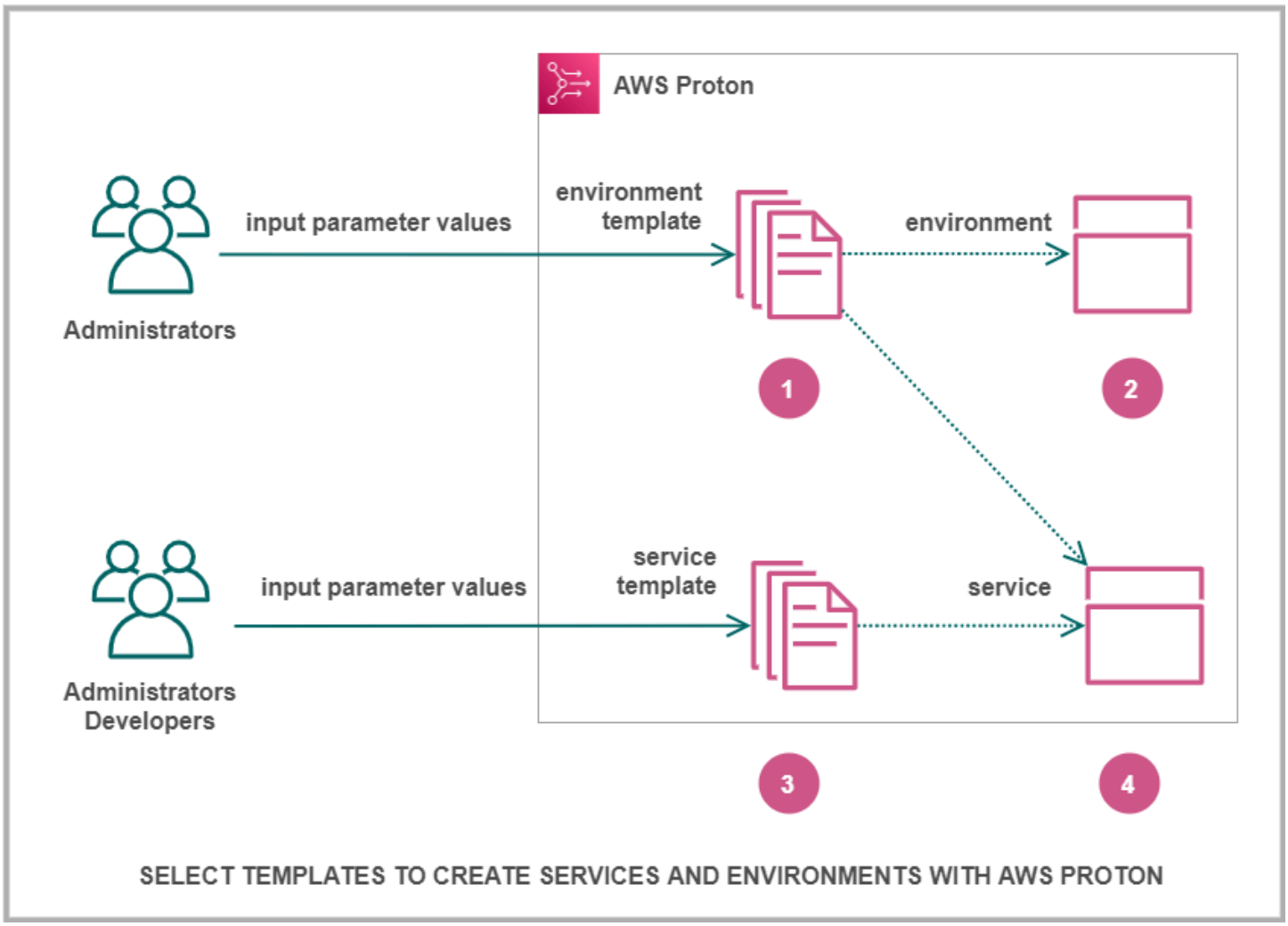
- aws-proton-terraform-sample-templates: esempi di pacchetti di modelli che utilizzano Terraform come linguaggio IAc. Puoi usare questi esempi per gli ambienti. Provisioning autogestito

Ciascuno di questi repository contiene un `README` file con informazioni complete sul contenuto e sulla struttura del repository. Ogni esempio contiene informazioni sul caso d'uso trattato dal modello, sull'architettura dell'esempio e sui parametri di input utilizzati dal modello.

Puoi utilizzare i modelli di questa libreria direttamente inserendo uno dei repository della libreria nel tuo GitHub account. In alternativa, utilizzate questi esempi come punto di partenza per sviluppare i vostri modelli di ambiente e servizio.

Come AWS Proton funziona

Con AWS Proton, esegui il provisioning degli ambienti e quindi dei servizi in esecuzione in tali ambienti. Gli ambienti e i servizi si basano rispettivamente su modelli di ambiente e di servizio scelti nella libreria di modelli AWS Proton con versioni.



1 Quand in qualità di amministratore, selezionate un modello di ambiente con AWS Proton, fornite i valori per i parametri di input richiesti.

2 AWS Proton utilizza il modello di ambiente e i valori dei parametri per eseguire il provisioning dell'ambiente.

3

Quand

in qualità di sviluppatore o amministratore, selezionate un modello di servizio con AWS Proton, fornite i valori per i parametri di input richiesti. È inoltre necessario selezionare un ambiente in cui distribuire l'applicazione o il servizio.

4

AWS Proton utilizza il modello di servizio e i valori del servizio e dei parametri di ambiente selezionati per fornire il servizio.

Fornisci i valori per i parametri di input per personalizzare il modello per il riutilizzo e per molteplici casi d'uso, applicazioni o servizi.

Per far sì che ciò funzioni, create pacchetti di modelli di ambiente o servizio e li caricate rispettivamente in modelli di ambiente o servizio registrati.

I pacchetti di modelli contengono tutto il AWS Proton necessario per fornire ambienti o servizi.

Quando si crea un modello di ambiente o di servizio, si carica un pacchetto di modelli che contiene i file parametrizzati di infrastruttura come codice (IaC) AWS Proton utilizzati per fornire ambienti o servizi.

Quando si seleziona un modello di ambiente o servizio per creare o aggiornare un ambiente o un servizio, si forniscono i valori per i parametri del file IaC del pacchetto di modelli.

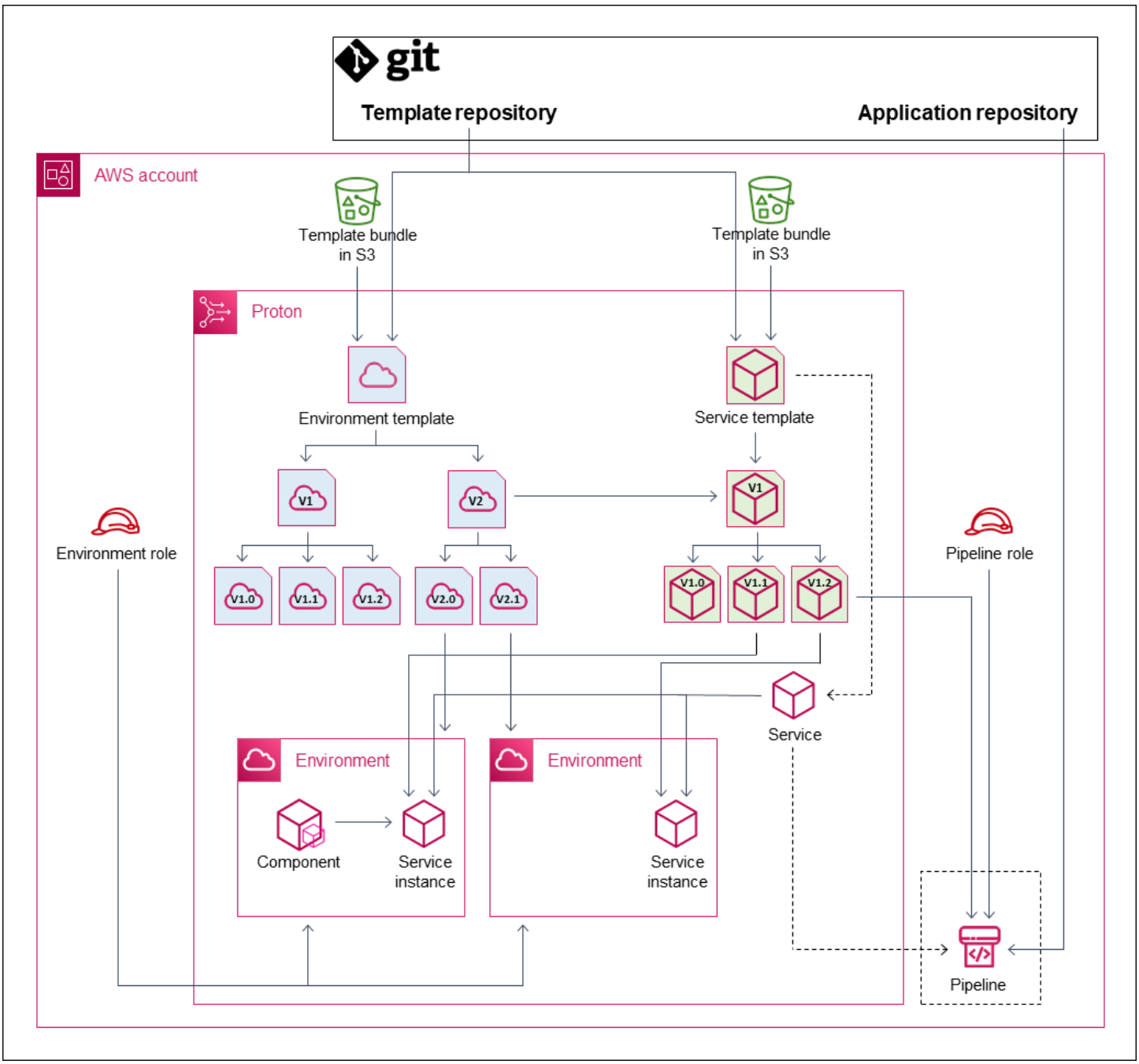
Argomenti

- AWS Proton oggetti
- In che modo AWS Proton provvede l'infrastruttura
- AWS Proton terminologia

AWS Proton oggetti

Il diagramma seguente mostra gli AWS Proton oggetti principali e la loro relazione con altri oggetti AWS e di terze parti. Le frecce rappresentano la direzione del flusso di dati (la direzione inversa della dipendenza).

Seguiamo il diagramma con brevi descrizioni e link di riferimento per questi oggetti. AWS Proton



- Modello di ambiente: una raccolta di versioni di modelli di ambiente che possono essere utilizzate per creare ambienti. AWS Proton

  Per ulteriori informazioni, consultare Creazione di modelli e pacchetti e Modelli.

- Versione del modello di ambiente: una versione specifica di un modello di ambiente. Accetta un pacchetto di modelli come input, da un bucket S3 o da un repository Git. Il pacchetto specifica Infrastructure as Code (IaC) e i relativi parametri di input per un ambiente. AWS Proton

Per ulteriori informazioni, consulta the section called “Versioni”, the section called “Pubblicare” e the section called “Configurazioni di sincronizzazione dei modelli”.

- Ambiente: l'insieme di risorse AWS infrastrutturali condivise e politiche di accesso in cui vengono distribuiti AWS Proton i servizi. AWS le risorse vengono fornite utilizzando una versione del modello di ambiente richiamata con valori di parametri specifici. Le politiche di accesso vengono fornite in un ruolo di servizio.

  Per ulteriori informazioni, consulta Ambienti.

- Modello di servizio: una raccolta di versioni dei modelli di servizio che possono essere utilizzate per creare AWS Proton servizi.

  Per ulteriori informazioni, consultare Creazione di modelli e pacchetti e Modelli.

- Versione del modello di servizio: una versione specifica di un modello di servizio. Accetta un pacchetto di modelli come input, da un bucket S3 o da un repository Git. Il pacchetto specifica Infrastructure as Code (IaC) e i relativi parametri di input per un servizio. AWS Proton

  Una versione del modello di servizio specifica inoltre questi vincoli sulle istanze di servizio in base alla versione:

  - Modelli di ambiente compatibili: le istanze possono essere eseguite solo in ambienti basati su questi modelli di ambiente compatibili.
  - Fonti di componenti supportate: i tipi di componenti che gli sviluppatori possono associare alle istanze.

  Per ulteriori informazioni, consulta the section called “Versioni”, the section called “Pubblicare” e the section called “Configurazioni di sincronizzazione dei modelli”.

- Servizio: una raccolta di istanze di servizio che eseguono un'applicazione utilizzando le risorse specificate in un modello di servizio e, facoltativamente, una pipeline CI/CD che distribuisce il codice dell'applicazione in queste istanze.

  Nel diagramma, la linea tratteggiata del modello Service indica che il servizio trasmette il modello alle istanze del servizio e alla pipeline.

  Per ulteriori informazioni, consulta Servizi.

- Istanza di servizio: l'insieme di risorse di AWS infrastruttura che eseguono un'applicazione in un ambiente specifico. AWS Proton AWS le risorse vengono fornite utilizzando una versione del modello di servizio richiamata con valori di parametri specifici.

Per ulteriori informazioni, consultare Servizi e the section called “Aggiorna istanza”.

- Pipeline: una pipeline CI/CD opzionale che distribuisce un'applicazione nelle istanze di un servizio, con politiche di accesso per il provisioning di questa pipeline. Le politiche di accesso vengono fornite in un ruolo di servizio. A un servizio non sempre è associata una AWS Proton pipeline: puoi scegliere di gestire le distribuzioni del codice delle app all'esterno di. AWS Proton

  Nel diagramma, la linea tratteggiata di Service e il riquadro tratteggiato che circonda Pipeline indicano che se scegli di gestire autonomamente le tue implementazioni CI/CD, la pipeline potrebbe non essere creata e la tua AWS Proton pipeline potrebbe non essere all'interno del tuo account. AWS

  Per ulteriori informazioni, consultare Servizi e the section called “Aggiorna pipeline”.
- Componente: estensione definita dallo sviluppatore a un'istanza di servizio. Speciifica le risorse AWS infrastrutturali aggiuntive di cui potrebbe aver bisogno una particolare applicazione, oltre alle risorse fornite dall'ambiente e dall'istanza del servizio. I team della piattaforma controllano l'infrastruttura che un componente può fornire attribuendo un ruolo all'ambiente.

  Per ulteriori informazioni, consulta Componenti.

In che modo AWS Proton provvede l'infrastruttura

AWS Proton può fornire l'infrastruttura in diversi modi:

- AWS-managed provisioning: AWS Proton richiama il motore di provisioning per conto dell'utente. Questo metodo supporta solo AWS CloudFormation pacchetti di modelli. Per ulteriori informazioni, consulta the section called “AWS CloudFormation file IAc”.
- CodeBuild provisioning: AWS Proton viene utilizzato AWS CodeBuild per eseguire i comandi di shell forniti dall'utente. I comandi sono in grado di leggere gli input che AWS Proton forniscono e sono responsabili del provisioning o del deprovisioning dell'infrastruttura e della generazione di valori di output. Un pacchetto di modelli per questo metodo include i comandi in un file manifest e tutti i programmi, script o altri file necessari per questi comandi.

  Come esempio di utilizzo del CodeBuild provisioning, è possibile includere codice che utilizza AWS Cloud Development Kit (AWS CDK) per il provisioning AWS delle risorse e un manifest che installa il CDK ed esegue il codice CDK.

  Per ulteriori informazioni, consulta the section called “CodeBuild pacchetto”.

> **Note**
>
> È possibile utilizzare il CodeBuild provisioning con ambienti e servizi. Al momento non è possibile effettuare il provisioning dei componenti in questo modo.

- Provisioning autogestito: AWS Proton invia una pull request (PR) a un repository fornito dall'utente, dove il sistema di implementazione dell'infrastruttura aziendale esegue il processo di provisioning. Questo metodo supporta solo i pacchetti di modelli Terraform. Per ulteriori informazioni, consulta the section called “file Terraform iAC”.

AWS Proton determina e imposta separatamente il metodo di provisioning per ogni ambiente e servizio. Quando si crea o si aggiorna un ambiente o un servizio, AWS Proton esamina il pacchetto di modelli fornito e determina il metodo di provisioning indicato dal pacchetto di modelli. A livello di ambiente, fornisci i parametri di cui l'ambiente e i suoi potenziali servizi potrebbero aver bisogno per i relativi metodi di provisioning: ruoli AWS Identity and Access Management (IAM), una connessione all'account di ambiente o un repository di infrastruttura.

Gli sviluppatori che forniscono AWS Proton un servizio hanno la stessa esperienza indipendentemente dal metodo di provisioning. Gli sviluppatori non devono conoscere il metodo di fornitura e non devono modificare nulla nel processo di fornitura del servizio. Il modello di servizio imposta il metodo di provisioning e ogni ambiente in cui uno sviluppatore distribuisce il servizio fornisce i parametri necessari per il provisioning delle istanze di servizio.

Il diagramma seguente riassume alcune caratteristiche principali dei diversi metodi di provisioning. Le sezioni che seguono la tabella forniscono dettagli su ciascun metodo.

Metodo di provisioning	Modelli	Fornito da	Stato monitorato da
AWS-gestito	manifesto, schema, file IAC () CloudFormation	AWS Proton (tramite CloudFormation)	AWS Proton (attraverso CloudFormation)
CodeBuild	manifest (con comandi), schema, dipendenze dei comandi (ad esempio AWS CDK codice)	AWS Proton (tramite CodeBuild)	AWS Proton (i tuoi comandi restituiscono lo stato tramite CodeBuild)

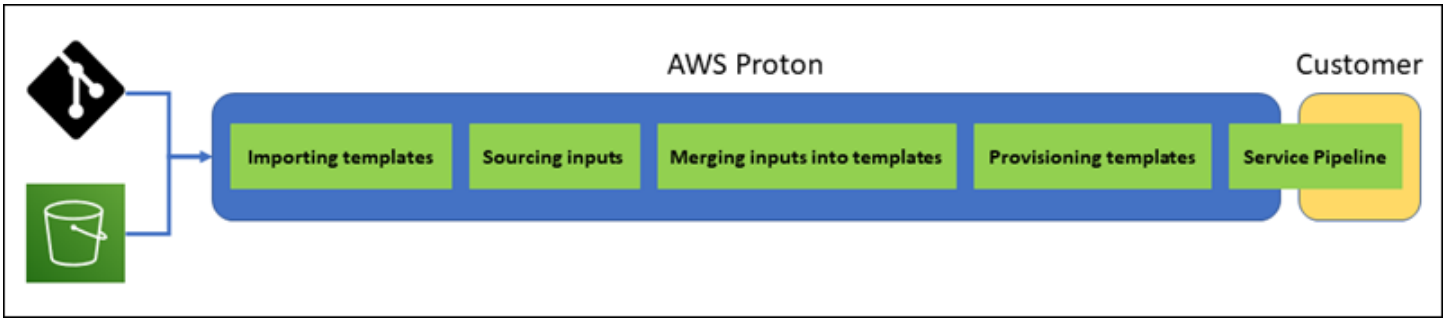
Metodo di provisioning	Modelli	Fornito da	Stato monitorato da
autogestito	manifest, schema, file IAc (Terraform)	Il tuo codice (tramite azioni Git)	Il tuo codice (passato AWS tramite chiamata API)

Come funziona il provisioning AWS gestito

Quando un ambiente o un servizio utilizza il provisioning AWS gestito, l'infrastruttura viene fornita come segue:

1. Un AWS Proton cliente (un amministratore o uno sviluppatore) crea la AWS Proton risorsa (un ambiente o un servizio). Il cliente seleziona un modello per la risorsa e fornisce i parametri richiesti. Per ulteriori informazioni, vedere la sezione seguente,the section called “Considerazioni sul provisioning gestito AWS”.
2. AWS Proton rende un AWS CloudFormation modello completo per il provisioning della risorsa.
3. AWS Proton chiamate AWS CloudFormation per avviare il provisioning utilizzando il modello renderizzato.
4. AWS Proton monitora continuamente la distribuzione. AWS CloudFormation
5. Una volta completato il provisioning, AWS Proton segnala gli errori in caso di guasto e acquisisce gli output del provisioning, come l'ID Amazon VPC, in caso di successo.

Il diagramma seguente mostra che AWS Proton si occupa direttamente della maggior parte di questi passaggi.



Considerazioni sul provisioning gestito AWS

- Ruolo di provisioning dell'infrastruttura: quando un ambiente o una qualsiasi delle istanze di servizio in esso in esecuzione possono utilizzare il provisioning AWS gestito, un amministratore deve configurare un ruolo IAM (direttamente o come parte di una connessione all'account di ambiente). AWS Proton AWS Proton utilizza questo ruolo per fornire l'infrastruttura di queste AWS risorse di provisioning gestite. Il ruolo deve disporre delle autorizzazioni necessarie AWS CloudFormation per creare tutte le risorse incluse nei modelli di tali risorse.

  Per ulteriori informazioni, consultare the section called “Ruoli IAM” e the section called “Esempi di policy relative ai ruoli di servizio”.
- Fornitura del servizio: quando uno sviluppatore implementa un'istanza di servizio che utilizza il provisioning AWS gestito nell'ambiente, AWS Proton utilizza il ruolo assegnato a tale ambiente per fornire l'infrastruttura per l'istanza di servizio. Gli sviluppatori non vedono questo ruolo e non possono cambiarlo.
- Servizio con pipeline: un modello di servizio che utilizza AWS-managed provisioning può includere una definizione di pipeline scritta nello schema YAML. AWS CloudFormation AWS Proton crea anche la pipeline chiamando. AWS CloudFormation Il ruolo AWS Proton utilizzato per creare una pipeline è separato dal ruolo per ogni singolo ambiente. Questo ruolo viene assegnato AWS Proton separatamente, una sola volta a livello di AWS account, e viene utilizzato per il provisioning e la gestione di tutte le pipeline AWS gestite. Questo ruolo dovrebbe avere le autorizzazioni per creare pipeline e altre risorse necessarie alle tue pipeline.

  Le seguenti procedure mostrano come fornire il ruolo della pipeline a. AWS Proton

  AWS Proton console

  Per fornire il ruolo della pipeline

  1. Nella AWS Proton console, nel riquadro di navigazione, scegli Impostazioni > Impostazioni account, quindi scegli Configura.
  2. Utilizza la sezione Pipeline AWS-managed role per configurare un ruolo pipeline nuovo o esistente per AWS il provisioning gestito.

  AWS Proton API

  Per fornire il ruolo della pipeline

  1. Usa l'azione UpdateAccountSettingsAPI.

2. Fornisci l'Amazon Resource Name (ARN) del tuo ruolo di servizio di pipeline nel parametro. `pipelineServiceRoleArn`

AWS CLI

Per fornire il ruolo della pipeline

Esegui il comando seguente:

```
$ aws proton update-account-settings \
    --pipeline-service-role-arn \
        "arn:aws:iam::123456789012:role/my-pipeline-role"
```

Come funziona il CodeBuild provisioning

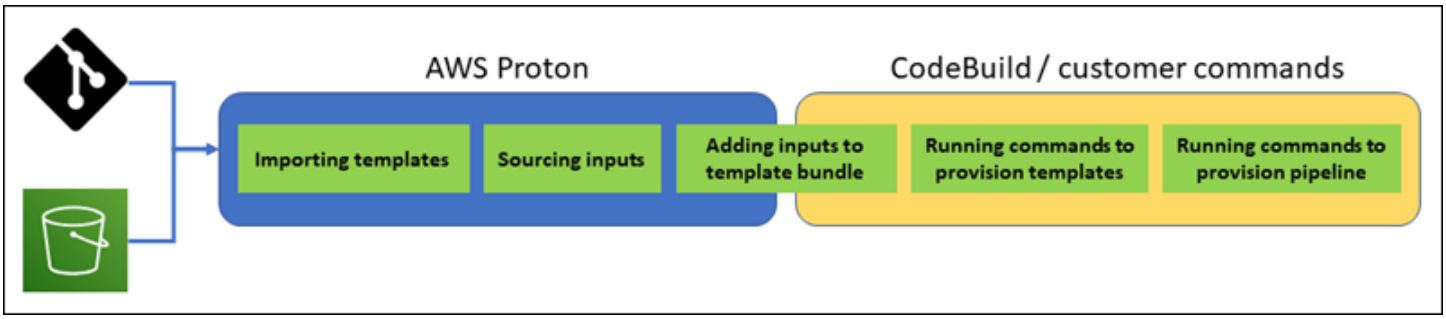
Quando un ambiente o un servizio utilizza il CodeBuild provisioning, l'infrastruttura viene fornita come segue:

1. Un AWS Proton cliente (un amministratore o uno sviluppatore) crea la AWS Proton risorsa (un ambiente o un servizio). Il cliente seleziona un modello per la risorsa e fornisce i parametri richiesti. Per ulteriori informazioni, vedere la sezione seguente,the section called “Considerazioni per il provisioning CodeBuild ”.
2. AWS Proton esegue il rendering di un file di input con i valori dei parametri di input per il provisioning della risorsa.
3. AWS Proton chiamate CodeBuild per iniziare un lavoro. Il CodeBuild job esegue i comandi della shell del cliente specificati nel modello. Questi comandi forniscono l'infrastruttura desiderata, mentre opzionalmente leggono i valori di input.
4. Al termine del provisioning, il comando del cliente finale restituisce lo stato di approvvigionamento CodeBuild e richiama l'azione dell'NotifyResourceDeploymentStatusChange AWS Proton API per fornire output, come l'ID Amazon VPC, se presenti.

> **Important**
>
> Assicurati che i comandi restituiscano correttamente lo stato di provisioning e forniscano gli output. CodeBuild In caso contrario, non sono in AWS Proton grado di tracciare

correttamente lo stato del provisioning e non possono fornire output corretti alle istanze di servizio.

Il diagramma seguente illustra i passaggi eseguiti e i passaggi AWS Proton eseguiti dai comandi all'interno di un processo. CodeBuild



Considerazioni per il provisioning CodeBuild

- Ruolo di provisioning dell'infrastruttura: quando un ambiente o una qualsiasi delle istanze di servizio in esso eseguite possono utilizzare il provisioning CodeBuild basato, un amministratore deve configurare un ruolo IAM (direttamente o come parte di una AWS Proton connessione all'account di ambiente). AWS Proton utilizza questo ruolo per fornire l'infrastruttura di queste CodeBuild risorse di provisioning. Il ruolo deve disporre delle autorizzazioni necessarie CodeBuild per creare tutte le risorse fornite dai comandi nei modelli di tali risorse.

  Per ulteriori informazioni, consultare the section called “Ruoli IAM” e the section called “Esempi di policy relative ai ruoli di servizio”.
- Fornitura del servizio: quando uno sviluppatore implementa un'istanza di servizio che utilizza il CodeBuild provisioning all'ambiente, AWS Proton utilizza il ruolo assegnato a tale ambiente per fornire l'infrastruttura per l'istanza di servizio. Gli sviluppatori non vedono questo ruolo e non possono cambiarlo.
- Servizio con pipeline: un modello di servizio che utilizza il CodeBuild provisioning può includere comandi per il provisioning di una pipeline. AWS Proton crea anche la pipeline chiamando. CodeBuild Il ruolo AWS Proton utilizzato per creare una pipeline è separato dal ruolo per ogni singolo ambiente. Questo ruolo viene assegnato AWS Proton separatamente, una sola volta a livello di AWS account, e viene utilizzato per il provisioning e la gestione di tutte le pipeline CodeBuild basate. Questo ruolo dovrebbe avere le autorizzazioni per creare pipeline e altre risorse necessarie alle tue pipeline.

Le seguenti procedure mostrano come fornire il ruolo della pipeline a. AWS Proton

AWS Proton console

Per fornire il ruolo della pipeline

1. Nella AWS Proton console, nel riquadro di navigazione, scegli Impostazioni > Impostazioni account, quindi scegli Configura.
2. Utilizza la sezione Codebuild pipeline provisioning role per configurare un ruolo pipeline nuovo o esistente per il provisioning. CodeBuild

AWS Proton API

Per fornire il ruolo della pipeline

1. Usa l'azione UpdateAccountSettingsAPI.
2. Fornisci l'Amazon Resource Name (ARN) del tuo ruolo di servizio di pipeline nel parametro. `pipelineCodebuildRoleArn`

AWS CLI

Per fornire il ruolo della pipeline

Esegui il comando seguente:

```
$ aws proton update-account-settings \
    --pipeline-codebuild-role-arn \
        "arn:aws:iam::123456789012:role/my-pipeline-role"
```

Come funziona il provisioning autogestito

Quando un ambiente è configurato per utilizzare il provisioning autogestito, l'infrastruttura viene fornita come segue:

1. Un AWS Proton cliente (un amministratore o uno sviluppatore) crea la AWS Proton risorsa (un ambiente o un servizio). Il cliente seleziona un modello per la risorsa e fornisce i parametri richiesti. Per un ambiente, il cliente fornisce anche un repository di infrastruttura collegata.

Per ulteriori informazioni, vedere la sezione seguente,the section called “Considerazioni per il provisioning autogestito”.

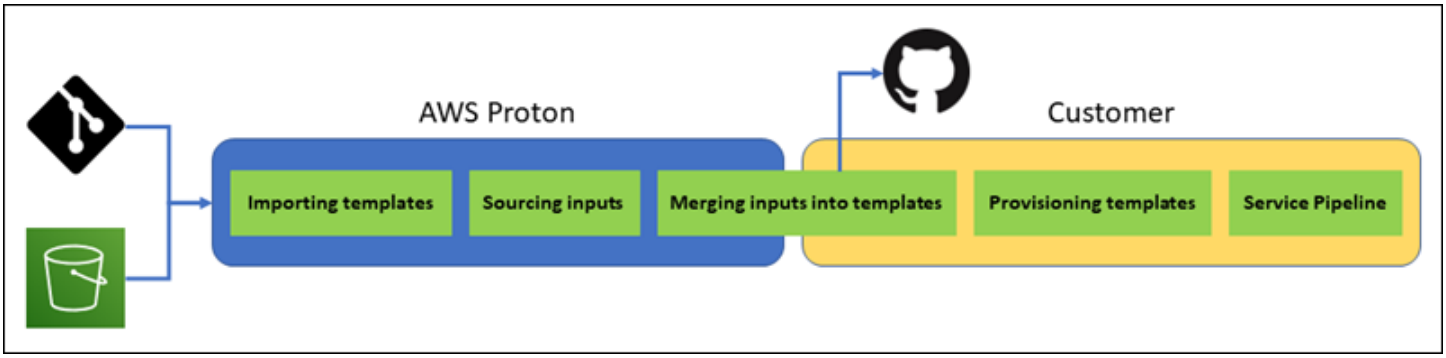
2. AWS Proton rende un modello Terraform completo. Consiste in uno o più file Terraform, potenzialmente in più cartelle, e un `.tfvars` file di variabili. AWS Proton scrive i valori dei parametri forniti durante la chiamata di creazione delle risorse in questo file di variabili.
3. AWS Proton invia un PR al repository dell'infrastruttura con il modello Terraform renderizzato.
4. Quando il cliente (amministratore o sviluppatore) unisce il PR, l'automazione del cliente attiva il motore di provisioning per avviare il provisioning dell'infrastruttura utilizzando il modello unito.

> **Note**
>
> Se il cliente (amministratore o sviluppatore) chiude il PR, AWS Proton riconosce il PR come chiuso e contrassegna l'implementazione come annullata.

5. Una volta completato il provisioning, l'automazione del cliente richiama l'azione dell'NotifyResourceDeploymentStatusChange AWS Proton API per indicare il completamento, fornire lo stato (successo o fallimento) e fornire output, come l'ID Amazon VPC, se presenti.

> **Important**
>
> Assicurati che il codice di automazione richiami lo stato e AWS Proton gli output del provisioning. In caso contrario, AWS Proton potresti considerare il provisioning in sospeso più a lungo del dovuto e continuare a mostrare lo stato In corso.

Il diagramma seguente illustra i passaggi eseguiti e i passaggi AWS Proton eseguiti dal proprio sistema di provisioning.



Considerazioni per il provisioning autogestito

- Archivio dell'infrastruttura: quando un amministratore configura un ambiente per il provisioning autogestito, deve fornire un repository di infrastruttura collegato. AWS Proton viene inviato PRs a questo repository per fornire l'infrastruttura dell'ambiente e tutte le istanze di servizio che vi vengono distribuite. L'azione di automazione di proprietà del cliente nel repository deve assumere un ruolo IAM con le autorizzazioni per creare tutte le risorse incluse nell'ambiente e nei modelli di servizio e un'identità che rifletta l'account di destinazione. AWS Per un esempio di GitHub azione che presuppone un ruolo, consulta Assumere un ruolo nella documentazione «Configure AWS Credentials» Action For Actions. GitHub
- Autorizzazioni: il codice di provisioning deve autenticarsi con un account secondo necessità (ad esempio, autenticarsi su un AWS account) e fornire l'autorizzazione per l'approvvigionamento delle risorse (ad esempio, fornire un ruolo).
- Fornitura del servizio: quando uno sviluppatore implementa un'istanza di servizio che utilizza il provisioning autogestito all'ambiente, AWS Proton invia un PR al repository associato all'ambiente per fornire l'infrastruttura per l'istanza del servizio. Gli sviluppatori non vedono il repository e non possono modificarlo.

  > **Note**
  >
  > Gli sviluppatori che creano servizi utilizzano lo stesso processo indipendentemente dal metodo di provisioning e la differenza viene astratta da essi. Tuttavia, con il provisioning autogestito, gli sviluppatori potrebbero riscontrare una risposta più lenta, poiché devono attendere che qualcuno (che potrebbe non essere loro stesso) unisca il PR nell'archivio dell'infrastruttura prima di iniziare il provisioning.

- Servizio con pipeline: un modello di servizio per un ambiente con provisioning autogestito può includere una definizione di pipeline (ad esempio, una pipeline), scritta in Terraform HCL. AWS CodePipeline Per consentire il provisioning AWS Proton di queste pipeline, un amministratore fornisce un repository di pipeline collegato a. AWS Proton Quando si effettua il provisioning di una pipeline, l'azione di automazione di proprietà del cliente nell'archivio deve assumere un ruolo IAM con le autorizzazioni per il provisioning della pipeline e un'identità che rifletta l'account di destinazione. AWS L'archivio e il ruolo della pipeline sono separati da quelli utilizzati per ogni singolo ambiente. L'archivio collegato viene fornito AWS Proton separatamente, una sola volta a livello di AWS account, e viene utilizzato per il provisioning e la gestione di tutte le pipeline. Il ruolo deve avere le autorizzazioni per creare pipeline e altre risorse necessarie alle tue pipeline.

Le seguenti procedure mostrano come fornire l'archivio e il ruolo della pipeline a. AWS Proton

AWS Proton console

Per fornire il ruolo della pipeline

1. Nella AWS Proton console, nel riquadro di navigazione, scegli Impostazioni > Impostazioni account, quindi scegli Configura.
2. Utilizza la sezione CI/CD pipeline repository per configurare un collegamento al repository nuovo o esistente.

AWS Proton API

Per fornire il ruolo della pipeline

1. Usa l'azione UpdateAccountSettingsAPI.
2. Fornisci il provider, il nome e il ramo del tuo repository di pipeline nel `pipelineProvisioningRepository` parametro.

AWS CLI

Per fornire il ruolo della pipeline

Esegui il comando seguente:

```
$ aws proton update-account-settings \
    --pipeline-provisioning-repository \
        "provider=GITHUB,name=my-pipeline-repo-name,branch=my-branch"
```

- Eliminazione delle risorse fornite autogestite: i moduli Terraform possono includere elementi di configurazione necessari per il funzionamento di Terraform, oltre alle definizioni delle risorse. Pertanto, non è AWS Proton possibile eliminare tutti i file Terraform per un'istanza di ambiente o servizio. Invece, AWS Proton contrassegna i file per l'eliminazione e ha aggiornato un flag nei metadati PR. La tua automazione può leggere quel flag e usarlo per attivare un comando terraform destroy.

AWS Proton terminologia

Modello di ambiente

Definisce l'infrastruttura condivisa, ad esempio un VPC o un cluster, utilizzata da più applicazioni o risorse.

Pacchetto di modelli di ambiente

Una raccolta di file caricati per creare e registrare un modello di ambiente. AWS Proton Un pacchetto di modelli di ambiente contiene quanto segue:

1. Un file di schema che definisce l'infrastruttura come parametri di input del codice.
2. Un file Infrastructure as Code (IaC) che definisce un'infrastruttura condivisa, ad esempio un VPC o un cluster, utilizzato da più applicazioni o risorse.
3. Un file manifesto che elenca il file IAc.

Ambiente

Infrastruttura condivisa fornita, ad esempio un VPC o un cluster, utilizzata da più applicazioni o risorse.

Modello di servizio

Definisce il tipo di infrastruttura necessaria per distribuire e mantenere un'applicazione o un microservizio in un ambiente.

Pacchetto di modelli di servizio

Una raccolta di file caricati per creare e registrare un modello di servizio. AWS Proton Un pacchetto di modelli di servizio contiene quanto segue:

1. Un file di schema che definisce i parametri di input dell'infrastruttura come codice (IaC).
2. Un file IaC che definisce l'infrastruttura necessaria per distribuire e gestire un'applicazione o un microservizio in un ambiente.
3. Un file manifesto che elenca il file IAc.
4. Facoltativo
   a. Un file IaC che definisce l'infrastruttura della pipeline di servizi.
   b. Un file manifest che elenca il file IAc.

Servizio

Infrastruttura fornita necessaria per distribuire e gestire un'applicazione o un microservizio in un ambiente.

Istanza del servizio

Infrastruttura predisposta che supporta un'applicazione o un microservizio in un ambiente.

Pipeline di servizi

Infrastruttura predisposta che supporta una pipeline.

Versione del modello

Una versione principale o secondaria di un modello. Per ulteriori informazioni, consulta Modelli con versione.

Parametri di input

Definito in un file di schema e utilizzato in un file Infrastructure as Code (IaC) in modo che il file IAc possa essere utilizzato ripetutamente e per una varietà di casi d'uso.

File di schema

Definisce l'infrastruttura come parametri di input del file di codice.

File Spec

Speciifica i valori per l'infrastruttura come parametri di input del file di codice, come definito in un file di schema.

File manifest

Elenca un'infrastruttura come file di codice.

Creazione di modelli e creazione di pacchetti per AWS Proton

AWS Proton fornisce risorse per te in base ai file Infrastructure as Code (IaC). Descrivi l'infrastruttura in file IAc riutilizzabili. Per rendere i file riutilizzabili per ambienti e applicazioni diversi, li si crea come modelli, si definiscono i parametri di input e si utilizzano questi parametri nelle definizioni IAc. Quando successivamente si crea una risorsa di provisioning (ambiente, istanza di servizio o componente), si AWS Proton utilizza un motore di rendering, che combina i valori di input con un modello per creare un file IAc pronto per il provisioning.

Gli amministratori creano la maggior parte dei modelli come pacchetti di modelli, quindi li caricano e li registrano. AWS Proton Il resto di questa pagina illustra questi pacchetti di modelli. AWS Proton I componenti definiti direttamente sono un'eccezione: gli sviluppatori li creano e forniscono direttamente i file modello IAc. Per ulteriori informazioni sui componenti, vedere. Componenti

Argomenti

- Pacchetti di modelli
- AWS Proton parametri
- AWS Proton infrastruttura come file di codice
- File di schema
- Raccogli i file modello per AWS Proton
- Considerazioni sul pacchetto di modelli

Pacchetti di modelli

In qualità di amministratore, puoi creare e registrare modelli con AWS Proton. Questi modelli vengono utilizzati per creare ambienti e servizi. Quando crei un servizio, effettua il AWS Proton provisioning e distribuisce le istanze del servizio in ambienti selezionati. Per ulteriori informazioni, consulta AWS Proton per i team di piattaforma.

Per creare e registrare un modello AWS Proton, è necessario caricare un pacchetto di modelli contenente i file Infrastructure as Code (IaC) AWS Proton necessari per il provisioning di un ambiente o di un servizio.

Un pacchetto di modelli contiene quanto segue:

- Un file Infrastructure as code (IaC) con un file YAML manifesto che elenca il file IaC.
- Un file YAML di schema per le definizioni dei parametri di input del file IAc.

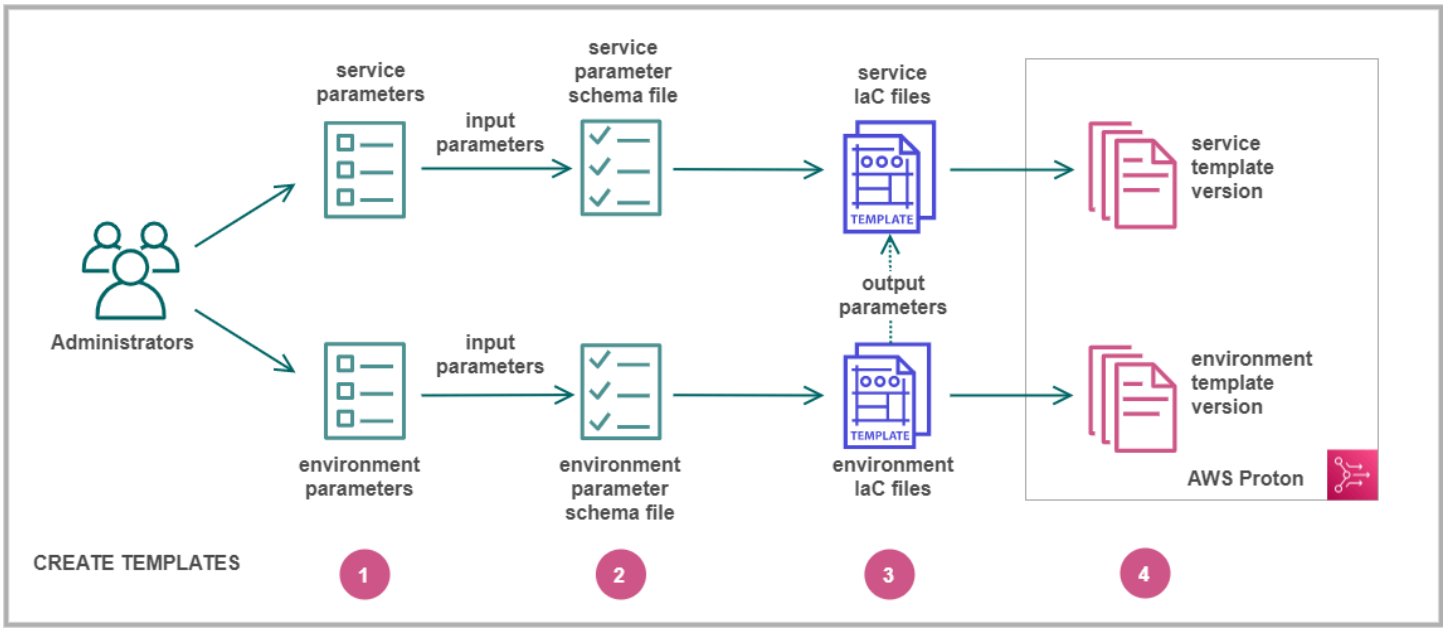
Un pacchetto di modelli di CloudFormation ambiente contiene un file IAc.

Un pacchetto CloudFormation di modelli di servizio contiene un file IAc per le definizioni delle istanze di servizio e un altro file IaC opzionale per la definizione di una pipeline.

I pacchetti di modelli di ambiente e servizio Terraform possono contenere più file IAc ciascuno.

AWS Proton richiede un file di schema dei parametri di input. Quando lo usi AWS CloudFormation per creare i tuoi file IAc, usi la sintassi Jinja per fare riferimento ai parametri di input. AWS Proton fornisce namespace di parametri che è possibile utilizzare per fare riferimento ai parametri nei file IAc.

Il diagramma seguente mostra un esempio di passaggi che è possibile eseguire per creare un modello per. AWS Proton



1

Identifi

i parametri di input.

2

Crea

un file di schema per definire i parametri di input.

3 Crea file IAc che fanno riferimento ai tuoi parametri di input. È possibile fare riferimento agli output dei file IAc di ambiente come input per i file IAc di servizio.

4 Registr una versione modello con AWS Proton e carica il tuo pacchetto di modelli.

AWS Proton parametri

È possibile definire e utilizzare i parametri nella propria infrastruttura come file di codice (IaC) per renderli flessibili e riutilizzabili. Leggete il valore di un parametro nei vostri file IAc facendo riferimento al nome del parametro nello spazio dei nomi dei AWS Proton parametri. AWS Proton inserisce i valori dei parametri nei file IAc renderizzati che genera durante l'approvvigionamento delle risorse. Per elaborare i parametri AWS CloudFormation IAc, utilizza Jinja. AWS Proton Per elaborare i parametri Terraform IAc, AWS Proton genera un file di valori dei parametri Terraform e si basa sulla capacità di parametrizzazione integrata in HCL.

ConCodeBuild approvvigionamento, AWS Proton genera un file di input che il codice può importare. Il file è un file JSON o HCL, a seconda di una proprietà nel manifesto del modello. Per ulteriori informazioni, consulta the section called “CodeBuild parametri di provisioning”.

È possibile fare riferimento ai parametri nei file IAc di ambiente, servizio e componente o al codice di provisioning con i seguenti requisiti:

- La lunghezza di ogni nome di parametro non supera i 100 caratteri.
- La lunghezza dello spazio dei nomi dei parametri e del nome della risorsa combinati non supera il limite di caratteri per il nome della risorsa.

AWS Proton il provisioning fallisce se queste quote vengono superate.

Tipi di parametro

I seguenti tipi di parametri sono disponibili come riferimento nei AWS Proton file IAc:

Parametro di input

Gli ambienti e le istanze di servizio possono accettare parametri di input definiti in un file di schema associato all'ambiente o al modello di servizio. È possibile fare riferimento ai parametri di

input di una risorsa nel file IAc della risorsa. I file IAc dei componenti possono fare riferimento ai parametri di input dell'istanza del servizio a cui è collegato il componente.

AWS Proton confronta i nomi dei parametri di input con il file di schema e li confronta con i parametri a cui si fa riferimento nei file IAc per inserire i valori di input forniti in un file di specifiche durante l'approvvigionamento delle risorse.

Parametro di output

È possibile definire gli output in qualsiasi file IAc. Un output può essere, ad esempio, un nome, un ID o un ARN di una delle risorse fornite dal modello oppure può essere un modo per passare attraverso uno degli input del modello. È possibile fare riferimento a questi output nei file IAc di altre risorse.

Nei file CloudFormation IAc, definite i parametri di output nel blocco. `Outputs:` In un file Terraform IAc, definisci ogni parametro di output utilizzando un'istruzione. `output`

Parametro di risorsa

AWS Proton crea automaticamente i parametri AWS Proton delle risorse. Questi parametri espongono le proprietà dell'oggetto AWS Proton risorsa. Un esempio di parametro di risorsa è`environment.name`.

Utilizzo AWS Proton dei parametri nei file IAc

Per leggere il valore di un parametro in un file IAc, fate riferimento al nome del parametro nello spazio AWS Proton dei nomi dei parametri. Per i file AWS CloudFormation IAc, utilizzate la sintassi Jinja e racchiudete il parametro con coppie di parentesi graffe e virgolette.

La tabella seguente mostra la sintassi di riferimento per ogni linguaggio di template supportato, con un esempio.

Linguaggio dei modelli	Sintassi	Esempio: input di ambiente denominato «VPC»
CloudFormation	`"{{ parameter-name }}"`	`"{{ environment.inputs.VPC }}"`
Terraform	`var.parameter-name`	`var.environment.inputs.VPC` Definizioni delle variabili Terraform generate

> **Note**
>
> Se utilizzi parametri CloudFormation dinamici nel tuo file IAc, devi evitarli per evitare errori di interpretazione errata di Jinja. Per ulteriori informazioni, consulta Risoluzione dei problemi AWS Proton

La tabella seguente elenca i nomi dei namespace per tutti i parametri delle risorse. AWS Proton Ogni tipo di file modello può utilizzare un sottoinsieme diverso dello spazio dei nomi dei parametri.

File modello	Tipo parametro	Nome del parametro	Descrizione
Ambiente	risorsa	`environment.` **`name`**	Nome ambiente
	input	`environment.inputs.` *`input-name`*	Input di ambiente definiti dallo schema
Servizio	risorsa	`environment.` **`name`** `environment.` **`account_id`**	Nome e ID dell'ambiente Account AWS
	output	`environment.outputs.` *`output-name`*	Output di file IAc di ambiente
	risorsa	`service.`**`branch_name`** `service.`**`name`** `service.`**`repository_connection_arn`** `service.`**`repository_id`**	Nome del servizio e archivio del codice
	risorsa	`service_instance.` **`name`**	Nome dell'istanza del servizio
	input	`service_instance.inputs.` *`input-name`*	Input dell'istanza di servizio definiti dallo schema

File modello	Tipo parametro	Nome del parametro	Descrizione
	risorsa	`service_instance.components.default.` **`name`**	Nome del componente predefinito allegato
	output	`service_instance.components.default.outputs.` *`output-name`*	Uscite di file IAc del componente predefinito allegate
Pipeline	risorsa	`service_instance.environment.` **`name`** `service_instance.environment.` **`account_id`**	Nome e ID dell'ambiente dell'istanza di servizio Account AWS
	output	`service_instance.environment.outputs.` *`output-name`*	Output dei file IAc dell'ambiente di istanza di servizio
	input	`pipeline.inputs.` *`input-name`*	Ingressi della pipeline definiti dallo schema
	risorsa	`service.`**`branch_name`** `service.`**`name`** `service.`**`repository_connection_arn`** `service.`**`repository_id`**	Nome del servizio e archivio del codice
	input	`service_instance.inputs.` *`input-name`*	Input delle istanze di servizio definiti dallo schema

File modello	Tipo parametro	Nome del parametro	Descrizione
	collezione	`{% for service_instance in service_instances %}...{% endfor %}`	Una raccolta di istanze di servizio che è possibile consultare in loop
Componente	risorsa	`environment.` **`name`** `environment.` **`account_id`**	Nome dell'ambiente e ID Account AWS dell'account
	output	`environment.outputs.` *`output-name`*	Output di file IAc di ambiente
	risorsa	`service.`**`branch_name`** `service.`**`name`** `service.`**`repository_connection_arn`** `service.`**`repository_id`**	Nome del servizio e archivio del codice (componenti allegati)
	risorsa	`service_instance.` **`name`**	Nome dell'istanza del servizio (componenti allegati)
	input	`service_instance.inputs.` *`input-name`*	Input dell'istanza di servizio definiti dallo schema (componenti collegati)
	risorsa	`component.` **`name`**	Nome componente

Per ulteriori informazioni ed esempi, vedere i sottoargomenti relativi ai parametri nei file modello IAc per diversi tipi di risorse e linguaggi di modello.

Argomenti

- Dettagli ed esempi dei parametri del file Environment CloudFormation IAc
- Dettagli ed esempi dei parametri del file Service CloudFormation IAc
- Dettagli ed esempi dei parametri del file CloudFormation IAc del componente
- filtri parametrici per CloudFormation file IAc
- CodeBuild dettagli ed esempi dei parametri di provisioning
- Dettagli ed esempi dei parametri del file Terraform infrastructure as code (IaC)

Dettagli ed esempi dei parametri del file Environment CloudFormation IAc

È possibile definire e fare riferimento ai parametri nell'infrastruttura ambientale come file di codice (IaC). Per una descrizione dettagliata dei AWS Proton parametri, dei tipi di parametri, dello spazio dei nomi dei parametri e di come utilizzare i parametri nei file IAc, vedere. the section called “Parametri”

Definire i parametri dell'ambiente

È possibile definire i parametri di input e output per i file IAc di ambiente.

- Parametri di input: definisci i parametri di input dell'ambiente nel tuo file di schema.

  L'elenco seguente include esempi di parametri di input ambientali per casi d'uso tipici.

  - Valori CIDR VPC
  - Impostazioni del bilanciamento del carico
  - Impostazioni del database
  - Un timeout per il controllo dello stato di salute

  In qualità di amministratore, puoi fornire valori per i parametri di input quando crei un ambiente:

  - Utilizza la console per compilare un modulo basato su schemi che AWS Proton fornisce.
  - Utilizza la CLI per fornire una specifica che includa i valori.
- Parametri di output: definisci gli output dell'ambiente nei file IAc del tuo ambiente. È quindi possibile fare riferimento a questi output nei file IAc di altre risorse.

Leggi i valori dei parametri nei file IAc di ambiente

È possibile leggere i parametri relativi all'ambiente nei file IAc di ambiente. Si legge il valore di un parametro facendo riferimento al nome del parametro nello spazio dei nomi del AWS Proton parametro.

- Parametri di input: legge un valore di input di ambiente facendo riferimento. `environment.inputs.`*`input-name`*
- Parametri delle AWS Proton risorse: leggi i parametri delle risorse facendo riferimento a nomi come. `environment.name`

> **Note**
>
> Nessun parametro di output di altre risorse è disponibile per i file IAc di ambiente.

File IAc di ambiente e servizio di esempio con parametri

L'esempio seguente mostra la definizione e il riferimento dei parametri in un file IAc di ambiente. L'esempio mostra quindi come è possibile fare riferimento ai parametri di output dell'ambiente definiti nel file IAc di ambiente in un file IAc di servizio.

Example File di ambiente IAc CloudFormation

Nota quanto segue in questo esempio:

- Lo spazio dei `environment.inputs.` nomi si riferisce ai parametri di input dell'ambiente.
- Il parametro Amazon EC2 Systems Manager (SSM) `StoreInputValue` concatena gli input dell'ambiente.
- L'`MyEnvParameterValue`output espone la stessa concatenazione di parametri di input di un parametro di output. Tre parametri di output aggiuntivi espongono inoltre i parametri di input singolarmente.
- Sei parametri di output aggiuntivi espongono le risorse messe a disposizione dall'ambiente.

```
Resources:
  StoreInputValue:
    Type: AWS::SSM::Parameter
    Properties:
      Type: String
      Value: "{{ environment.inputs.my_sample_input }}
 {{ environment.inputs.my_other_sample_input}}
 {{ environment.inputs.another_optional_input }}"
              # input parameter references
```

```
# These output values are available to service infrastructure as code files as outputs,
 when given the
# the 'environment.outputs' namespace, for example,
 service_instance.environment.outputs.ClusterName.
Outputs:
  MyEnvParameterValue:                                    # output definition
    Value: !GetAtt StoreInputValue.Value
  MySampleInputValue:                                     # output definition
    Value: "{{ environment.inputs.my_sample_input }}"         #   input parameter
 reference
  MyOtherSampleInputValue:                                # output definition
    Value: "{{ environment.inputs.my_other_sample_input }}"   #   input parameter
 reference
  AnotherOptionalInputValue:                              # output definition
    Value: "{{ environment.inputs.another_optional_input }}"  #   input parameter
 reference
  ClusterName:                                            # output definition
    Description: The name of the ECS cluster
    Value: !Ref 'ECSCluster'                              #   provisioned resource
  ECSTaskExecutionRole:                                   # output definition
    Description: The ARN of the ECS role
    Value: !GetAtt 'ECSTaskExecutionRole.Arn'             #   provisioned resource
  VpcId:                                                  # output definition
    Description: The ID of the VPC that this stack is deployed in
    Value: !Ref 'VPC'                                     #   provisioned resource
  PublicSubnetOne:                                        # output definition
    Description: Public subnet one
    Value: !Ref 'PublicSubnetOne'                         #   provisioned resource
  PublicSubnetTwo:                                        # output definition
    Description: Public subnet two
    Value: !Ref 'PublicSubnetTwo'                         #   provisioned resource
  ContainerSecurityGroup:                                 # output definition
    Description: A security group used to allow Fargate containers to receive traffic
    Value: !Ref 'ContainerSecurityGroup'                  #   provisioned resource
```

Example File CloudFormation IAc di servizio

Il `environment.outputs.` namespace si riferisce agli output di ambiente di un file IAc di ambiente. Ad esempio, il nome `environment.outputs.ClusterName` legge il valore del parametro di output dell'ambiente. `ClusterName`

```
AWSTemplateFormatVersion: '2010-09-09'
```

```
Description: Deploy a service on AWS Fargate, hosted in a public subnet, and accessible
 via a public load balancer.
Mappings:
  TaskSize:
    x-small:
      cpu: 256
      memory: 512
    small:
      cpu: 512
      memory: 1024
    medium:
      cpu: 1024
      memory: 2048
    large:
      cpu: 2048
      memory: 4096
    x-large:
      cpu: 4096
      memory: 8192
Resources:
  # A log group for storing the stdout logs from this service's containers
  LogGroup:
    Type: AWS::Logs::LogGroup
    Properties:
      LogGroupName: '{{service_instance.name}}' # resource parameter

  # The task definition. This is a simple metadata description of what
  # container to run, and what resource requirements it has.
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: '{{service_instance.name}}' # resource parameter
      Cpu: !FindInMap [TaskSize, {{service_instance.inputs.task_size}}, cpu] # input
 parameter
      Memory: !FindInMap [TaskSize, {{service_instance.inputs.task_size}}, memory]
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      ExecutionRoleArn: '{{environment.outputs.ECSTaskExecutionRole}}'  # output
 reference to an environment infrastructure code file
      TaskRoleArn: !Ref "AWS::NoValue"
      ContainerDefinitions:
        - Name: '{{service_instance.name}}'  # resource parameter
          Cpu: !FindInMap [TaskSize, {{service_instance.inputs.task_size}}, cpu]
```

```
          Memory: !FindInMap [TaskSize, {{service_instance.inputs.task_size}}, memory]
          Image: '{{service_instance.inputs.image}}'
          PortMappings:
            - ContainerPort: '{{service_instance.inputs.port}}' # input parameter
          LogConfiguration:
            LogDriver: 'awslogs'
            Options:
              awslogs-group: '{{service_instance.name}}' # resource parameter
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: '{{service_instance.name}}' # resource parameter

  # The service_instance. The service is a resource which allows you to run multiple
  # copies of a type of task, and gather up their logs and metrics, as well
  # as monitor the number of running tasks and replace any that have crashed
  Service:
    Type: AWS::ECS::Service
    DependsOn: LoadBalancerRule
    Properties:
      ServiceName: '{{service_instance.name}}'  # resource parameter
      Cluster: '{{environment.outputs.ClusterName}}' # output reference to an
 environment infrastructure as code file
      LaunchType: FARGATE
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 75
      DesiredCount: '{{service_instance.inputs.desired_count}}'# input parameter
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - '{{environment.outputs.ContainerSecurityGroup}}' # output reference to an
 environment infrastructure as code file
          Subnets:
            - '{{environment.outputs.PublicSubnetOne}}' # output reference to an
 environment infrastructure as code file
            - '{{environment.outputs.PublicSubnetTwo}}' # output reference to an
 environment infrastructure as code file
      TaskDefinition: !Ref 'TaskDefinition'
      LoadBalancers:
        - ContainerName: '{{service_instance.name}}'  # resource parameter
          ContainerPort: '{{service_instance.inputs.port}}' # input parameter
          TargetGroupArn: !Ref 'TargetGroup'
[...]
```

Dettagli ed esempi dei parametri del file Service CloudFormation IAc

È possibile definire e fare riferimento ai parametri nell'infrastruttura di servizio e di pipeline come file di codice (IaC). Per una descrizione dettagliata dei AWS Proton parametri, dei tipi di parametri, dello spazio dei nomi dei parametri e di come utilizzare i parametri nei file IAc, vedere. the section called “Parametri”

Definire i parametri del servizio

È possibile definire i parametri di input e output per i file IAc di servizio.

- Parametri di input: definisci i parametri di input dell'istanza di servizio nel file di schema.

  L'elenco seguente include esempi di parametri di input del servizio per casi d'uso tipici.

  - Porta
  - Dimensioni processo
  - Immagine
  - Conteggio desiderato
  - File Docker
  - Comando di test unitario

  Fornisci valori per i parametri di input quando crei un servizio:

  - Utilizza la console per compilare un modulo basato su schemi che AWS Proton fornisce.
  - Utilizza la CLI per fornire una specifica che includa i valori.
- Parametri di output: definisci gli output delle istanze di servizio nei file IAc del servizio. È quindi possibile fare riferimento a questi output nei file IAc di altre risorse.

Leggi i valori dei parametri nei file IAc di servizio

È possibile leggere i parametri relativi al servizio e ad altre risorse nei file IAc di servizio. È possibile leggere il valore di un parametro facendo riferimento al nome del parametro nello spazio dei nomi del AWS Proton parametro.

- Parametri di input: legge il valore di input di un'istanza di servizio facendo riferimento. `service_instance.inputs.input-name`
- Parametri delle AWS Proton risorse: legge i parametri delle risorse facendo riferimento a nomi come `service.nameservice_instance.name`, e. `environment.name`

- Parametri di output: leggi gli output di altre risorse facendo riferimento `environment.outputs.`*`output-name`* a o. `service_instance.components.default.outputs.`*`output-name`*

Esempio di file IAc di servizio con parametri

L'esempio seguente è un frammento di un file IAc di servizio CloudFormation . Il `environment.outputs.` namespace si riferisce agli output del file IAc di ambiente. Lo spazio dei `service_instance.inputs.` nomi si riferisce ai parametri di input dell'istanza di servizio. La `service_instance.name` proprietà si riferisce a un parametro di AWS Proton risorsa.

```
Resources:
  StoreServiceInstanceInputValue:
    Type: AWS::SSM::Parameter
    Properties:
      Type: String
      Value: "{{ service.name }} {{ service_instance.name }}
 {{ service_instance.inputs.my_sample_service_instance_required_input }}
 {{ service_instance.inputs.my_sample_service_instance_optional_input }}
 {{ environment.outputs.MySampleInputValue }}
 {{ environment.outputs.MyOtherSampleInputValue }}"
              #  resource parameter references               # input parameter
 references
                                          # output references to an environment
 infrastructure as code file
Outputs:
  MyServiceInstanceParameter:                                                  #
 output definition
    Value: !Ref StoreServiceInstanceInputValue
  MyServiceInstanceRequiredInputValue:                                         #
 output definition
    Value: "{{ service_instance.inputs.my_sample_service_instance_required_input }}"  #
 input parameter reference
  MyServiceInstanceOptionalInputValue:                                         #
 output definition
    Value: "{{ service_instance.inputs.my_sample_service_instance_optional_input }}"  #
 input parameter reference
  MyServiceInstancesEnvironmentSampleOutputValue:                              #
 output definition
    Value: "{{ environment.outputs.MySampleInputValue }}"                      #
 output reference to an environment IaC file
```

```
 MyServiceInstancesEnvironmentOtherSampleOutputValue:                    #
output definition
   Value: "{{ environment.outputs.MyOtherSampleInputValue }}"            #
output reference to an environment IaC file
```

Dettagli ed esempi dei parametri del file CloudFormation IAc del componente

È possibile definire e fare riferimento ai parametri nell'infrastruttura dei componenti come file di codice (IaC). Per una descrizione dettagliata dei AWS Proton parametri, dei tipi di parametri, dello spazio dei nomi dei parametri e di come utilizzare i parametri nei file IAc, vedere. the section called “Parametri” Per ulteriori informazioni sui componenti, vedere. Componenti

Definire i parametri di output dei componenti

È possibile definire i parametri di output nei file IAc dei componenti. È quindi possibile fare riferimento a queste uscite nei file IAc di servizio.

> **Note**
>
> Non è possibile definire input per i file IAc dei componenti. I componenti collegati possono ricevere input dall'istanza di servizio a cui sono collegati. I componenti scollegati non dispongono di input.

Leggi i valori dei parametri nei file IAc dei componenti

È possibile leggere i parametri relativi al componente e ad altre risorse nei file IAc dei componenti. È possibile leggere il valore di un parametro facendo riferimento al nome del parametro nello spazio dei nomi del AWS Proton parametro.

- Parametri di input: legge il valore di input di un'istanza di servizio allegata mediante riferimento. `service_instance.inputs.`*`input-name`*
- Parametri AWS Proton delle risorse: legge i parametri delle risorse facendo riferimento a nomi come`component.name`, `service.name`e `service_instance.name`, e. `environment.name`
- Parametri di output: legge gli output dell'ambiente mediante riferimento. `environment.outputs.`*`output-name`*

File IAc di esempio per componenti e servizi con parametri

L'esempio seguente mostra un componente che fornisce un bucket Amazon Simple Storage Service (Amazon S3) e la relativa policy di accesso ed espone gli Amazon Resource Names ARNs () di entrambe le risorse come output dei componenti. Un modello IAc di servizio aggiunge gli output dei componenti come variabili di ambiente del contenitore di un'attività Amazon Elastic Container Service (Amazon ECS) per rendere gli output disponibili per il codice in esecuzione nel contenitore e aggiunge la policy di accesso al bucket al ruolo dell'attività. Il nome del bucket si basa sui nomi dell'ambiente, del servizio, dell'istanza del servizio e del componente, il che significa che il bucket è associato a un'istanza specifica del modello di componente che estende un'istanza di servizio specifica. Gli sviluppatori possono creare più componenti personalizzati basati su questo modello di componenti, per fornire bucket Amazon S3 per diverse istanze di servizio ed esigenze funzionali.

L'esempio mostra come utilizzare la `{{ ... }}` sintassi Jinja per fare riferimento ai parametri dei componenti e ad altri parametri di risorse nel file IAc del servizio. È possibile utilizzare `{% if ... %}` le istruzioni per aggiungere blocchi di istruzioni solo quando un componente è collegato all'istanza del servizio. Le `proton_cfn_*` parole chiave sono filtri che è possibile utilizzare per disinfettare e formattare i valori dei parametri di output. Per ulteriori informazioni sui filtri, consultare the section called “CloudFormation filtri parametrici”.

In qualità di amministratore, sei l'autore del file modello IAc del servizio.

Example file CloudFormation IAc di servizio che utilizza un componente

```
# service/instance_infrastructure/cloudformation.yaml

Resources:
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      TaskRoleArn: !Ref TaskRole
      ContainerDefinitions:
        - Name: '{{service_instance.name}}'
          # ...
          {% if service_instance.components.default.outputs | length > 0 %}
          Environment:
            {{ service_instance.components.default.outputs |
                proton_cfn_ecs_task_definition_formatted_env_vars }}
          {% endif %}

  # ...
```

```
  TaskRole:
    Type: AWS::IAM::Role
    Properties:
      # ...
      ManagedPolicyArns:
        - !Ref BaseTaskRoleManagedPolicy
        {{ service_instance.components.default.outputs
            | proton_cfn_iam_policy_arns }}

  # Basic permissions for the task
  BaseTaskRoleManagedPolicy:
    Type: AWS::IAM::ManagedPolicy
    Properties:
      # ...
```

In qualità di sviluppatore, sei l'autore del file modello IAc del componente.

Example file componente CloudFormation IAc

```
# cloudformation.yaml

# A component that defines an S3 bucket and a policy for accessing the bucket.
Resources:
  S3Bucket:
    Type: 'AWS::S3::Bucket'
    Properties:
      BucketName: '{{environment.name}}-{{service.name}}-{{service_instance.name}}-
{{component.name}}'
  S3BucketAccessPolicy:
    Type: AWS::IAM::ManagedPolicy
    Properties:
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Action:
              - 's3:Get*'
              - 's3:List*'
              - 's3:PutObject'
            Resource: !GetAtt S3Bucket.Arn
Outputs:
  BucketName:
    Description: "Bucket to access"
```

```
    Value: !GetAtt S3Bucket.Arn
  BucketAccessPolicyArn:
    Value: !Ref S3BucketAccessPolicy
```

Quando AWS Proton esegue il rendering di un AWS CloudFormation modello per l'istanza del servizio e sostituisce tutti i parametri con valori effettivi, il modello potrebbe avere l'aspetto del file seguente.

Example l'istanza di servizio ha CloudFormation reso il file IAc

```
Resources:
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      TaskRoleArn: !Ref TaskRole
      ContainerDefinitions:
        - Name: '{{service_instance.name}}'
          # ...
          Environment:
            - Name: BucketName
              Value: arn:aws:s3:us-
east-1:123456789012:environment_name-service_name-service_instance_name-component_name
            - Name: BucketAccessPolicyArn
              Value: arn:aws:iam::123456789012:policy/cfn-generated-policy-name
  # ...

  TaskRole:
    Type: AWS::IAM::Role
    Properties:
      # ...
      ManagedPolicyArns:
        - !Ref BaseTaskRoleManagedPolicy
        - arn:aws:iam::123456789012:policy/cfn-generated-policy-name

  # Basic permissions for the task
  BaseTaskRoleManagedPolicy:
    Type: AWS::IAM::ManagedPolicy
    Properties:
      # ...
```

filtri parametrici per CloudFormation file IAc

Quando si fanno riferimenti ai AWS Proton parametri nei file AWS CloudFormation IAc, è possibile utilizzare i modificatori Jinja noti come filtri per convalidare, filtrare e formattare i valori dei parametri prima che vengano inseriti nel modello renderizzato. Le convalide dei filtri sono particolarmente utili quando si fa riferimento ai parametri di output dei componenti, poiché la creazione e il collegamento dei componenti vengono eseguiti dagli sviluppatori e un amministratore che utilizza gli output dei componenti in un modello di istanza di servizio potrebbe volerne verificare l'esistenza e la validità. Tuttavia, puoi utilizzare i filtri in qualsiasi file Jinja IAc.

Le sezioni seguenti descrivono e definiscono i filtri parametrici disponibili e forniscono esempi. AWS Proton definisce la maggior parte di questi filtri. Il `default` filtro è un filtro integrato Jinja.

Proprietà dell'ambiente di formattazione per le attività di Amazon ECS

Dichiarazione

```
dict # proton_cfn_ecs_task_definition_formatted_env_vars (raw: boolean = True) # YAML
 list of dicts
```

Descrizione

Questo filtro formatta un elenco di output da utilizzare in una proprietà Environment nella `ContainerDefinition` sezione di una definizione di attività Amazon Elastic Container Service (Amazon ECS) Elastic Container Service (Amazon ECS).

Imposta su `raw` per `False` convalidare anche il valore del parametro. In questo caso, è necessario che il valore corrisponda all'espressione `^[a-zA-Z0-9_-]*$` regolare. Se il valore non supera questa convalida, il rendering del modello fallisce.

Esempio

Con il seguente modello di componente personalizzato:

```
Resources:
  # ...
Outputs:
  Output1:
    Description: "Example component output 1"
    Value: hello
```

```
  Output2:
    Description: "Example component output 2"
    Value: world
```

E il seguente modello di servizio:

```
Resources:
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      # ...
      ContainerDefinitions:
        - Name: MyServiceName
          # ...
          Environment:
            {{ service_instance.components.default.outputs
              | proton_cfn_ecs_task_definition_formatted_env_vars }}
```

Il modello di servizio reso è il seguente:

```
Resources:
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      # ...
      ContainerDefinitions:
        - Name: MyServiceName
          # ...
          Environment:
            - Name: Output1
              Value: hello
            - Name: Output2
              Value: world
```

Proprietà di ambiente di formattazione per le funzioni Lambda

Dichiarazione

```
dict # proton_cfn_lambda_function_formatted_env_vars (raw: boolean = True) # YAML dict
```

Descrizione

Questo filtro formatta un elenco di output da utilizzare in una proprietà Environment nella `Properties` sezione di una definizione di AWS Lambda funzione.

Imposta su `raw` `False` per convalidare anche il valore del parametro. In questo caso, è necessario che il valore corrisponda all'espressione `^[a-zA-Z0-9_-]*$` regolare. Se il valore non supera questa convalida, il rendering del modello fallisce.

Esempio

Con il seguente modello di componente personalizzato:

```
Resources:
  # ...
Outputs:
  Output1:
    Description: "Example component output 1"
    Value: hello
  Output2:
    Description: "Example component output 2"
    Value: world
```

E il seguente modello di servizio:

```
Resources:
  Lambda:
    Type: AWS::Lambda::Function
    Properties:
      Environment:
        Variables:
          {{ service_instance.components.default.outputs
            | proton_cfn_lambda_function_formatted_env_vars }}
```

Il modello di servizio reso è il seguente:

```
Resources:
  Lambda:
    Type: AWS::Lambda::Function
    Properties:
      Environment:
        Variables:
          Output1: hello
          Output2: world
```

Estrai la policy IAM ARNs da includere nei ruoli IAM

Dichiarazione

```
dict # proton_cfn_iam_policy_arns # YAML list
```

Descrizione

Questo filtro formatta un elenco di output da utilizzare in una ManagedPolicyArns proprietà nella `Properties` sezione di una definizione di ruolo AWS Identity and Access Management (IAM). Il filtro utilizza l'espressione regolare `^arn:[a-zA-Z-]+:iam::\d{12}:policy/` per estrarre una politica IAM valida ARNs dall'elenco dei parametri di output. È possibile utilizzare questo filtro per aggiungere le politiche nei valori dei parametri di output a una definizione di ruolo IAM in un modello di servizio.

Esempio

Con il seguente modello di componente personalizzato:

```
Resources:
  # ...
  ExamplePolicy1:
    Type: AWS::IAM::ManagedPolicy
    Properties:
      # ...
  ExamplePolicy2:
    Type: AWS::IAM::ManagedPolicy
    Properties:
      # ...

  # ...

Outputs:
  Output1:
    Description: "Example component output 1"
    Value: hello
  Output2:
    Description: "Example component output 2"
    Value: world
  PolicyArn1:
    Description: "ARN of policy 1"
    Value: !Ref ExamplePolicy1
```

```
  PolicyArn2:
    Description: "ARN of policy 2"
    Value: !Ref ExamplePolicy2
```

E il seguente modello di servizio:

```
Resources:

  # ...

  TaskRole:
    Type: AWS::IAM::Role
    Properties:
      # ...
      ManagedPolicyArns:
        - !Ref BaseTaskRoleManagedPolicy
        {{ service_instance.components.default.outputs
            | proton_cfn_iam_policy_arns }}

  # Basic permissions for the task
  BaseTaskRoleManagedPolicy:
    Type: AWS::IAM::ManagedPolicy
    Properties:
      # ...
```

Il modello di servizio reso è il seguente:

```
Resources:

  # ...

  TaskRole:
    Type: AWS::IAM::Role
    Properties:
      # ...
      ManagedPolicyArns:
        - !Ref BaseTaskRoleManagedPolicy
        - arn:aws:iam::123456789012:policy/cfn-generated-policy-name-1
        - arn:aws:iam::123456789012:policy/cfn-generated-policy-name-2

  # Basic permissions for the task
  BaseTaskRoleManagedPolicy:
    Type: AWS::IAM::ManagedPolicy
```

```
    Properties:
      # ...
```

Disinfetta i valori delle proprietà

Dichiarazione

```
string # proton_cfn_sanitize # string
```

Descrizione

Si tratta di un filtro generico. Utilizzatelo per convalidare la sicurezza del valore di un parametro. Il filtro verifica che il valore corrisponda all'espressione regolare `^[a-zA-Z0-9_-]*$` o sia un Amazon Resource Name (ARN) valido. Se il valore non supera questa convalida, il rendering del modello non riesce.

Esempio

Con il seguente modello di componente personalizzato:

```
Resources:
  # ...
Outputs:
  Output1:
    Description: "Example of valid output"
    Value: "This-is_valid_37"
  Output2:
    Description: "Example incorrect output"
    Value: "this::is::incorrect"
  SomeArn:
    Description: "Example ARN"
    Value: arn:aws:some-service::123456789012:some-resource/resource-name
```

- Il seguente riferimento in un modello di servizio:

  ```
  # ...
    {{ service_instance.components.default.outputs.Output1
      | proton_cfn_sanitize }}
  ```

  Viene visualizzato come segue:

```
# ...
  This-is_valid_37
```

- Il seguente riferimento in un modello di servizio:

```
# ...
  {{ service_instance.components.default.outputs.Output2
    | proton_cfn_sanitize }}
```

  Risultati con il seguente errore di rendering:

```
Illegal character(s) detected in "this::is::incorrect". Must match regex ^[a-zA-
Z0-9_-]*$ or be a valid ARN
```

- Il seguente riferimento in un modello di servizio:

```
# ...
  {{ service_instance.components.default.outputs.SomeArn
    | proton_cfn_sanitize }}
```

  Viene visualizzato come segue:

```
# ...
  arn:aws:some-service::123456789012:some-resource/resource-name
```

Fornite valori predefiniti per riferimenti inesistenti

Descrizione

Il `default` filtro fornisce un valore predefinito quando non esiste un riferimento allo spazio dei nomi. Utilizzatelo per scrivere modelli robusti in grado di eseguire il rendering senza errori anche quando manca il parametro a cui fate riferimento.

Esempio

Il riferimento seguente in un modello di servizio fa sì che il rendering del modello non riesca se all'istanza del servizio non è associato un componente definito direttamente (predefinito) o se il componente allegato non ha un output denominato`test`.

```
# ...
  {{ service_instance.components.default.outputs.test }}
```

Per evitare questo problema, aggiungi il `default` filtro.

```
# ...
  {{ service_instance.components.default.outputs.test | default("[optional-value]") }}
```

CodeBuild dettagli ed esempi dei parametri di provisioning

È possibile definire i parametri nei modelli per AWS Proton le risorse CodeBuild basate e fare riferimento a tali parametri nel codice di provisioning. Per una descrizione dettagliata dei AWS Proton parametri, dei tipi di parametro, dello spazio dei nomi dei parametri e di come utilizzare i parametri nei file IAc, vedere. the section called “Parametri”

> **Note**
>
> È possibile utilizzare il CodeBuild provisioning con ambienti e servizi. Al momento non è possibile effettuare il provisioning dei componenti in questo modo.

Parametri di input

Quando crei una AWS Proton risorsa, come un ambiente o un servizio, fornisci valori per i parametri di input definiti nel file di schema del modello. Quando la risorsa che crei utilizzaCodeBuild approvvigionamento, AWS Proton rende questi valori di input in un file di input. Il codice di provisioning può importare e ottenere valori dei parametri da questo file.

Per un esempio di CodeBuild modello, vedithe section called “CodeBuild pacchetto”. Per ulteriori informazioni sui file manifest, consulta the section called “Manifesta e concludi”.

L'esempio seguente è un file di input JSON generato durante il provisioning CodeBuild basato su base di un'istanza di servizio.

Esempio: utilizzo di with provisioning AWS CDK CodeBuild

```
{
  "service_instance": {
    "name": "my-service-staging",
    "inputs": {
```

```
      "port": "8080",
      "task_size": "medium"
    }
  },
  "service": {
    "name": "my-service"
  },
  "environment": {
    "account_id": "123456789012",
    "name": "my-env-staging",
    "outputs": {
      "vpc-id": "hdh2323423"
    }
  }
}
```

Parametri di output

Per comunicare gli output del provisioning delle risorse a AWS Proton, il codice di provisioning può generare un file JSON denominato `proton-outputs.json` con i valori per i parametri di output definiti nel file di schema del modello. Ad esempio, il `cdk deploy` comando ha l'`--outputs-file`argomento che indica di generare un file JSON AWS CDK con output di provisioning. Se la tua risorsa utilizza il AWS CDK, specifica il seguente comando nel manifesto del modello: CodeBuild

```
aws proton notify-resource-deployment-status-change
```

AWS Proton cerca questo file JSON. Se il file esiste dopo che il codice di provisioning è stato completato correttamente, ne AWS Proton legge i valori dei parametri di output.

Dettagli ed esempi dei parametri del file Terraform infrastructure as code (IaC)

Puoi includere le variabili di input Terraform nei `variable.tf` file del tuo pacchetto di modelli. Puoi anche creare uno schema per creare variabili AWS Proton gestite. AWS Proton crea una variabile `.tf files` dal tuo file di schema. Per ulteriori informazioni, consulta the section called “file Terraform iAC”.

Per fare riferimento alle AWS Proton variabili definite dallo schema nella vostra infrastruttura`.tf files`, utilizzate i AWS Proton namespace mostrati nella tabella Parametri e namespace per Terraform IAc. Per esempio, è possibile utilizzare `var.environment.inputs.vpc_cidr`.

All'interno delle virgolette, racchiudi queste variabili tra parentesi singole e aggiungi il simbolo del dollaro davanti alla prima parentesi (ad esempio,). “`${var.environment.inputs.vpc_cidr}`”

L'esempio seguente mostra come utilizzare i namespace per includere parametri in un ambiente. AWS Proton `.tf file`

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
  // This tells terraform to store the state file in s3 at the location
  // s3://terraform-state-bucket/tf-os-sample/terraform.tfstate
  backend "s3" {
    bucket = "terraform-state-bucket"
    key    = "tf-os-sample/terraform.tfstate"
    region = "us-east-1"
  }
}

// Configure the AWS Provider
provider "aws" {
  region = "us-east-1"
  default_tags {
    tags = var.proton_tags
  }
}

resource "aws_ssm_parameter" "my_ssm_parameter" {
  name  = "my_ssm_parameter"
  type  = "String"
  // Use the Proton environment.inputs. namespace
  value = var.environment.inputs.ssm_parameter_value
}
```

AWS Proton infrastruttura come file di codice

Le parti principali del pacchetto di modelli sono i file Infrastructure as Code (IaC) che definiscono le risorse e le proprietà dell'infrastruttura da fornire. AWS CloudFormation e altre infrastrutture come motori di codice utilizzano questi tipi di file per fornire risorse infrastrutturali.

> **Note**
>
> Un file IAc può essere utilizzato anche indipendentemente dai pacchetti di modelli, come input diretto per componenti definiti direttamente. Per ulteriori informazioni sui componenti, vedere. Componenti

AWS Proton attualmente supporta due tipi di file IAc:

- CloudFormationfile: utilizzati per il AWS provisioning gestito. AWS Proton utilizza Jinja oltre al formato di file CloudFormation modello per la parametrizzazione.
- File Terraform HCL: utilizzati per il provisioning autogestito. HCL supporta nativamente la parametrizzazione.

Non è possibile effettuare il provisioning AWS Proton delle risorse utilizzando una combinazione di metodi di provisioning. È necessario utilizzare l'uno o l'altro. Non è possibile implementare un servizio di provisioning AWS gestito in un ambiente di provisioning autogestito o viceversa.

Per ulteriori informazioni, consulta the section called “Metodi di approvvigionamento”, Ambienti, Servizi e Componenti.

AWS CloudFormation file IAc

Scopri come utilizzare l' AWS CloudFormation infrastruttura come file di codice con. AWS Proton AWS CloudFormation è un servizio Infrastructure as Code (IaC) che ti aiuta a modellare e configurare AWS le tue risorse. Definisci le risorse dell'infrastruttura in modelli, utilizzando Jinja oltre al formato di file CloudFormation modello per la parametrizzazione. AWS Proton espande i parametri e rende il modello completo. CloudFormation CloudFormation fornisce le risorse definite come CloudFormation stack. Per ulteriori informazioni, consulta Cosa contiene la AWS CloudFormation Guida per l' AWS CloudFormation utente.

AWS Proton supporta il AWS provisioning gestito per CloudFormation IaC.

Inizia con la tua infrastruttura esistente come file di codice

È possibile adattare i propri file Infrastructure as Code (IaC) esistenti per utilizzarli con AWS Proton.

I seguenti AWS CloudFormation esempi, Esempio 1 ed Esempio 2, rappresentano i vostri file CloudFormation IAc esistenti. CloudFormation può usare questi file per creare due CloudFormation pile diverse.

Nell'esempio 1, il file CloudFormation IAc è configurato per fornire l'infrastruttura da condividere tra le applicazioni container. In questo esempio, vengono aggiunti parametri di input in modo da poter utilizzare lo stesso file IAc per creare più set di infrastrutture predisposte. Ogni set può avere nomi diversi insieme a un set diverso di valori CIDR per VPC e sottorete. In qualità di amministratore o sviluppatore, si forniscono valori per questi parametri quando si utilizza un file IaC per il provisioning delle risorse dell'infrastruttura. CloudFormation Per comodità, nell'esempio questi parametri di input sono contrassegnati da commenti e citati più volte. Gli output sono definiti alla fine del modello. Possono essere referenziati in altri file CloudFormation IAc.

Nell'Esempio 2, il file CloudFormation IAc è configurato per distribuire un'applicazione nell'infrastruttura fornita dall'Esempio 1. I parametri sono commentati per comodità.

Esempio 1: file CloudFormation IAc

```
AWSTemplateFormatVersion: '2010-09-09'
Description: AWS Fargate cluster running containers in a public subnet. Only supports
             public facing load balancer, and public service discovery namespaces.
Parameters:
   VpcCIDR:       # input parameter
        Description: CIDR for VPC
        Type: String
        Default: "10.0.0.0/16"
   SubnetOneCIDR: # input parameter
        Description: CIDR for SubnetOne
        Type: String
        Default: "10.0.0.0/24"
   SubnetTwoCIDR: # input parameters
        Description: CIDR for SubnetTwo
        Type: String
        Default: "10.0.1.0/24"
Resources:
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      EnableDnsSupport: true
      EnableDnsHostnames: true
      CidrBlock:
        Ref: 'VpcCIDR'
```

```
# Two public subnets, where containers will have public IP addresses
PublicSubnetOne:
  Type: AWS::EC2::Subnet
  Properties:
    AvailabilityZone:
       Fn::Select:
       - 0
       - Fn::GetAZs: {Ref: 'AWS::Region'}
    VpcId: !Ref 'VPC'
    CidrBlock:
       Ref: 'SubnetOneCIDR'
    MapPublicIpOnLaunch: true

PublicSubnetTwo:
  Type: AWS::EC2::Subnet
  Properties:
    AvailabilityZone:
       Fn::Select:
       - 1
       - Fn::GetAZs: {Ref: 'AWS::Region'}
    VpcId: !Ref 'VPC'
    CidrBlock:
      Ref: 'SubnetTwoCIDR'
    MapPublicIpOnLaunch: true

# Setup networking resources for the public subnets. Containers
# in the public subnets have public IP addresses and the routing table
# sends network traffic via the internet gateway.
InternetGateway:
  Type: AWS::EC2::InternetGateway
GatewayAttachement:
  Type: AWS::EC2::VPCGatewayAttachment
  Properties:
    VpcId: !Ref 'VPC'
    InternetGatewayId: !Ref 'InternetGateway'
PublicRouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId: !Ref 'VPC'
PublicRoute:
  Type: AWS::EC2::Route
  DependsOn: GatewayAttachement
  Properties:
```

```
      RouteTableId: !Ref 'PublicRouteTable'
      DestinationCidrBlock: '0.0.0.0/0'
      GatewayId: !Ref 'InternetGateway'
  PublicSubnetOneRouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId: !Ref PublicSubnetOne
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetTwoRouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId: !Ref PublicSubnetTwo
      RouteTableId: !Ref PublicRouteTable

  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster

  # A security group for the containers we will run in Fargate.
  # Rules are added to this security group based on what ingress you
  # add for the cluster.
  ContainerSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Access to the Fargate containers
      VpcId: !Ref 'VPC'

  # This is a role which is used by the ECS tasks themselves.
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
        - Effect: Allow
          Principal:
            Service: [ecs-tasks.amazonaws.com]
          Action: ['sts:AssumeRole']
      Path: /
      ManagedPolicyArns:
        - 'arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy'

# These output values will be available to other templates to use.
Outputs:
  ClusterName:                                          # output
```

```
    Description: The name of the ECS cluster
    Value: !Ref 'ECSCluster'
    Export:
      Name:
        Fn::Sub: "${AWS::StackName}-ECSCluster"
  ECSTaskExecutionRole:                                    # output
    Description: The ARN of the ECS role
    Value: !GetAtt 'ECSTaskExecutionRole.Arn'
    Export:
      Name:
        Fn::Sub: "${AWS::StackName}-ECSTaskExecutionRole"
  VpcId:                                                   # output
    Description: The ID of the VPC that this stack is deployed in
    Value: !Ref 'VPC'
    Export:
      Name:
        Fn::Sub: "${AWS::StackName}-VPC"
  PublicSubnetOne:                                         # output
    Description: Public subnet one
    Value: !Ref 'PublicSubnetOne'
    Export:
      Name:
        Fn::Sub: "${AWS::StackName}-PublicSubnetOne"
  PublicSubnetTwo:                                         # output
    Description: Public subnet two
    Value: !Ref 'PublicSubnetTwo'
    Export:
      Name:
        Fn::Sub: "${AWS::StackName}-PublicSubnetTwo"
  ContainerSecurityGroup:                                  # output
    Description: A security group used to allow Fargate containers to receive traffic
    Value: !Ref 'ContainerSecurityGroup'
    Export:
      Name:
        Fn::Sub: "${AWS::StackName}-ContainerSecurityGroup"
```

Esempio 2: file CloudFormation IAc

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Deploy a service on AWS Fargate, hosted in a public subnet, and accessible
 via a public load balancer.
Parameters:
    ContainerPortInput:  # input parameter
```

```
        Description: The port to route traffic to
        Type: Number
        Default: 80
    TaskCountInput:      # input parameter
        Description: The default number of Fargate tasks you want running
        Type: Number
        Default: 1
    TaskSizeInput:       # input parameter
        Description: The size of the task you want to run
        Type: String
        Default: x-small
    ContainerImageInput: # input parameter
        Description: The name/url of the container image
        Type: String
        Default: "public.ecr.aws/z9d2n7e1/nginx:1.19.5"
    TaskNameInput:       # input parameter
        Description: Name for your task
        Type: String
        Default: "my-fargate-instance"
    StackName:           # input parameter
        Description: Name of the environment stack to deploy to
        Type: String
        Default: "my-fargate-environment"
Mappings:
  TaskSizeMap:
    x-small:
      cpu: 256
      memory: 512
    small:
      cpu: 512
      memory: 1024
    medium:
      cpu: 1024
      memory: 2048
    large:
      cpu: 2048
      memory: 4096
    x-large:
      cpu: 4096
      memory: 8192
Resources:
  # A log group for storing the stdout logs from this service's containers
  LogGroup:
    Type: AWS::Logs::LogGroup
```

```
    Properties:
      LogGroupName:
        Ref: 'TaskNameInput' # input parameter

  # The task definition. This is a simple metadata description of what
  # container to run, and what resource requirements it has.
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: !Ref 'TaskNameInput'
      Cpu: !FindInMap [TaskSizeMap, !Ref 'TaskSizeInput', cpu]
      Memory: !FindInMap [TaskSizeMap, !Ref 'TaskSizeInput', memory]
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      ExecutionRoleArn:
        Fn::ImportValue:
          !Sub "${StackName}-ECSTaskExecutionRole"    # output parameter from another
CloudFormation template
             awslogs-region: !Ref 'AWS::Region'
             awslogs-stream-prefix: !Ref 'TaskNameInput'


  # The service_instance. The service is a resource which allows you to run multiple
  # copies of a type of task, and gather up their logs and metrics, as well
  # as monitor the number of running tasks and replace any that have crashed
  Service:
    Type: AWS::ECS::Service
    DependsOn: LoadBalancerRule
    Properties:
      ServiceName: !Ref 'TaskNameInput'
      Cluster:
        Fn::ImportValue:
          !Sub "${StackName}-ECSCluster"  # output parameter from another
CloudFormation template
      LaunchType: FARGATE
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 75
      DesiredCount: !Ref 'TaskCountInput'
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
```

```
            - Fn::ImportValue:
                !Sub "${StackName}-ContainerSecurityGroup"    # output parameter from
another CloudFormation template
          Subnets:
            - Fn::ImportValue:r CloudFormation template
      TaskRoleArn: !Ref "AWS::NoValue"
      ContainerDefinitions:
        - Name: !Ref 'TaskNameInput'
          Cpu: !FindInMap [TaskSizeMap, !Ref 'TaskSizeInput', cpu]
          Memory: !FindInMap [TaskSizeMap, !Ref 'TaskSizeInput', memory]
          Image: !Ref 'ContainerImageInput'             # input parameter
          PortMappings:
            - ContainerPort: !Ref 'ContainerPortInput'  # input parameter

          LogConfiguration:
            LogDriver: 'awslogs'
            Options:
              awslogs-group: !Ref 'TaskNameInput'
                !Sub "${StackName}-PublicSubnetOne"    # output parameter from another
CloudFormation template
            - Fn::ImportValue:
                !Sub "${StackName}-PublicSubnetTwo"    # output parameter from another
CloudFormation template
      TaskDefinition: !Ref 'TaskDefinition'
      LoadBalancers:
        - ContainerName: !Ref 'TaskNameInput'
          ContainerPort: !Ref 'ContainerPortInput'  # input parameter
          TargetGroupArn: !Ref 'TargetGroup'

  # A target group. This is used for keeping track of all the tasks, and
  # what IP addresses / port numbers they have. You can query it yourself,
  # to use the addresses yourself, but most often this target group is just
  # connected to an application load balancer, or network load balancer, so
  # it can automatically distribute traffic across all the targets.
  TargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      HealthCheckIntervalSeconds: 6
      HealthCheckPath: /
      HealthCheckProtocol: HTTP
      HealthCheckTimeoutSeconds: 5
      HealthyThresholdCount: 2
      TargetType: ip
      Name: !Ref 'TaskNameInput'
```

```
      Port: !Ref 'ContainerPortInput'
      Protocol: HTTP
      UnhealthyThresholdCount: 2
      VpcId:
        Fn::ImportValue:
          !Sub "${StackName}-VPC"    # output parameter from another CloudFormation
 template

  # Create a rule on the load balancer for routing traffic to the target group
  LoadBalancerRule:
    Type: AWS::ElasticLoadBalancingV2::ListenerRule
    Properties:
      Actions:
        - TargetGroupArn: !Ref 'TargetGroup'
          Type: 'forward'
      Conditions:
        - Field: path-pattern
          Values:
            - '*'
      ListenerArn: !Ref PublicLoadBalancerListener
      Priority: 1

  # Enable autoscaling for this service
  ScalableTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    DependsOn: Service
    Properties:
      ServiceNamespace: 'ecs'
      ScalableDimension: 'ecs:service:DesiredCount'
      ResourceId:
        Fn::Join:
          - '/'
          - - service
            - Fn::ImportValue:
                !Sub "${StackName}-ECSCluster"
            - !Ref 'TaskNameInput'
      MinCapacity: 1
      MaxCapacity: 10
      RoleARN: !Sub arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService

  # Create scaling policies for the service
  ScaleDownPolicy:
```

```
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    DependsOn: ScalableTarget
    Properties:
      PolicyName:
        Fn::Join:
          - '/'
          - - scale
            - !Ref 'TaskNameInput'
            - down
      PolicyType: StepScaling
      ResourceId:
        Fn::Join:
          - '/'
          - - service
            - Fn::ImportValue:
                !Sub "${StackName}-ECSCluster"  # output parameter from another
CloudFormation template
            - !Ref 'TaskNameInput'
      ScalableDimension: 'ecs:service:DesiredCount'
      ServiceNamespace: 'ecs'
      StepScalingPolicyConfiguration:
        AdjustmentType: 'ChangeInCapacity'
        StepAdjustments:
          - MetricIntervalUpperBound: 0
            ScalingAdjustment: -1
        MetricAggregationType: 'Average'
        Cooldown: 60

  ScaleUpPolicy:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    DependsOn: ScalableTarget
    Properties:
      PolicyName:
        Fn::Join:
          - '/'
          - - scale
            - !Ref 'TaskNameInput'
            - up
      PolicyType: StepScaling
      ResourceId:
        Fn::Join:
          - '/'
          - - service
            - Fn::ImportValue:
```

```
                !Sub "${StackName}-ECSCluster"
            - !Ref 'TaskNameInput'
      ScalableDimension: 'ecs:service:DesiredCount'
      ServiceNamespace: 'ecs'
      StepScalingPolicyConfiguration:
        AdjustmentType: 'ChangeInCapacity'
        StepAdjustments:
          - MetricIntervalLowerBound: 0
            MetricIntervalUpperBound: 15
            ScalingAdjustment: 1
          - MetricIntervalLowerBound: 15
            MetricIntervalUpperBound: 25
            ScalingAdjustment: 2
          - MetricIntervalLowerBound: 25
            ScalingAdjustment: 3
        MetricAggregationType: 'Average'
        Cooldown: 60

  # Create alarms to trigger these policies
  LowCpuUsageAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmName:
        Fn::Join:
          - '-'
          - - low-cpu
            - !Ref 'TaskNameInput'
      AlarmDescription:
        Fn::Join:
          - ' '
          - - "Low CPU utilization for service"
            - !Ref 'TaskNameInput'
      MetricName: CPUUtilization
      Namespace: AWS/ECS
      Dimensions:
        - Name: ServiceName
          Value: !Ref 'TaskNameInput'
        - Name: ClusterName
          Value:
            Fn::ImportValue:
              !Sub "${StackName}-ECSCluster"
      Statistic: Average
      Period: 60
      EvaluationPeriods: 1
```

```
      Threshold: 20
      ComparisonOperator: LessThanOrEqualToThreshold
      AlarmActions:
        - !Ref ScaleDownPolicy

  HighCpuUsageAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmName:
        Fn::Join:
          - '-'
          - - high-cpu
            - !Ref 'TaskNameInput'
      AlarmDescription:
        Fn::Join:
          - ' '
          - - "High CPU utilization for service"
            - !Ref 'TaskNameInput'
      MetricName: CPUUtilization
      Namespace: AWS/ECS
      Dimensions:
        - Name: ServiceName
          Value: !Ref 'TaskNameInput'
        - Name: ClusterName
          Value:
            Fn::ImportValue:
              !Sub "${StackName}-ECSCluster"
      Statistic: Average
      Period: 60
      EvaluationPeriods: 1
      Threshold: 70
      ComparisonOperator: GreaterThanOrEqualToThreshold
      AlarmActions:
        - !Ref ScaleUpPolicy

  EcsSecurityGroupIngressFromPublicALB:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      Description: Ingress from the public ALB
      GroupId:
        Fn::ImportValue:
          !Sub "${StackName}-ContainerSecurityGroup"
      IpProtocol: -1
      SourceSecurityGroupId: !Ref 'PublicLoadBalancerSG'
```

```
  # Public load balancer, hosted in public subnets that is accessible
  # to the public, and is intended to route traffic to one or more public
  # facing services. This is used for accepting traffic from the public
  # internet and directing it to public facing microservices
  PublicLoadBalancerSG:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Access to the public facing load balancer
      VpcId:
        Fn::ImportValue:
          !Sub "${StackName}-VPC"
      SecurityGroupIngress:
          # Allow access to ALB from anywhere on the internet
          - CidrIp: 0.0.0.0/0
            IpProtocol: -1

  PublicLoadBalancer:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Scheme: internet-facing
      LoadBalancerAttributes:
      - Key: idle_timeout.timeout_seconds
        Value: '30'
      Subnets:
        # The load balancer is placed into the public subnets, so that traffic
        # from the internet can reach the load balancer directly via the internet
gateway
        - Fn::ImportValue:
            !Sub "${StackName}-PublicSubnetOne"
        - Fn::ImportValue:
            !Sub "${StackName}-PublicSubnetTwo"
      SecurityGroups: [!Ref 'PublicLoadBalancerSG']

  PublicLoadBalancerListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    DependsOn:
      - PublicLoadBalancer
    Properties:
      DefaultActions:
        - TargetGroupArn: !Ref 'TargetGroup'
          Type: 'forward'
      LoadBalancerArn: !Ref 'PublicLoadBalancer'
      Port: 80
```

```
      Protocol: HTTP
# These output values will be available to other templates to use.
Outputs:
  ServiceEndpoint:        # output
    Description: The URL to access the service
    Value: !Sub "http://${PublicLoadBalancer.DNSName}"
```

È possibile adattare questi file per utilizzarli con AWS Proton.

Trasforma la tua infrastruttura sotto forma di codice in AWS Proton

Con lievi modifiche, puoi utilizzare l'Esempio 1 come file Infrastructure as Code (IaC) per un pacchetto di modelli di ambiente da utilizzare per AWS Proton distribuire un ambiente (come mostrato nell'Esempio 3).

Invece di utilizzare CloudFormation i parametri, utilizzate la sintassi Jinja per fare riferimento ai parametri che avete definito in un file di schema basato su Open API. Questi parametri di input vengono commentati per comodità e citati più volte nel file IAc. In questo modo, AWS Proton può controllare e controllare i valori dei parametri. Può anche abbinare e inserire i valori dei parametri di output in un file IAc con i parametri in un altro file IAc.

In qualità di amministratore, è possibile aggiungere lo spazio dei AWS Proton `environment.inputs.` nomi ai parametri di input. Quando si fa riferimento agli output dei file IAc di ambiente in un file IAc di servizio, è possibile aggiungere lo spazio dei `environment.outputs.` nomi agli output (ad esempio,). `environment.outputs.ClusterName` Infine, li racchiudi con parentesi graffe e virgolette.

Con queste modifiche, i tuoi file CloudFormation IAc possono essere utilizzati da. AWS Proton

Esempio 3: infrastruttura AWS Proton ambientale come file di codice

```
AWSTemplateFormatVersion: '2010-09-09'
Description: AWS Fargate cluster running containers in a public subnet. Only supports
             public facing load balancer, and public service discovery prefixes.
Mappings:
  # The VPC and subnet configuration is passed in via the environment spec.
  SubnetConfig:
    VPC:
      CIDR: '{{ environment.inputs.vpc_cidr}}'        # input parameter
    PublicOne:
      CIDR: '{{ environment.inputs.subnet_one_cidr}}' # input parameter
    PublicTwo:
```

```
      CIDR: '{{ environment.inputs.subnet_two_cidr}}' # input parameter
Resources:
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      EnableDnsSupport: true
      EnableDnsHostnames: true
      CidrBlock: !FindInMap ['SubnetConfig', 'VPC', 'CIDR']

  # Two public subnets, where containers will have public IP addresses
  PublicSubnetOne:
    Type: AWS::EC2::Subnet
    Properties:
      AvailabilityZone:
         Fn::Select:
         - 0
         - Fn::GetAZs: {Ref: 'AWS::Region'}
      VpcId: !Ref 'VPC'
      CidrBlock: !FindInMap ['SubnetConfig', 'PublicOne', 'CIDR']
      MapPublicIpOnLaunch: true

  PublicSubnetTwo:
    Type: AWS::EC2::Subnet
    Properties:
      AvailabilityZone:
         Fn::Select:
         - 1
         - Fn::GetAZs: {Ref: 'AWS::Region'}
      VpcId: !Ref 'VPC'
      CidrBlock: !FindInMap ['SubnetConfig', 'PublicTwo', 'CIDR']
      MapPublicIpOnLaunch: true

  # Setup networking resources for the public subnets. Containers
  # in the public subnets have public IP addresses and the routing table
  # sends network traffic via the internet gateway.
  InternetGateway:
    Type: AWS::EC2::InternetGateway
  GatewayAttachement:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      VpcId: !Ref 'VPC'
      InternetGatewayId: !Ref 'InternetGateway'
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
```

```
    Properties:
      VpcId: !Ref 'VPC'
  PublicRoute:
    Type: AWS::EC2::Route
    DependsOn: GatewayAttachement
    Properties:
      RouteTableId: !Ref 'PublicRouteTable'
      DestinationCidrBlock: '0.0.0.0/0'
      GatewayId: !Ref 'InternetGateway'
  PublicSubnetOneRouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId: !Ref PublicSubnetOne
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetTwoRouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      SubnetId: !Ref PublicSubnetTwo
      RouteTableId: !Ref PublicRouteTable

  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster

  # A security group for the containers we will run in Fargate.
  # Rules are added to this security group based on what ingress you
  # add for the cluster.
  ContainerSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Access to the Fargate containers
      VpcId: !Ref 'VPC'

  # This is a role which is used by the ECS tasks themselves.
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
        - Effect: Allow
          Principal:
            Service: [ecs-tasks.amazonaws.com]
          Action: ['sts:AssumeRole']
      Path: /
```

```
      ManagedPolicyArns:
        - 'arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy'

# These output values are available to service infrastructure as code files as outputs,
 when given the
# the 'service_instance.environment.outputs.' namespace, for example,
 service_instance.environment.outputs.ClusterName.

Outputs:
  ClusterName:                                          # output
    Description: The name of the ECS cluster
    Value: !Ref 'ECSCluster'
  ECSTaskExecutionRole:                                 # output
    Description: The ARN of the ECS role
    Value: !GetAtt 'ECSTaskExecutionRole.Arn'
  VpcId:                                                # output
    Description: The ID of the VPC that this stack is deployed in
    Value: !Ref 'VPC'
  PublicSubnetOne:                                      # output
    Description: Public subnet one
    Value: !Ref 'PublicSubnetOne'
  PublicSubnetTwo:                                      # output
    Description: Public subnet two
    Value: !Ref 'PublicSubnetTwo'
  ContainerSecurityGroup:                               # output
    Description: A security group used to allow Fargate containers to receive traffic
    Value: !Ref 'ContainerSecurityGroup'
```

I file IAc nell'Esempio 1 e nell'Esempio 3 producono CloudFormation pile leggermente diverse. I parametri vengono visualizzati in modo diverso nei file modello dello stack. Il file modello di CloudFormation pila Example 1 mostra le etichette dei parametri (chiavi) nella vista del modello di pila. Il file modello dello stack di AWS Proton CloudFormation infrastruttura Example 3 mostra i valori dei parametri. AWS Proton i parametri di input non vengono visualizzati nella visualizzazione dei parametri CloudFormation dello stack della console.

Nell'Esempio 4, il file IAc del AWS Proton servizio corrisponde all'Esempio 2.

Esempio 4: file IAc dell'istanza di AWS Proton servizio

```
AWSTemplateFormatVersion: '2010-09-09'
Description: Deploy a service on AWS Fargate, hosted in a public subnet, and accessible
 via a public load balancer.
Mappings:
```

```
  TaskSize:
    x-small:
      cpu: 256
      memory: 512
    small:
      cpu: 512
      memory: 1024
    medium:
      cpu: 1024
      memory: 2048
    large:
      cpu: 2048
      memory: 4096
    x-large:
      cpu: 4096
      memory: 8192
Resources:
  # A log group for storing the stdout logs from this service's containers
  LogGroup:
    Type: AWS::Logs::LogGroup
    Properties:
      LogGroupName: '{{service_instance.name}}' # resource parameter

  # The task definition. This is a simple metadata description of what
  # container to run, and what resource requirements it has.
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: '{{service_instance.name}}'
      Cpu: !FindInMap [TaskSize, {{service_instance.inputs.task_size}}, cpu] # input
 parameter
      Memory: !FindInMap [TaskSize, {{service_instance.inputs.task_size}}, memory]
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      ExecutionRoleArn: '{{environment.outputs.ECSTaskExecutionRole}}' # output from an
 environment infrastructure as code file
      TaskRoleArn: !Ref "AWS::NoValue"
      ContainerDefinitions:
        - Name: '{{service_instance.name}}'
          Cpu: !FindInMap [TaskSize, {{service_instance.inputs.task_size}}, cpu]
          Memory: !FindInMap [TaskSize, {{service_instance.inputs.task_size}}, memory]
          Image: '{{service_instance.inputs.image}}'
          PortMappings:
```

```
            - ContainerPort: '{{service_instance.inputs.port}}' # input parameter
          LogConfiguration:
            LogDriver: 'awslogs'
            Options:
              awslogs-group: '{{service_instance.name}}'
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: '{{service_instance.name}}'

 # The service_instance. The service is a resource which allows you to run multiple
 # copies of a type of task, and gather up their logs and metrics, as well
 # as monitor the number of running tasks and replace any that have crashed
 Service:
   Type: AWS::ECS::Service
   DependsOn: LoadBalancerRule
   Properties:
     ServiceName: '{{service_instance.name}}'
     Cluster: '{{environment.outputs.ClusterName}}' # output from an environment
infrastructure as code file
     LaunchType: FARGATE
     DeploymentConfiguration:
       MaximumPercent: 200
       MinimumHealthyPercent: 75
     DesiredCount: '{{service_instance.inputs.desired_count}}'       # input parameter
     NetworkConfiguration:
       AwsvpcConfiguration:
         AssignPublicIp: ENABLED
         SecurityGroups:
           - '{{environment.outputs.ContainerSecurityGroup}}' # output from an
environment infrastructure as code file
         Subnets:
           - '{{environment.outputs.PublicSubnetOne}}'        # output from an
environment infrastructure as code file
           - '{{environment.outputs.PublicSubnetTwo}}'
     TaskDefinition: !Ref 'TaskDefinition'
     LoadBalancers:
       - ContainerName: '{{service_instance.name}}'
         ContainerPort: '{{service_instance.inputs.port}}'
         TargetGroupArn: !Ref 'TargetGroup'

 # A target group. This is used for keeping track of all the tasks, and
 # what IP addresses / port numbers they have. You can query it yourself,
 # to use the addresses yourself, but most often this target group is just
 # connected to an application load balancer, or network load balancer, so
 # it can automatically distribute traffic across all the targets.
```

```
  TargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      HealthCheckIntervalSeconds: 6
      HealthCheckPath: /
      HealthCheckProtocol: HTTP
      HealthCheckTimeoutSeconds: 5
      HealthyThresholdCount: 2
      TargetType: ip
      Name: '{{service_instance.name}}'
      Port: '{{service_instance.inputs.port}}'
      Protocol: HTTP
      UnhealthyThresholdCount: 2
      VpcId: '{{environment.outputs.VpcId}}' # output from an environment
infrastructure as code file

  # Create a rule on the load balancer for routing traffic to the target group
  LoadBalancerRule:
    Type: AWS::ElasticLoadBalancingV2::ListenerRule
    Properties:
      Actions:
        - TargetGroupArn: !Ref 'TargetGroup'
          Type: 'forward'
      Conditions:
        - Field: path-pattern
          Values:
            - '*'
      ListenerArn: !Ref PublicLoadBalancerListener
      Priority: 1

  # Enable autoscaling for this service
  ScalableTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    DependsOn: Service
    Properties:
      ServiceNamespace: 'ecs'
      ScalableDimension: 'ecs:service:DesiredCount'
      ResourceId:
        Fn::Join:
          - '/'
          - - service
            - '{{environment.outputs.ClusterName}}' # output from an environment
infrastructure as code file
            - '{{service_instance.name}}'
```

```
      MinCapacity: 1
      MaxCapacity: 10
      RoleARN: !Sub arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService

  # Create scaling policies for the service
  ScaleDownPolicy:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    DependsOn: ScalableTarget
    Properties:
      PolicyName:
        Fn::Join:
          - '/'
          - - scale
            - '{{service_instance.name}}'
            - down
      PolicyType: StepScaling
      ResourceId:
        Fn::Join:
          - '/'
          - - service
            - '{{environment.outputs.ClusterName}}'
            - '{{service_instance.name}}'
      ScalableDimension: 'ecs:service:DesiredCount'
      ServiceNamespace: 'ecs'
      StepScalingPolicyConfiguration:
        AdjustmentType: 'ChangeInCapacity'
        StepAdjustments:
          - MetricIntervalUpperBound: 0
            ScalingAdjustment: -1
        MetricAggregationType: 'Average'
        Cooldown: 60

  ScaleUpPolicy:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    DependsOn: ScalableTarget
    Properties:
      PolicyName:
        Fn::Join:
          - '/'
          - - scale
            - '{{service_instance.name}}'
            - up
```

```
      PolicyType: StepScaling
      ResourceId:
        Fn::Join:
          - '/'
          - - service
            - '{{environment.outputs.ClusterName}}'
            - '{{service_instance.name}}'
      ScalableDimension: 'ecs:service:DesiredCount'
      ServiceNamespace: 'ecs'
      StepScalingPolicyConfiguration:
        AdjustmentType: 'ChangeInCapacity'
        StepAdjustments:
          - MetricIntervalLowerBound: 0
            MetricIntervalUpperBound: 15
            ScalingAdjustment: 1
          - MetricIntervalLowerBound: 15
            MetricIntervalUpperBound: 25
            ScalingAdjustment: 2
          - MetricIntervalLowerBound: 25
            ScalingAdjustment: 3
        MetricAggregationType: 'Average'
        Cooldown: 60

  # Create alarms to trigger these policies
  LowCpuUsageAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmName:
        Fn::Join:
          - '-'
          - - low-cpu
            - '{{service_instance.name}}'
      AlarmDescription:
        Fn::Join:
          - ' '
          - - "Low CPU utilization for service"
            - '{{service_instance.name}}'
      MetricName: CPUUtilization
      Namespace: AWS/ECS
      Dimensions:
        - Name: ServiceName
          Value: '{{service_instance.name}}'
        - Name: ClusterName
          Value:
```

```
            '{{environment.outputs.ClusterName}}'
      Statistic: Average
      Period: 60
      EvaluationPeriods: 1
      Threshold: 20
      ComparisonOperator: LessThanOrEqualToThreshold
      AlarmActions:
        - !Ref ScaleDownPolicy

  HighCpuUsageAlarm:
    Type: AWS::CloudWatch::Alarm
    Properties:
      AlarmName:
        Fn::Join:
          - '-'
          - - high-cpu
            - '{{service_instance.name}}'
      AlarmDescription:
        Fn::Join:
          - ' '
          - - "High CPU utilization for service"
            - '{{service_instance.name}}'
      MetricName: CPUUtilization
      Namespace: AWS/ECS
      Dimensions:
        - Name: ServiceName
          Value: '{{service_instance.name}}'
        - Name: ClusterName
          Value:
            '{{environment.outputs.ClusterName}}'
      Statistic: Average
      Period: 60
      EvaluationPeriods: 1
      Threshold: 70
      ComparisonOperator: GreaterThanOrEqualToThreshold
      AlarmActions:
        - !Ref ScaleUpPolicy

  EcsSecurityGroupIngressFromPublicALB:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      Description: Ingress from the public ALB
      GroupId: '{{environment.outputs.ContainerSecurityGroup}}'
      IpProtocol: -1
```

```
      SourceSecurityGroupId: !Ref 'PublicLoadBalancerSG'

  # Public load balancer, hosted in public subnets that is accessible
  # to the public, and is intended to route traffic to one or more public
  # facing services. This is used for accepting traffic from the public
  # internet and directing it to public facing microservices
  PublicLoadBalancerSG:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Access to the public facing load balancer
      VpcId: '{{environment.outputs.VpcId}}'
      SecurityGroupIngress:
          # Allow access to ALB from anywhere on the internet
          - CidrIp: 0.0.0.0/0
            IpProtocol: -1

  PublicLoadBalancer:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Scheme: internet-facing
      LoadBalancerAttributes:
      - Key: idle_timeout.timeout_seconds
        Value: '30'
      Subnets:
        # The load balancer is placed into the public subnets, so that traffic
        # from the internet can reach the load balancer directly via the internet
 gateway
        - '{{environment.outputs.PublicSubnetOne}}'
        - '{{environment.outputs.PublicSubnetTwo}}'
      SecurityGroups: [!Ref 'PublicLoadBalancerSG']

  PublicLoadBalancerListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    DependsOn:
      - PublicLoadBalancer
    Properties:
      DefaultActions:
        - TargetGroupArn: !Ref 'TargetGroup'
          Type: 'forward'
      LoadBalancerArn: !Ref 'PublicLoadBalancer'
      Port: 80
      Protocol: HTTP
Outputs:
  ServiceEndpoint:         # output
```

```
    Description: The URL to access the service
    Value: !Sub "http://${PublicLoadBalancer.DNSName}"
```

Nell'esempio 5, il file IAc della AWS Proton pipeline effettua il provisioning dell'infrastruttura della pipeline per supportare le istanze di servizio fornite dall'Esempio 4.

Esempio 5: file IAc della pipeline di servizio AWS Proton

```
Resources:
  ECRRepo:
    Type: AWS::ECR::Repository
    DeletionPolicy: Retain
  BuildProject:
    Type: AWS::CodeBuild::Project
    Properties:
      Artifacts:
        Type: CODEPIPELINE
      Environment:
        ComputeType: BUILD_GENERAL1_SMALL
        Image: aws/codebuild/amazonlinux2-x86_64-standard:3.0
        PrivilegedMode: true
        Type: LINUX_CONTAINER
        EnvironmentVariables:
        - Name: repo_name
          Type: PLAINTEXT
          Value: !Ref ECRRepo
        - Name: service_name
          Type: PLAINTEXT
          Value: '{{ service.name }}'    # resource parameter
      ServiceRole:
        Fn::GetAtt:
          - PublishRole
          - Arn
      Source:
        BuildSpec:
          Fn::Join:
            - ""
            - - >-
                {
                  "version": "0.2",
                  "phases": {
                    "install": {
                      "runtime-versions": {
```

```
                        "docker": 18
                      },
                      "commands": [
                        "pip3 install --upgrade --user awscli",
                        "echo
 'f6bd1536a743ab170b35c94ed4c7c4479763356bd543af5d391122f4af852460  yq_linux_amd64' >
 yq_linux_amd64.sha",
                        "wget https://github.com/mikefarah/yq/releases/download/3.4.0/
yq_linux_amd64",
                        "sha256sum -c yq_linux_amd64.sha",
                        "mv yq_linux_amd64 /usr/bin/yq",
                        "chmod +x /usr/bin/yq"
                      ]
                    },
                    "pre_build": {
                      "commands": [
                        "cd $CODEBUILD_SRC_DIR",
                        "$(aws ecr get-login --no-include-email --region
 $AWS_DEFAULT_REGION)",
                        "{{ pipeline.inputs.unit_test_command }}",    # input parameter
                      ]
                    },
                    "build": {
                      "commands": [
                        "IMAGE_REPO_NAME=$repo_name",
                        "IMAGE_TAG=$CODEBUILD_BUILD_NUMBER",
                        "IMAGE_ID=
              - Ref: AWS::AccountId
              - >-
                .dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:
$IMAGE_TAG",
                        "docker build -t $IMAGE_REPO_NAME:$IMAGE_TAG -f
 {{ pipeline.inputs.dockerfile }} .",     # input parameter
                        "docker tag $IMAGE_REPO_NAME:$IMAGE_TAG $IMAGE_ID;",
                        "docker push $IMAGE_ID"
                      ]
                    },
                    "post_build": {
                      "commands": [
                        "aws proton --region $AWS_DEFAULT_REGION get-service --name
 $service_name | jq -r .service.spec > service.yaml",
                        "yq w service.yaml 'instances[*].spec.image' \"$IMAGE_ID\" >
 rendered_service.yaml"
                      ]
```

```
                    }
                  },
                  "artifacts": {
                    "files": [
                      "rendered_service.yaml"
                    ]
                  }
                }
        Type: CODEPIPELINE
      EncryptionKey:
        Fn::GetAtt:
          - PipelineArtifactsBucketEncryptionKey
          - Arn
{% for service_instance in service_instances %}
  Deploy{{loop.index}}Project:
    Type: AWS::CodeBuild::Project
    Properties:
      Artifacts:
        Type: CODEPIPELINE
      Environment:
        ComputeType: BUILD_GENERAL1_SMALL
        Image: aws/codebuild/amazonlinux2-x86_64-standard:3.0
        PrivilegedMode: false
        Type: LINUX_CONTAINER
        EnvironmentVariables:
        - Name: service_name
          Type: PLAINTEXT
          Value:  '{{service.name}}'          # resource parameter
        - Name: service_instance_name
          Type: PLAINTEXT
          Value: '{{service_instance.name}}'  # resource parameter
      ServiceRole:
        Fn::GetAtt:
          - DeploymentRole
          - Arn
      Source:
        BuildSpec: >-
          {
            "version": "0.2",
            "phases": {
              "build": {
                "commands": [
                  "pip3 install --upgrade --user awscli",
```

```
                  "aws proton --region $AWS_DEFAULT_REGION update-service-instance
 --deployment-type CURRENT_VERSION --name $service_instance_name --service-name
 $service_name --spec file://rendered_service.yaml",
                  "aws proton --region $AWS_DEFAULT_REGION wait service-instance-
deployed --name $service_instance_name --service-name $service_name"
                ]
              }
            }
          }
        Type: CODEPIPELINE
      EncryptionKey:
        Fn::GetAtt:
          - PipelineArtifactsBucketEncryptionKey
          - Arn
{% endfor %}
  # This role is used to build and publish an image to ECR
  PublishRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Action: sts:AssumeRole
            Effect: Allow
            Principal:
              Service: codebuild.amazonaws.com
        Version: "2012-10-17"
  PublishRoleDefaultPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyDocument:
        Statement:
          - Action:
              - logs:CreateLogGroup
              - logs:CreateLogStream
              - logs:PutLogEvents
            Effect: Allow
            Resource:
              - Fn::Join:
                  - ""
                  - - "arn:"
                    - Ref: AWS::Partition
                    - ":logs:"
                    - Ref: AWS::Region
                    - ":"
```

```
                - Ref: AWS::AccountId
                - :log-group:/aws/codebuild/
                - Ref: BuildProject
            - Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":logs:"
                  - Ref: AWS::Region
                  - ":"
                  - Ref: AWS::AccountId
                  - :log-group:/aws/codebuild/
                  - Ref: BuildProject
                  - :*
        - Action:
            - codebuild:CreateReportGroup
            - codebuild:CreateReport
            - codebuild:UpdateReport
            - codebuild:BatchPutTestCases
          Effect: Allow
          Resource:
            Fn::Join:
              - ""
              - - "arn:"
                - Ref: AWS::Partition
                - ":codebuild:"
                - Ref: AWS::Region
                - ":"
                - Ref: AWS::AccountId
                - :report-group/
                - Ref: BuildProject
                - -*
        - Action:
            - ecr:GetAuthorizationToken
          Effect: Allow
          Resource: "*"
        - Action:
            - ecr:BatchCheckLayerAvailability
            - ecr:CompleteLayerUpload
            - ecr:GetAuthorizationToken
            - ecr:InitiateLayerUpload
            - ecr:PutImage
            - ecr:UploadLayerPart
          Effect: Allow
```

```
            Resource:
              Fn::GetAtt:
                - ECRRepo
                - Arn
          - Action:
              - proton:GetService
            Effect: Allow
            Resource: "*"
          - Action:
              - s3:GetObject*
              - s3:GetBucket*
              - s3:List*
              - s3:DeleteObject*
              - s3:PutObject*
              - s3:Abort*
            Effect: Allow
            Resource:
              - Fn::GetAtt:
                  - PipelineArtifactsBucket
                  - Arn
              - Fn::Join:
                  - ""
                  - - Fn::GetAtt:
                        - PipelineArtifactsBucket
                        - Arn
                    - /*
          - Action:
              - kms:Decrypt
              - kms:DescribeKey
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Resource:
              Fn::GetAtt:
                - PipelineArtifactsBucketEncryptionKey
                - Arn
          - Action:
              - kms:Decrypt
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Resource:
```

```
            Fn::GetAtt:
              - PipelineArtifactsBucketEncryptionKey
              - Arn
      Version: "2012-10-17"
    PolicyName: PublishRoleDefaultPolicy
    Roles:
      - Ref: PublishRole

DeploymentRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Statement:
        - Action: sts:AssumeRole
          Effect: Allow
          Principal:
            Service: codebuild.amazonaws.com
      Version: "2012-10-17"
DeploymentRoleDefaultPolicy:
  Type: AWS::IAM::Policy
  Properties:
    PolicyDocument:
      Statement:
        - Action:
            - logs:CreateLogGroup
            - logs:CreateLogStream
            - logs:PutLogEvents
          Effect: Allow
          Resource:
            - Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":logs:"
                  - Ref: AWS::Region
                  - ":"
                  - Ref: AWS::AccountId
                  - :log-group:/aws/codebuild/Deploy*Project*
            - Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":logs:"
                  - Ref: AWS::Region
```

```
                - ":"
                - Ref: AWS::AccountId
                - :log-group:/aws/codebuild/Deploy*Project:*
          - Action:
              - codebuild:CreateReportGroup
              - codebuild:CreateReport
              - codebuild:UpdateReport
              - codebuild:BatchPutTestCases
            Effect: Allow
            Resource:
              Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":codebuild:"
                  - Ref: AWS::Region
                  - ":"
                  - Ref: AWS::AccountId
                  - :report-group/Deploy*Project
                  - -*
          - Action:
              - proton:UpdateServiceInstance
              - proton:GetServiceInstance
            Effect: Allow
            Resource: "*"
          - Action:
              - s3:GetObject*
              - s3:GetBucket*
              - s3:List*
            Effect: Allow
            Resource:
              - Fn::GetAtt:
                  - PipelineArtifactsBucket
                  - Arn
              - Fn::Join:
                  - ""
                  - - Fn::GetAtt:
                        - PipelineArtifactsBucket
                        - Arn
                    - /*
          - Action:
              - kms:Decrypt
              - kms:DescribeKey
            Effect: Allow
```

```
            Resource:
              Fn::GetAtt:
                - PipelineArtifactsBucketEncryptionKey
                - Arn
          - Action:
              - kms:Decrypt
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Resource:
              Fn::GetAtt:
                - PipelineArtifactsBucketEncryptionKey
                - Arn
        Version: "2012-10-17"
      PolicyName: DeploymentRoleDefaultPolicy
      Roles:
        - Ref: DeploymentRole
  PipelineArtifactsBucketEncryptionKey:
    Type: AWS::KMS::Key
    Properties:
      KeyPolicy:
        Statement:
          - Action:
              - kms:Create*
              - kms:Describe*
              - kms:Enable*
              - kms:List*
              - kms:Put*
              - kms:Update*
              - kms:Revoke*
              - kms:Disable*
              - kms:Get*
              - kms:Delete*
              - kms:ScheduleKeyDeletion
              - kms:CancelKeyDeletion
              - kms:GenerateDataKey
              - kms:TagResource
              - kms:UntagResource
            Effect: Allow
            Principal:
              AWS:
                Fn::Join:
                  - ""
```

```
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":iam::"
                  - Ref: AWS::AccountId
                  - :root
            Resource: "*"
          - Action:
              - kms:Decrypt
              - kms:DescribeKey
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Principal:
              AWS:
                Fn::GetAtt:
                  - PipelineRole
                  - Arn
            Resource: "*"
          - Action:
              - kms:Decrypt
              - kms:DescribeKey
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Principal:
              AWS:
                Fn::GetAtt:
                  - PublishRole
                  - Arn
            Resource: "*"
          - Action:
              - kms:Decrypt
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Principal:
              AWS:
                Fn::GetAtt:
                  - PublishRole
                  - Arn
            Resource: "*"
```

```
          - Action:
              - kms:Decrypt
              - kms:DescribeKey
            Effect: Allow
            Principal:
              AWS:
                Fn::GetAtt:
                  - DeploymentRole
                  - Arn
            Resource: "*"
          - Action:
              - kms:Decrypt
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Principal:
              AWS:
                Fn::GetAtt:
                  - DeploymentRole
                  - Arn
            Resource: "*"
        Version: "2012-10-17"
    UpdateReplacePolicy: Delete
    DeletionPolicy: Delete
  PipelineArtifactsBucket:
    Type: AWS::S3::Bucket
    Properties:
      VersioningConfiguration:
        Status: Enabled
      BucketEncryption:
        ServerSideEncryptionConfiguration:
          - ServerSideEncryptionByDefault:
              KMSMasterKeyID:
                Fn::GetAtt:
                  - PipelineArtifactsBucketEncryptionKey
                  - Arn
              SSEAlgorithm: aws:kms
      PublicAccessBlockConfiguration:
        BlockPublicAcls: true
        BlockPublicPolicy: true
        IgnorePublicAcls: true
        RestrictPublicBuckets: true
    UpdateReplacePolicy: Retain
```

```
    DeletionPolicy: Retain
  PipelineArtifactsBucketEncryptionKeyAlias:
    Type: AWS::KMS::Alias
    Properties:
      AliasName: 'alias/codepipeline-encryption-key-{{ service.name }}'
      TargetKeyId:
        Fn::GetAtt:
          - PipelineArtifactsBucketEncryptionKey
          - Arn
    UpdateReplacePolicy: Delete
    DeletionPolicy: Delete
  PipelineRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Action: sts:AssumeRole
            Effect: Allow
            Principal:
              Service: codepipeline.amazonaws.com
        Version: "2012-10-17"
  PipelineRoleDefaultPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyDocument:
        Statement:
          - Action:
              - s3:GetObject*
              - s3:GetBucket*
              - s3:List*
              - s3:DeleteObject*
              - s3:PutObject*
              - s3:Abort*
            Effect: Allow
            Resource:
              - Fn::GetAtt:
                  - PipelineArtifactsBucket
                  - Arn
              - Fn::Join:
                  - ""
                  - - Fn::GetAtt:
                        - PipelineArtifactsBucket
                        - Arn
                    - /*
```

```
          - Action:
              - kms:Decrypt
              - kms:DescribeKey
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Resource:
              Fn::GetAtt:
                - PipelineArtifactsBucketEncryptionKey
                - Arn
          - Action: codestar-connections:*
            Effect: Allow
            Resource: "*"
          - Action: sts:AssumeRole
            Effect: Allow
            Resource:
              Fn::GetAtt:
                - PipelineBuildCodePipelineActionRole
                - Arn
          - Action: sts:AssumeRole
            Effect: Allow
            Resource:
              Fn::GetAtt:
                - PipelineDeployCodePipelineActionRole
                - Arn
        Version: "2012-10-17"
      PolicyName: PipelineRoleDefaultPolicy
      Roles:
        - Ref: PipelineRole
  Pipeline:
    Type: AWS::CodePipeline::Pipeline
    Properties:
      RoleArn:
        Fn::GetAtt:
          - PipelineRole
          - Arn
      Stages:
        - Actions:
            - ActionTypeId:
                Category: Source
                Owner: AWS
                Provider: CodeStarSourceConnection
                Version: "1"
```

```
            Configuration:
              ConnectionArn: '{{ service.repository_connection_arn }}'
              FullRepositoryId: '{{ service.repository_id }}'
              BranchName: '{{ service.branch_name }}'
            Name: Checkout
            OutputArtifacts:
              - Name: Artifact_Source_Checkout
            RunOrder: 1
        Name: Source
      - Actions:
          - ActionTypeId:
              Category: Build
              Owner: AWS
              Provider: CodeBuild
              Version: "1"
            Configuration:
              ProjectName:
                Ref: BuildProject
            InputArtifacts:
              - Name: Artifact_Source_Checkout
            Name: Build
            OutputArtifacts:
              - Name: BuildOutput
            RoleArn:
              Fn::GetAtt:
                - PipelineBuildCodePipelineActionRole
                - Arn
            RunOrder: 1
        Name: Build {%- for service_instance in service_instances %}
      - Actions:
          - ActionTypeId:
              Category: Build
              Owner: AWS
              Provider: CodeBuild
              Version: "1"
            Configuration:
              ProjectName:
                Ref: Deploy{{loop.index}}Project
            InputArtifacts:
              - Name: BuildOutput
            Name: Deploy
            RoleArn:
              Fn::GetAtt:
                - PipelineDeployCodePipelineActionRole
```

```
                  - Arn
              RunOrder: 1
          Name: 'Deploy{{service_instance.name}}'
{%- endfor %}
      ArtifactStore:
        EncryptionKey:
          Id:
            Fn::GetAtt:
              - PipelineArtifactsBucketEncryptionKey
              - Arn
          Type: KMS
        Location:
          Ref: PipelineArtifactsBucket
        Type: S3
    DependsOn:
      - PipelineRoleDefaultPolicy
      - PipelineRole
  PipelineBuildCodePipelineActionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Action: sts:AssumeRole
            Effect: Allow
            Principal:
              AWS:
                Fn::Join:
                  - ""
                  - - "arn:"
                    - Ref: AWS::Partition
                    - ":iam::"
                    - Ref: AWS::AccountId
                    - :root
        Version: "2012-10-17"
  PipelineBuildCodePipelineActionRoleDefaultPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyDocument:
        Statement:
          - Action:
              - codebuild:BatchGetBuilds
              - codebuild:StartBuild
              - codebuild:StopBuild
            Effect: Allow
```

```
          Resource:
            Fn::GetAtt:
              - BuildProject
              - Arn
      Version: "2012-10-17"
    PolicyName: PipelineBuildCodePipelineActionRoleDefaultPolicy
    Roles:
      - Ref: PipelineBuildCodePipelineActionRole
PipelineDeployCodePipelineActionRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Statement:
        - Action: sts:AssumeRole
          Effect: Allow
          Principal:
            AWS:
              Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":iam::"
                  - Ref: AWS::AccountId
                  - :root
      Version: "2012-10-17"
PipelineDeployCodePipelineActionRoleDefaultPolicy:
  Type: AWS::IAM::Policy
  Properties:
    PolicyDocument:
      Statement:
        - Action:
            - codebuild:BatchGetBuilds
            - codebuild:StartBuild
            - codebuild:StopBuild
          Effect: Allow
          Resource:
            Fn::Join:
              - ""
              - - "arn:"
                - Ref: AWS::Partition
                - ":codebuild:"
                - Ref: AWS::Region
                - ":"
                - Ref: AWS::AccountId
```

```
                  - ":project/Deploy*"
        Version: "2012-10-17"
      PolicyName: PipelineDeployCodePipelineActionRoleDefaultPolicy
      Roles:
        - Ref: PipelineDeployCodePipelineActionRole
Outputs:
  PipelineEndpoint:
    Description: The URL to access the pipeline
    Value: !Sub "https://${AWS::Region}.console.aws.amazon.com/codesuite/codepipeline/
pipelines/${Pipeline}/view?region=${AWS::Region}"

                ]
              }
            }
          }
        Type: CODEPIPELINE
      EncryptionKey:
        Fn::GetAtt:
          - PipelineArtifactsBucketEncryptionKey
          - Arn
{% endfor %}
  # This role is used to build and publish an image to ECR
  PublishRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Action: sts:AssumeRole
            Effect: Allow
            Principal:
              Service: codebuild.amazonaws.com
        Version: "2012-10-17"
  PublishRoleDefaultPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyDocument:
        Statement:
          - Action:
              - logs:CreateLogGroup
              - logs:CreateLogStream
              - logs:PutLogEvents
            Effect: Allow
            Resource:
              - Fn::Join:
```

```
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":logs:"
                  - Ref: AWS::Region
                  - ":"
                  - Ref: AWS::AccountId
                  - :log-group:/aws/codebuild/
                  - Ref: BuildProject
              - Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":logs:"
                  - Ref: AWS::Region
                  - ":"
                  - Ref: AWS::AccountId
                  - :log-group:/aws/codebuild/
                  - Ref: BuildProject
                  - :*
          - Action:
              - codebuild:CreateReportGroup
              - codebuild:CreateReport
              - codebuild:UpdateReport
              - codebuild:BatchPutTestCases
            Effect: Allow
            Resource:
              Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":codebuild:"
                  - Ref: AWS::Region
                  - ":"
                  - Ref: AWS::AccountId
                  - :report-group/
                  - Ref: BuildProject
                  - -*
          - Action:
              - ecr:GetAuthorizationToken
            Effect: Allow
            Resource: "*"
          - Action:
              - ecr:BatchCheckLayerAvailability
```

```
            - ecr:CompleteLayerUpload
            - ecr:GetAuthorizationToken
            - ecr:InitiateLayerUpload
            - ecr:PutImage
            - ecr:UploadLayerPart
          Effect: Allow
          Resource:
            Fn::GetAtt:
              - ECRRepo
              - Arn
        - Action:
            - proton:GetService
          Effect: Allow
          Resource: "*"
        - Action:
            - s3:GetObject*
            - s3:GetBucket*
            - s3:List*
            - s3:DeleteObject*
            - s3:PutObject*
            - s3:Abort*
          Effect: Allow
          Resource:
            - Fn::GetAtt:
                - PipelineArtifactsBucket
                - Arn
            - Fn::Join:
                - ""
                - - Fn::GetAtt:
                      - PipelineArtifactsBucket
                      - Arn
                  - /*
        - Action:
            - kms:Decrypt
            - kms:DescribeKey
            - kms:Encrypt
            - kms:ReEncrypt*
            - kms:GenerateDataKey*
          Effect: Allow
          Resource:
            Fn::GetAtt:
              - PipelineArtifactsBucketEncryptionKey
              - Arn
        - Action:
```

```
              - kms:Decrypt
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Resource:
              Fn::GetAtt:
                - PipelineArtifactsBucketEncryptionKey
                - Arn
      Version: "2012-10-17"
    PolicyName: PublishRoleDefaultPolicy
    Roles:
      - Ref: PublishRole

DeploymentRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Statement:
        - Action: sts:AssumeRole
          Effect: Allow
          Principal:
            Service: codebuild.amazonaws.com
      Version: "2012-10-17"
DeploymentRoleDefaultPolicy:
  Type: AWS::IAM::Policy
  Properties:
    PolicyDocument:
      Statement:
        - Action:
            - logs:CreateLogGroup
            - logs:CreateLogStream
            - logs:PutLogEvents
          Effect: Allow
          Resource:
            - Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":logs:"
                  - Ref: AWS::Region
                  - ":"
                  - Ref: AWS::AccountId
                  - :log-group:/aws/codebuild/Deploy*Project*
```

```
            - Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":logs:"
                  - Ref: AWS::Region
                  - ":"
                  - Ref: AWS::AccountId
                  - :log-group:/aws/codebuild/Deploy*Project:*
        - Action:
            - codebuild:CreateReportGroup
            - codebuild:CreateReport
            - codebuild:UpdateReport
            - codebuild:BatchPutTestCases
          Effect: Allow
          Resource:
            Fn::Join:
              - ""
              - - "arn:"
                - Ref: AWS::Partition
                - ":codebuild:"
                - Ref: AWS::Region
                - ":"
                - Ref: AWS::AccountId
                - :report-group/Deploy*Project
                - -*
        - Action:
            - proton:UpdateServiceInstance
            - proton:GetServiceInstance
          Effect: Allow
          Resource: "*"
        - Action:
            - s3:GetObject*
            - s3:GetBucket*
            - s3:List*
          Effect: Allow
          Resource:
            - Fn::GetAtt:
                - PipelineArtifactsBucket
                - Arn
            - Fn::Join:
                - ""
                - - Fn::GetAtt:
                      - PipelineArtifactsBucket
```

```
                      - Arn
                  - /*
          - Action:
              - kms:Decrypt
              - kms:DescribeKey
            Effect: Allow
            Resource:
              Fn::GetAtt:
                - PipelineArtifactsBucketEncryptionKey
                - Arn
          - Action:
              - kms:Decrypt
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Resource:
              Fn::GetAtt:
                - PipelineArtifactsBucketEncryptionKey
                - Arn
        Version: "2012-10-17"
      PolicyName: DeploymentRoleDefaultPolicy
      Roles:
        - Ref: DeploymentRole
  PipelineArtifactsBucketEncryptionKey:
    Type: AWS::KMS::Key
    Properties:
      KeyPolicy:
        Statement:
          - Action:
              - kms:Create*
              - kms:Describe*
              - kms:Enable*
              - kms:List*
              - kms:Put*
              - kms:Update*
              - kms:Revoke*
              - kms:Disable*
              - kms:Get*
              - kms:Delete*
              - kms:ScheduleKeyDeletion
              - kms:CancelKeyDeletion
              - kms:GenerateDataKey
              - kms:TagResource
```

```
            - kms:UntagResource
          Effect: Allow
          Principal:
            AWS:
              Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":iam::"
                  - Ref: AWS::AccountId
                  - :root
          Resource: "*"
        - Action:
            - kms:Decrypt
            - kms:DescribeKey
            - kms:Encrypt
            - kms:ReEncrypt*
            - kms:GenerateDataKey*
          Effect: Allow
          Principal:
            AWS:
              Fn::GetAtt:
                - PipelineRole
                - Arn
          Resource: "*"
        - Action:
            - kms:Decrypt
            - kms:DescribeKey
            - kms:Encrypt
            - kms:ReEncrypt*
            - kms:GenerateDataKey*
          Effect: Allow
          Principal:
            AWS:
              Fn::GetAtt:
                - PublishRole
                - Arn
          Resource: "*"
        - Action:
            - kms:Decrypt
            - kms:Encrypt
            - kms:ReEncrypt*
            - kms:GenerateDataKey*
          Effect: Allow
```

```
            Principal:
              AWS:
                Fn::GetAtt:
                  - PublishRole
                  - Arn
            Resource: "*"
          - Action:
              - kms:Decrypt
              - kms:DescribeKey
            Effect: Allow
            Principal:
              AWS:
                Fn::GetAtt:
                  - DeploymentRole
                  - Arn
            Resource: "*"
          - Action:
              - kms:Decrypt
              - kms:Encrypt
              - kms:ReEncrypt*
              - kms:GenerateDataKey*
            Effect: Allow
            Principal:
              AWS:
                Fn::GetAtt:
                  - DeploymentRole
                  - Arn
            Resource: "*"
        Version: "2012-10-17"
    UpdateReplacePolicy: Delete
    DeletionPolicy: Delete
  PipelineArtifactsBucket:
    Type: AWS::S3::Bucket
    Properties:
      BucketEncryption:
        ServerSideEncryptionConfiguration:
          - ServerSideEncryptionByDefault:
              KMSMasterKeyID:
                Fn::GetAtt:
                  - PipelineArtifactsBucketEncryptionKey
                  - Arn
              SSEAlgorithm: aws:kms
      PublicAccessBlockConfiguration:
        BlockPublicAcls: true
```

```
        BlockPublicPolicy: true
        IgnorePublicAcls: true
        RestrictPublicBuckets: true
    UpdateReplacePolicy: Retain
    DeletionPolicy: Retain
  PipelineArtifactsBucketEncryptionKeyAlias:
    Type: AWS::KMS::Alias
    Properties:
      AliasName: 'alias/codepipeline-encryption-key-{{ service.name }}'     # resource
parameter
      TargetKeyId:
        Fn::GetAtt:
          - PipelineArtifactsBucketEncryptionKey
          - Arn
    UpdateReplacePolicy: Delete
    DeletionPolicy: Delete
  PipelineRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Action: sts:AssumeRole
            Effect: Allow
            Principal:
              Service: codepipeline.amazonaws.com
        Version: "2012-10-17"
  PipelineRoleDefaultPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyDocument:
        Statement:
          - Action:
              - s3:GetObject*
              - s3:GetBucket*
              - s3:List*
              - s3:DeleteObject*
              - s3:PutObject*
              - s3:Abort*
            Effect: Allow
            Resource:
              - Fn::GetAtt:
                  - PipelineArtifactsBucket
                  - Arn
              - Fn::Join:
```

```
                - ""
                - - Fn::GetAtt:
                      - PipelineArtifactsBucket
                      - Arn
                  - /*
        - Action:
            - kms:Decrypt
            - kms:DescribeKey
            - kms:Encrypt
            - kms:ReEncrypt*
            - kms:GenerateDataKey*
          Effect: Allow
          Resource:
            Fn::GetAtt:
              - PipelineArtifactsBucketEncryptionKey
              - Arn
        - Action: codestar-connections:*
          Effect: Allow
          Resource: "*"
        - Action: sts:AssumeRole
          Effect: Allow
          Resource:
            Fn::GetAtt:
              - PipelineBuildCodePipelineActionRole
              - Arn
        - Action: sts:AssumeRole
          Effect: Allow
          Resource:
            Fn::GetAtt:
              - PipelineDeployCodePipelineActionRole
              - Arn
      Version: "2012-10-17"
    PolicyName: PipelineRoleDefaultPolicy
    Roles:
      - Ref: PipelineRole
Pipeline:
  Type: AWS::CodePipeline::Pipeline
  Properties:
    RoleArn:
      Fn::GetAtt:
        - PipelineRole
        - Arn
    Stages:
      - Actions:
```

```
            - ActionTypeId:
                Category: Source
                Owner: AWS
                Provider: CodeStarSourceConnection
                Version: "1"
              Configuration:
                ConnectionArn: '{{ service.repository_connection_arn }}'   # resource
parameter
                FullRepositoryId: '{{ service.repository_id }}'            # resource
parameter
                BranchName: '{{ service.branch_name }}'                    # resource
parameter
              Name: Checkout
              OutputArtifacts:
                - Name: Artifact_Source_Checkout
              RunOrder: 1
          Name: Source
        - Actions:
            - ActionTypeId:
                Category: Build
                Owner: AWS
                Provider: CodeBuild
                Version: "1"
              Configuration:
                ProjectName:
                  Ref: BuildProject
              InputArtifacts:
                - Name: Artifact_Source_Checkout
              Name: Build
              OutputArtifacts:
                - Name: BuildOutput
              RoleArn:
                Fn::GetAtt:
                  - PipelineBuildCodePipelineActionRole
                  - Arn
              RunOrder: 1
          Name: Build {%- for service_instance in service_instances %}
        - Actions:
            - ActionTypeId:
                Category: Build
                Owner: AWS
                Provider: CodeBuild
                Version: "1"
              Configuration:
```

```
                ProjectName:
                  Ref: Deploy{{loop.index}}Project
              InputArtifacts:
                - Name: BuildOutput
              Name: Deploy
              RoleArn:
                Fn::GetAtt:
                  - PipelineDeployCodePipelineActionRole
                  - Arn
              RunOrder: 1
          Name: 'Deploy{{service_instance.name}}'         # resource parameter
{%- endfor %}
      ArtifactStore:
        EncryptionKey:
          Id:
            Fn::GetAtt:
              - PipelineArtifactsBucketEncryptionKey
              - Arn
          Type: KMS
        Location:
          Ref: PipelineArtifactsBucket
        Type: S3
    DependsOn:
      - PipelineRoleDefaultPolicy
      - PipelineRole
  PipelineBuildCodePipelineActionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Action: sts:AssumeRole
            Effect: Allow
            Principal:
              AWS:
                Fn::Join:
                  - ""
                  - - "arn:"
                    - Ref: AWS::Partition
                    - ":iam::"
                    - Ref: AWS::AccountId
                    - :root
        Version: "2012-10-17"
  PipelineBuildCodePipelineActionRoleDefaultPolicy:
    Type: AWS::IAM::Policy
```

```
    Properties:
      PolicyDocument:
        Statement:
          - Action:
              - codebuild:BatchGetBuilds
              - codebuild:StartBuild
              - codebuild:StopBuild
            Effect: Allow
            Resource:
              Fn::GetAtt:
                - BuildProject
                - Arn
        Version: "2012-10-17"
      PolicyName: PipelineBuildCodePipelineActionRoleDefaultPolicy
      Roles:
        - Ref: PipelineBuildCodePipelineActionRole
  PipelineDeployCodePipelineActionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Action: sts:AssumeRole
            Effect: Allow
            Principal:
              AWS:
                Fn::Join:
                  - ""
                  - - "arn:"
                    - Ref: AWS::Partition
                    - ":iam::"
                    - Ref: AWS::AccountId
                    - :root
        Version: "2012-10-17"
  PipelineDeployCodePipelineActionRoleDefaultPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyDocument:
        Statement:
          - Action:
              - codebuild:BatchGetBuilds
              - codebuild:StartBuild
              - codebuild:StopBuild
            Effect: Allow
            Resource:
```

```
              Fn::Join:
                - ""
                - - "arn:"
                  - Ref: AWS::Partition
                  - ":codebuild:"
                  - Ref: AWS::Region
                  - ":"
                  - Ref: AWS::AccountId
                  - ":project/Deploy*"
        Version: "2012-10-17"
      PolicyName: PipelineDeployCodePipelineActionRoleDefaultPolicy
      Roles:
        - Ref: PipelineDeployCodePipelineActionRole
Outputs:
  PipelineEndpoint:
    Description: The URL to access the pipeline
    Value: !Sub "https://${AWS::Region}.console.aws.amazon.com/codesuite/codepipeline/
pipelines/${Pipeline}/view?region=${AWS::Region}"
```

CodeBuild pacchetto di modelli di provisioning

Con il CodeBuild provisioning, invece di utilizzare i modelli IAc per renderizzare i file IAc ed eseguirli utilizzando un motore di provisioning IaC, AWS Proton esegue semplicemente i comandi della shell. A tale scopo, AWS Proton crea un AWS CodeBuild progetto per l'ambiente, nell'account di ambiente, e avvia un processo per eseguire i comandi per ogni AWS Proton creazione o aggiornamento delle risorse. Quando si crea un pacchetto di modelli, si fornisce un manifesto che specifica i comandi di provisioning e deprovisioning dell'infrastruttura e tutti i programmi, gli script e gli altri file necessari a tali comandi. I comandi sono in grado di leggere gli input che AWS Proton forniscono e sono responsabili del provisioning o del deprovisioning dell'infrastruttura e della generazione di valori di output.

Il manifesto specifica inoltre come AWS Proton deve essere visualizzato il file di input da cui il codice può immettere e da cui ottenere i valori di input. Può essere renderizzato in JSON o HCL. Per ulteriori informazioni sui parametri di input, vedere. the section called “CodeBuild parametri di provisioning” Per ulteriori informazioni sui file manifest, consulta the section called “Manifesta e concludi”.

> **Note**
>
> È possibile utilizzare il CodeBuild provisioning con ambienti e servizi. Al momento non è possibile effettuare il provisioning dei componenti in questo modo.

Esempio: utilizzo di AWS CDK with CodeBuild provisioning

Come esempio di utilizzo del CodeBuild provisioning, è possibile includere codice che utilizza le AWS risorse AWS Cloud Development Kit (AWS CDK) per fornire (distribuire) e deprovisionare (distruggere) e un manifest che installa il CDK ed esegue il codice CDK.

Nelle sezioni seguenti sono elencati i file di esempio che è possibile includere in un pacchetto di modelli di provisioning che effettua il CodeBuild provisioning di un ambiente utilizzando. AWS CDK

Manifesto

Il seguente file manifesto specifica il CodeBuild provisioning e include i comandi necessari per installare e utilizzare il file di output AWS CDK, l'elaborazione dei file di output e la restituzione degli output. AWS Proton

Example infrastruttura/manifest.yaml

```
infrastructure:
  templates:
    - rendering_engine: codebuild
      settings:
        image: aws/codebuild/amazonlinux2-x86_64-standard:4.0
        runtimes:
          nodejs: 16
        provision:
          - npm install
          - npm run build
          - npm run cdk bootstrap
          - npm run cdk deploy -- --require-approval never --outputs-file proton-
outputs.json
          - jq 'to_entries | map_values(.value) | add | to_entries | map({key:.key,
 valueString:.value})' < proton-outputs.json > outputs.json
          - aws proton notify-resource-deployment-status-change --resource-arn
 $RESOURCE_ARN --status IN_PROGRESS --outputs file://./outputs.json
        deprovision:
          - npm install
          - npm run build
          - npm run cdk destroy
        project_properties:
          VpcConfig:
            VpcId: "{{ environment.inputs.codebuild_vpc_id }}"
            Subnets: "{{ environment.inputs.codebuild_subnets }}"
```

```
            SecurityGroupIds: "{{ environment.inputs.codebuild_security_groups }}"
```

Schema

Il seguente file di schema definisce i parametri per l'ambiente. Il AWS CDK codice può fare riferimento ai valori di questi parametri durante la distribuzione.

Example schema/schema.yaml

```
schema:
  format:
    openapi: "3.0.0"
  environment_input_type: "MyEnvironmentInputType"
  types:
    MyEnvironmentInputType:
      type: object
      description: "Input properties for my environment"
      properties:
        my_sample_input:
          type: string
          description: "This is a sample input"
          default: "hello world"
        my_other_sample_input:
          type: string
          description: "Another sample input"
      required:
        - my_other_sample_input
```

AWS CDK file

I seguenti file sono un esempio di progetto CDK Node.js.

Example infrastruttura/package.json

```
{
  "name": "ProtonEnvironment",
  "version": "0.1.0",
  "bin": {
    "ProtonEnvironmente": "bin/ProtonEnvironment.js"
  },
  "scripts": {
    "build": "tsc",
    "watch": "tsc -w",
```

```
    "test": "jest",
    "cdk": "cdk"
  },
  "devDependencies": {
    "@types/jest": "^28.1.7",
    "@types/node": "18.7.6",
    "jest": "^28.1.3",
    "ts-jest": "^28.0.8",
    "aws-cdk": "2.37.1",
    "ts-node": "^10.9.1",
    "typescript": "~4.7.4"
  },
  "dependencies": {
    "aws-cdk-lib": "2.37.1",
    "constructs": "^10.1.77",
    "source-map-support": "^0.5.21"
  }
}
```

Example infrastruttura/tsconfig.json

```
{
  "compilerOptions": {
    "target": "ES2018",
    "module": "commonjs",
    "lib": [
      "es2018"
    ],
    "declaration": true,
    "strict": true,
    "noImplicitAny": true,
    "strictNullChecks": true,
    "noImplicitThis": true,
    "alwaysStrict": true,
    "noUnusedLocals": false,
    "noUnusedParameters": false,
    "noImplicitReturns": true,
    "noFallthroughCasesInSwitch": false,
    "inlineSourceMap": true,
    "inlineSources": true,
    "experimentalDecorators": true,
    "strictPropertyInitialization": false,
    "resolveJsonModule": true,
```

```
    "esModuleInterop": true,
    "typeRoots": [
      "./node_modules/@types"
    ]
  },
  "exclude": [
    "node_modules",
    "cdk.out"
  ]
}
```

Example infrastruttura/cdk.json

```
{
  "app": "npx ts-node --prefer-ts-exts bin/ProtonEnvironment.ts",
  "outputsFile": "proton-outputs.json",
  "watch": {
    "include": [
      "**"
    ],
    "exclude": [
      "README.md",
      "cdk*.json",
      "**/*.d.ts",
      "**/*.js",
      "tsconfig.json",
      "package*.json",
      "yarn.lock",
      "node_modules",
      "test"
    ]
  },
  "context": {
    "@aws-cdk/aws-apigateway:usagePlanKeyOrderInsensitiveId": true,
    "@aws-cdk/core:stackRelativeExports": true,
    "@aws-cdk/aws-rds:lowercaseDbIdentifier": true,
    "@aws-cdk/aws-lambda:recognizeVersionProps": true,
    "@aws-cdk/aws-cloudfront:defaultSecurityPolicyTLSv1.2_2021": true,
    "@aws-cdk-containers/ecs-service-extensions:enableDefaultLogDriver": true,
    "@aws-cdk/aws-ec2:uniqueImdsv2TemplateName": true,
    "@aws-cdk/core:target-partitions": [
      "aws",
      "aws-cn"
```

```
    ]
  }
}
```

Example infrastructure/bin/ProtonEnvironment.ts

```
#!/usr/bin/env node
import 'source-map-support/register';
import * as cdk from 'aws-cdk-lib';
import { ProtonEnvironmentStack } from '../lib/ProtonEnvironmentStack';

const app = new cdk.App();
new ProtonEnvironmentStack(app, 'ProtonEnvironmentStack', {});
```

Example infrastructure/lib/ProtonEnvironmentStack.ts

```
import { Stack, StackProps } from 'aws-cdk-lib';
import { Construct } from 'constructs';
import * as cdk from 'aws-cdk-lib';
import * as ssm from 'aws-cdk-lib/aws-ssm';
import input from '../proton-inputs.json';

export class ProtonEnvironmentStack extends Stack {
  constructor(scope: Construct, id: string, props?: StackProps) {
    super(scope, id, { ...props, stackName: process.env.STACK_NAME });

    const ssmParam = new ssm.StringParameter(this, "ssmParam", {
      stringValue: input.environment.inputs.my_sample_input,
      parameterName: `${process.env.STACK_NAME}-Param`,
      tier: ssm.ParameterTier.STANDARD
    })

    new cdk.CfnOutput(this, 'ssmParamOutput', {
      value: ssmParam.parameterName,
      description: 'The name of the ssm parameter',
      exportName: `${process.env.STACK_NAME}-Param`
    });
  }
}
```

File di input renderizzato

Quando si crea un ambiente utilizzando un modello di provisioning CodeBuild basato su un modello di provisioning, AWS Proton esegue il rendering di un file di input con i valori dei parametri di input forniti dall'utente. Il codice può fare riferimento a questi valori. Il file seguente è un esempio di file di input renderizzato.

Example infrastruttura/proton-inputs.json

```
{
  "environment": {
    "name": "myenv",
    "inputs": {
      "my_sample_input": "10.0.0.0/16",
      "my_other_sample_input": "11.0.0.0/16"
    }
  }
}
```

file Terraform iAC

Scopri come utilizzare l'infrastruttura Terraform come file di codice (IaC) con. AWS ProtonTerraform è un motore IaC open source ampiamente utilizzato sviluppato da. HashiCorp I moduli Terraform sono sviluppati nel linguaggio HashiCorp HCL e supportano diversi fornitori di infrastrutture di backend, tra cui Amazon Web Services.

AWS Proton supporta il provisioning autogestito per Terraform IAc.

Per un esempio completo di un repository di provisioning che risponde alle pull request e implementa il provisioning dell'infrastruttura, consulta il modello di automazione Terraform Actions per on. OpenSource GitHub AWS Proton GitHub

Come funziona il provisioning autogestito con i file bundle di modelli Terraform IAc:

1. Quando crei un ambiente dai pacchetti di modelli Terraform, AWS Proton compila i tuoi file con la console o i parametri di input. `.tf spec file`
2. Effettua una richiesta pull per unire i file IAc compilati nel repository con cui ti sei registrato. AWS Proton
3. Se la richiesta viene approvata, AWS Proton attende lo stato di approvvigionamento fornito.
4. Se la richiesta viene rifiutata, la creazione dell'ambiente viene annullata.

5. Se la pull request scade, la creazione dell'ambiente non è completa.

AWS Proton con considerazioni su Terraform IAc:

- AWS Proton non gestisce il provisioning di Terraform.
- È necessario registrare un repository di provisioning con. AWS Proton AWS Proton effettua richieste pull su questo repository.
- È necessario creare una CodeStar connessione per connettersi al AWS Proton repository di provisioning.
- Per effettuare il provisioning da file IaC AWS Proton compilati, è necessario rispondere alle AWS Proton pull request. AWS Proton effettua richieste pull dopo azioni di creazione e aggiornamento dell'ambiente e del servizio. Per ulteriori informazioni, consultare AWS Proton ambienti e AWS Proton servizi.
- Per effettuare il provisioning di una pipeline da file IAc AWS Proton compilati, è necessario creare un repository di pipeline CI/CD.
- L'automazione del provisioning basata su pull request deve includere passaggi per notificare eventuali AWS Proton modifiche allo stato delle risorse assegnate. AWS Proton Puoi usare l'API. AWS Proton NotifyResourceDeploymentStatusChange
- Non è possibile distribuire servizi, pipeline e componenti creati da file CloudFormation IAc in ambienti creati da file Terraform IAc.
- Non è possibile distribuire servizi, pipeline e componenti creati da file Terraform IAc in ambienti creati da file IAc. CloudFormation

Quando prepari i file Terraform IAc per AWS Proton, alleghi dei namespace alle variabili di input, come mostrato negli esempi seguenti. Per ulteriori informazioni, consulta Parametri.

Esempio 1: file Terraform IAc di ambiente AWS Proton

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
  // This tells terraform to store the state file in s3 at the location
```

```
  // s3://terraform-state-bucket/tf-os-sample/terraform.tfstate
  backend "s3" {
    bucket = "terraform-state-bucket"
    key    = "tf-os-sample/terraform.tfstate"
    region = "us-east-1"
  }
}

// Configure the AWS Provider
provider "aws" {
  region = "us-east-1"
  default_tags {
    tags = var.proton_tags
  }
}

resource "aws_ssm_parameter" "my_ssm_parameter" {
  name  = "my_ssm_parameter"
  type  = "String"
  // Use the Proton environment.inputs. namespace
  value = var.environment.inputs.ssm_parameter_value
}
```

Infrastruttura compilata come codice

Quando crei un ambiente o un servizio, AWS Proton compila l'infrastruttura come file di codice con console o `spec file` input. Crea `proton.auto.tfvars.json` file per `proton.`*`resource-type`*`.variables.tf` i tuoi input che possono essere utilizzati da Terraform, come mostrato negli esempi seguenti. Questi file si trovano in un repository specificato in una cartella che corrisponde al nome dell'ambiente o dell'istanza del servizio.

L'esempio mostra come AWS Proton include i tag nella definizione e nei valori delle variabili e come è possibile propagare questi AWS Proton tag alle risorse assegnate. Per ulteriori informazioni, consulta the section called “Propagazione dei tag alle risorse assegnate”.

Esempio 2: file IAc compilati per un ambiente denominato «dev».

dev/environment.tf:

```
terraform {
  required_providers {
    aws = {
```

```
      source  = "hashicorp/aws"
      version = "~> 3.0"
    }
  }
  // This tells terraform to store the state file in s3 at the location
  // s3://terraform-state-bucket/tf-os-sample/terraform.tfstate
  backend "s3" {
    bucket = "terraform-state-bucket"
    key    = "tf-os-sample/terraform.tfstate"
    region = "us-east-1"
  }
}

// Configure the AWS Provider
provider "aws" {
  region = "us-east-1"
  default_tags {
    tags = var.proton_tags
  }
}

resource "aws_ssm_parameter" "my_ssm_parameter" {
  name  = "my_ssm_parameter"
  type  = "String"
  // Use the Proton environment.inputs. namespace
  value = var.environment.inputs.ssm_parameter_value
}
```

dev/proton.environment.variables.ttf:

```
variable "environment" {
  type = object({
    inputs = map(string)
    name = string
  })
}

variable "proton_tags" {
  type = map(string)
  default = null
}
```

dev/proton.auto.tfvars.json:

```
{
  "environment": {
    "name": "dev",
    "inputs": {
      "ssm_parameter_value": "MyNewParamValue"
    }
  }

  "proton_tags" : {
    "proton:account" : "123456789012",
    "proton:template" : "arn:aws:proton:us-east-1:123456789012:environment-template/
fargate-env",
    "proton:environment" : "arn:aws:proton:us-east-1:123456789012:environment/dev"
  }
}
```

Percorsi del repository

AWS Proton utilizza gli input della console o delle specifiche provenienti dall'ambiente o dal servizio, crea azioni per trovare il repository e il percorso in cui collocare i file IAc compilati. I valori di input vengono passati a parametri di input con namespace.

AWS Proton supporta due layout di percorso del repository. Negli esempi seguenti, i percorsi sono denominati in base ai parametri delle risorse con namespace di due ambienti. Ogni ambiente dispone di istanze di servizio di due servizi e le istanze di servizio di uno dei servizi hanno componenti definiti direttamente.

Tipo di risorsa	Parametro del nome	=	Nome risorsa
Ambiente	`environment.name`		`"env-prod"`
Ambiente	`environment.name`	=	`"env-staged"`
Servizio	`service.name`		`"service-one"`

Tipo di risorsa	Parametro del nome	=	Nome risorsa
Istanza del servizio	`service_instance.name`		`"instance-one-prod"`
Istanza del servizio	`service_instance.name`		`"instance-one-staged"`
Servizio	`service.name`		`"service-two"`
Istanza del servizio	`service_instance.name`		`"instance-two-prod"`
Componente	`service_instance.components.default.name`		`"component-prod"`
Istanza del servizio	`service_instance.name`		`"instance-two-staged"`
Componente	`service_instance.components.default.name`		`"component-staged"`

Layout 1

Se AWS Proton trova il repository specificato con una `environments` cartella, crea una cartella che include i file IAc compilati e viene denominata con. `environment.name`

Se AWS Proton trova il repository specificato con una `environments` cartella che contiene un nome di cartella che corrisponde a un nome di ambiente compatibile con le istanze di servizio, crea una cartella che include i file IAc dell'istanza compilata e viene denominata con. `service_instance.name`

```
/repo
    /environments
        /env-prod                           # environment folder
            main.tf
            proton.environment.variables.tf
            proton.auto.tfvars.json

            /service-one-instance-one-prod    # instance folder
                main.tf
                proton.service_instance.variables.tf
                proton.auto.tfvars.json

            /service-two-instance-two-prod    # instance folder
                main.tf
                proton.service_instance.variables.tf
                proton.auto.tfvars.json

            /component-prod                   # component folder
                main.tf
                proton.component.variables.tf
                proton.auto.tfvars.json

        /env-staged                         # environment folder
            main.tf
            proton.variables.tf
            proton.auto.tfvars.json

            /service-one-instance-one-staged  # instance folder
                main.tf
                proton.service_instance.variables.tf
                proton.auto.tfvars.json

            /service-two-instance-two-staged  # instance folder
                main.tf
                proton.service_instance.variables.tf
                proton.auto.tfvars.json

            /component-staged                 # component folder
                main.tf
                proton.component.variables.tf
                proton.auto.tfvars.json
```

Layout 2

Se AWS Proton trova il repository specificato senza una `environments` cartella, crea una `environment.name` cartella in cui collocare i file IAc dell'ambiente compilato.

Se AWS Proton trova il repository specificato con un nome di cartella che corrisponde a un nome di ambiente compatibile con l'istanza di servizio, crea una `service_instance.name` cartella in cui localizza i file IAc dell'istanza compilata.

```
/repo
    /env-prod                           # environment folder
        main.tf
        proton.environment.variables.tf
        proton.auto.tfvars.json

        /service-one-instance-one-prod    # instance folder
            main.tf
            proton.service_instance.variables.tf
            proton.auto.tfvars.json

        /service-two-instance-two-prod    # instance folder
            main.tf
            proton.service_instance.variables.tf
            proton.auto.tfvars.json

        /component-prod                   # component folder
            main.tf
            proton.component.variables.tf
            proton.auto.tfvars.json

    /env-staged                           # environment folder
        main.tf
        proton.variables.tf
        proton.auto.tfvars.json

        /service-one-instance-one-staged  # instance folder
            main.tf
            proton.service_instance.variables.tf
            proton.auto.tfvars.json

        /service-two-instance-two-staged  # instance folder
            main.tf
            proton.service_instance.variables.tf
```

```
        proton.auto.tfvars.json

    /component-staged                # component folder
        main.tf
        proton.component.variables.tf
        proton.auto.tfvars.json
```

File di schema

In qualità di amministratore, quando utilizzi la sezione Open API Data Models (schemas) per definire un file YAML dello schema di parametri per il tuo pacchetto di modelli, AWS Proton puoi convalidare gli input dei valori dei parametri in base ai requisiti definiti nello schema.

Per ulteriori informazioni sui formati e sulle parole chiave disponibili, consulta la sezione Oggetto Schema di OpenAPI.

Requisiti dello schema per i pacchetti di modelli di ambiente

Lo schema deve seguire la sezione Data Models (schemas) di OpenAPI in formato YAML. Deve inoltre far parte del pacchetto di modelli di ambiente.

Per lo schema dell'ambiente, devi includere le intestazioni formattate per stabilire che stai utilizzando la sezione Data Models (schemas) dell'Open API. Nei seguenti esempi di schemi di ambiente, queste intestazioni vengono visualizzate nelle prime tre righe.

Un `environment_input_type` deve essere incluso e definito con un nome fornito dall'utente. Negli esempi seguenti, questo è definito alla riga 5. Definendo questo parametro, lo si associa a una risorsa AWS Proton ambientale.

Per seguire il modello di schema Open API, è necessario include`retypes`. Nell'esempio seguente, questa è la riga 6.

Di seguito`types`, è necessario definire un `environment_input_type` tipo. I parametri di input per l'ambiente vengono definiti come proprietà di`environment_input_type`. È necessario includere almeno una proprietà con un nome che corrisponda ad almeno un parametro elencato nell'infrastruttura di ambiente come file di codice (IaC) associato allo schema.

Quando create un ambiente e fornite valori di parametro personalizzati, AWS Proton utilizza il file di schema per abbinarli, convalidarli e inserirli nei parametri tra parentesi graffe del file IAc associato.

CloudFormation Per ogni proprietà (parametro), fornite una e. `name` `type` Facoltativamente, fornisci anche un `description`default, e`pattern`.

I parametri definiti per lo schema del modello di ambiente standard di esempio seguente includono `vpc_cidr`subnet_one_cidr, e `subnet_two_cidr` con la `default` parola chiave e i valori predefiniti. Quando si crea un ambiente con questo schema di bundle di modelli di ambiente, è possibile accettare i valori predefiniti o fornire valori personalizzati. Se un parametro non ha un valore predefinito ed è elencato come `required` proprietà (parametro), è necessario fornire i relativi valori quando si crea un ambiente.

Il secondo esempio di schema di modello di ambiente standard elenca il `required` parametro`my_other_sample_input`.

È possibile creare uno schema per due tipi di modelli di ambiente. Per ulteriori informazioni, consulta Registra e pubblica modelli.

- Modelli di ambiente standard

  Nell'esempio seguente, viene definito un tipo di input di ambiente con una descrizione e proprietà di input. Questo esempio di schema può essere utilizzato con il file AWS Proton CloudFormation IAc mostrato nell'Esempio 3.

  Schema di esempio per un modello di ambiente standard:

```
schema:                           # required
  format:                         # required
    openapi: "3.0.0"              # required
  # required              defined by administrator
  environment_input_type: "PublicEnvironmentInput"
  types:                          # required
    # defined by administrator
    PublicEnvironmentInput:
      type: object
      description: "Input properties for my environment"
      properties:
        vpc_cidr:                  # parameter
          type: string
          description: "This CIDR range for your VPC"
          default: 10.0.0.0/16
          pattern: ([0-9]{1,3}\.){3}[0-9]{1,3}($|/(16|24))
        subnet_one_cidr:           # parameter
          type: string
```

```
          description: "The CIDR range for subnet one"
          default: 10.0.0.0/24
          pattern: ([0-9]{1,3}\.){3}[0-9]{1,3}($|/(16|24))
        subnet_two_cidr:            # parameter
          type: string
          description: "The CIDR range for subnet one"
          default: 10.0.1.0/24
          pattern: ([0-9]{1,3}\.){3}[0-9]{1,3}($|/(16|24))
```

Schema di esempio per un modello di ambiente standard che include un `required` parametro:

```
schema:                           # required
  format:                         # required
    openapi: "3.0.0"              # required
  # required              defined by administrator
  environment_input_type: "MyEnvironmentInputType"
  types:                          # required
    # defined by administrator
    MyEnvironmentInputType:
      type: object
      description: "Input properties for my environment"
      properties:
        my_sample_input:          # parameter
          type: string
          description: "This is a sample input"
          default: "hello world"
        my_other_sample_input:    # parameter
          type: string
          description: "Another sample input"
        another_optional_input:   # parameter
          type: string
          description: "Another optional input"
          default: "!"
      required:
        - my_other_sample_input
```

- Modelli di ambiente gestiti dal cliente

  Nell'esempio seguente, lo schema include solo un elenco di output che replicano gli output dell'IaC utilizzato per il provisioning dell'infrastruttura gestita dal cliente. È necessario definire i tipi di valori di output solo come stringhe (non elenchi, array o altri tipi). Ad esempio, il frammento di codice successivo mostra la sezione degli output di un modello esterno. AWS CloudFormation Questo è

tratto dal modello mostrato nell'Esempio 1. Può essere utilizzato per creare un'infrastruttura esterna gestita dai clienti per un servizio AWS Proton Fargate creato dall'Esempio 4.

> **Important**
>
> In qualità di amministratore, è necessario assicurarsi che l'infrastruttura fornita e gestita e tutti i parametri di output siano compatibili con i modelli di ambiente gestito dal cliente associati. AWS Proton non puoi tenere conto delle modifiche per tuo conto perché queste modifiche non sono visibili a AWS Proton. Le incoerenze provocano errori.

Esempi di output di file CloudFormation IaC per un modello di ambiente gestito dal cliente:

```
// Cloudformation Template Outputs
[...]
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref 'ECSCluster'
  ECSTaskExecutionRole:
    Description: The ARN of the ECS role
    Value: !GetAtt 'ECSTaskExecutionRole.Arn'
  VpcId:
    Description: The ID of the VPC that this stack is deployed in
    Value: !Ref 'VPC'
[...]
```

Lo schema per il corrispondente pacchetto di modelli di ambiente gestito dal AWS Proton cliente è illustrato nell'esempio seguente. Ogni valore di output è definito come una stringa.

Schema di esempio per un modello di ambiente gestito dal cliente:

```
schema:                           # required
  format:                         # required
    openapi: "3.0.0"              # required
  # required              defined by administrator
  environment_input_type: "EnvironmentOutput"
  types:                          # required
    # defined by administrator
    EnvironmentOutput:
      type: object
```

```
      description: "Outputs of the environment"
      properties:
        ClusterName:               # parameter
          type: string
          description: "The name of the ECS cluster"
        ECSTaskExecutionRole:      # parameter
          type: string
          description: "The ARN of the ECS role"
        VpcId:                     # parameter
          type: string
          description: "The ID of the VPC that this stack is deployed in"
[...]
```

Requisiti dello schema per i pacchetti di modelli di servizio

Lo schema deve seguire la sezione Data Models (schemas) di OpenAPI in formato YAML, come mostrato negli esempi seguenti. È necessario fornire un file di schema nel pacchetto di modelli di servizio.

Nei seguenti esempi di schemi di servizio, è necessario includere le intestazioni formattate. Nell'esempio seguente, si tratta delle prime tre righe. Questo serve a stabilire che stai utilizzando la sezione Data Models (schemas) dell'Open API.

A `service_input_type` deve essere incluso e definito con un nome fornito dall'utente. Nell'esempio seguente, si trova nella riga 5. Ciò associa i parametri a una risorsa AWS Proton di servizio.

Una pipeline di AWS Proton servizi è inclusa per impostazione predefinita quando si utilizza la console o la CLI per creare un servizio. Quando includi una pipeline di servizi per il tuo servizio, devi includerla `pipeline_input_type` con un nome fornito. Nell'esempio seguente, si tratta della riga 7. Non includere questo parametro se non includi una pipeline AWS Proton di servizi. Per ulteriori informazioni, consulta Registra e pubblica modelli.

Per seguire il modello di schema Open API, è necessario includere `types` Nel seguente esempio, questo è nella riga 9.

Di seguito`types`, è necessario definire un `service_input_type` tipo. I parametri di input per il servizio vengono definiti come proprietà di`service_input_type`. È necessario includere almeno una proprietà con un nome che corrisponda ad almeno un parametro elencato nel file Service Infrastructure as Code (IaC) associato allo schema.

Per definire una pipeline di servizi, al di sotto della vostra `service_input_type` definizione, è necessario definire un. `pipeline_input_type` Come sopra, è necessario includere almeno una proprietà con un nome che corrisponda ad almeno un parametro elencato in un file IAc di pipeline associato allo schema. Non includete questa definizione se non includete una pipeline di AWS Proton servizi.

Quando, in qualità di amministratore o sviluppatore, create un servizio e fornite valori di parametri personalizzati, AWS Proton utilizzate il file di schema per abbinarli, convalidarli e inserirli nei parametri rinforzati ricci del file CloudFormation IAc associato. Per ogni proprietà (parametro), fornisci a e a. `name` `type` Facoltativamente, fornisci anche un `description``default`, e`pattern`.

I parametri definiti per lo schema di esempio includono `port` `task_size` e `image` con la `default` parola chiave e i valori predefiniti. `desired_count` Quando si crea un servizio con questo schema di bundle di modelli di servizio, è possibile accettare i valori predefiniti o fornire valori personalizzati. Anche il parametro `unique_name` è incluso nell'esempio e non ha un valore predefinito. È elencato come `required` proprietà (parametro). In qualità di amministratore o sviluppatore, è necessario fornire valori per `required` i parametri quando si creano servizi.

Se desideri creare un modello di servizio con una pipeline di servizi, includilo nello schema. `pipeline_input_type`

File di schema di servizio di esempio per un servizio che include una pipeline AWS Proton di servizi.

Questo esempio di schema può essere utilizzato con i file AWS Proton IAc mostrati nell'Esempio 4 e nell'Esempio 5. È inclusa una pipeline di servizi.

```
schema:                           # required
  format:                         # required
    openapi: "3.0.0"              # required
  # required           defined by administrator
  service_input_type: "LoadBalancedServiceInput"
  # only include if including AWS Proton service pipeline, defined by administrator
  pipeline_input_type: "PipelineInputs"

  types:                          # required
    # defined by administrator
    LoadBalancedServiceInput:
      type: object
      description: "Input properties for a loadbalanced Fargate service"
      properties:
        port:                     # parameter
```

```
        type: number
        description: "The port to route traffic to"
        default: 80
        minimum: 0
        maximum: 65535
      desired_count:            # parameter
        type: number
        description: "The default number of Fargate tasks you want running"
        default: 1
        minimum: 1
      task_size:                # parameter
        type: string
        description: "The size of the task you want to run"
        enum: ["x-small", "small", "medium", "large", "x-large"]
        default: "x-small"
      image:                    # parameter
        type: string
        description: "The name/url of the container image"
        default: "public.ecr.aws/z9d2n7e1/nginx:1.19.5"
        minLength: 1
        maxLength: 200
      unique_name:              # parameter
        type: string
        description: "The unique name of your service identifier. This will be used
to name your log group, task definition and ECS service"
        minLength: 1
        maxLength: 100
    required:
      - unique_name
  # defined by administrator
  PipelineInputs:
    type: object
    description: "Pipeline input properties"
    properties:
      dockerfile:               # parameter
        type: string
        description: "The location of the Dockerfile to build"
        default: "Dockerfile"
        minLength: 1
        maxLength: 100
      unit_test_command:        # parameter
        type: string
        description: "The command to run to unit test the application code"
        default: "echo 'add your unit test command here'"
```

```
          minLength: 1
          maxLength: 200
```

Se desiderate creare un modello di servizio senza una pipeline di servizi, non includetelo `pipeline_input_type` nello schema, come illustrato nell'esempio seguente.

File di schema di servizio di esempio per un servizio che non include una pipeline AWS Proton di servizi

```
schema:                           # required
  format:                         # required
    openapi: "3.0.0"              # required
  # required            defined by administrator
  service_input_type: "MyServiceInstanceInputType"

  types:                          # required
    # defined by administrator
    MyServiceInstanceInputType:
      type: object
      description: "Service instance input properties"
      required:
        - my_sample_service_instance_required_input
      properties:
        my_sample_service_instance_optional_input:   # parameter
          type: string
          description: "This is a sample input"
          default: "hello world"
        my_sample_service_instance_required_input:   # parameter
          type: string
          description: "Another sample input"
```

Raccogli i file modello per AWS Proton

Dopo aver preparato i file di ambiente e infrastruttura di servizio come codice (IaC) e i rispettivi file di schema, è necessario organizzarli in directory. È inoltre necessario creare un file YAML manifesto. Il file manifest elenca i file IaC in una directory, il motore di rendering e il linguaggio dei modelli usato per sviluppare IaC in questo modello.

> **Note**
>
> Un file manifest può essere utilizzato anche indipendentemente dai pacchetti di modelli, come input diretto per componenti definiti direttamente. In questo caso, specifica sempre un singolo file modello IAc, sia per Terraform che CloudFormation per Terraform. Per ulteriori informazioni sui componenti, vedere. Componenti

Il file manifesto deve rispettare il formato e il contenuto illustrati nell'esempio seguente.

CloudFormation formato del file manifesto:

Con CloudFormation, si elenca un singolo file.

```
infrastructure:
  templates:
    - file: "cloudformation.yaml"
      rendering_engine: jinja
      template_language: cloudformation
```

Formato di file manifest Terraform:

Con terraform, puoi elencare esplicitamente un singolo file o usare il wildcard * per elencare ciascuno dei file in una directory.

> **Note**
>
> Il carattere jolly include solo i file i cui nomi terminano con. `.tf` Gli altri file vengono ignorati.

```
infrastructure:
  templates:
    - file: "*"
      rendering_engine: hcl
      template_language: terraform
```

CodeBuildformato di file manifesto di provisioning basato:

Con il provisioning CodeBuild basato, si specificano i comandi shell di provisioning e deprovisioning.

> **Note**
>
> Oltre al manifesto, il pacchetto deve includere tutti i file da cui dipendono i comandi.

Il seguente manifesto di esempio utilizza il provisioning CodeBuild basato per fornire (distribuire) e deprovisionare (distruggere) risorse utilizzando (). AWS Cloud Development Kit (AWS CDK) AWS CDK Il pacchetto di modelli deve includere anche il codice CDK.

Durante il provisioning, AWS Proton crea un file di input con valori per i parametri di input definiti nello schema del modello con il nome. `proton-input.json`

```
infrastructure:
  templates:
    - rendering_engine: codebuild
      settings:
        image: aws/codebuild/amazonlinux2-x86_64-standard:4.0
        runtimes:
          nodejs: 16
        provision:
          - npm install
          - npm run build
          - npm run cdk bootstrap
          - npm run cdk deploy -- --require-approval never --outputs-file proton-
outputs.json
          - jq 'to_entries | map_values(.value) | add | to_entries | map({key:.key,
 valueString:.value})' < proton-outputs.json > outputs.json
          - aws proton notify-resource-deployment-status-change --resource-arn
 $RESOURCE_ARN --status IN_PROGRESS --outputs file://./outputs.json
        deprovision:
          - npm install
          - npm run build
          - npm run cdk destroy
        project_properties:
          VpcConfig:
            VpcId: "{{ environment.inputs.codebuild_vpc_id }}"
            Subnets: "{{ environment.inputs.codebuild_subnets }}"
            SecurityGroupIds: "{{ environment.inputs.codebuild_security_groups }}"
```

Dopo aver configurato le directory e i file manifest per il tuo ambiente o pacchetto di modelli di servizio, comprimi con gzip le directory in un tar ball e le carichi in un bucket Amazon Simple

Storage Service (Amazon S3) AWS Proton dove puoi recuperarle o in un repository Git per la sincronizzazione dei modelli.

Quando crei una versione secondaria di un ambiente o di un modello di servizio con cui ti sei registrato AWS Proton, fornisci il percorso dell'ambiente o del pacchetto di modelli di servizio tar ball che si trova nel tuo bucket S3. AWS Proton lo salva con la nuova versione secondaria del modello. È possibile selezionare la nuova versione secondaria del modello con cui creare o aggiornare ambienti o servizi AWS Proton.

Riepilogo del pacchetto di modelli di ambiente

Esistono due tipi di pacchetti di modelli di ambiente per i quali è possibile creare. AWS Proton

- Per creare un pacchetto di modelli di ambiente per un modello di ambiente standard, organizzate lo schema, i file di infrastruttura come codice (IaC) e il file manifest nelle directory, come illustrato nella seguente struttura di directory dei pacchetti di modelli di ambiente.
- Per creare un pacchetto di modelli di ambiente per un modello di ambiente gestito dal cliente, fornite solo il file di schema e la directory. Non includere la directory e i file dell'infrastruttura. AWS Proton genera un errore se la directory e i file dell'infrastruttura sono inclusi.

Per ulteriori informazioni, consulta Registra e pubblica modelli.

CloudFormation struttura delle cartelle del pacchetto di template di ambiente:

```
/schema
  schema.yaml
/infrastructure
  manifest.yaml
  cloudformation.yaml
```

Struttura della directory del pacchetto dei modelli di ambiente Terraform:

```
/schema
  schema.yaml
/infrastructure
  manifest.yaml
  environment.tf
```

Riepilogo del pacchetto di modelli di servizio

Per creare un pacchetto di modelli di servizio, è necessario organizzare lo schema, i file IaC (Infrastructure as Code) e i file manifest in directory, come mostrato nell'esempio di struttura di directory del Service Template Bundle.

Se non includete una pipeline di servizi nel vostro pacchetto di modelli, non includete la directory della pipeline e i file e impostateli `"pipelineProvisioning": "CUSTOMER_MANAGED"` quando create il modello di servizio da associare a questo pacchetto di modelli.

> **Note**
>
> Non è possibile modificare `pipelineProvisioning` dopo la creazione del modello di servizio.

Per ulteriori informazioni, consulta Registra e pubblica modelli.

CloudFormation struttura delle cartelle del pacchetto di modelli di servizio:

```
/schema
  schema.yaml
/instance_infrastructure
  manifest.yaml
  cloudformation.yaml
/pipeline_infrastructure
  manifest.yaml
  cloudformation.yaml
```

Struttura della directory del pacchetto dei modelli di servizio Terraform:

```
/schema
  schema.yaml
/instance_infrastructure
  manifest.yaml
  instance.tf
/pipeline_infrastructure
  manifest.yaml
  pipeline.tf
```

Considerazioni sul pacchetto di modelli

- File Infrastructure as code (IaC)

  AWS Proton verifica i modelli per verificare il formato di file corretto. Tuttavia, AWS Proton non verifica la presenza di errori logici, di dipendenza e di sviluppo dei modelli. Ad esempio, supponiamo di aver specificato la creazione di un bucket Amazon S3 nel file AWS CloudFormation IAc come parte del modello di servizio o ambiente. Viene creato un servizio basato su tali modelli. Supponiamo ora che a un certo punto si desideri eliminare il servizio. Se il bucket S3 specificato non è vuoto e il file CloudFormation IAc non lo contrassegna come `Retain` nel`DeletionPolicy`, l'operazione di eliminazione del AWS Proton servizio fallisce.
- Raggruppa i limiti e il formato delle dimensioni dei file
  - I limiti relativi alla dimensione, al numero e alla dimensione dei nomi dei file del pacchetto sono disponibili all'indirizzo. AWS Proton quote
  - Le directory di file del pacchetto modello vengono compresse con gzip in una sfera tar e collocate in un bucket Amazon Simple Storage Service (Amazon S3).
  - Ogni file del pacchetto deve essere un file YAML in formato valido.
- Crittografia del pacchetto di modelli di bucket S3

  Se desideri crittografare i dati sensibili nei tuoi pacchetti di modelli e archiviati nel bucket S3, usa le chiavi SSE-S3 o SSE-KMS per consentirne il recupero. AWS Proton

AWS Proton modelli

Per aggiungere il tuo pacchetto di modelli alla tua libreria di AWS Proton modelli, crea una versione secondaria di modelli e registrala con AWS Proton. Quando crei il modello, fornisci il nome del bucket Amazon S3 e il percorso per il tuo pacchetto di modelli. Dopo la pubblicazione, i modelli possono essere selezionati dai membri del team della piattaforma e dagli sviluppatori. Dopo averli selezionati, AWS Proton utilizza il modello per creare e fornire infrastrutture e applicazioni.

In qualità di amministratore, puoi creare e registrare un modello di ambiente con AWS Proton. Questo modello di ambiente può quindi essere utilizzato per distribuire più ambienti. Ad esempio, può essere utilizzato per distribuire ambienti «dev», «staging» e «prod». L'ambiente «dev» potrebbe includere un VPC con sottoreti private e una politica di accesso restrittiva a tutte le risorse. Gli output ambientali possono essere utilizzati come input per i servizi.

È possibile creare e registrare modelli di ambiente per creare due diversi tipi di ambienti. Sia tu che gli sviluppatori potete utilizzare AWS Proton per distribuire servizi a entrambi i tipi.

- Registra e pubblica un modello di ambiente standard da AWS Proton utilizzare per creare un ambiente standard per il provisioning e la gestione dell'infrastruttura ambientale.
- Registra e pubblica un modello di ambiente gestito dal cliente da AWS Proton utilizzare per creare un ambiente gestito dal cliente che si connette all'infrastruttura predisposta esistente. AWS Proton non gestisce l'infrastruttura predisposta esistente.

È possibile creare e registrare modelli di servizio con AWS Proton cui distribuire servizi negli ambienti. È necessario creare un AWS Proton ambiente prima di potervi distribuire un servizio.

L'elenco seguente descrive come creare e gestire i modelli con AWS Proton.

- (Facoltativo) Prepara un ruolo IAM per controllare l'accesso degli sviluppatori alle chiamate AWS Proton API e ai ruoli dei servizi AWS Proton IAM. Per ulteriori informazioni, consulta the section called “Ruoli IAM”.
- Componi un pacchetto di modelli. Per ulteriori informazioni, consulta Pacchetti di modelli.
- Crea e registra un modello AWS Proton dopo che il pacchetto di modelli è stato composto, compresso e salvato in un bucket Amazon S3. Puoi farlo nella console o utilizzando il. AWS CLI
- Testa e usa il modello per creare e gestire AWS Proton le risorse assegnate dopo la registrazione. AWS Proton

- Crea e gestisci le versioni principali e secondarie del modello per tutta la durata del modello.

Puoi gestire le versioni del modello manualmente o con configurazioni di sincronizzazione dei modelli:

- Usa la AWS Proton console e AWS CLI crea una nuova versione secondaria o principale.
- Crea una configurazione di sincronizzazione dei modelli che consenta di creare AWS Proton automaticamente una nuova versione secondaria o principale quando rileva una modifica al tuo pacchetto di modelli in un repository da te definito.

Per ulteriori informazioni, consulta The AWS Proton Service API Reference.

Argomenti

- Modelli con versione
- Registra e pubblica modelli
- Visualizza i dati del modello
- Aggiornare un modello
- Eliminare i modelli
- Configurazioni di sincronizzazione dei modelli
- Configurazioni di sincronizzazione dei servizi

Modelli con versione

In qualità di amministratore o membro di un team della piattaforma, definisci, crei e gestisci una libreria di modelli con versioni che vengono utilizzati per fornire le risorse dell'infrastruttura. Esistono due tipi di versioni dei modelli: versioni secondarie e versioni principali.

- Versioni secondarie: modifiche al modello con uno schema compatibile con le versioni precedenti. Queste modifiche non richiedono allo sviluppatore di fornire nuove informazioni durante l'aggiornamento alla nuova versione del modello.

  Quando tenti di apportare una modifica secondaria alla versione, AWS Proton fa del suo meglio per determinare se lo schema della nuova versione è retrocompatibile con le versioni secondarie precedenti del modello. Se il nuovo schema non è compatibile con le versioni precedenti, la registrazione della nuova versione secondaria AWS Proton non riesce.

> **Note**
>
> La compatibilità è determinata esclusivamente in base allo schema. AWS Proton non verifica se il file Template Bundle Infrastructure as Code (IAc) è retrocompatibile con le versioni minori precedenti. Ad esempio, AWS Proton non verifica se il nuovo file IAc causa modifiche sostanziali alle applicazioni in esecuzione sull'infrastruttura fornita da una precedente versione secondaria del modello.

- Versioni principali: modifiche al modello che potrebbero non essere compatibili con le versioni precedenti. Queste modifiche richiedono in genere nuovi input da parte dello sviluppatore e spesso comportano modifiche allo schema del modello.

  A volte puoi scegliere di designare una modifica compatibile con le versioni precedenti come versione principale in base al modello operativo del tuo team.

Il modo in cui si AWS Proton determina se una richiesta di versione del modello riguarda una versione secondaria o principale dipende dal modo in cui vengono tracciate le modifiche al modello:

- Quando si effettua esplicitamente una richiesta per creare una nuova versione del modello, si richiede una versione principale specificando un numero di versione principale e si richiede una versione secondaria non specificando un numero di versione principale.
- Quando utilizzi la sincronizzazione dei modelli (e quindi non effettui richieste esplicite di versione del modello), AWS Proton tenta di creare nuove versioni secondarie per le modifiche al modello che si verificano nel file YAML esistente. AWS Proton crea una versione principale quando crei una nuova directory per la nuova modifica del modello (ad esempio, passa dalla v1 alla v2).

> **Note**
>
> La registrazione di una nuova versione secondaria basata sulla sincronizzazione dei modelli continua a fallire se AWS Proton determina che la modifica non è compatibile con le versioni precedenti.

Quando pubblichi una nuova versione di un modello, questa diventa la versione consigliata se è la versione principale e secondaria più recente. Le nuove AWS Proton risorse vengono create

utilizzando la nuova versione consigliata e AWS Proton richiede agli amministratori di utilizzare la nuova versione e aggiornare AWS Proton le risorse esistenti che utilizzano una versione obsoleta.

Registra e pubblica modelli

È possibile registrare e pubblicare modelli di ambiente e servizio con AWS Proton, come descritto nelle sezioni seguenti.

È possibile creare una nuova versione di un modello con la console o AWS CLI.

In alternativa, puoi utilizzare la console o AWS CLI creare un modello e configurare una sincronizzazione dei modelli per esso. Questa configurazione consente la AWS Proton sincronizzazione dai pacchetti di modelli che si trovano nei repository git registrati che hai definito. Ogni volta che viene inviato al tuo repository un commit che modifica uno dei tuoi pacchetti di modelli, viene creata una nuova versione minore o principale del modello, se la versione non esiste già. Per ulteriori informazioni sui prerequisiti e sui requisiti di configurazione della sincronizzazione dei modelli, consulta. Configurazioni di sincronizzazione dei modelli

Registrare e pubblicare modelli di ambiente

È possibile registrare e pubblicare i seguenti tipi di modelli di ambiente.

- Registra e pubblica un modello di ambiente standard da AWS Proton utilizzare per distribuire e gestire l'infrastruttura ambientale.
- Registra e pubblica un modello di ambiente gestito dal cliente da AWS Proton utilizzare per connettersi all'infrastruttura predisposta esistente che gestisci. AWS Proton non gestisce l'infrastruttura predisposta esistente.

> **Important**
>
> In qualità di amministratore, assicurati che l'infrastruttura fornita e gestita e tutti i parametri di output siano compatibili con i modelli di ambiente gestiti dal cliente associati. AWS Proton non posso tenere conto delle modifiche per tuo conto perché queste modifiche non sono visibili a AWS Proton. Le incoerenze provocano errori.

È possibile utilizzare la console o il AWS CLI per registrare e pubblicare un modello di ambiente.

AWS Management Console

Usa la console per registrare e pubblicare un nuovo modello di ambiente.

1. Nella AWS Proton console, scegli Modelli di ambiente.
2. Scegli Crea modello di ambiente.
3. Nella pagina Crea modello di ambiente, nella sezione Opzioni modello, scegli una delle due opzioni di modello disponibili.
   - Crea un modello per il provisioning di nuovi ambienti.
   - Crea un modello per utilizzare l'infrastruttura predisposta che gestisci.
4. Se hai scelto Crea un modello per il provisioning di nuovi ambienti, nella sezione Sorgente del pacchetto di modelli, scegli una delle tre opzioni di origine del pacchetto di modelli disponibili. Per ulteriori informazioni sui requisiti e i prerequisiti per la sincronizzazione dei modelli, consulta. Configurazioni di sincronizzazione dei modelli
   - Utilizza uno dei nostri pacchetti di modelli di esempio.
   - Usa il tuo pacchetto di modelli.
   - Sincronizza i modelli da Git.
5. Fornisci un percorso per un pacchetto di modelli.
   a. Se hai scelto Usa uno dei nostri pacchetti di modelli di esempio:

      Nella sezione Pacchetto di modelli di esempio, seleziona un pacchetto di modelli di esempio.
   b. Se hai scelto Sincronizza modelli da Git, nella sezione Codice sorgente:

      i. Seleziona il repository per la configurazione di sincronizzazione dei modelli.
      ii. Inserisci il nome del ramo del repository da cui eseguire la sincronizzazione.
      iii. (Facoltativo) Immettete il nome di una directory per limitare la ricerca del vostro pacchetto di modelli.
   c. Altrimenti, nella sezione sulla posizione del pacchetto S3, fornisci un percorso al tuo pacchetto di modelli.
6. Nella sezione Dettagli del modello.
   a. Inserisci un Nome del modello.
   b. (Facoltativo) Inserite un nome da visualizzare per il modello.

c. (Facoltativo) Inserite una descrizione del modello per il modello di ambiente.

7. (Facoltativo) Seleziona la casella di controllo Personalizza le impostazioni di crittografia (avanzate) nella sezione Impostazioni di crittografia per fornire la tua chiave di crittografia.
8. (Facoltativo) Nella sezione Tag, scegli Aggiungi nuovo tag e inserisci una chiave e un valore per creare un tag gestito dal cliente.
9. Scegli il modello Create Environment.

   Ora ti trovi su una nuova pagina che mostra lo stato e i dettagli del tuo nuovo modello di ambiente. Questi dettagli includono un elenco di AWS tag gestiti dai clienti. AWS Proton genera automaticamente tag AWS gestiti per te quando crei AWS Proton risorse. Per ulteriori informazioni, consulta AWS Proton risorse e etichettatura.
10. Lo stato di un nuovo modello di ambiente inizia nello stato Bozza. Tu e altre persone con `proton:CreateEnvironment` autorizzazioni potete visualizzarlo e accedervi. Segui il passaggio successivo per rendere il modello disponibile agli altri.
11. Nella sezione Versioni del modello, scegli il pulsante di opzione a sinistra della versione secondaria del modello che hai appena creato (1.0). In alternativa, puoi scegliere Pubblica nell'avviso informativo e saltare il passaggio successivo.
12. Nella sezione Versioni dei modelli, scegli Pubblica.
13. Lo stato del modello cambia in Pubblicato. Poiché è la versione più recente del modello, è la versione consigliata.
14. Nel riquadro di navigazione, seleziona Modelli di ambiente per visualizzare un elenco dei modelli e dei dettagli dell'ambiente.

Usa la console per registrare nuove versioni principali e secondarie di un modello di ambiente.

Per ulteriori informazioni, consulta Modelli con versione.

1. Nella AWS Proton console, scegli Modelli di ambiente.
2. Nell'elenco dei modelli di ambiente, scegli il nome del modello di ambiente per cui desideri creare una versione principale o secondaria.
3. Nella visualizzazione dettagliata del modello di ambiente, scegli Crea nuova versione nella sezione Versioni del modello.
4. Nella pagina Crea una nuova versione del modello di ambiente, nella sezione Sorgente del pacchetto di modelli, scegli una delle due opzioni di origine del pacchetto di modelli disponibili.

- Utilizza uno dei nostri pacchetti di modelli di esempio.
- Usa il tuo pacchetto di modelli.

5. Fornisci un percorso per il pacchetto di modelli selezionato.
   - Se hai scelto Usa uno dei nostri pacchetti di modelli di esempio, nella sezione Pacchetto di modelli di esempio, seleziona un pacchetto di modelli di esempio.
   - Se hai scelto Usa il tuo pacchetto di modelli, nella sezione Posizione del pacchetto S3, scegli il percorso del tuo pacchetto di modelli.
6. Nella sezione Dettagli del modello.
   a. (Facoltativo) Inserite un nome da visualizzare per il modello.
   b. (Facoltativo) Inserire una descrizione del modello per il modello di servizio.
7. Nella sezione Dettagli del modello, scegli una delle seguenti opzioni.
   - Per creare una versione secondaria, mantieni vuota la casella di controllo Seleziona per creare una nuova versione principale.
   - Per creare una versione principale, seleziona la casella di controllo Seleziona per creare una nuova versione principale.
8. Continua con i passaggi della console per creare la nuova versione secondaria o principale e scegli Crea nuova versione.

AWS CLI

Utilizzate la CLI per registrare e pubblicare un nuovo modello di ambiente, come illustrato nei passaggi seguenti.

1. Crea un modello di ambiente OR gestito dal cliente standard specificando l'area, il nome, il nome visualizzato (opzionale) e la descrizione (opzionale).
   a. Crea un modello di ambiente standard.

      Esegui il comando seguente:

```
$ aws proton create-environment-template \
    --name "simple-env" \
    --display-name "Fargate" \
```

```
    --description "VPC with public access"
```

Risposta:

```
{
    "environmentTemplate": {
        "arn": "arn:aws:proton:region-id:123456789012:environment-template/
simple-env",
        "createdAt": "2020-11-11T23:02:45.336000+00:00",
        "description": "VPC with public access",
        "displayName": "VPC",
        "lastModifiedAt": "2020-11-11T23:02:45.336000+00:00",
        "name": "simple-env"
    }
}
```

b. Crea un modello di ambiente gestito dal cliente aggiungendo il `provisioning` parametro con valore`CUSTOMER_MANAGED`.

Esegui il comando seguente:

```
$ aws proton create-environment-template \
    --name "simple-env" \
    --display-name "Fargate" \
    --description "VPC with public access" \
    --provisioning "CUSTOMER_MANAGED"
```

Risposta:

```
{
    "environmentTemplate": {
        "arn": "arn:aws:proton:region-id:123456789012:environment-template/
simple-env",
        "createdAt": "2020-11-11T23:02:45.336000+00:00",
        "description": "VPC with public access",
        "displayName": "VPC",
        "lastModifiedAt": "2020-11-11T23:02:45.336000+00:00",
        "name": "simple-env",
        "provisioning": "CUSTOMER_MANAGED"
    }
}
```

2. Crea una versione secondaria 0 della versione principale 1 del modello di ambiente

   Questo e i passaggi rimanenti sono gli stessi sia per i modelli di ambiente standard che per quelli gestiti dal cliente.

   Includi il nome del modello, la versione principale, il nome e la chiave del bucket S3 per il bucket che contiene il pacchetto di modelli di ambiente.

   Esegui il comando seguente:

```
$ aws proton create-environment-template-version \
    --template-name "simple-env" \
    --description "Version 1" \
    --source s3="{bucket=your_s3_bucket, key=your_s3_key}"
```

   Risposta:

```
{
    "environmentTemplateVersion": {
        "arn": "arn:aws:proton:region-id:123456789012:environment-template/
simple-env:1.0",
        "createdAt": "2020-11-11T23:02:47.763000+00:00",
        "description": "Version 1",
        "lastModifiedAt": "2020-11-11T23:02:47.763000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "status": "REGISTRATION_IN_PROGRESS",
        "templateName": "simple-env"
    }
}
```

3. Usa il comando get per controllare lo stato delle registrazioni.

   Esegui il comando seguente:

```
$ aws proton get-environment-template-version \
    --template-name "simple-env" \
    --major-version "1" \
    --minor-version "0"
```

   Risposta:

```
{
    "environment": {
        "arn": "arn:aws:proton:region-id:123456789012:environment-template/
simple-env:1.0",
        "createdAt": "2020-11-11T23:02:47.763000+00:00",
        "description": "Version 1",
        "lastModifiedAt": "2020-11-11T23:02:47.763000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "recommendedMinorVersion": "0",
        "schema": "schema:\n  format:\n    openapi: \"3.0.0\"\n
 environment_input_type: \"MyEnvironmentInputType\"\n  types:\n
 MyEnvironmentInputType:\n      type: object\n      description: \"Input
 properties for my environment\"\n      properties:\n        my_sample_input:\n
          type: string\n          description: \"This is a sample input\"\n
      default: \"hello world\"\n        my_other_sample_input:\n          type:
 string\n          description: \"Another sample input\"\n      required:\n
   - my_other_sample_input\n",
        "status": "DRAFT",
        "statusMessage": "",
        "templateName": "simple-env"
    }
}
```

4. Pubblica la versione secondaria 0 della versione principale 1 del modello di ambiente fornendo il nome del modello e la versione principale e secondaria. Questa versione è la `Recommended` versione.

   Esegui il comando seguente:

   ```
   $ aws proton update-environment-template-version \
       --template-name "simple-env" \
       --major-version "1" \
       --minor-version "0" \
       --status "PUBLISHED"
   ```

   Risposta:

   ```
   {
       "environmentTemplateVersion": {
   ```

```
        "arn": "arn:aws:proton:region-id:123456789012:environment-template/
simple-env:1.0",
        "createdAt": "2020-11-11T23:02:47.763000+00:00",
        "description": "Version 1",
        "lastModifiedAt": "2020-11-11T23:02:54.610000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "recommendedMinorVersion": "0",
        "schema": "schema:\n  format:\n    openapi: \"3.0.0\"\n
 environment_input_type: \"MyEnvironmentInputType\"\n  types:\n
 MyEnvironmentInputType:\n      type: object\n      description: \"Input
 properties for my environment\"\n      properties:\n        my_sample_input:\n
          type: string\n          description: \"This is a sample input\"\n
      default: \"hello world\"\n        my_other_sample_input:\n          type:
 string\n          description: \"Another sample input\"\n      required:\n
   - my_other_sample_input\n",
        "status": "PUBLISHED",
        "statusMessage": "",
        "templateName": "simple-env"
    }
}
```


Dopo aver creato un nuovo modello utilizzando AWS CLI, puoi visualizzare un elenco di AWS tag gestiti dai clienti. AWS Proton genera automaticamente tag AWS gestiti per te. Puoi anche modificare e creare tag gestiti dai clienti utilizzando AWS CLI. Per ulteriori informazioni, consulta AWS Proton risorse e etichettatura.

Esegui il comando seguente:

```
$ aws proton list-tags-for-resource \
    --resource-arn "arn:aws:proton:region-id:123456789012:environment-
template/simple-env"
```

Registrare e pubblicare modelli di servizio

Quando si crea una versione del modello di servizio, si specifica un elenco di modelli di ambiente compatibili. In questo modo, quando gli sviluppatori selezionano un modello di servizio, possono scegliere in quale ambiente distribuire il servizio.

Prima di creare un servizio da un modello di servizio o prima di pubblicare un modello di servizio, verificate che gli ambienti siano distribuiti dai modelli di ambiente compatibili elencati.

Non è possibile aggiornare un servizio alla nuova versione principale se viene distribuito in un ambiente creato da un modello di ambiente compatibile rimosso.

Per aggiungere o rimuovere modelli di ambiente compatibili per una versione del modello di servizio, è necessario crearne una nuova versione principale.

È possibile utilizzare la console o il AWS CLI per registrare e pubblicare un modello di servizio.

AWS Management Console

Utilizza la console per registrare e pubblicare un nuovo modello di servizio.

1. Nella AWS Proton console, scegli Modelli di servizio.
2. Scegli Crea modello di servizio.
3. Nella pagina Crea modello di servizio, nella sezione Origine del pacchetto di modelli, scegli una delle opzioni di modello disponibili.
   - Usa il tuo pacchetto di modelli.
   - Sincronizza i modelli da Git.
4. Fornisci un percorso per un pacchetto di modelli.
   a. Se hai scelto Sincronizza modelli da Git, nella sezione Archivio del codice sorgente:
      i. Seleziona l'archivio per la configurazione di sincronizzazione dei modelli.
      ii. Inserisci il nome del ramo del repository da cui eseguire la sincronizzazione.
      iii. (Facoltativo) Immettete il nome di una directory per limitare la ricerca del vostro pacchetto di modelli.
   b. Altrimenti, nella sezione sulla posizione del pacchetto S3, fornisci un percorso al tuo pacchetto di modelli.
5. Nella sezione Dettagli del modello.
   a. Inserisci un Nome del modello.
   b. (Facoltativo) Inserite un nome da visualizzare per il modello.
   c. (Facoltativo) Inserire una descrizione del modello per il modello di servizio.

6. Nella sezione Modelli di ambiente compatibili, scegli da un elenco di modelli di ambiente compatibili.

7. (Facoltativo) Nella sezione Impostazioni di crittografia, scegli Personalizza le impostazioni di crittografia (avanzate) per fornire la tua chiave di crittografia.

8. (Facoltativo) Nella sezione Pipeline:

   Se non includi una definizione di pipeline di servizio nel modello di servizio, deseleziona la casella di controllo Pipeline - opzionale nella parte inferiore della pagina. Non è possibile modificare questa impostazione dopo la creazione del modello di servizio. Per ulteriori informazioni, consulta Pacchetti di modelli.

9. (Facoltativo) Nella sezione Origini dei componenti supportate, per Sorgenti dei componenti, scegliete Direttamente definito per abilitare il collegamento di componenti definiti direttamente alle istanze del servizio.

10. (Facoltativo) Nella sezione Tag, scegli Aggiungi nuovo tag e inserisci una chiave e un valore per creare un tag gestito dal cliente.

11. Scegli Crea un modello di servizio.

    Ora ti trovi su una nuova pagina che mostra lo stato e i dettagli del tuo nuovo modello di servizio. Questi dettagli includono un elenco di AWS tag gestiti dai clienti. AWS Proton genera automaticamente tag AWS gestiti per te quando crei AWS Proton risorse. Per ulteriori informazioni, consulta AWS Proton risorse e etichettatura.

12. Lo stato di un nuovo modello di servizio inizia nello stato Bozza. Tu e altre persone con `proton:CreateService` autorizzazioni potete visualizzarlo e accedervi. Segui il passaggio successivo per rendere il modello disponibile agli altri.

13. Nella sezione Versioni del modello, scegli il pulsante di opzione a sinistra della versione secondaria del modello che hai appena creato (1.0). In alternativa, puoi scegliere Pubblica nell'avviso informativo e saltare il passaggio successivo.

14. Nella sezione Versioni dei modelli, scegli Pubblica.

15. Lo stato del modello cambia in Pubblicato. Poiché è la versione più recente del modello, è la versione consigliata.

16. Nel riquadro di navigazione, seleziona Modelli di servizio per visualizzare un elenco dei modelli e dei dettagli del servizio.

Utilizza la console per registrare nuove versioni principali e secondarie di un modello di servizio.

Per ulteriori informazioni, consulta Modelli con versione.

1. Nella AWS Proton console, scegli Service Templates.
2. Nell'elenco dei modelli di servizio, scegli il nome del modello di servizio per cui desideri creare una versione principale o secondaria.
3. Nella visualizzazione dettagliata del modello di servizio, scegli Crea nuova versione nella sezione Versioni del modello.
4. Nella pagina Crea una nuova versione del modello di servizio, nella sezione Sorgente del pacchetto, seleziona Usa il tuo pacchetto di modelli.
5. Nella sezione Posizione del pacchetto S3, scegli il percorso del tuo pacchetto modello.
6. Nella sezione Dettagli del modello.
   a. (Facoltativo) Inserite un nome da visualizzare per il modello.
   b. (Facoltativo) Inserire una descrizione del modello per il modello di servizio.
7. Nella sezione Dettagli del modello, scegli una delle seguenti opzioni.
   - Per creare una versione secondaria, mantieni vuota la casella di controllo Seleziona per creare una nuova versione principale.
   - Per creare una versione principale, seleziona la casella di controllo Seleziona per creare una nuova versione principale.
8. Continua con i passaggi della console per creare la nuova versione secondaria o principale e scegli Crea nuova versione.

AWS CLI

Per creare un modello di servizio che distribuisca un servizio senza una pipeline di servizi, aggiungi il parametro e il `create-service-template` valore `--pipeline-provisioning "CUSTOMER_MANAGED"` al comando. Configura i tuoi pacchetti di modelli come descritto nella Pacchetti di modelli creazione e. Requisiti dello schema per i pacchetti di modelli di servizio

> **Note**
>
> Non è possibile apportare modifiche `pipelineProvisioning` dopo la creazione del modello di servizio.

1. Utilizza la CLI per registrare e pubblicare un nuovo modello di servizio, con o senza una pipeline di servizi, come illustrato nei passaggi seguenti.

   a. Crea un modello di servizio con una pipeline di servizi utilizzando la CLI.

      Specificate il nome, il nome visualizzato (opzionale) e la descrizione (opzionale).

      Esegui il comando seguente:

      ```
      $ aws proton create-service-template \
          --name "fargate-service" \
          --display-name "Fargate" \
          --description "Fargate-based Service"
      ```

      Risposta:

      ```
      {
          "serviceTemplate": {
              "arn": "arn:aws:proton:region-id:123456789012:service-template/
      fargate-service",
              "createdAt": "2020-11-11T23:02:55.551000+00:00",
              "description": "Fargate-based Service",
              "displayName": "Fargate",
              "lastModifiedAt": "2020-11-11T23:02:55.551000+00:00",
              "name": "fargate-service"
          }
      }
      ```

   b. Crea un modello di servizio senza una pipeline di servizi.

      Add `--pipeline-provisioning`.

      Eseguite il comando seguente:

      ```
      $ aws proton create-service-template \
          --name "fargate-service" \
          --display-name "Fargate" \
          --description "Fargate-based Service" \
          --pipeline-provisioning "CUSTOMER_MANAGED"
      ```

      Risposta:

```
{
    "serviceTemplate": {
        "arn": "arn:aws:proton:region-id:123456789012:service-template/
fargate-service",
        "createdAt": "2020-11-11T23:02:55.551000+00:00",
        "description": "Fargate-based Service",
        "displayName": "Fargate",
        "lastModifiedAt": "2020-11-11T23:02:55.551000+00:00",
        "name": "fargate-service",
        "pipelineProvisioning": "CUSTOMER_MANAGED"
    }
}
```

2. Crea una versione secondaria 0 della versione principale 1 del modello di servizio.

   Includi il nome del modello, i modelli di ambiente compatibili, la versione principale e il nome e la chiave del bucket S3 per il bucket che contiene il pacchetto di modelli di servizio.

   Esegui il comando seguente:

```
$ aws proton create-service-template-version \
    --template-name "fargate-service" \
    --description "Version 1" \
    --source s3="{bucket=your_s3_bucket, key=your_s3_key}" \
    --compatible-environment-templates '[{"templateName":"simple-
env","majorVersion":"1"}]'
```

   Risposta:

```
{
    "serviceTemplateMinorVersion": {
        "arn": "arn:aws:proton:region-id:123456789012:service-template/fargate-
service:1.0",
        "compatibleEnvironmentTemplates": [
            {
                "majorVersion": "1",
                "templateName": "simple-env"
            }
        ],
        "createdAt": "2020-11-11T23:02:57.912000+00:00",
        "description": "Version 1",
```

```
        "lastModifiedAt": "2020-11-11T23:02:57.912000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "status": "REGISTRATION_IN_PROGRESS",
        "templateName": "fargate-service"
    }
}
```

3. Usa il comando get per controllare lo stato delle registrazioni.

   Esegui il seguente comando:

```
$ aws proton get-service-template-version \
    --template-name "fargate-service" \
    --major-version "1" \
    --minor-version "0"
```

   Risposta:

```
{
    "serviceTemplateMinorVersion": {
        "arn": "arn:aws:proton:us-east-1:123456789012:service-template/fargate-
service:1.0",
        "compatibleEnvironmentTemplates": [
            {
                "majorVersion": "1",
                "templateName": "simple-env"
            }
        ],
        "createdAt": "2020-11-11T23:02:57.912000+00:00",
        "description": "Version 1",
        "lastModifiedAt": "2020-11-11T23:02:57.912000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "schema": "schema:\n  format:\n    openapi: \"3.0.0\"\n
 pipeline_input_type: \"MyPipelineInputType\"\n  service_input_type:
 \"MyServiceInstanceInputType\"\n\n  types:\n    MyPipelineInputType:\n
    type: object\n      description: \"Pipeline input properties\"\n
 required:\n        - my_sample_pipeline_required_input\n      properties:\n
        my_sample_pipeline_optional_input:\n          type: string\n
   description: \"This is a sample input\"\n          default: \"hello world
\"\n        my_sample_pipeline_required_input:\n          type: string\n
```

```
     description: \"Another sample input\"\n\n    MyServiceInstanceInputType:
\n      type: object\n      description: \"Service instance input properties
\"\n      required:\n        - my_sample_service_instance_required_input\n
    properties:\n        my_sample_service_instance_optional_input:\n
   type: string\n          description: \"This is a sample input\"\n
 default: \"hello world\"\n        my_sample_service_instance_required_input:\n
         type: string\n          description: \"Another sample input\"",
        "status": "DRAFT",
        "statusMessage": "",
        "templateName": "fargate-service"
    }
}
```

4. Pubblica il modello di servizio utilizzando il comando update per modificare lo stato in. `"PUBLISHED"`

   Esegui il comando seguente:

```
$ aws proton update-service-template-version \
    --template-name "fargate-service" \
    --description "Version 1" \
    --major-version "1" \
    --minor-version "0" \
    --status "PUBLISHED"
```

   Risposta:

```
{
    "serviceTemplateVersion": {
        "arn": "arn:aws:proton:region-id:123456789012:service-template/fargate-
service:1.0",
        "compatibleEnvironmentTemplates": [
            {
                "majorVersion": "1",
                "templateName": "simple-env"
            }
        ],
        "createdAt": "2020-11-11T23:02:57.912000+00:00",
        "description": "Version 1",
        "lastModifiedAt": "2020-11-11T23:02:57.912000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "recommendedMinorVersion": "0",
```

```
        "schema": "schema:\n  format:\n    openapi: \"3.0.0\"\n
 pipeline_input_type: \"MyPipelineInputType\"\n  service_input_type:
 \"MyServiceInstanceInputType\"\n\n  types:\n    MyPipelineInputType:\n
    type: object\n      description: \"Pipeline input properties\"\n
 required:\n        - my_sample_pipeline_required_input\n      properties:\n
        my_sample_pipeline_optional_input:\n          type: string\n
 description: \"This is a sample input\"\n          default: \"hello pipeline
\"\n        my_sample_pipeline_required_input:\n          type: string\n
     description: \"Another sample input\"\n\n    MyServiceInstanceInputType:
\n      type: object\n      description: \"Service instance input properties
\"\n      required:\n        - my_sample_service_instance_required_input\n
    properties:\n        my_sample_service_instance_optional_input:\n
   type: string\n          description: \"This is a sample input\"\n
 default: \"hello world\"\n        my_sample_service_instance_required_input:\n
         type: string\n          description: \"Another sample input\"\n",
        "status": "PUBLISHED",
        "statusMessage": "",
        "templateName": "fargate-service"
    }
}
```

5. Verifica che sia AWS Proton stata pubblicata la versione 1.0 utilizzando il comando get per recuperare i dati di dettaglio del modello di servizio.

   Esegui il comando seguente:

```
$ aws proton get-service-template-version \
    --template-name "fargate-service" \
    --major-version "1" \
    --minor-version "0"
```

   Risposta:

```
{
    "serviceTemplateMinorVersion": {
        "arn": "arn:aws:proton:us-east-1:123456789012:service-template/fargate-
service:1.0",
        "compatibleEnvironmentTemplates": [
            {
                "majorVersion": "1",
                "templateName": "simple-env"
            }
        ],
```

```
        "createdAt": "2020-11-11T23:02:57.912000+00:00",
        "description": "Version 1",
        "lastModifiedAt": "2020-11-11T23:03:04.767000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "schema": "schema:\n  format:\n    openapi: \"3.0.0\"\n
 pipeline_input_type: \"MyPipelineInputType\"\n  service_input_type:
 \"MyServiceInstanceInputType\"\n\n  types:\n    MyPipelineInputType:\n
    type: object\n      description: \"Pipeline input properties\"\n
 required:\n        - my_sample_pipeline_required_input\n      properties:\n
        my_sample_pipeline_optional_input:\n          type: string\n
   description: \"This is a sample input\"\n          default: \"hello world
\"\n        my_sample_pipeline_required_input:\n          type: string\n
     description: \"Another sample input\"\n\n    MyServiceInstanceInputType:
\n      type: object\n      description: \"Service instance input properties
\"\n      required:\n        - my_sample_service_instance_required_input\n
    properties:\n        my_sample_service_instance_optional_input:\n
   type: string\n          description: \"This is a sample input\"\n
 default: \"hello world\"\n        my_sample_service_instance_required_input:\n
         type: string\n          description: \"Another sample input\"",
        "status": "PUBLISHED",
        "statusMessage": "",
        "templateName": "fargate-service"
    }
}
```


Visualizza i dati del modello

È possibile visualizzare elenchi di modelli con dettagli e visualizzare singoli modelli con dati di dettaglio utilizzando la AWS Proton console e AWS CLI.

I dati del modello di ambiente gestito dal cliente includono il `provisioned` parametro con il valore`CUSTOMER_MANAGED`.

Se un modello di servizio non include una pipeline di servizi, i dati del modello di servizio includono il `pipelineProvisioning` parametro con il valore`CUSTOMER_MANAGED`.

Per ulteriori informazioni, consulta Registra e pubblica modelli.

È possibile utilizzare la console o il AWS CLI per elencare e visualizzare i dati del modello.

AWS Management Console

Usa la console per elencare e visualizzare i modelli.

1. Per visualizzare un elenco di modelli, scegli (Ambiente o Servizio) modelli.
2. Per visualizzare i dati di dettaglio, scegli il nome di un modello.

   Visualizza i dati di dettaglio del modello, un elenco delle versioni principali e secondarie del modello, un elenco delle AWS Proton risorse che sono state distribuite utilizzando le versioni del modello e i tag del modello.

   La versione principale e la versione secondaria consigliate sono etichettate come Consigliate.

AWS CLI

Usa il AWS CLI per elencare e visualizzare i modelli.

Esegui il comando seguente:

```
$ aws proton get-environment-template-version \
    --template-name "simple-env" \
    --major-version "1" \
    --minor-version "0"
```

Risposta:

```
{
    "environmentTemplateVersion": {
        "arn": "arn:aws:proton:region-id:123456789012:environment-template/simple-
env:1.0",
        "createdAt": "2020-11-10T18:35:08.293000+00:00",
        "description": "Version 1",
        "lastModifiedAt": "2020-11-10T18:35:11.162000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "recommendedMinorVersion": "0",
        "schema": "schema:\n  format:\n    openapi: \"3.0.0\"\n
 environment_input_type: \"MyEnvironmentInputType\"\n  types:\n
 MyEnvironmentInputType:\n      type: object\n      description: \"Input properties
 for my environment\"\n      properties:\n        my_sample_input:\n
  type: string\n          description: \"This is a sample input\"\n
 default: \"hello world\"\n        my_other_sample_input:\n          type: string
```

```
\n          description: \"Another sample input\"\n      required:\n        -
 my_other_sample_input\n",
        "status": "DRAFT",
        "statusMessage": "",
        "templateName": "simple-env"
    }
}
```

Esegui il comando seguente:

```
$ aws proton list-environment-templates
```

Risposta:

```
{
    "templates": [
        {
            "arn": "arn:aws:proton:region-id:123456789012:environment-template/
simple-env-3",
            "createdAt": "2020-11-10T18:35:05.763000+00:00",
            "description": "VPC with Public Access",
            "displayName": "VPC",
            "lastModifiedAt": "2020-11-10T18:35:05.763000+00:00",
            "name": "simple-env-3",
            "recommendedVersion": "1.0"
        },
        {
            "arn": "arn:aws:proton:region-id:123456789012:environment-template/
simple-env-1",
            "createdAt": "2020-11-10T00:14:06.881000+00:00",
            "description": "Some SSM Parameters",
            "displayName": "simple-env-1",
            "lastModifiedAt": "2020-11-10T00:14:06.881000+00:00",
            "name": "simple-env-1",
            "recommendedVersion": "1.0"
        }
    ]
}
```

Visualizza una versione secondaria di un modello di servizio.

Esegui il comando seguente:

```
$ aws proton get-service-template-version \
    --template-name "fargate-service" \
    --major-version "1" \
    --minor-version "0"
```

Risposta:


```
{
    "serviceTemplateMinorVersion": {
        "arn": "arn:aws:proton:us-east-1:123456789012:service-template/fargate-
service:1.0",
        "compatibleEnvironmentTemplates": [
            {
                "majorVersion": "1",
                "templateName": "simple-env"
            }
        ],
        "createdAt": "2020-11-11T23:02:57.912000+00:00",
        "description": "Version 1",
        "lastModifiedAt": "2020-11-11T23:02:57.912000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "schema": "schema:\n  format:\n    openapi: \"3.0.0\"\n
 pipeline_input_type: \"MyPipelineInputType\"\n  service_input_type:
 \"MyServiceInstanceInputType\"\n\n  types:\n    MyPipelineInputType:\n
    type: object\n      description: \"Pipeline input properties\"\n
 required:\n        - my_sample_pipeline_required_input\n      properties:\n
       my_sample_pipeline_optional_input:\n          type: string\n
 description: \"This is a sample input\"\n          default: \"hello world\"\n
  my_sample_pipeline_required_input:\n          type: string\n          description:
 \"Another sample input\"\n\n    MyServiceInstanceInputType:\n      type: object
\n      description: \"Service instance input properties\"\n      required:\n
       - my_sample_service_instance_required_input\n      properties:\n
 my_sample_service_instance_optional_input:\n          type: string\n
 description: \"This is a sample input\"\n          default: \"hello world\"\n
     my_sample_service_instance_required_input:\n          type: string\n
 description: \"Another sample input\"",
        "status": "DRAFT",
        "statusMessage": "",
        "templateName": "fargate-service"
    }
}
```

Visualizza un modello di servizio senza una pipeline di servizio, come illustrato nel comando e risposta di esempio successivo.

Esegui il comando seguente:

```
$ aws proton get-service-template \
    --name "simple-svc-template-cli"
```

Risposta:

```
{
    "serviceTemplate": {
        "arn": "arn:aws:proton:region-id:123456789012:service-template/simple-svc-
template-cli",
        "createdAt": "2021-02-18T15:38:57.949000+00:00",
        "displayName": "simple-svc-template-cli",
        "lastModifiedAt": "2021-02-18T15:38:57.949000+00:00",
        "status": "DRAFT",
        "name": "simple-svc-template-cli",
        "pipelineProvisioning": "CUSTOMER_MANAGED"
    }
}
```

Aggiornare un modello

È possibile aggiornare un modello come descritto nell'elenco seguente.

- Modifica l'`description`or `display name` di un modello quando usi la console o AWS CLI. Non puoi modificare il file `name` di un modello.
- Aggiorna lo stato di una versione secondaria del modello quando utilizzi la console o AWS CLI. È possibile modificare lo stato solo da `DRAFT` a`PUBLISHED`.
- Modifica il nome visualizzato e la descrizione di una versione secondaria o principale di un modello quando utilizzi il AWS CLI.

AWS Management Console

Modifica la descrizione e il nome visualizzato del modello utilizzando la console come descritto nei passaggi seguenti.

Nell'elenco dei modelli.

1. Nella AWS Proton console, scegli (Ambiente o Servizio) Modelli.
2. Nell'elenco dei modelli, scegli il pulsante di opzione a sinistra del modello di cui desideri aggiornare la descrizione o il nome visualizzato.
3. Scegli Azioni, quindi Modifica.
4. Nella pagina Modifica (ambiente o servizio) del modello, nella sezione Dettagli del modello, inserisci le modifiche nel modulo e scegli Salva modifiche.

Modifica lo stato di una versione secondaria di un modello utilizzando la console per pubblicare un modello come descritto di seguito. È possibile modificare lo stato solo da `DRAFT` a`PUBLISHED`.

Nella pagina dei dettagli del modello (ambiente o servizio).

1. Nella AWS Proton console, scegli i modelli (Ambiente o Servizio).
2. Nell'elenco dei modelli, scegli il nome del modello di cui desideri aggiornare lo stato di una versione secondaria da Bozza a Pubblicata.
3. Nella pagina di dettaglio del modello (ambiente o servizio), nella sezione Versioni del modello, seleziona il pulsante di opzione a sinistra della versione secondaria che desideri pubblicare.
4. Scegli Pubblica nella sezione Versioni del modello. Lo stato cambia da Bozza a Pubblicato.

AWS CLI

Il comando e la risposta di esempio seguenti mostrano come modificare la descrizione di un modello di ambiente.

Esegui il comando seguente.

```
$ aws proton update-environment-template \
    --name "simple-env" \
    --description "A single VPC with public access"
```

Risposta:

```
{
    "environmentTemplate": {
```

```
        "arn": "arn:aws:proton:region-id:123456789012:environment-template/simple-
env",
        "createdAt": "2020-11-28T22:02:10.651000+00:00",
        "description": "A single VPC with public access",
        "displayName": "simple-env",
        "lastModifiedAt": "2020-11-29T16:11:18.956000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "recommendedMinorVersion": "0",
        "schema": "schema:\n  format:\n    openapi: \"3.0.0\"\n
 environment_input_type: \"MyEnvironmentInputType\"\n  types:\n
 MyEnvironmentInputType:\n      type: object\n      description: \"Input properties
 for my environment\"\n      properties:\n        my_sample_input:\n
  type: string\n          description: \"This is a sample input\"\n
 default: \"hello world\"\n        my_other_sample_input:\n          type: string
\n          description: \"Another sample input\"\n      required:\n        -
 my_other_sample_input\n",
        "status": "PUBLISHED",
        "statusMessage": "",
        "templateName": "simple-env"
    }
}
```


È inoltre possibile utilizzare il AWS CLI per aggiornare i modelli di servizio. Vedi Registrare e pubblicare modelli di servizio il passaggio 5 per un esempio di aggiornamento dello stato di una versione secondaria di un modello di servizio.

Eliminare i modelli

I modelli possono essere eliminati utilizzando la console e AWS CLI.

È possibile eliminare una versione secondaria di un modello di ambiente se non sono presenti ambienti distribuiti in quella versione.

È possibile eliminare una versione secondaria di un modello di servizio se non sono presenti istanze di servizio o pipeline distribuite in quella versione. La pipeline può essere distribuita su una versione del modello diversa rispetto all'istanza del servizio. Ad esempio, se l'istanza del servizio viene aggiornata alla versione 1.1 dalla 1.0 e la pipeline è ancora distribuita alla versione 1.0, non è possibile eliminare il modello di servizio 1.0.

AWS Management Console

Puoi utilizzare la console per eliminare l'intero modello o singole versioni secondarie e principali di un modello.

Utilizza la console per eliminare i modelli come segue.

> **Note**
>
> Quando si utilizza la console per eliminare i modelli.
>
> - Quando si elimina l'intero modello, si eliminano anche le versioni principali e secondarie del modello.

Nell'elenco dei modelli (di ambiente o servizio).

1. Nella AWS Proton console, scegli (Ambiente o Servizio) Modelli.
2. Nell'elenco dei modelli, seleziona il pulsante di opzione a sinistra del modello che desideri eliminare.

   Puoi eliminare un intero modello solo se non ci sono AWS Proton risorse distribuite nelle relative versioni.
3. Scegli Azioni, quindi Elimina per eliminare l'intero modello.
4. Una finestra modale richiede di confermare l'azione di eliminazione.
5. Segui le istruzioni e scegli Sì, elimina.

Nella pagina dei dettagli del modello (ambiente o servizio).

1. Nella AWS Proton console, scegli Modelli (Ambiente o Servizio).
2. Nell'elenco dei modelli, scegli il nome del modello che desideri eliminare completamente o eliminane le singole versioni principali o secondarie.
3. Per eliminare l'intero modello.

   È possibile eliminare un intero modello solo se non ci sono AWS Proton risorse distribuite nelle relative versioni.

   a. Scegli Elimina, nell'angolo in alto a destra della pagina.

b. Una finestra modale richiede di confermare l'azione di eliminazione.

c. Segui le istruzioni e scegli Sì, elimina.

4. Per eliminare le versioni principali o secondarie di un modello.

   È possibile eliminare una versione secondaria di un modello solo se non ci sono AWS Proton risorse distribuite in quella versione.

   a. Nella sezione Versioni del modello, seleziona il pulsante di opzione a sinistra della versione che desideri eliminare.

   b. Scegli Elimina nella sezione Versioni del modello.

   c. Una finestra modale richiede di confermare l'azione di eliminazione.

   d. Segui le istruzioni e scegli Sì, elimina.

AWS CLI

AWS CLI le operazioni di eliminazione di un modello non includono l'eliminazione di altre versioni di un modello. Quando si utilizza AWS CLI, eliminare i modelli con le seguenti condizioni.

- Elimina un intero modello se non esistono versioni principali o secondarie del modello.
- Elimina una versione principale quando elimini l'ultima versione secondaria rimanente.
- Elimina una versione secondaria di un modello se non ci sono AWS Proton risorse distribuite in quella versione.
- Elimina la versione secondaria consigliata di un modello se non esistono altre versioni secondarie del modello e non ci sono AWS Proton risorse distribuite in quella versione.

I comandi e le risposte di esempio seguenti mostrano come utilizzare i modelli AWS CLI per eliminare.

Esegui il comando seguente:

```
$ aws proton delete-environment-template-version \
    --template-name "simple-env" \
    --major-version "1" \
    --minor-version "0"
```

Risposta:

```
{
    "environmentTemplateVersion": {
        "arn": "arn:aws:proton:region-id:123456789012:environment-template/simple-
env:1.0",
        "createdAt": "2020-11-11T23:02:47.763000+00:00",
        "description": "Version 1",
        "lastModifiedAt": "2020-11-11T23:02:54.610000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "status": "PUBLISHED",
        "statusMessage": "",
        "templateName": "simple-env"
    }
}
```

Esegui il comando seguente:

```
$ aws proton delete-environment-template \
    --name "simple-env"
```

Risposta:

```
{
    "environmentTemplate": {
        "arn": "arn:aws:proton:region-id:123456789012:environment-template/simple-
env",
        "createdAt": "2020-11-11T23:02:45.336000+00:00",
        "description": "VPC with Public Access",
        "displayName": "VPC",
        "lastModifiedAt": "2020-11-12T00:23:22.339000+00:00",
        "name": "simple-env",
        "recommendedVersion": "1.0"
    }
}
```

Esegui il comando seguente:

```
$ aws proton delete-service-template-version \
    --template-name "fargate-service" \
    --major-version "1" \
    --minor-version "0"
```

Risposta:

```
{
    "serviceTemplateVersion": {
        "arn": "arn:aws:proton:region-id:123456789012:service-template/fargate-
service:1.0",
        "compatibleEnvironmentTemplates": [{"majorVersion": "1", "templateName":
 "simple-env"}],
        "createdAt": "2020-11-28T22:07:05.798000+00:00",
        "lastModifiedAt": "2020-11-28T22:19:05.368000+00:00",
        "majorVersion": "1",
        "minorVersion": "0",
        "status": "PUBLISHED",
        "statusMessage": "",
        "templateName": "fargate-service"
    }
}
```

Configurazioni di sincronizzazione dei modelli

Scopri come configurare un modello per consentire la AWS Proton sincronizzazione dai pacchetti di modelli che si trovano nei repository git registrati da te definiti. Quando un commit viene inviato al tuo repository, AWS Proton verifica la presenza di modifiche ai pacchetti di modelli di repository. Se rileva una modifica al pacchetto di modelli, viene creata una nuova versione secondaria o principale del modello, se la versione non esiste già. AWS Proton attualmente supporta GitHub GitHub Enterprise e. BitBucket

Invio di un commit a un pacchetto di modelli sincronizzato

Quando invii un commit a un ramo monitorato da uno dei tuoi modelli, AWS Proton clona il tuo repository e determina quali modelli deve sincronizzare. Scansiona i file nella tua directory per trovare le directory che corrispondono alla convenzione di. `{template-name}/{major-version}/`

Dopo aver AWS Proton determinato quali modelli e versioni principali sono associati al repository e al ramo, inizia a provare a sincronizzare tutti quei modelli in parallelo.

Durante ogni sincronizzazione con un particolare modello, verifica AWS Proton innanzitutto se il contenuto della directory dei modelli è cambiato dall'ultima sincronizzazione riuscita. Se il contenuto non è cambiato, AWS Proton salta la registrazione di un pacchetto duplicato. Ciò garantisce la creazione di una nuova versione secondaria del modello se il contenuto del pacchetto di modelli

cambia. Se il contenuto del pacchetto di modelli è cambiato, il pacchetto viene registrato con. AWS Proton

Dopo la registrazione del pacchetto di modelli, AWS Proton monitora lo stato della registrazione fino al completamento della registrazione.

È possibile effettuare una sola sincronizzazione per una particolare versione secondaria e principale del modello alla volta. Tutti i commit che potrebbero essere stati inviati mentre era in corso una sincronizzazione vengono raggruppati in batch. I commit in batch vengono sincronizzati dopo il completamento del tentativo di sincronizzazione precedente.

Modelli di servizio di sincronizzazione

AWS Proton può sincronizzare modelli di ambiente e di servizio dal tuo repository git. Per sincronizzare i modelli di servizio, aggiungi un file aggiuntivo denominato `.template-registration.yaml` in ciascuna directory delle versioni principali del tuo pacchetto di modelli. Questo file contiene dettagli aggiuntivi AWS Proton necessari quando crea una versione del modello di servizio a seguito di un commit: ambienti compatibili e fonti di componenti supportate.

Il percorso completo del file è*`service-template-name`*`/`*`major-version`*`/.template-registration.yaml`. Per ulteriori informazioni, consulta the section called “Modelli di servizio di sincronizzazione”.

Considerazioni sulla configurazione della sincronizzazione dei modelli

Esamina le seguenti considerazioni sull'utilizzo delle configurazioni di sincronizzazione dei modelli.

- I repository non devono superare i 250 MB.
- Per configurare la sincronizzazione dei modelli, collega innanzitutto il repository a. AWS Proton Per ulteriori informazioni, consulta the section called “Creazione di un collegamento di repository”.
- Quando viene creata una nuova versione del modello da un modello sincronizzato, è nello `DRAFT` stato.
- Viene creata una nuova versione secondaria di un modello se si verifica una delle seguenti condizioni:
  - I contenuti del pacchetto di modelli sono diversi da quelli dell'ultima versione secondaria del modello sincronizzata.
  - L'ultima versione secondaria del modello precedentemente sincronizzata è stata eliminata.
- La sincronizzazione non può essere messa in pausa.

- Sia le nuove versioni secondarie che quelle principali vengono sincronizzate automaticamente.
- I nuovi modelli di primo livello non possono essere creati mediante configurazioni di sincronizzazione dei modelli.
- Non è possibile sincronizzare un modello da più repository con una configurazione di sincronizzazione dei modelli.
- Non puoi usare tag al posto dei rami.
- Quando si crea un modello di servizio, si specificano modelli di ambiente compatibili.
- È possibile creare un modello di ambiente e aggiungerlo come ambiente compatibile per il modello di servizio nello stesso commit.
- Le sincronizzazioni con una singola versione principale del modello vengono eseguite una alla volta. Durante una sincronizzazione, se vengono rilevati nuovi commit, questi vengono raggruppati in batch e applicati al termine della sincronizzazione attiva. Le sincronizzazioni con diverse versioni principali dei modelli avvengono in parallelo.
- Se modifichi il ramo da cui vengono sincronizzati i modelli, tutte le sincronizzazioni in corso dal ramo precedente vengono completate per prime. Quindi la sincronizzazione inizia dal nuovo ramo.
- Se modifichi l'archivio da cui esegui la sincronizzazione dei modelli, tutte le sincronizzazioni in corso dal vecchio repository potrebbero non riuscire o essere completate. Dipende dalla fase della sincronizzazione in cui si trovano.

Per ulteriori informazioni, consulta The AWS Proton Service API Reference.

Argomenti

- Crea una configurazione di sincronizzazione dei modelli
- Visualizza i dettagli di configurazione della sincronizzazione dei modelli
- Modifica una configurazione di sincronizzazione dei modelli
- Eliminare una configurazione di sincronizzazione dei modelli

Crea una configurazione di sincronizzazione dei modelli

Scopri come creare una configurazione di sincronizzazione dei modelli con AWS Proton.

Crea i prerequisiti di configurazione per la sincronizzazione dei modelli:

- Hai collegato un repository con. AWS Proton

- Un pacchetto di modelli si trova nel tuo repository.

Il link al repository è composto da quanto segue:

- Una CodeConnections connessione che AWS Proton autorizza l'accesso al repository e la sottoscrizione alle relative notifiche.
- Un ruolo collegato al servizio. Quando colleghi il tuo repository, il ruolo collegato al servizio viene creato automaticamente.

Prima di creare la prima configurazione di sincronizzazione dei modelli, inserisci un pacchetto di modelli nel tuo repository, come mostrato nel seguente layout di directory.

```
/templates/                                                 # subdirectory (optional)
/templates/my-env-template/                                 # template name
/templates/my-env-template/v1/                              # template version
/templates/my-env-template/v1/infrastructure/               # template bundle
/templates/my-env-template/v1/schema/
```

Dopo aver creato la prima configurazione di sincronizzazione dei modelli, le nuove versioni dei modelli vengono create automaticamente quando invii un commit che aggiunge un pacchetto di modelli aggiornato in una nuova versione (ad esempio, sotto). `/my-env-template/v2/`

```
/templates/                                                 # subdirectory (optional)
/templates/my-env-template/                                 # template name
/templates/my-env-template/v1/                              # template version
/templates/my-env-template/v1/infrastructure/               # template bundle
/templates/my-env-template/v1/schema/
/templates/my-env-template/v2/
/templates/my-env-template/v2/infrastructure/
/templates/my-env-template/v2/schema/
```

Puoi includere nuove versioni di pacchetti di modelli per uno o più modelli configurati per la sincronizzazione in un unico commit. AWS Proton crea una nuova versione del modello per ogni nuova versione del pacchetto di modelli inclusa nel commit.

Dopo aver creato la configurazione di sincronizzazione dei modelli, puoi comunque creare manualmente nuove versioni del modello nella console o caricando i pacchetti di modelli AWS CLI da un bucket S3. La sincronizzazione dei modelli funziona solo in una direzione: dal tuo repository a. AWS Proton Le versioni dei modelli create manualmente non vengono sincronizzate.

Dopo aver impostato una configurazione di sincronizzazione dei modelli, AWS Proton ascolta le modifiche al tuo repository. Ogni volta che viene inserita una modifica, cerca una cartella con lo stesso nome del modello. Quindi cerca all'interno di quella directory tutte le directory che assomigliano alle versioni principali. AWS Proton registra il pacchetto di modelli nella versione principale del modello corrispondente. Le nuove versioni sono sempre disponibili nello `DRAFT` stato. Puoi pubblicare le nuove versioni con la console o AWS CLI.

Ad esempio, supponiamo di avere un modello chiamato `my-env-template` configurato per la sincronizzazione da `my-repo/templates` on branch `main` con il seguente layout.

```
/code
/code/service.go
README.md
/templates/
/templates/my-env-template/
/templates/my-env-template/v1/
/templates/my-env-template/v1/infrastructure/
/templates/my-env-template/v1/schema/
/templates/my-env-template/v2/
/templates/my-env-template/v2/infrastructure/
/templates/my-env-template/v2/schema/
```

AWS Proton sincronizza il contenuto di `/templates/my-env-template/v1/` to `my-env-template:1` e il contenuto di `/templates/my-env-template/v2/` to. `my-env-template:2` Se non esistono già, crea queste versioni principali.

AWS Proton ha trovato la prima directory che corrisponde al nome del modello. È possibile limitare le AWS Proton ricerche nelle directory specificando un `subdirectoryPath` quando si crea o si modifica una configurazione di sincronizzazione dei modelli. Ad esempio, è possibile specificare per. `/production-templates/` `subdirectoryPath`

È possibile creare una configurazione di sincronizzazione dei modelli utilizzando la console o la CLI.

AWS Management Console

Crea un modello e una configurazione di sincronizzazione dei modelli utilizzando la console.

1. Nella AWS Proton console, scegli Modelli di ambiente.
2. Scegli Crea modello di ambiente.

3. Nella pagina Crea modello di ambiente, nella sezione Opzioni modello, scegli Crea un modello per il provisioning di nuovi ambienti.
4. Nella sezione Sorgente del pacchetto di modelli, scegli Sincronizza modelli da Git.
5. Nella sezione Archivio del codice sorgente:
   a. Per Repository, seleziona l'archivio collegato che contiene il tuo pacchetto di modelli.
   b. Per Branch, seleziona un ramo del repository da cui eseguire la sincronizzazione.
   c. (Facoltativo) Per la directory dei pacchetti di modelli, inserite il nome di una directory per delimitare la ricerca del pacchetto di modelli.
6. Nella sezione Dettagli del modello.
   a. Inserisci un Nome del modello.
   b. (Facoltativo) Inserite un nome da visualizzare per il modello.
   c. (Facoltativo) Inserite una descrizione del modello per il modello di ambiente.
7. (Facoltativo) Seleziona la casella di controllo Personalizza le impostazioni di crittografia (avanzate) nella sezione Impostazioni di crittografia per fornire la tua chiave di crittografia.
8. (Facoltativo) Nella sezione Tag, scegli Aggiungi nuovo tag e inserisci una chiave e un valore per creare un tag gestito dal cliente.
9. Scegli il modello Create Environment.

   Ora ti trovi su una nuova pagina che mostra lo stato e i dettagli del tuo nuovo modello di ambiente. Questi dettagli includono un elenco di tag AWS gestiti e gestiti dai clienti. AWS Proton genera automaticamente tag AWS gestiti per te quando crei AWS Proton risorse. Per ulteriori informazioni, consulta AWS Proton risorse e etichettatura.
10. Nella pagina dei dettagli del modello, scegli la scheda Sincronizzazione per visualizzare i dati di dettaglio della configurazione della sincronizzazione del modello.
11. Scegli la scheda Versioni del modello per visualizzare le versioni del modello con i dettagli sullo stato.
12. Lo stato di un nuovo modello di ambiente inizia nello stato Bozza. Tu e altre persone con `proton:CreateEnvironment` autorizzazioni potete visualizzarlo e accedervi. Segui il passaggio successivo per rendere il modello disponibile agli altri.
13. Nella sezione Versioni del modello, scegli il pulsante di opzione a sinistra della versione secondaria del modello che hai appena creato (1.0). In alternativa, puoi scegliere Pubblica nell'avviso informativo e saltare il passaggio successivo.

14. Nella sezione Versioni dei modelli, scegli Pubblica.
15. Lo stato del modello cambia in Pubblicato. È la versione più recente e consigliata del modello.
16. Nel riquadro di navigazione, seleziona Modelli di ambiente per visualizzare un elenco dei modelli e dei dettagli dell'ambiente.

La procedura per creare un modello di servizio e una configurazione di sincronizzazione dei modelli è simile.

AWS CLI

Crea un modello e una configurazione di sincronizzazione dei modelli utilizzando AWS CLI.

1. Crea un modello. In questo esempio, viene creato un modello di ambiente.

   Esegui il comando seguente.

   ```
   $ aws proton create-environment-template \
       --name "env-template"
   ```

   La risposta è la seguente.

   ```
   {
       "environmentTemplate": {
           "arn": "arn:aws:proton:us-east-1:123456789012:environment-template/env-
   template",
           "createdAt": "2021-11-07T23:32:43.045000+00:00",
           "displayName": "env-template",
           "lastModifiedAt": "2021-11-07T23:32:43.045000+00:00",
           "name": "env-template",
           "status": "DRAFT",
           "templateName": "env-template"
       }
   }
   ```

2. Crea la configurazione di sincronizzazione dei modelli con AWS CLI fornendo quanto segue:

   - Il modello con cui vuoi sincronizzarti. Dopo aver creato la configurazione di sincronizzazione dei modelli, puoi comunque crearne nuove versioni manualmente nella console o con AWS CLI.
   - Il nome del modello.

- Il tipo di modello.
- L'archivio collegato da cui desideri eseguire la sincronizzazione.
- Il fornitore di repository collegato.
- La filiale in cui si trova il pacchetto di modelli.
- (Facoltativo) Il percorso della directory contenente il pacchetto di modelli. Per impostazione predefinita, AWS Proton cerca la prima directory che corrisponde al nome del modello.

Esegui il comando seguente.

```
$ aws proton create-template-sync-config \
    --template-name "env-template" \
    --template-type "ENVIRONMENT" \
    --repository-name "myrepos/templates" \
    --repository-provider "GITHUB" \
    --branch "main" \
    --subdirectory "env-template/"
```

La risposta è la seguente.

```
{
    "templateSyncConfigDetails": {
        "branch": "main",
        "repositoryName": "myrepos/templates",
        "repositoryProvider": "GITHUB",
        "subdirectory": "templates",
        "templateName": "env-template",
        "templateType": "ENVIRONMENT"
    }
}
```

3. Per pubblicare la versione del modello, consultaRegistra e pubblica modelli.

Modelli di servizio di sincronizzazione

Gli esempi precedenti mostrano come sincronizzare i modelli di ambiente. I modelli di servizio sono simili. Per sincronizzare i modelli di servizio, aggiungi un file aggiuntivo denominato `.template-registration.yaml` a ciascuna directory delle versioni principali del pacchetto di modelli. Questo file contiene dettagli aggiuntivi AWS Proton necessari quando crea una versione del modello di

servizio a seguito di un commit. Quando crei esplicitamente una versione di modello di servizio utilizzando la AWS Proton console o l'API, fornisci questi dettagli come input e questo file sostituisce questi input per la sincronizzazione del modello.

```
./templates/                                                  # subdirectory (optional)
 /templates/my-svc-template/                                  # service template name
 /templates/my-svc-template/v1/                               # service template version
 /templates/my-svc-template/v1/.template-registration.yaml    # service template version
 properties
 /templates/my-svc-template/v1/instance_infrastructure/       # template bundle
 /templates/my-svc-template/v1/schema/
```

Il `.template-registration.yaml` file contiene i seguenti dettagli:

- Ambienti compatibili [richiesto]: gli ambienti basati su questi modelli di ambiente e le versioni principali sono compatibili con i servizi basati su questa versione del modello di servizio.
- Fonti di componenti supportate [opzionale]: i componenti che utilizzano queste fonti sono compatibili con i servizi basati su questa versione del modello di servizio. Se non specificato, i componenti non possono essere collegati a questi servizi. Per ulteriori informazioni sui componenti, vedereComponenti.

La sintassi YAML del file è la seguente:

```
compatible_environments:
  - env-templ-name:major-version
  - ...
supported_component_sources:
  - DIRECTLY_DEFINED
```

Specificate una o più combinazioni di modelli di ambiente/versioni principali. `supported_component_sources`La specificazione è facoltativa e l'unico valore supportato è`DIRECTLY_DEFINED`.

Example .template-registration.yaml

In questo esempio, la versione del modello di servizio è compatibile con le versioni principali 1 e 2 del modello di ambiente. `my-env-template` È anche compatibile con le versioni principali 1 e 3 del modello di `another-env-template` ambiente. Il file non

specificasupported_component_sources, quindi i componenti non possono essere allegati ai servizi basati su questa versione del modello di servizio.

```
compatible_environments:
  - my-env-template:1
  - my-env-template:2
  - another-env-template:1
  - another-env-template:3
```

> **Note**
>
> In precedenza, AWS Proton definiva un file diverso.compatible-envs, per specificare ambienti compatibili. AWS Proton supporta ancora quel file e il suo formato per la compatibilità con le versioni precedenti. Non è più consigliabile utilizzarlo, perché non è estensibile e non può supportare funzionalità più recenti come i componenti.

Visualizza i dettagli di configurazione della sincronizzazione dei modelli

Visualizza i dati di dettaglio della configurazione di sincronizzazione dei modelli utilizzando la console o la CLI.

AWS Management Console

Utilizza la console per visualizzare i dettagli di configurazione della sincronizzazione dei modelli.

1. Nel riquadro di navigazione, scegli (Ambiente o Servizio) modelli.
2. Per visualizzare i dati di dettaglio, scegli il nome di un modello per cui hai creato una configurazione di sincronizzazione dei modelli.
3. Nella pagina dei dettagli del modello, seleziona la scheda Sincronizzazione per visualizzare i dati di dettaglio della configurazione di sincronizzazione del modello.

AWS CLI

Utilizza il AWS CLI per visualizzare un modello sincronizzato.

Esegui il comando seguente.

```
$ aws proton get-template-sync-config \
```

```
    --template-name "svc-template" \
    --template-type "SERVICE"
```

La risposta è la seguente.

```
{
    "templateSyncConfigDetails": {
        "branch": "main",
        "repositoryProvider": "GITHUB",
        "repositoryName": "myrepos/myrepo",
        "subdirectory": "svc-template",
        "templateName": "svc-template",
        "templateType": "SERVICE"
    }
}
```

Usa AWS CLI per ottenere lo stato di sincronizzazione del modello.

Per`template-version`, inserisci la versione principale del modello.

Esegui il comando seguente.

```
$ aws proton get-template-sync-status \
    --template-name "env-template" \
    --template-type "ENVIRONMENT" \
    --template-version "1"
```

Modifica una configurazione di sincronizzazione dei modelli

È possibile modificare qualsiasi parametro di configurazione di sincronizzazione del modello tranne `template-name` e`template-type`.

Scopri come modificare una configurazione di sincronizzazione dei modelli utilizzando la console o la CLI.

AWS Management Console

Modifica un ramo di configurazione di sincronizzazione dei modelli utilizzando la console.

Nell'elenco dei modelli.

1. Nella AWS Proton console, scegli (Ambiente o Servizio) Modelli.

2. Nell'elenco dei modelli, scegli il nome del modello con la configurazione di sincronizzazione dei modelli che desideri modificare.
3. Nella pagina dei dettagli del modello, scegli la scheda Sincronizzazione modello.
4. Nella sezione Dettagli di sincronizzazione del modello, scegli Modifica.
5. Nella pagina Modifica, nella sezione Archivio del codice sorgente, per Branch, seleziona un ramo, quindi scegli Salva configurazione.

AWS CLI

Il comando e la risposta di esempio seguenti mostrano come modificare una configurazione di sincronizzazione dei modelli **`branch`** utilizzando la CLI.

Esegui il comando seguente.

```
$ aws proton update-template-sync-config \
    --template-name "env-template" \
    --template-type "ENVIRONMENT" \
    --repository-provider "GITHUB" \
    --repository-name "myrepos/templates" \
    --branch "fargate" \
    --subdirectory "env-template"
```

La risposta è la seguente.

```
{
    "templateSyncConfigDetails": {
        "branch": "fargate",
        "repositoryProvider": "GITHUB",
        "repositoryName": "myrepos/myrepo",
        "subdirectory": "templates",
        "templateName": "env-template",
        "templateType": "ENVIRONMENT"
    }
}
```

Allo stesso modo è possibile utilizzare il AWS CLI per aggiornare i modelli di servizio sincronizzati.

Eliminare una configurazione di sincronizzazione dei modelli

Eliminare una configurazione di sincronizzazione dei modelli utilizzando la console o la CLI.

AWS Management Console

Elimina una configurazione di sincronizzazione dei modelli utilizzando la console.

1. Nella pagina dei dettagli del modello, scegli la scheda Sincronizzazione.
2. Nella sezione Dettagli di sincronizzazione, scegli Disconnetti.

AWS CLI

I comandi e le risposte di esempio seguenti mostrano come utilizzare per AWS CLI eliminare le configurazioni dei modelli sincronizzati.

Esegui il comando seguente.

```
$ aws proton delete-template-sync-config \
    --template-name "env-template" \
    --template-type "ENVIRONMENT"
```

La risposta è la seguente.

```
{
    "templateSyncConfig": {
        "templateName": "env-template",
        "templateType": "ENVIRONMENT"
    }
}
```

Configurazioni di sincronizzazione dei servizi

Con la sincronizzazione dei servizi, puoi configurare e distribuire i tuoi AWS Proton servizi utilizzando Git. Puoi usare service sync per gestire le distribuzioni e gli aggiornamenti iniziali del tuo AWS Proton servizio con una configurazione definita in un repository Git. Tramite Git, puoi utilizzare funzionalità come il monitoraggio delle versioni e le richieste pull per configurare, gestire e distribuire i tuoi servizi. Service sync combina AWS Proton e Git per aiutarti a fornire un'infrastruttura standardizzata definita e gestita tramite AWS Proton modelli. Gestisce le definizioni dei servizi nel tuo repository Git e

riduce il cambio di strumento. Rispetto all'utilizzo esclusivo di Git, la standardizzazione dei modelli e dell'implementazione AWS Proton consente di dedicare meno tempo alla gestione dell'infrastruttura. AWS Proton offre inoltre una maggiore trasparenza e verificabilità sia per gli sviluppatori che per i team di piattaforma.

AWS Proton File OPS

Il `proton-ops` file definisce dove si AWS Proton trova il file delle specifiche utilizzato per aggiornare l'istanza del servizio. Definisce inoltre in quale ordine aggiornare le istanze del servizio e quando promuovere le modifiche da un'istanza all'altra.

Il `proton-ops` file supporta la sincronizzazione di un'istanza del servizio utilizzando il file spec, o più file di specifiche, presenti nell'archivio collegato. È possibile farlo definendo un blocco di sincronizzazione nel `proton-ops` file, come nell'esempio seguente.

Esempio. /configuration/proton-ops.yaml:

```
sync:
  services:
      frontend-svc:
          alpha:
              branch: dev
              spec: ./frontend-svc/test/frontend-spec.yaml
          beta:
              branch: dev
              spec: ./frontend-svc/test/frontend-spec.yaml
          gamma:
              branch: pre-prod
              spec: ./frontend-svc/pre-prod/frontend-spec.yaml
          prod-one:
              branch: prod
              spec: ./frontend-svc/prod/frontend-spec-second.yaml
          prod-two:
              branch: prod
              spec: ./frontend-svc/prod/frontend-spec-second.yaml
          prod-three:
              branch: prod
              spec: ./frontend-svc/prod/frontend-spec-second.yaml
```

Nell'esempio precedente, `frontend-svc` è il nome del servizio e,,,, e `alpha` sono le istanze. `beta` `gamma` `prod-one` `prod-two` `prod-three`

Il `spec` file può essere composto da tutte le istanze o da un sottoinsieme delle istanze definite all'interno del file. `proton-ops` Tuttavia, come minimo, deve avere l'istanza definita all'interno del ramo e le specifiche da cui viene sincronizzata. Se le istanze non sono definite nel `proton-ops` file, con il ramo e la posizione del `spec` file specifici, la sincronizzazione del servizio non creerà o aggiornerà tali istanze.

Gli esempi seguenti mostrano l'aspetto dei `spec` file. Ricorda che il `proton-ops` file viene sincronizzato da questi `spec` file.

Esempio **`./frontend-svc/test/frontend-spec.yaml`**:

```
proton: "ServiceSpec"
instances:
- name: "alpha"
  environment: "frontend-env"
  spec:
    port: 80
    desired_count: 1
    task_size: "x-small"
    image: "public.ecr.aws/z9d2n7e1/nginx:1.21.0"
- name: "beta"
  environment: "frontend-env"
  spec:
    port: 80
    desired_count: 1
    task_size: "x-small"
    image: "public.ecr.aws/z9d2n7e1/nginx:1.21.0"
```

Esempio **`./frontend-svc/pre-prod/frontend-spec.yaml`**:

```
proton: "ServiceSpec"
instances:
- name: "gamma"
  environment: "frontend-env"
  spec:
    port: 80
    desired_count: 1
    task_size: "x-small"
    image: "public.ecr.aws/z9d2n7e1/nginx:1.21.0"
```

Esempio **`./frontend-svc/prod/frontend-spec-second.yaml`**:

```
proton: "ServiceSpec"
instances:
- name: "prod-one"
  environment: "frontend-env"
  spec:
    port: 80
    desired_count: 1
    task_size: "x-small"
    image: "public.ecr.aws/z9d2n7e1/nginx:1.21.0"
- name: "prod-two"
  environment: "frontend-env"
  spec:
    port: 80
    desired_count: 1
    task_size: "x-small"
    image: "public.ecr.aws/z9d2n7e1/nginx:1.21.0"
- name: "prod-three"
  environment: "frontend-env"
  spec:
    port: 80
    desired_count: 1
    task_size: "x-small"
    image: "public.ecr.aws/z9d2n7e1/nginx:1.21.0"
```

Se un'istanza non si sincronizza e il problema persiste durante il tentativo di sincronizzazione, chiamare l'GetServiceInstanceSyncStatusAPI può aiutare a risolvere il problema.

> **Note**
>
> I clienti che utilizzano la sincronizzazione del servizio sono ancora limitati da AWS Proton limiti.

Bloccanti

Sincronizzando il servizio utilizzando AWS Proton service sync, puoi aggiornare le specifiche del servizio e creare e aggiornare le istanze del servizio dal tuo repository Git. Tuttavia, in alcuni momenti potrebbe essere necessario aggiornare manualmente un servizio o un'istanza tramite o. AWS Management Console AWS CLI

AWS Proton aiuta a evitare di sovrascrivere eventuali modifiche manuali apportate tramite il AWS Management Console sistema operativo AWS CLI, ad esempio l'aggiornamento di un'istanza del servizio o l'eliminazione di un'istanza del servizio. A tal fine, crea AWS Proton automaticamente un blocco della sincronizzazione dei servizi disabilitando la sincronizzazione del servizio quando rileva una modifica manuale.

Per ottenere tutti i blocker associati a un servizio, è necessario eseguire le seguenti operazioni in ordine per ciascuno `serviceInstance` associato al servizio:

- Chiama l'`getServiceSyncBlockerSummary`API solo con. `serviceName`
- Chiama l'`getServiceSyncBlockerSummary`API con `serviceName` `eserviceInstanceName`.

Ciò restituisce un elenco dei bloccanti più recenti e lo stato ad essi associato. Se alcuni bloccanti sono contrassegnati come ATTIVI, è necessario risolverli chiamando l'`UpdateServiceSyncBlocker`API con `blockerId` e `resolvedReason` per ognuno di essi.

Se aggiorni o crei manualmente un'istanza del servizio, AWS Proton crea un blocco della sincronizzazione dei servizi sull'istanza del servizio. AWS Proton continua a sincronizzare tutte le altre istanze del servizio, ma disabilita la sincronizzazione di questa istanza di servizio fino alla risoluzione del blocco. Se elimini un'istanza del servizio da un servizio, AWS Proton crea un blocco della sincronizzazione dei servizi sul servizio. Ciò AWS Proton impedisce la sincronizzazione di qualsiasi istanza del servizio fino a quando il blocco non è stato risolto.

Dopo aver installato tutti i bloccanti attivi, è necessario risolverli chiamando l'`UpdateServiceSyncBlocker`API con `blockerId` e `resolvedReason` per ciascuno dei bloccanti attivi.

Utilizzando AWS Management Console, è possibile determinare se la sincronizzazione di un servizio è disabilitata accedendo alla scheda Service Sync AWS Proton e selezionandola. Se il servizio o le istanze del servizio sono bloccati, viene visualizzato il pulsante Abilita. Per abilitare la sincronizzazione del servizio, scegli Abilita.

Argomenti

- Crea una configurazione di sincronizzazione del servizio
- Visualizza i dettagli di configurazione per una sincronizzazione del servizio
- Modifica una configurazione di sincronizzazione del servizio
- Eliminare una configurazione di sincronizzazione del servizio

Crea una configurazione di sincronizzazione del servizio

Puoi creare una configurazione di sincronizzazione del servizio utilizzando la console o AWS CLI.

AWS Management Console

1. Nella pagina Scegli un modello di servizio, seleziona un modello e scegli Configura.
2. Nella pagina Configura servizio, nella sezione Dettagli del servizio, inserisci un nuovo nome di servizio.
3. (Facoltativo) Inserisci una descrizione per il servizio.
4. Nella sezione Archivio del codice sorgente dell'applicazione, scegli Scegli un repository Git collegato per selezionare un repository a cui ti sei già collegato. AWS Proton Se non hai già un repository collegato, scegli Collega un altro repository Git e segui le istruzioni in Creare un link al tuo repository.
5. Per Repository, scegli il nome del tuo repository di codice sorgente dall'elenco.
6. Per Branch, scegli il nome del ramo del repository per il tuo codice sorgente dall'elenco.
7. (Facoltativo) Nella sezione Tag, scegli Aggiungi nuovo tag e inserisci una chiave e un valore per creare un tag gestito dal cliente.
8. Scegli Next (Successivo).
9. Nella pagina Configura istanze del servizio, nella sezione Sorgente della definizione del servizio, seleziona Sincronizza il servizio da Git.
10. Nella sezione File di definizione del servizio, se desideri AWS Proton creare il tuo `proton-ops` file, seleziona Voglio che AWS Proton crei i file. Con questa opzione, AWS Proton crea il `proton-ops` file `spec` and nelle posizioni specificate. Seleziona Sto fornendo i miei file per creare il tuo file OPS.
11. Nella sezione Service definition repository, scegli Scegli un repository Git collegato per selezionare un repository a cui ti sei già collegato. AWS Proton
12. Per Nome del repository, scegli il nome del tuo repository di codice sorgente dall'elenco.
13. Per **`proton-ops`**il ramo di file, scegli il nome del tuo ramo dall'elenco in cui AWS Proton verranno inseriti il tuo file OPS e spec.
14. Nella sezione Istanze di servizio, ogni campo viene compilato automaticamente in base ai valori del file. `proton-ops`
15. Scegli Avanti e controlla i dati inseriti.
16. Scegli Create (Crea).

AWS CLI

Crea una configurazione di sincronizzazione del servizio utilizzando AWS CLI

- Esegui il comando seguente.

```
$ aws proton create-service-sync-config \
    --resource "service-arn" \
    --repository-provider "GITHUB" \
    --repository "example/proton-sync-service" \
    --ops-file-branch "main" \
    --proton-ops-file "./configuration/custom-proton-ops.yaml" (optional)
```

La risposta è la seguente.

```
{
    "serviceSyncConfig": {
        "branch": "main",
        "filePath": "./configuration/custom-proton-ops.yaml",
        "repositoryName": "example/proton-sync-service",
        "repositoryProvider": "GITHUB",
        "serviceName": "service name"
    }
}
```

Visualizza i dettagli di configurazione per una sincronizzazione del servizio

È possibile visualizzare i dati dei dettagli di configurazione per una sincronizzazione del servizio utilizzando la console o AWS CLI.

AWS Management Console

Utilizza la console per visualizzare i dettagli di configurazione per una sincronizzazione del servizio

1. Nel riquadro di navigazione, scegli Servizi.
2. Per visualizzare i dati di dettaglio, scegli il nome di un servizio per cui hai creato una configurazione di sincronizzazione del servizio.

3. Nella pagina dei dettagli del servizio, seleziona la scheda Service sync per visualizzare i dati di dettaglio della configurazione per la sincronizzazione del servizio.

AWS CLI

Utilizza il AWS CLI per ottenere un servizio sincronizzato.

Esegui il comando seguente.

```
$ aws proton get-service-sync-config \
    --service-name "service name"
```

La risposta è la seguente.

```
{
    "serviceSyncConfig": {
        "branch": "main",
        "filePath": "./configuration/custom-proton-ops.yaml",
        "repositoryName": "example/proton-sync-service",
        "repositoryProvider": "GITHUB",
        "serviceName": "service name"
    }
}
```

Usa il AWS CLI per ottenere lo stato di sincronizzazione del servizio.

Esegui il comando seguente.

```
$ aws proton get-service-sync-status \
    --service-name "service name"
```

Modifica una configurazione di sincronizzazione del servizio

Puoi modificare una configurazione di sincronizzazione del servizio utilizzando la console o AWS CLI.

AWS Management Console

Modifica una configurazione di sincronizzazione del servizio utilizzando la console.

1. Nel riquadro di navigazione, scegli Servizi.
2. Per visualizzare i dati di dettaglio, scegli il nome di un servizio per cui hai creato una configurazione di sincronizzazione del servizio.
3. Nella pagina dei dettagli del servizio, scegli la scheda Sincronizzazione del servizio.
4. Nella sezione Sincronizzazione del servizio, scegli Modifica.
5. Nella pagina Modifica, aggiorna le informazioni che desideri modificare, quindi scegli Salva.

AWS CLI

Il comando e la risposta di esempio seguenti mostrano come modificare una configurazione di sincronizzazione del servizio utilizzando AWS CLI.

Esegui il comando seguente.

```
$ aws proton update-service-sync-config \
    --service-name "service name" \
    --repository-provider "GITHUB" \
    --repository "example/proton-sync-service" \
    --ops-file-branch "main" \
    --ops-file "./configuration/custom-proton-ops.yaml"
```

La risposta è la seguente.

```
{
    "serviceSyncConfig": {
        "branch": "main",
        "filePath": "./configuration/custom-proton-ops.yaml",
        "repositoryName": "example/proton-sync-service",
        "repositoryProvider": "GITHUB",
        "serviceName": "service name"
    }
}
```

Eliminare una configurazione di sincronizzazione del servizio

È possibile eliminare una configurazione di sincronizzazione del servizio utilizzando la console o AWS CLI.

AWS Management Console

Eliminare una configurazione di sincronizzazione del servizio utilizzando la console

1. Nella pagina dei dettagli del servizio, scegli la scheda Sincronizzazione del servizio.
2. Nella sezione Dettagli di sincronizzazione del servizio, scegli Disconnetti per disconnettere il repository. Dopo la disconnessione del repository, non sincronizziamo più il servizio da quel repository.

AWS CLI

I comandi e le risposte di esempio seguenti mostrano come utilizzare le configurazioni sincronizzate del servizio AWS CLI per eliminare.

Esegui il comando seguente.

```
$ aws proton delete-service-sync-config \
    --service-name "service name"
```

La risposta è la seguente.

```
{
    "serviceSyncConfig": {
        "branch": "main",
        "filePath": "./configuration/custom-proton-ops.yaml",
        "repositoryName": "example/proton-sync-service",
        "repositoryProvider": "GITHUB",
        "serviceName": "service name"
    }
}
```

> **Note**
>
> La sincronizzazione dei servizi non elimina le istanze del servizio. Elimina solo la configurazione.

AWS Proton ambienti

Infatti AWS Proton, un ambiente rappresenta l'insieme di risorse e politiche condivise in cui vengono distribuiti i servizi. AWS Proton Possono contenere tutte le risorse che dovrebbero essere condivise tra le istanze AWS Proton del servizio. Queste risorse possono includere cluster e VPCs sistemi di bilanciamento del carico condivisi o gateway API. È necessario creare un AWS Proton ambiente prima di potervi distribuire un servizio.

Questa sezione descrive come gestire gli ambienti utilizzando le operazioni di creazione, visualizzazione, aggiornamento ed eliminazione. Per >informazioni aggiuntive, consulta The AWS Proton Service API Reference.

Argomenti

- Ruoli IAM
- Creazione di un ambiente
- Visualizza i dati ambientali
- Aggiorna un ambiente
- Eliminare un ambiente
- Connessioni agli account di ambiente
- Ambienti gestiti dal cliente
- CodeBuild creazione di ruoli di provisioning

Ruoli IAM

Con AWS Proton, fornisci i ruoli e AWS KMS le chiavi IAM per le AWS risorse che possiedi e gestisci. Questi vengono successivamente applicati e utilizzati dalle risorse possedute e gestite dagli sviluppatori. Crei un ruolo IAM per controllare l'accesso del tuo team di sviluppatori all' AWS Proton API.

AWS Proton ruolo di servizio

Quando crei un nuovo ambiente, fornisci un ruolo di servizio IAM correlato. Il ruolo contiene tutte le autorizzazioni necessarie per aggiornare tutta l'infrastruttura fornita definita sia nei modelli di ambiente che nei modelli di servizio. Per esempi di ruolo, vedere. AWS Proton ruolo di servizio

per il provisioning utilizzando AWS CloudFormation Se si utilizzano connessioni con account di ambiente e account di ambiente, si crea il ruolo in un account di ambiente selezionato. Per ulteriori informazioni, consultare Crea un ambiente in un account ed esegui il provisioning in un altro account e Connessioni agli account di ambiente.

Il modo in cui viene fornito questo ruolo di servizio e chi lo assume dipendono dal metodo di provisioning dell'ambiente.

- AWS-provisioning gestito: fornisci il ruolo a AWS Proton, direttamente durante la creazione di un ambiente o indirettamente tramite connessioni all'account. AWS Proton assume il ruolo nell'account pertinente per fornire l'ambiente e l'infrastruttura di servizio.
- Provisioning autogestito: è responsabilità dell'utente configurare l'automazione del provisioning per assumere un ruolo appropriato utilizzando le credenziali appropriate quando una pull request (PR) attiva un'azione di provisioning. Per un esempio di GitHub azione che presuppone un ruolo, consulta Assumere un ruolo nella documentazione «Configure Credentials» Action For Actions. AWS GitHub

Per ulteriori informazioni sui metodi di provisioning, vedere. the section called “Metodi di approvvigionamento”

Creazione di un ambiente

Impara a creare AWS Proton ambienti.

Puoi creare un AWS Proton ambiente in due modi:

- Crea, gestisci ed esegui il provisioning di un ambiente standard utilizzando un modello di ambiente standard. AWS Proton fornisce l'infrastruttura per il tuo ambiente.
- Connect AWS Proton all'infrastruttura gestita dal cliente utilizzando un modello di ambiente gestito dal cliente. Effettua il provisioning delle tue risorse condivise all'esterno di AWS Proton e quindi fornisci output di provisioning utilizzabili. AWS Proton

È possibile scegliere uno dei diversi approcci di provisioning quando si crea un ambiente.

- AWS provisioning gestito: creazione, gestione e provisioning di un ambiente in un unico account. AWS Proton effettua il provisioning del tuo ambiente.

  Questo metodo supporta solo modelli di codice di CloudFormation infrastruttura (IaC).

- AWS provisioning gestito su un altro account: in un unico account di gestione, crea e gestisci un ambiente fornito in un altro account con connessioni di account di ambiente. AWS Proton effettua il provisioning dell'ambiente nell'altro account. Per ulteriori informazioni, consultare Crea un ambiente in un account ed esegui il provisioning in un altro account e Connessioni agli account di ambiente.

  Questo metodo supporta solo i modelli CloudFormation IaC.
- Provisioning autogestito: AWS Proton invia le richieste pull di provisioning a un repository collegato con la propria infrastruttura di provisioning.

  Questo metodo supporta solo i modelli Terraform IAc.
- CodeBuild provisioning: AWS Proton viene utilizzato AWS CodeBuild per eseguire i comandi di shell forniti dall'utente. I comandi sono in grado di leggere gli input che AWS Proton forniscono e sono responsabili del provisioning o del deprovisioning dell'infrastruttura e della generazione di valori di output. Un pacchetto di modelli per questo metodo include i comandi in un file manifest e tutti i programmi, script o altri file necessari per questi comandi.

  Come esempio di utilizzo del CodeBuild provisioning, è possibile includere codice che utilizza AWS Cloud Development Kit (AWS CDK) per il provisioning AWS delle risorse e un manifest che installa il CDK ed esegue il codice CDK.

  Per ulteriori informazioni, consulta the section called “CodeBuild pacchetto”.

  > **Note**
  >
  > È possibile utilizzare il CodeBuild provisioning con ambienti e servizi. Al momento non è possibile effettuare il provisioning dei componenti in questo modo.

Con il provisioning AWS gestito (sia nello stesso account che su un altro account), AWS Proton effettua chiamate dirette per il rifornimento delle risorse.

Con il provisioning autogestito, AWS Proton effettua richieste pull per fornire file IaC compilati utilizzati dal motore IAc per l'approvvigionamento delle risorse.

Per ulteriori informazioni, consulta the section called “Metodi di approvvigionamento”, the section called “Pacchetti di modelli” e the section called “Requisiti dello schema ambientale”.

Argomenti

- Crea e fornisci un ambiente standard nello stesso account

- Crea un ambiente in un account ed esegui il provisioning in un altro account
- Crea ed esegui il provisioning di un ambiente utilizzando il provisioning autogestito

Crea e fornisci un ambiente standard nello stesso account

Utilizza la console o AWS CLI per creare ed eseguire il provisioning di un ambiente in un unico account. Il provisioning è gestito da AWS.

AWS Management Console

Utilizza la console per creare ed effettuare il provisioning di un ambiente in un unico account

1. Nella AWS Proton console, scegli Ambienti.
2. Seleziona Create environment (Crea ambiente).
3. Nella pagina Scegli un modello di ambiente, seleziona un modello e scegli Configura.
4. Nella pagina Configura ambiente, nella sezione Provisioning, scegli Provisioning AWS gestito.
5. Nella sezione Account di distribuzione, scegli Questo. Account AWS
6. Nella pagina Configura ambiente, nella sezione Impostazioni dell'ambiente, inserisci un nome per l'ambiente.
7. (Facoltativo) Inserisci una descrizione per l'ambiente.
8. Nella sezione Ruoli ambientali, seleziona il ruolo di AWS Proton servizio di cui hai creatoConfigurazione dei ruoli AWS Proton di servizio.
9. (Facoltativo) Nella sezione Ruolo del componente, selezionate un ruolo di servizio che consenta ai componenti definiti direttamente di funzionare nell'ambiente e definisca le risorse a cui possono fornire. Per ulteriori informazioni, consulta Componenti.
10. (Facoltativo) Nella sezione Tag, scegli Aggiungi nuovo tag e inserisci una chiave e un valore per creare un tag gestito dal cliente.
11. Scegli Next (Successivo).
12. Nella pagina Configurazione delle impostazioni personalizzate dell'ambiente, devi inserire i valori per i `required` parametri. È possibile inserire valori per i `optional` parametri o utilizzare i valori predefiniti, se disponibili.
13. Scegliete Avanti e controllate i dati inseriti.
14. Scegli Create (Crea).

Visualizza i dettagli e lo stato dell'ambiente, nonché i tag AWS gestiti e i tag gestiti dal cliente per il tuo ambiente.

15. Nel riquadro di navigazione, selezionare Compute environments (Ambienti di calcolo).

    Una nuova pagina mostra un elenco degli ambienti insieme allo stato e ad altri dettagli dell'ambiente.

AWS CLI

Utilizza il AWS CLI per creare ed eseguire il provisioning di un ambiente in un unico account.

Per creare un ambiente, è necessario specificare l'ARN del ruolo di AWS Proton servizio, il percorso del file delle specifiche, il nome dell'ambiente, l'ARN del modello di ambiente, le versioni principali e secondarie e la descrizione (opzionale).

Gli esempi successivi mostrano un file di specifiche YAML formattato che specifica i valori per due input definiti nel file di schema del modello di ambiente. È possibile utilizzare il `get-environment-template-minor-version` comando per visualizzare lo schema del modello di ambiente.

```
proton: EnvironmentSpec
spec:
  my_sample_input: "the first"
  my_other_sample_input: "the second"
```

Creare un ambiente eseguendo il comando seguente.

```
$ aws proton create-environment \
    --name "MySimpleEnv" \
    --template-name simple-env \
    --template-major-version 1 \
    --proton-service-role-arn "arn:aws:iam::123456789012:role/AWS ProtonServiceRole"
 \
    --spec "file://env-spec.yaml"
```

Risposta:

```
{
```

```
    "environment": {
        "arn": "arn:aws:proton:region-id:123456789012:environment/MySimpleEnv",
        "createdAt": "2020-11-11T23:03:05.405000+00:00",
        "deploymentStatus": "IN_PROGRESS",
        "lastDeploymentAttemptedAt": "2020-11-11T23:03:05.405000+00:00",
        "name": "MySimpleEnv",
        "protonServiceRoleArn": "arn:aws:iam::123456789012:role/ProtonServiceRole",
        "templateName": "simple-env"
    }
}
```

Dopo aver creato un nuovo ambiente, è possibile visualizzare un elenco di AWS tag gestiti dal cliente, come illustrato nel seguente comando di esempio. AWS Proton genera automaticamente tag AWS gestiti per te. Puoi anche modificare e creare tag gestiti dai clienti utilizzando AWS CLI. Per ulteriori informazioni, consulta AWS Proton risorse e etichettatura.

Comando:

```
$ aws proton list-tags-for-resource \
    --resource-arn "arn:aws:proton:region-id:123456789012:environment/MySimpleEnv"
```

Crea un ambiente in un account ed esegui il provisioning in un altro account

Utilizza la console o AWS CLI per creare un ambiente standard in un account di gestione che esegue il provisioning dell'infrastruttura ambientale in un altro account. Il provisioning è gestito da AWS.

Prima di utilizzare la console o la CLI, completa i seguenti passaggi.

1. Identifica Account AWS IDs l'account di gestione e ambiente e copiali per un uso successivo.
2. Nell'account di ambiente, crea un ruolo AWS Proton di servizio con autorizzazioni minime per la creazione dell'ambiente. Per ulteriori informazioni, consulta AWS Proton ruolo di servizio per il provisioning utilizzando AWS CloudFormation.

AWS Management Console

Usa la console per creare un ambiente in un account ed effettuare il provisioning in un altro.

1. Nell'account di ambiente, crea una connessione all'account di ambiente e usala per inviare una richiesta di connessione all'account di gestione.

   a. Nella AWS Proton console, scegli Environment account connections nel pannello di navigazione.

   b. Nella pagina Connessioni agli account Environment, scegli Richiedi di connessione.

   > **Note**
   >
   > Verifica che l'ID dell'account elencato nell'intestazione della pagina di connessione all'account Environment corrisponda all'ID dell'account di ambiente preidentificato.

   c. Nella pagina Richiesta di connessione, nella sezione Ruolo ambientale, seleziona Ruolo di servizio esistente e il nome del ruolo di servizio che hai creato per l'ambiente.

   d. Nella sezione Connect to management account, inserisci l'ID dell'account di gestione e un nome di ambiente per il tuo AWS Proton ambiente. Copia il nome per un uso successivo.

   e. Scegli Richiedi di connessione nell'angolo in basso a destra della pagina.

   f. La richiesta viene visualizzata come in sospeso nella tabella delle connessioni di ambiente inviate a un account di gestione e una finestra di dialogo mostra come accettare la richiesta dall'account di gestione.

2. Nell'account di gestione, accetta una richiesta di connessione dall'account di ambiente.

   a. Accedi al tuo account di gestione e scegli Environment account connections nella AWS Proton console.

   b. Nella pagina Connessioni all'account Environment, nella tabella Richieste di connessione all'account Environment, seleziona la connessione all'account di ambiente con l'ID dell'account di ambiente che corrisponde all'ID dell'account di ambiente preidentificato.

> **Note**
>
> Verifica che l'ID dell'account elencato nell'intestazione della pagina di connessione all'account Environment corrisponda all'ID dell'account di gestione preidentificato.

c. Scegliere Accept (Accetta). Lo stato cambia da PENDING a CONNECTED.

3. Nell'account di gestione, crea un ambiente.

a. Nel riquadro di navigazione, scegli Modelli di ambiente.

b. Nella pagina Modelli di ambiente, scegli Crea modello di ambiente.

c. Nella pagina Scegli un modello di ambiente, scegli un modello di ambiente.

d. Nella pagina Configura ambiente, nella sezione Provisioning, scegli Provisioning AWS gestito.

e. Nella sezione Account di distribuzione, scegli Altro AWS account;.

f. Nella sezione Dettagli dell'ambiente, seleziona la connessione all'account Environment e il nome dell'ambiente.

g. Scegli Next (Successivo).

h. Compila i moduli e scegli Avanti fino a raggiungere la pagina Rivedi e crea.

i. Rivedi e scegli Crea ambiente.

AWS CLI

Utilizza il AWS CLI per creare un ambiente in un account e il provisioning in un altro.

Nell'account di ambiente, create una connessione all'account di ambiente e richiedete la connessione eseguendo il comando seguente.

```
$ aws proton create-environment-account-connection \
    --environment-name "simple-env-connected" \
    --role-arn "arn:aws:iam::222222222222:role/service-role/env-account-proton-service-role" \
    --management-account-id "111111111111"
```

Risposta:

```
{
    "environmentAccountConnection": {
        "arn": "arn:aws:proton:region-id:222222222222:environment-account-
connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "environmentAccountId": "222222222222",
        "environmentName": "simple-env-connected",
        "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "lastModifiedAt": "2021-04-28T23:13:50.847000+00:00",
        "managementAccountId": "111111111111",
        "requestedAt": "2021-04-28T23:13:50.847000+00:00",
        "roleArn": "arn:aws:iam::222222222222:role/service-role/env-account-proton-
service-role",
        "status": "PENDING"
    }
}
```

Nell'account di gestione, accetta la richiesta di connessione all'account di ambiente eseguendo il comando seguente.

```
$ aws proton accept-environment-account-connection \
    --id "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Risposta:

```
{
    "environmentAccountConnection": {
        "arn": "arn:aws:proton:region-id:222222222222:environment-account-
connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "environmentAccountId": "222222222222",
        "environmentName": "simple-env-connected",
        "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "lastModifiedAt": "2021-04-28T23:15:33.486000+00:00",
        "managementAccountId": "111111111111",
        "requestedAt": "2021-04-28T23:13:50.847000+00:00",
        "roleArn": "arn:aws:iam::222222222222:role/service-role/env-account-proton-
service-role",
        "status": "CONNECTED"
    }
}
```

Visualizza la connessione all'account di ambiente eseguendo il comando seguente.

```
$ aws proton get-environment-account-connection \
    --id "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Risposta:

```
{
    "environmentAccountConnection": {
        "arn": "arn:aws:proton:region-id:222222222222:environment-account-
connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "environmentAccountId": "222222222222",
        "environmentName": "simple-env-connected",
        "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "lastModifiedAt": "2021-04-28T23:15:33.486000+00:00",
        "managementAccountId": "111111111111",
        "requestedAt": "2021-04-28T23:13:50.847000+00:00",
        "roleArn": "arn:aws:iam::222222222222:role/service-role/env-account-proton-
service-role",
        "status": "CONNECTED"
    }
}
```

Nell'account di gestione, crea un ambiente eseguendo il comando seguente.

```
$ aws proton create-environment \
    --name "simple-env-connected" \
    --template-name simple-env-template \
    --template-major-version "1" \
    --template-minor-version "1" \
    --spec "file://simple-env-template/specs/original.yaml" \
    --environment-account-connection-id "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Risposta:

```
{
    "environment": {
        "arn": "arn:aws:proton:region-id:111111111111:environment/simple-env-
connected",
        "createdAt": "2021-04-28T23:02:57.944000+00:00",
        "deploymentStatus": "IN_PROGRESS",
        "environmentAccountConnectionId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "environmentAccountId": "222222222222",
        "lastDeploymentAttemptedAt": "2021-04-28T23:02:57.944000+00:00",
```

```
        "name": "simple-env-connected",
        "templateName": "simple-env-template"
    }
}
```

Crea ed esegui il provisioning di un ambiente utilizzando il provisioning autogestito

Quando si utilizza il provisioning autogestito, AWS Proton invia le richieste pull di provisioning a un repository collegato con la propria infrastruttura di provisioning. Le pull request avviano un flusso di lavoro personalizzato, che richiama AWS i servizi al provisioning dell'infrastruttura.

Considerazioni sul provisioning autogestito:

- Prima di creare un ambiente, configurate una directory di risorse del repository per il provisioning autogestito. Per ulteriori informazioni, consulta AWS Proton infrastruttura come file di codice.
- Dopo aver creato l'ambiente, AWS Proton attende di ricevere notifiche asincrone sullo stato del provisioning dell'infrastruttura. Il codice di provisioning deve utilizzare l' AWS Proton `NotifyResourceStateChange`API per inviare queste notifiche asincrone a. AWS Proton

È possibile utilizzare il provisioning autogestito nella console o con. AWS CLI Gli esempi seguenti mostrano come utilizzare il provisioning autogestito con Terraform.

AWS Management Console

Usa la console per creare un ambiente Terraform utilizzando il provisioning autogestito.

1. Nella AWS Proton console, scegli Ambienti.
2. Seleziona Create environment (Crea ambiente).
3. Nella pagina Scegli un modello di ambiente, seleziona un modello Terraform e scegli Configura.
4. Nella pagina Configura ambiente, nella sezione Provisioning, scegli Provisioning autogestito.
5. Nella sezione dei dettagli del repository di Provisioning:

    a. Se non hai ancora collegato il tuo repository di provisioning a AWS Proton, scegli Nuovo repository, scegli uno dei provider di repository e quindi, per la CodeStarconnessione, scegli una delle tue connessioni.

> **Note**
>
> Se non disponi ancora di una connessione all'account del provider di repository pertinente, scegli Aggiungi una nuova connessione. CodeStar Quindi, crea una connessione, quindi scegli il pulsante di aggiornamento accanto al menu di CodeStar connessione. Ora dovresti essere in grado di scegliere la tua nuova connessione nel menu.

Se hai già collegato il tuo repository a AWS Proton, scegli Deposito esistente.

b. Per Nome del repository, scegli un repository. Il menu a discesa mostra i repository collegati per il repository esistente o l'elenco dei repository nell'account del provider per Nuovo repository.

c. Per Nome del ramo, scegli uno dei rami del repository.

6. Nella sezione Impostazioni dell'ambiente, inserisci un nome per l'ambiente.
7. (Facoltativo) Inserisci una descrizione per l'ambiente.
8. (Facoltativo) Nella sezione Tag, scegli Aggiungi nuovo tag e inserisci una chiave e un valore per creare un tag gestito dal cliente.
9. Scegli Next (Successivo).
10. Nella pagina Configurazione delle impostazioni personalizzate dell'ambiente, devi inserire i valori per i `required` parametri. È possibile inserire valori per i `optional` parametri o utilizzare i valori predefiniti, se disponibili.
11. Scegliete Avanti e controllate i dati inseriti.
12. Scegli Crea per inviare una pull request.

    - Se approvi la pull request, la distribuzione è in corso.
    - Se si rifiuta la pull request, la creazione dell'ambiente viene annullata.
    - Se la pull request scade, la creazione dell'ambiente non è completa.
13. Visualizza i dettagli e lo stato dell'ambiente, nonché i tag AWS gestiti e i tag gestiti dal cliente per il tuo ambiente.
14. Nel riquadro di navigazione, selezionare Compute environments (Ambienti di calcolo).

    Una nuova pagina mostra un elenco degli ambienti insieme allo stato e ad altri dettagli dell'ambiente.

AWS CLI

Quando si crea un ambiente utilizzando il provisioning autogestito, si aggiunge il `provisioningRepository` parametro e si omettono i `ProtonServiceRoleArn` parametri and. `environmentAccountConnectionId`

Utilizzate il AWS CLI per creare un ambiente Terraform con provisioning autogestito.

1. Crea un ambiente e invia una pull request al repository per la revisione e l'approvazione.

   Gli esempi successivi mostrano un file di specifiche YAML formattato che definisce i valori per due input in base al file di schema del modello di ambiente. È possibile utilizzare il `get-environment-template-minor-version` comando per visualizzare lo schema del modello di ambiente.

   Specifiche:

   ```
   proton: EnvironmentSpec
   spec:
     ssm_parameter_value: "test"
   ```

   Crea un ambiente eseguendo il comando seguente.

   ```
   $ aws proton create-environment \
       --name "pr-environment" \
       --template-name "pr-env-template" \
       --template-major-version "1" \
       --provisioning-repository="branch=main,name=myrepos/env-
   repo,provider=GITHUB" \
       --spec "file://env-spec.yaml"
   ```

   Risposta: >

   ```
   {
       "environment": {
           "arn": "arn:aws:proton:region-id:123456789012:environment/pr-
   environment",
           "createdAt": "2021-11-18T17:06:58.679000+00:00",
           "deploymentStatus": "IN_PROGRESS",
           "lastDeploymentAttemptedAt": "2021-11-18T17:06:58.679000+00:00",
           "name": "pr-environment",
   ```

```
        "provisioningRepository": {
            "arn": "arn:aws:proton:region-id:123456789012:repository/
github:myrepos/env-repo",
            "branch": "main",
            "name": "myrepos/env-repo",
            "provider": "GITHUB"
        },
        "templateName": "pr-env-template"
    }
```

2. Esamina la richiesta.

   - Se approvi la richiesta, il rifornimento è in corso.
   - Se si rifiuta la richiesta, la creazione dell'ambiente viene annullata.
   - Se la richiesta pull scade, la creazione dell'ambiente non è completa.

3. Fornisci in modo asincrono lo stato di provisioning a. AWS Proton L'esempio seguente notifica AWS Proton l'esito positivo del provisioning.

```
$ aws proton notify-resource-deployment-status-change \
    --resource-arn "arn:aws:proton:region-id:123456789012:environment/pr-
environment" \
    --status "SUCCEEDED"
```

Visualizza i dati ambientali

È possibile visualizzare i dati di dettaglio dell'ambiente utilizzando la AWS Proton console o il AWS CLI.

AWS Management Console

È possibile visualizzare elenchi di ambienti con dettagli e singoli ambienti con dati di dettaglio utilizzando la AWS Proton console.

1. Per visualizzare un elenco dei tuoi ambienti, scegli Ambienti nel pannello di navigazione.
2. Per visualizzare i dati di dettaglio, scegli il nome di un ambiente.

   Visualizza i dati di dettaglio dell'ambiente.

AWS CLI

Usa i dettagli dell'ambiente AWS CLI get o list.

Esegui il comando seguente:

```
$ aws proton get-environment \
    --name "MySimpleEnv"
```

Risposta:

```
{
    "environment": {
        "arn": "arn:aws:proton:region-id:123456789012:environment/MySimpleEnv",
        "createdAt": "2020-11-11T23:03:05.405000+00:00",
        "deploymentStatus": "SUCCEEDED",
        "lastDeploymentAttemptedAt": "2020-11-11T23:03:05.405000+00:00",
        "lastDeploymentSucceededAt": "2020-11-11T23:03:05.405000+00:00",
        "name": "MySimpleEnv",
        "protonServiceRoleArn": "arn:aws:iam::123456789012:role/ProtonServiceRole",
        "spec": "proton: EnvironmentSpec\nspec:\n  my_sample_input: \"the first\"\n
 my_other_sample_input: \"the second\"\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
        "templateName": "simple-env"
    }
}
```

Aggiorna un ambiente

Se l' AWS Proton ambiente è associato a una connessione con un account di ambiente, non aggiornare né includere il `protonServiceRoleArn` parametro per aggiornare o connettersi a una connessione a un account di ambiente.

È possibile eseguire l'aggiornamento a una nuova connessione all'account di ambiente solo se sono soddisfatte entrambe le seguenti condizioni:

- La connessione all'account di ambiente è stata creata nello stesso account di ambiente in cui è stata creata la connessione all'account di ambiente corrente.
- >La connessione all'account di ambiente è associata all'ambiente corrente.

Se l'ambiente non è associato a una connessione all'account di ambiente, non aggiornare o includere il `environmentAccountConnectionId` parametro.

È possibile aggiornare il `protonServiceRoleArn` parametro e il valore `environmentAccountConnectionId` o. Non puoi aggiornare entrambi.

Se il tuo ambiente utilizza il provisioning autogestito, non aggiornare il `provisioning-repository` parametro e ometti i `environmentAccountConnectionId` parametri and. `protonServiceRoleArn`

Esistono quattro modalità per aggiornare un ambiente, come descritto nell'elenco seguente. Quando si utilizza AWS CLI, il `deployment-type` campo definisce la modalità. Quando si utilizza la console, queste modalità sono mappate alle azioni Modifica, Aggiorna, Aggiorna secondarie e Aggiorna principali che si trovano in Azioni.

`NONE`

In questa modalità, non si verifica una distribuzione. Vengono aggiornati solo i parametri dei metadati richiesti.

`CURRENT_VERSION`

In questa modalità, l'ambiente viene distribuito e aggiornato con le nuove specifiche fornite dall'utente. Vengono aggiornati solo i parametri richiesti. Non includere i parametri della versione minore o principale quando lo usi`deployment-type`.

`MINOR_VERSION`

In questa modalità, l'ambiente viene distribuito e aggiornato con la versione secondaria pubblicata, consigliata (più recente) della versione principale corrente in uso per impostazione predefinita. È inoltre possibile specificare una versione secondaria diversa della versione principale corrente in uso.

`MAJOR_VERSION`

In questa modalità, l'ambiente viene distribuito e aggiornato con la versione principale e secondaria pubblicata, consigliata (più recente) del modello corrente per impostazione predefinita. È inoltre possibile specificare una versione principale diversa che sia superiore alla versione principale in uso e una versione secondaria (opzionale).

Argomenti

- Aggiornare un ambiente di provisioning AWS gestito
- Aggiornare un ambiente di provisioning autogestito
- Annullare l'implementazione di un ambiente in corso

Aggiornare un ambiente di provisioning AWS gestito

Il provisioning standard è supportato solo dagli ambienti che forniscono con. AWS CloudFormation

Usa la console o AWS CLI per aggiornare il tuo ambiente.

AWS Management Console

Aggiorna un ambiente utilizzando la console come illustrato nei passaggi seguenti.

1. Scegli uno dei 2 passaggi seguenti.
   a. Nell'elenco degli ambienti.
      i. Nella AWS Proton console, scegli Ambienti.
      ii. Nell'elenco degli ambienti, scegli il pulsante di opzione a sinistra dell'ambiente che desideri aggiornare.
   b. Nella pagina dei dettagli dell'ambiente della console.
      i. Nella AWS Proton console, scegli Ambienti.
      ii. Nell'elenco degli ambienti, scegli il nome dell'ambiente che desideri aggiornare.
2. Scegli uno dei 4 passaggi successivi per aggiornare l'ambiente.
   a. Per apportare una modifica che non richieda la distribuzione nell'ambiente.
      i. Ad esempio, per modificare una descrizione.

         Scegli Modifica.
      ii. Compila il modulo e scegli Avanti.
      iii. Controlla la modifica e scegli Aggiorna.
   b. Per aggiornare solo gli input dei metadati.
      i. Scegli Azioni e poi Aggiorna.

ii. Compila il modulo e scegli Modifica.

iii. Compila i moduli e scegli Avanti fino a raggiungere la pagina di revisione.

iv. Controlla gli aggiornamenti e scegli Aggiorna.

c. Per effettuare un aggiornamento a una nuova versione secondaria del relativo modello di ambiente.

i. Scegli Azioni e poi Aggiorna secondario.

ii. Compila il modulo e scegli Avanti.

iii. Compila i moduli e scegli Avanti fino a raggiungere la pagina di revisione.

iv. Controlla gli aggiornamenti e scegli Aggiorna.

d. Per effettuare un aggiornamento a una nuova versione principale del relativo modello di ambiente.

i. Scegli Azioni e poi Aggiorna principale.

ii. Compila il modulo e scegli Avanti.

iii. Compila i moduli e scegli Avanti fino a raggiungere la pagina di revisione.

iv. Controlla gli aggiornamenti e scegli Aggiorna.

AWS CLI

Usa il AWS Proton AWS CLI per aggiornare un ambiente a una nuova versione secondaria.

Esegui il comando seguente per aggiornare l'ambiente:

```
$ aws proton update-environment \
        --name "MySimpleEnv" \
        --deployment-type "MINOR_VERSION" \
        --template-major-version "1" \
        --template-minor-version "1" \
        --proton-service-role-arn arn:aws:iam::123456789012:role/service-
role/ProtonServiceRole \
        --spec "file:///spec.yaml"
```

Risposta:

```
{
    "environment": {
        "arn": "arn:aws:proton:region-id:123456789012:environment/MySimpleEnv",
```

```
        "createdAt": "2021-04-02T17:29:55.472000+00:00",
        "deploymentStatus": "IN_PROGRESS",
        "lastDeploymentAttemptedAt": "2021-04-02T17:48:26.307000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T17:29:55.472000+00:00",
        "name": "MySimpleEnv",
        "protonServiceRoleArn": "arn:aws:iam::123456789012:role/service-role/
ProtonServiceRole",
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
        "templateName": "simple-env"
    }
}
```

Esegui il comando seguente per ottenere e confermare lo stato:

```
$ aws proton get-environment \
        --name "MySimpleEnv"
```

Risposta:

```
{
    "environment": {
        "arn": "arn:aws:proton:region-id:123456789012:environment/MySimpleEnv",
        "createdAt": "2021-04-02T17:29:55.472000+00:00",
        "deploymentStatus": "SUCCEEDED",
        "environmentName": "MySimpleEnv",
        "lastDeploymentAttemptedAt": "2021-04-02T17:48:26.307000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T17:48:26.307000+00:00",
        "protonServiceRoleArn": "arn:aws:iam::123456789012:role/service-role/
ProtonServiceRole",
        "spec": "proton: EnvironmentSpec\n\nspec:\n  my_sample_input: hello\n
 my_other_sample_input: everybody\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "simple-env"
    }
}
```

Aggiornare un ambiente di provisioning autogestito

Il provisioning autogestito è supportato solo dagli ambienti che utilizzano Terraform.

Usa la console o AWS CLI per aggiornare il tuo ambiente.

AWS Management Console

Aggiorna un ambiente utilizzando la console come illustrato nei passaggi seguenti.

1. Scegli uno dei 2 passaggi seguenti.
   a. Nell'elenco degli ambienti.
      i. Nella AWS Proton console, scegli Ambienti.
      ii. Nell'elenco degli ambienti, scegli il pulsante di opzione a sinistra del modello di ambiente che desideri aggiornare.
   b. Nella pagina dei dettagli dell'ambiente della console.
      i. Nella AWS Proton console, scegli Ambienti.
      ii. Nell'elenco degli ambienti, scegli il nome dell'ambiente che desideri aggiornare.
2. Scegli uno dei 4 passaggi successivi per aggiornare l'ambiente.
   a. Per apportare una modifica che non richieda la distribuzione nell'ambiente.
      i. Ad esempio, per modificare una descrizione.

         Scegli Modifica.
      ii. Compila il modulo e scegli Avanti.
      iii. Controlla la modifica e scegli Aggiorna.
   b. Per aggiornare solo gli input dei metadati.
      i. Scegli Azioni e poi Aggiorna.
      ii. Compila il modulo e scegli Modifica.
      iii. Compila i moduli e scegli Avanti fino a raggiungere la pagina di revisione.
      iv. Controlla gli aggiornamenti e scegli Aggiorna.
   c. Per effettuare un aggiornamento a una nuova versione secondaria del relativo modello di ambiente.
      i. Scegli Azioni e poi Aggiorna secondario.
      ii. Compila il modulo e scegli Avanti.
      iii. Compila i moduli e scegli Avanti fino a raggiungere la pagina di revisione.

iv. Controlla gli aggiornamenti e scegli Aggiorna.

d. Per effettuare un aggiornamento a una nuova versione principale del relativo modello di ambiente.

i. Scegli Azioni e poi Aggiorna principale.

ii. Compila il modulo e scegli Avanti.

iii. Compila i moduli e scegli Avanti fino a raggiungere la pagina di revisione.

iv. Controlla gli aggiornamenti e scegli Aggiorna.

AWS CLI

Usa il AWS CLI per aggiornare un ambiente Terraform a una nuova versione minore con provisioning autogestito.

1. Esegui il seguente comando per aggiornare il tuo ambiente:

```
$ aws proton update-environment \
    --name "pr-environment" \
    --deployment-type "MINOR_VERSION" \
    --template-major-version "1" \
    --template-minor-version "1" \
    --provisioning-repository "branch=main,name=myrepos/env-
repo,provider=GITHUB" \
    --spec "file://env-spec-mod.yaml"
```

Risposta:

```
{
    "environment": {
        "arn": "arn:aws:proton:region-id:123456789012:environment/pr-
environment",
        "createdAt": "2021-11-18T21:09:15.745000+00:00",
        "deploymentStatus": "IN_PROGRESS",
        "lastDeploymentAttemptedAt": "2021-11-18T21:25:41.998000+00:00",
        "lastDeploymentSucceededAt": "2021-11-18T21:09:15.745000+00:00",
        "name": "pr-environment",
        "provisioningRepository": {
            "arn": "arn:aws:proton:region-id:123456789012:repository/
github:myrepos/env-repo",
```

```
            "branch": "main",
            "name": "myrepos/env-repo",
            "provider": "GITHUB"
        },
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
        "templateName": "pr-env-template"
    }
}
```

2. Esegui il comando seguente per ottenere e confermare lo stato:

```
$ aws proton get-environment \
    --name "pr-environment"
```

Risposta:

```
{
    "environment": {
        "arn": "arn:aws:proton:region-id:123456789012:environment/pr-
environment",
        "createdAt": "2021-11-18T21:09:15.745000+00:00",
        "deploymentStatus": "SUCCEEDED",
        "lastDeploymentAttemptedAt": "2021-11-18T21:25:41.998000+00:00",
        "lastDeploymentSucceededAt": "2021-11-18T21:25:41.998000+00:00",
        "name": "pr-environment",
        "provisioningRepository": {
            "arn": "arn:aws:proton:region-id:123456789012:repository/
github:myrepos/env-repo",
            "branch": "main",
            "name": "myrepos/env-repo",
            "provider": "GITHUB"
        },
        "spec": "proton: EnvironmentSpec\nspec:\n   ssm_parameter_value: \"test
\"\n ssm_another_parameter_value: \"update\"\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "pr-env-template"
    }
}
```

3. Controlla la pull request che è stata inviata da AWS Proton.

- Se approvi la richiesta, il provisioning è in corso.
- Se si rifiuta la richiesta, la creazione dell'ambiente viene annullata.
- Se la richiesta pull scade, la creazione dell'ambiente non è completa.

4. Fornisci lo stato di approvvigionamento a. AWS Proton

```
$ aws proton notify-resource-deployment-status-change \
    --resource-arn "arn:aws:proton:region-id:123456789012:environment/pr-environment" \
    --status "SUCCEEDED"
```

Annullare l'implementazione di un ambiente in corso

È possibile tentare di annullare la distribuzione di un aggiornamento dell'ambiente se `deploymentStatus` è attiva`IN_PROGRESS`. AWS Proton tenta di annullare la distribuzione. L'annullamento riuscito non è garantito.

Quando annulli la distribuzione di un aggiornamento, AWS Proton tenta di annullare la distribuzione come indicato nei passaggi seguenti.

Con AWS-managed provisioning, AWS Proton effettua le seguenti operazioni:

- Imposta lo stato di distribuzione su. `CANCELLING`
- Interrompe la distribuzione in corso ed elimina tutte le nuove risorse create dalla distribuzione quando`IN_PROGRESS`.
- Imposta lo stato di distribuzione su`CANCELLED`.
- Riporta lo stato della risorsa allo stato precedente all'avvio della distribuzione.

Con il provisioning autogestito, AWS Proton effettua le seguenti operazioni:

- Tenta di chiudere la pull request per impedire l'unione delle modifiche al repository.
- Imposta lo stato di distribuzione `CANCELLED` se la pull request è stata chiusa con successo.

Per istruzioni su come annullare la distribuzione di un ambiente, consulta CancelEnvironmentDeployment l'AWS Proton API Reference.

È possibile utilizzare la console o la CLI per annullare gli ambienti in corso.

AWS Management Console

Utilizza la console per annullare la distribuzione di un aggiornamento dell'ambiente, come illustrato nei passaggi seguenti.

1. Nella AWS Proton console, scegli Ambienti nel riquadro di navigazione.
2. Nell'elenco degli ambienti, scegli il nome dell'ambiente con l'aggiornamento di distribuzione che desideri annullare.
3. Se lo stato di distribuzione dell'aggiornamento è In corso, nella pagina dei dettagli dell'ambiente, scegli Azioni e quindi Annulla distribuzione.
4. Una finestra modale richiede di confermare l'annullamento. Scegli Annulla distribuzione.
5. Lo stato di distribuzione degli aggiornamenti è impostato su Annullamento e quindi su Annullato per completare l'annullamento.

AWS CLI

Utilizzate il AWS Proton AWS CLI per annullare la distribuzione di un aggiornamento dell'ambiente IN_PROGRESS a una nuova versione secondaria 2.

Nel modello utilizzato per questo esempio è inclusa una condizione di attesa in modo che l'annullamento inizi prima del completamento della distribuzione dell'aggiornamento.

Esegui il comando seguente per annullare l'aggiornamento:

```
$ aws proton cancel-environment-deployment \
        --environment-name "MySimpleEnv"
```

Risposta:

```
{
    "environment": {
        "arn": "arn:aws:proton:region-id:123456789012:environment/MySimpleEnv",
        "createdAt": "2021-04-02T17:29:55.472000+00:00",
        "deploymentStatus": "CANCELLING",
        "lastDeploymentAttemptedAt": "2021-04-02T18:15:10.243000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T17:48:26.307000+00:00",
        "name": "MySimpleEnv",
        "protonServiceRoleArn": "arn:aws:iam::123456789012:role/service-role/
ProtonServiceRole",
```

```
        "spec": "proton: EnvironmentSpec\n\nspec:\n  my_sample_input: hello\n
 my_other_sample_input: everybody\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "simple-env"
    }
}
```

Esegui il comando seguente per ottenere e confermare lo stato:»

```
$ aws proton get-environment \
        --name "MySimpleEnv"
```

Risposta:

```
{
    "environment": {
        "arn": "arn:aws:proton:region-id:123456789012:environment/MySimpleEnv",
        "createdAt": "2021-04-02T17:29:55.472000+00:00",
        "deploymentStatus": "CANCELLED",
        "deploymentStatusMessage": "User initiated cancellation.",
        "lastDeploymentAttemptedAt": "2021-04-02T18:15:10.243000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T17:48:26.307000+00:00",
        "name": "MySimpleEnv",
        "protonServiceRoleArn": "arn:aws:iam::123456789012:role/service-role/
ProtonServiceRole",
        "spec": "proton: EnvironmentSpec\n\nspec:\n  my_sample_input: hello\n
 my_other_sample_input: everybody\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "simple-env"
    }
}
```

Eliminare un ambiente

È possibile eliminare un AWS Proton ambiente utilizzando la AWS Proton console o il AWS CLI.

> **Note**
>
> Non è possibile eliminare un ambiente a cui è associato alcun componente. Per eliminare un ambiente di questo tipo, è necessario innanzitutto eliminare tutti i componenti in esecuzione nell'ambiente. Per ulteriori informazioni sui componenti, vedereComponenti.

AWS Management Console

Eliminare un ambiente utilizzando la console come descritto nelle due opzioni seguenti.

Nell'elenco degli ambienti.

1. Nella AWS Proton console, scegli Ambienti.
2. Nell'elenco degli ambienti, seleziona il pulsante di opzione a sinistra dell'ambiente che desideri eliminare.
3. Scegli Azioni, quindi Elimina.
4. Un modale richiede di confermare l'azione di eliminazione.
5. Segui le istruzioni e scegli Sì, elimina.

Nella pagina dei dettagli dell'ambiente.

1. Nella AWS Proton console, scegli Ambienti.
2. Nell'elenco degli ambienti, scegli il nome dell'ambiente che desideri eliminare.
3. Nella pagina dei dettagli dell'ambiente, scegli Azioni, quindi Elimina.
4. Una finestra modale ti chiede di confermare che desideri eliminare.
5. Segui le istruzioni e scegli Sì, elimina.

AWS CLI

Usa il AWS CLI per eliminare un ambiente.

Non eliminare un ambiente se nell'ambiente vengono distribuiti servizi o istanze di servizio.

Esegui il comando seguente:

```
$ aws proton delete-environment \
```

```
    --name "MySimpleEnv"
```

Risposta:

```
{
    "environment": {
        "arn": "arn:aws:proton:region-id:123456789012:environment/MySimpleEnv",
        "createdAt": "2021-04-02T17:29:55.472000+00:00",
        "deploymentStatus": "DELETE_IN_PROGRESS",
        "lastDeploymentAttemptedAt": "2021-04-02T17:48:26.307000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T17:48:26.307000+00:00",
        "name": "MySimpleEnv",
        "protonServiceRoleArn": "arn:aws:iam::123456789012:role/ProtonServiceRole",
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "simple-env"
    }
}
```

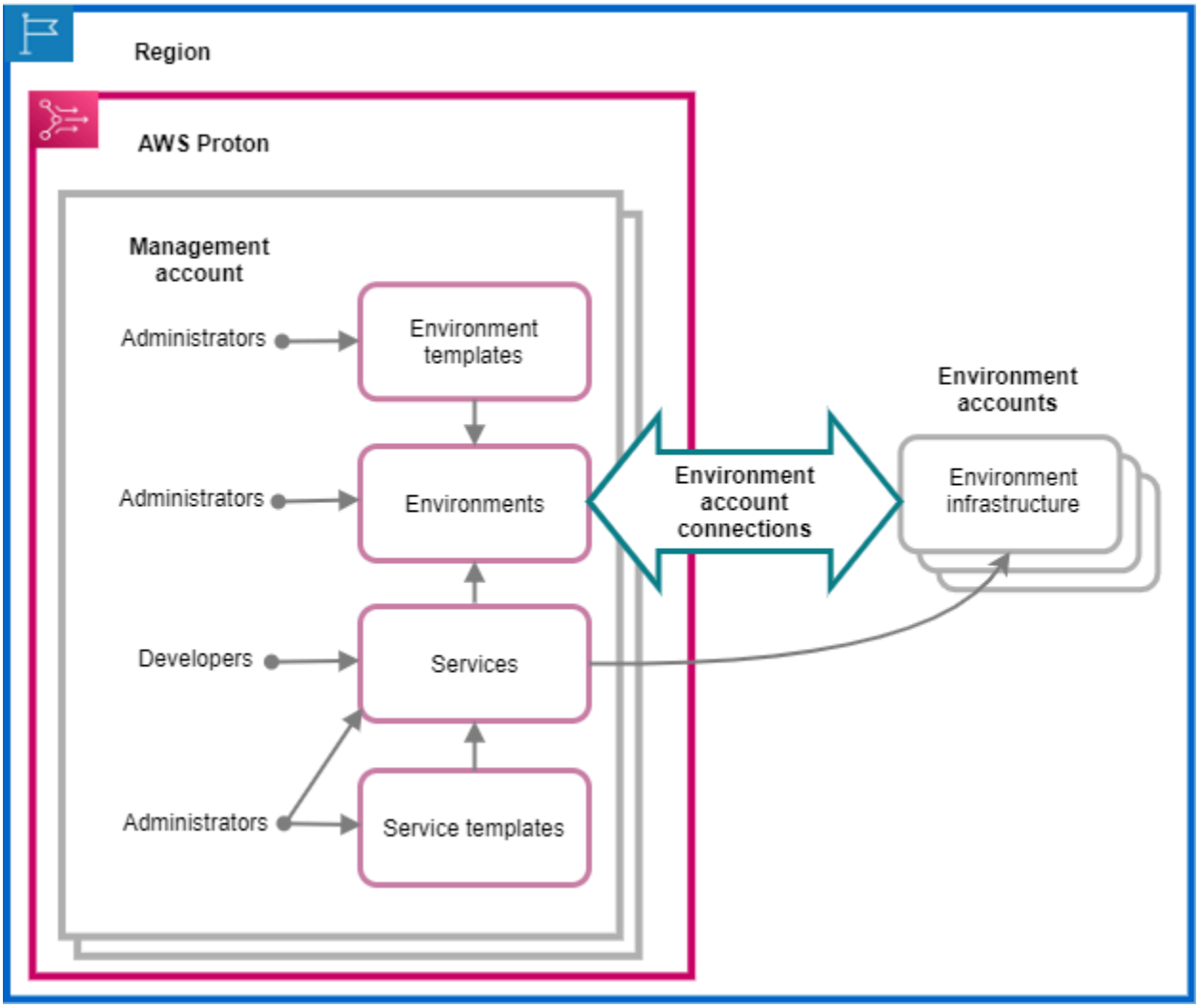
Connessioni agli account di ambiente

Panoramica

Scopri come creare e gestire un AWS Proton ambiente in un account e fornire le relative risorse di infrastruttura in un altro account. Questo può aiutare a migliorare la visibilità e l'efficienza su larga scala. Le connessioni agli account di ambiente supportano solo il provisioning standard con AWS CloudFormation infrastruttura come codice.

> **Note**
>
> Le informazioni contenute in questo argomento sono rilevanti per gli ambienti configurati con il provisioning AWS gestito. Con ambienti configurati con il provisioning autogestito, AWS Proton non effettua direttamente il provisioning dell'infrastruttura. Al contrario, invia richieste pull (PRs) al repository per il provisioning. È tua responsabilità assicurarti che il codice di automazione assuma l'identità e il ruolo corretti.
> Per ulteriori informazioni sui metodi di provisioning, consulta. the section called “Metodi di approvvigionamento”

Terminologia



Con le connessioni agli account di AWS Proton ambiente, è possibile creare un AWS Proton ambiente da un account ed eseguire il provisioning della relativa infrastruttura in un altro account.

Gestione dell'account

L'account singolo in cui, in qualità di amministratore, crei un AWS Proton ambiente che fornisce le risorse dell'infrastruttura in un altro account di ambiente.

Account ambientale

Un account in cui viene eseguito il provisioning dell'infrastruttura ambientale, quando si crea un AWS Proton ambiente in un altro account.

Connessione all'account di ambiente

Una connessione bidirezionale sicura tra un account di gestione e un account di ambiente. Mantiene le autorizzazioni e le autorizzazioni come descritto ulteriormente nelle sezioni seguenti.

Quando si crea una connessione con un account di ambiente in una regione specifica, solo gli account di gestione della stessa regione possono visualizzare e utilizzare la connessione all'account di ambiente. Ciò significa che l' AWS Proton ambiente creato nell'account di gestione e l'infrastruttura di ambiente fornita nell'account di ambiente devono trovarsi nella stessa regione.

Considerazioni sulla connessione all'account di ambiente

- È necessaria una connessione all'account di ambiente per ogni ambiente di cui si desidera effettuare il provisioning in un account di ambiente.
- Per informazioni sulle quote di connessione degli account di ambiente, vedereAWS Proton quote.

Assegnazione di tag

Nell'account di ambiente, utilizza la console o i tag gestiti dal cliente AWS CLI per visualizzare e gestire la connessione all'account di ambiente. AWS i tag gestiti non vengono generati per le connessioni agli account di ambiente. Per ulteriori informazioni, consulta Assegnazione di tag.

Crea un ambiente in un account ed esegui il provisioning della relativa infrastruttura in un altro account

Per creare ed eseguire il provisioning di un ambiente da un singolo account di gestione, configura un account di ambiente per un ambiente che intendi creare.

Avvia l'account di ambiente e crea una connessione.

Nell'account di ambiente, crea un ruolo di AWS Proton servizio limitato solo alle autorizzazioni necessarie per il provisioning delle risorse dell'infrastruttura dell'ambiente. Per ulteriori informazioni, consulta AWS Proton ruolo di servizio per il provisioning utilizzando AWS CloudFormation.

Quindi, crea e invia una richiesta di connessione all'account di ambiente al tuo account di gestione. Quando la richiesta viene accettata, AWS Proton puoi utilizzare il ruolo IAM associato che consente il provisioning delle risorse ambientali nell'account di ambiente associato.

Nell'account di gestione, accetta o rifiuta la connessione all'account di ambiente.

Nell'account di gestione, accetta o rifiuta la richiesta di connessione all'account di ambiente. Non puoi eliminare una connessione all'account di ambiente dal tuo account di gestione.

Se accetti la richiesta, AWS Proton possono utilizzare il ruolo IAM associato che consente il provisioning delle risorse nell'account di ambiente associato.

Le risorse dell'infrastruttura ambientale vengono fornite nell'account di ambiente associato. È possibile utilizzarle AWS Proton APIs per accedere e gestire l'ambiente e le relative risorse di infrastruttura solo dal proprio account di gestione. Per ulteriori informazioni, consultare Crea un ambiente in un account ed esegui il provisioning in un altro account e Aggiorna un ambiente.

Dopo aver rifiutato una richiesta, non è possibile accettare o utilizzare la connessione all'account di ambiente rifiutata.

> **Note**
>
> Non puoi rifiutare una connessione a un account di ambiente connesso a un ambiente. Per rifiutare la connessione all'account di ambiente, devi prima eliminare l'ambiente associato.

Nell'account di ambiente, accedi alle risorse dell'infrastruttura predisposte.

Nell'account di ambiente, è possibile visualizzare e accedere alle risorse dell'infrastruttura assegnate. Ad esempio, è possibile utilizzare le azioni CloudFormation API per monitorare e ripulire gli stack, se necessario. Non è possibile utilizzare le azioni AWS Proton API per accedere o gestire l' AWS Proton ambiente utilizzato per il provisioning delle risorse dell'infrastruttura.

Nell'account di ambiente, è possibile eliminare le connessioni degli account di ambiente che sono state create nell'account di ambiente. Non puoi accettarle o rifiutarle. Se elimini una connessione all'account di ambiente utilizzata da un AWS Proton ambiente, non AWS Proton sarai in grado di gestire le risorse dell'infrastruttura dell'ambiente finché non verrà accettata una nuova connessione di ambiente per l'account di ambiente e l'ambiente denominato. Sei responsabile della pulizia delle risorse assegnate che rimangono senza una connessione all'ambiente.

Usa la console o la CLI per gestire le connessioni degli account dell'ambiente

È possibile utilizzare la console o la CLI per creare e gestire le connessioni degli account di ambiente.

AWS Management Console

Utilizza la console per creare una connessione all'account di ambiente e inviare una richiesta all'account di gestione come illustrato nei passaggi successivi.

1. Decidi un nome per l'ambiente che intendi creare nel tuo account di gestione o scegli il nome di un ambiente esistente che richiede una connessione all'account di ambiente.
2. In un account di ambiente, nella AWS Proton console, scegli Environment account connections nel pannello di navigazione.
3. Nella pagina Connessioni all'account Environment, scegli Richiedi di connessione.

   > **Note**
   >
   > Verifica l'ID dell'account elencato nell'intestazione della pagina di connessione all'account Environment. Assicurati che corrisponda all'ID dell'account di ambiente in cui desideri eseguire il provisioning dell'ambiente denominato.

4. Nella pagina Richiesta di connessione:

   a. Nella sezione Connect to management account, inserisci l'ID dell'account di gestione e il nome dell'ambiente che hai inserito nel passaggio 1.

   b. Nella sezione Ruolo ambientale, scegli Nuovo ruolo di servizio e crea AWS Proton automaticamente un nuovo ruolo per te. In alternativa, seleziona Ruolo di servizio esistente e il nome del ruolo di servizio creato in precedenza.

      > **Note**
      >
      > Il ruolo creato AWS Proton automaticamente per te dispone di ampie autorizzazioni. Ti consigliamo di limitare il ruolo alle autorizzazioni necessarie per fornire le risorse dell'infrastruttura dell'ambiente. Per ulteriori informazioni, consulta AWS Proton ruolo di servizio per il provisioning utilizzando AWS CloudFormation.

   c. (Facoltativo) Nella sezione Tag, scegli Aggiungi nuovo tag per creare un tag gestito dal cliente per la connessione all'account di ambiente.

   d. Scegli Richiedi la connessione.

5. La tua richiesta viene visualizzata come in sospeso nella tabella Environment delle connessioni inviate a un account di gestione e un modulo ti consente di sapere come accettare la richiesta dall'account di gestione.

Accetta o rifiuta una richiesta di connessione all'account di ambiente.

1. In un account di gestione, nella AWS Proton console, scegli Environment account connections nel pannello di navigazione.
2. Nella pagina Connessioni all'account Environment, nella tabella Richieste di connessione all'account Environment, scegli la richiesta di connessione ambientale da accettare o rifiutare.

> **Note**
>
> Verifica l'ID dell'account elencato nell'intestazione della pagina di connessione all'account Environment. Assicurati che corrisponda all'ID dell'account di gestione associato alla connessione dell'account di ambiente da rifiutare. Dopo aver rifiutato questa connessione all'account di ambiente, non puoi accettare o utilizzare la connessione all'account di ambiente rifiutata.

3. Scegli Rifiuta o Accetta.
   - Se hai selezionato Rifiuta, lo stato passa da In sospeso a Rifiutato.
   - Se hai selezionato Accetta, lo stato passa da In sospeso a connesso.

Eliminare una connessione all'account di ambiente.

1. In un account di ambiente, nella AWS Proton console, scegli Connessioni all'account di ambiente nel riquadro di navigazione.

> **Note**
>
> Verifica l'ID dell'account elencato nell'intestazione della pagina di connessione all'account Environment. Assicurati che corrisponda all'ID dell'account di gestione associato alla connessione dell'account di ambiente da rifiutare. Dopo aver eliminato questa connessione all'account di ambiente, non è AWS Proton possibile gestire le risorse dell'infrastruttura di ambiente nell'account di ambiente. Può gestirla solo

dopo che una nuova connessione all'account di ambiente per l'account di ambiente e l'ambiente denominato sono stati accettati dall'account di gestione.

2. Nella pagina Connessioni all'account di ambiente, nella sezione Richieste inviate per la connessione all'account di gestione, scegli Elimina.
3. Una finestra modale ti chiede di confermare che desideri eliminare. Scegliere Delete (Elimina).

AWS CLI

Decidi un nome per l'ambiente che intendi creare nel tuo account di gestione o scegli il nome di un ambiente esistente che richiede una connessione all'account di ambiente.

Crea una connessione all'account di ambiente in un account di ambiente.

Esegui il comando seguente:

```
$ aws proton create-environment-account-connection \
    --environment-name "simple-env-connected" \
    --role-arn "arn:aws:iam::222222222222:role/service-role/env-account-proton-service-role" \
    --management-account-id "111111111111"
```

Risposta:

```
{
    "environmentAccountConnection": {
        "arn": "arn:aws:proton:region-id:222222222222:environment-account-connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "environmentAccountId": "222222222222",
        "environmentName": "simple-env-connected",
        "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "lastModifiedAt": "2021-04-28T23:13:50.847000+00:00",
        "managementAccountId": "111111111111",
        "requestedAt": "2021-04-28T23:13:50.847000+00:00",
        "roleArn": "arn:aws:iam::222222222222:role/service-role/env-account-proton-service-role",
        "status": "PENDING"
    }
}
```

Accetta o rifiuta una connessione con un account di ambiente in un account di gestione, come illustrato nel comando e nella risposta seguenti.

> **Note**
>
> Se rifiuti questa connessione all'account di ambiente, non sarai in grado di accettare o utilizzare la connessione all'account di ambiente rifiutata.

Se si specifica Rifiuta, lo stato passa da In sospeso a Rifiutato.

Se si specifica Accept, lo stato passa da In sospeso a connesso.

Esegui il comando seguente per accettare la connessione all'account di ambiente:

```
$ aws proton accept-environment-account-connection \
    --id "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Risposta:

```
{
    "environmentAccountConnection": {
        "arn": "arn:aws:proton:region-id:222222222222:environment-account-
connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "environmentAccountId": "222222222222",
        "environmentName": "simple-env-connected",
        "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "lastModifiedAt": "2021-04-28T23:15:33.486000+00:00",
        "managementAccountId": "111111111111",
        "requestedAt": "2021-04-28T23:13:50.847000+00:00",
        "roleArn": "arn:aws:iam::222222222222:role/service-role/env-account-proton-
service-role",
        "status": "CONNECTED"
    }
}
```

Esegui il comando seguente per rifiutare la connessione all'account di ambiente:

```
$ aws proton reject-environment-account-connection \
    --id "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Risposta:

```
{
    "environmentAccountConnection": {
        "arn": "arn:aws:proton:us-east-1:222222222222:environment-account-
connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "status": "REJECTED",
        "environmentAccountId": "222222222222",
        "environmentName": "simple-env-reject",
        "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "lastModifiedAt": "2021-04-28T23:13:50.847000+00:00",
        "managementAccountId": "111111111111",
        "requestedAt": "2021-04-28T23:13:50.847000+00:00",
        "roleArn": "arn:aws:iam::222222222222:role/service-role/env-account-proton-
service-role"
    }
}
```

Visualizza le connessioni di un account di ambiente. È possibile ottenere o elencare le connessioni degli account di ambiente.

Esegui il seguente comando get:

```
$ aws proton get-environment-account-connection \
    --id "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Risposta:

```
{
    "environmentAccountConnection": {
        "arn": "arn:aws:proton:region-id:222222222222:environment-account-
connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "environmentAccountId": "222222222222",
        "environmentName": "simple-env-connected",
        "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "lastModifiedAt": "2021-04-28T23:15:33.486000+00:00",
        "managementAccountId": "111111111111",
        "requestedAt": "2021-04-28T23:13:50.847000+00:00",
        "roleArn": "arn:aws:iam::222222222222:role/service-role/env-account-proton-
service-role",
        "status": "CONNECTED"
    }
```

```
}
```

Elimina una connessione a un account di ambiente in un account di ambiente.

> **Note**
>
> Se si elimina questa connessione all'account di ambiente, AWS Proton non sarà possibile gestire le risorse dell'infrastruttura di ambiente nell'account di ambiente fino a quando non sarà stata accettata una nuova connessione di ambiente per l'account di ambiente e denominata ambiente. Sei responsabile della pulizia delle risorse assegnate che rimangono senza una connessione all'ambiente.

Esegui il comando seguente:

```
$ aws proton delete-environment-account-connection \
    --id "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Risposta:

```
{
    "environmentAccountConnection": {
        "arn": "arn:aws:proton:us-east-1:222222222222:environment-account-
connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "environmentAccountId": "222222222222",
        "environmentName": "simple-env-connected",
        "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "lastModifiedAt": "2021-04-28T23:13:50.847000+00:00",
        "managementAccountId": "111111111111",
        "requestedAt": "2021-04-28T23:13:50.847000+00:00",
        "roleArn": "arn:aws:iam::222222222222:role/service-role/env-account-proton-
service-role",
        "status": "CONNECTED"
    }
}
```

Ambienti gestiti dal cliente

Con gli ambienti gestiti dai clienti, puoi utilizzare l'infrastruttura esistente, come un VPC, che hai già implementato come ambiente. AWS Proton Durante l'utilizzo di ambienti gestiti dal cliente, è possibile fornire le proprie risorse condivise all'esterno di. AWS Proton Tuttavia, è comunque possibile consentire di AWS Proton utilizzare gli output di provisioning pertinenti come input per i AWS Proton servizi una volta implementati. Se gli output possono cambiare, AWS Proton è in grado di accettare gli aggiornamenti. AWS Proton non è in grado di modificare direttamente l'ambiente, tuttavia, poiché il provisioning è gestito all'esterno di. AWS Proton

Dopo aver creato l'ambiente, sei responsabile di fornire gli stessi output AWS Proton che sarebbero stati creati se AWS Proton avessi creato l'ambiente, ad esempio i nomi dei cluster Amazon ECS o Amazon VPC. IDs

Con questa funzionalità, puoi distribuire e aggiornare le risorse di AWS Proton servizio da un modello di servizio a questo AWS Proton ambiente. Tuttavia, l'ambiente stesso non viene modificato tramite gli aggiornamenti del modello in AWS Proton. Sei responsabile dell'esecuzione degli aggiornamenti all'ambiente e dell'aggiornamento di tali output in AWS Proton.

Puoi avere più ambienti in un unico account, costituiti da un mix di ambienti AWS Proton gestiti e gestiti dal cliente. Puoi anche collegare un secondo account e utilizzare un AWS Proton modello nell'account principale per eseguire distribuzioni e aggiornamenti di ambienti e servizi in quel secondo account collegato.

Come utilizzare gli ambienti gestiti dal cliente

La prima cosa che gli amministratori devono fare è registrare un modello di ambiente importato e gestito dal cliente. Non fornire manifesti o file di infrastruttura nel pacchetto di modelli. Fornisci solo lo schema.

Lo schema seguente delinea un elenco di output utilizzando il formato API aperto e replica gli output di un modello. AWS CloudFormation

> **Important**
>
> Per gli output sono consentiti solo input di stringhe.

L'esempio seguente è un frammento delle sezioni di output di un AWS CloudFormation modello per un modello Fargate corrispondente.

```
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref 'ECSCluster'
  ECSTaskExecutionRole:
    Description: The ARN of the ECS role
    Value: !GetAtt 'ECSTaskExecutionRole.Arn'
  VpcId:
    Description: The ID of the VPC that this stack is deployed in
    Value: !Ref 'VPC'
[...]
```

Lo schema per l'ambiente AWS Proton importato corrispondente è simile al seguente. Non fornire valori predefiniti nello schema.

```
schema:
  format:
    openapi: "3.0.0"
  environment_input_type: "EnvironmentOutput"
  types:
    EnvironmentOutput:
      type: object
      description: "Outputs of the environment"
      properties:
        ClusterName:
          type: string
          description: "The name of the ECS cluster"
        ECSTaskExecutionRole:
          type: string
          description: "The ARN of the ECS role"
        VpcId:
          type: string
          description: "The ID of the VPC that this stack is deployed in"
[...]
```

Al momento della registrazione del modello, indichi che questo modello è importato e fornisce la posizione del bucket Amazon S3 per il pacchetto. AWS Proton verifica che lo schema contenga solo parametri del modello `environment_input_type` e non contenga alcun parametro prima di inserire il AWS CloudFormation modello in bozza.

Fornite quanto segue per creare un ambiente importato.

- Un ruolo IAM da utilizzare durante le distribuzioni.
- Una specifica con i valori per gli output richiesti.

È possibile fornire entrambi tramite la console o AWS CLI utilizzando un processo simile alla distribuzione di un ambiente normale.

CodeBuild creazione di ruoli di provisioning

Gli strumenti Infrastructure as a Code (IaaC) come AWS CloudFormation Terraform richiedono autorizzazioni per i diversi tipi di risorse. AWS Ad esempio, se un modello IaaC dichiara un bucket Amazon S3, necessita delle autorizzazioni per creare, leggere, aggiornare ed eliminare i bucket Amazon S3. Limitare i ruoli alle autorizzazioni minime richieste è considerata una best practice di sicurezza. Data l'ampiezza delle AWS risorse, è difficile creare politiche con privilegi minimi per i modelli IaaC, soprattutto quando le risorse gestite da tali modelli possono cambiare in seguito. Ad esempio, nelle ultime modifiche apportate a un modello gestito da AWS Proton, si aggiunge una risorsa del database RDS.

La configurazione delle autorizzazioni corrette aiuta a semplificare le implementazioni del tuo IAc. AWS Proton CodeBuild Il provisioning esegue comandi CLI arbitrari forniti dal cliente in un CodeBuild progetto situato nell'account del cliente. In genere, questi comandi creano ed eliminano l'infrastruttura utilizzando uno strumento Infrastructure as Code (IaaC) come. AWS CDK Quando una AWS risorsa distribuisce il cui modello utilizza CodeBuild Provisioning, AWS avvia una compilazione in un CodeBuild progetto gestito da. AWS Viene passato un ruolo a CodeBuild, che CodeBuild presuppone l'esecuzione di comandi. Questo ruolo, denominato CodeBuild Provisioning Role, viene fornito dal cliente e contiene le autorizzazioni necessarie per il provisioning dell'infrastruttura. È pensato per essere assunto solo da CodeBuild e nemmeno AWS Proton può assumerlo.

Creazione del ruolo

Il ruolo CodeBuild Provisioning può essere creato nella console IAM o in. AWS CLI Per crearlo in: AWS CLI

```
aws iam create-role --role-name AWSProtonCodeBuildProvisioning --assume-role-
policy-document '{"Version":"2012-10-17","Statement":[{"Effect":"Allow","Principal":
{"Service":"codebuild.amazonaws.com"},"Action":"sts:AssumeRole"}]}'
aws iam attach-role-policy --role-name AWSProtonCodeBuildProvisioning --policy-arn
 arn:aws:iam::aws:policy/AWSProtonCodeBuildProvisioningBasicAccess
```

Questo allega anche il`AWSProtonCodeBuildProvisioningBasicAccess`, che contiene le autorizzazioni minime necessarie al CodeBuild servizio per eseguire una build.

Se preferisci usare la console, assicurati di quanto segue quando crei il ruolo:

1. Per entità attendibile, seleziona AWS servizio, quindi seleziona CodeBuild.
2. Nel passaggio Aggiungi autorizzazioni, seleziona `AWSProtonCodeBuildProvisioningBasicAccess` e tutte le altre politiche che desideri allegare.

Accesso amministratore

Se alleghi la `AdministratorAccess` policy al CodeBuild Provisioning Role, garantirai che qualsiasi modello IaaC non fallirà a causa della mancanza di autorizzazioni. Significa anche che chiunque sia in grado di creare un modello di ambiente o un modello di servizio può eseguire azioni a livello di amministratore, anche se l'utente non è un amministratore. AWS Proton non ne consiglia l'utilizzo `AdministatorAccess` con il Provisioning Role CodeBuild . Se decidi di utilizzarlo `AdministratorAccess` con il CodeBuild Provisioning Role, fallo in un ambiente sandbox.

Puoi creare un ruolo `AdministratorAccess` nella console IAM o eseguendo questo comando:

```
aws iam create-role --role-name AWSProtonCodeBuildProvisioning --assume-role-
policy-document '{"Version":"2012-10-17","Statement":[{"Effect":"Allow","Principal":
{"Service":"codebuild.amazonaws.com"},"Action":"sts:AssumeRole"}]}'
aws iam attach-role-policy --role-name AWSProtonCodeBuildProvisioning --policy-arn
 arn:aws:iam::aws:policy/AdministratorAccess
```

Creazione di un ruolo con ambito minimo

Se desideri creare un ruolo con autorizzazioni minime, esistono diversi approcci:

- Esegui la distribuzione con autorizzazioni di amministratore, quindi riduci l'ambito del ruolo. Ti consigliamo di utilizzare IAM Access Analyzer.
- Utilizza le policy gestite per consentire l'accesso ai servizi che intendi utilizzare.

AWS CDK

Se utilizzi AWS CDK with e AWS Proton l'hai eseguito `cdk bootstrap` su ogni account/area di ambiente, esiste già un ruolo per. `cdk deploy` In questo caso, allega la seguente politica al ruolo di CodeBuild provisioning:

```
{
    "Action": "sts:AssumeRole",
    "Resource": [
        "arn:aws:iam::account-id:role/cdk-*-deploy-role-*",
        "arn:aws:iam::account-id:role/cdk-*-file-publishing-role-*"
    ],
    "Effect": "Allow"
}
```

VPC personalizzato

Se decidi di eseguire l'esecuzione CodeBuild in un VPC personalizzato, avrai bisogno delle seguenti autorizzazioni per il tuo ruolo: CodeBuild

```
{
    "Effect": "Allow",
    "Action": [
        "ec2:CreateNetworkInterface"
    ],
    "Resource": [
        "arn:aws:ec2:region:account-id:network-interface/*",
        "arn:aws:ec2:region:account-id:subnet/*",
        "arn:aws:ec2:region:account-id:security-group/*"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "ec2:DeleteNetworkInterface"
    ],
    "Resource": [
        "arn:aws:ec2:region:account-id:*/*"
    ]
},
{
    "Effect": "Allow",
    "Action": [
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeNetworkInterfaces",
```

```
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeVpcs"
    ],
    "Resource": "*"
},
{
    "Effect": "Allow",
    "Action": [
        "ec2:CreateNetworkInterfacePermission"
    ],
    "Resource": "arn:aws:ec2:region:account-id:network-interface/*",
    "Condition": {
        "StringEquals": {
            "ec2:AuthorizedService": "codebuild.amazonaws.com"
        }
    }
}
```

Puoi anche utilizzare la politica `AmazonEC2FullAccess` gestita, sebbene includa autorizzazioni che potrebbero non essere necessarie. Per allegare la policy gestita utilizzando la CLI:

```
aws iam create-role --role-name AWSProtonCodeBuildProvisioning --assume-role-
policy-document '{"Version":"2012-10-17","Statement":[{"Effect":"Allow","Principal":
{"Service":"codebuild.amazonaws.com"},"Action":"sts:AssumeRole"}]}'
aws iam attach-role-policy --role-name AWSProtonCodeBuildProvisioning --policy-arn
 arn:aws:iam::aws:policy/AdministratorAccess
```

AWS Proton servizi

Un AWS Proton servizio è un'istanza di un modello di servizio, che normalmente include diverse istanze di servizio e una pipeline. Un'istanza AWS Proton di servizio è un'istanza di un modello di servizio in un ambiente specifico. Un modello di servizio è una definizione completa dell'infrastruttura e della pipeline di servizi opzionale per un servizio. AWS Proton

Dopo aver distribuito le istanze del servizio, è possibile aggiornarle inserendo il codice sorgente nella pipeline CI/CD oppure aggiornando il servizio alle nuove versioni del relativo modello di servizio. AWS Proton ti avvisa quando diventano disponibili nuove versioni del relativo modello di servizio in modo da poter aggiornare i servizi a una nuova versione. Quando il servizio viene aggiornato, AWS Proton ridistribuisce il servizio e le istanze del servizio.

Questo capitolo mostra come gestire i servizi utilizzando le operazioni di creazione, visualizzazione, aggiornamento ed eliminazione. Per ulteriori informazioni, consulta The AWS Proton Service API Reference.

Argomenti

- Creazione di un servizio
- Visualizza i dati del servizio
- Modifica un servizio
- Eliminazione di un servizio
- Visualizza i dati dell'istanza del servizio
- Aggiornare un'istanza di servizio
- Aggiorna una pipeline di servizi

Creazione di un servizio

Per distribuire un'applicazione con AWS Proton, in qualità di sviluppatore, devi creare un servizio e fornire i seguenti input.

1. Il nome di un modello di AWS Proton servizio pubblicato dal team della piattaforma.
2. Un nome per il servizio.
3. Il numero di istanze del servizio che si desidera distribuire.

4. Una selezione di ambienti che desideri utilizzare.
5. Una connessione al tuo repository di codice se utilizzi un modello di servizio che include una pipeline di servizi (opzionale).

Cosa c'è in un servizio?

Quando crei un AWS Proton servizio, puoi scegliere tra due diversi tipi di modelli di servizio:

- Un modello di servizio che include una pipeline di servizi (impostazione predefinita).
- Un modello di servizio che non include una pipeline di servizi.

È necessario creare almeno un'istanza del servizio quando si crea il servizio.

Un'istanza di servizio e una pipeline opzionale sono associate a un servizio. È possibile creare o eliminare una pipeline solo nel contesto delle azioni di creazione ed eliminazione del servizio. Per informazioni su come aggiungere e rimuovere istanze da un servizio, consulta. Modifica un servizio

> **Note**
>
> L'ambiente è configurato per il provisioning AWS o per la gestione automatica. AWS Proton fornisce servizi in un ambiente utilizzando lo stesso metodo di provisioning utilizzato dall'ambiente. Lo sviluppatore che crea o aggiorna le istanze del servizio non vede la differenza e la sua esperienza è la stessa in entrambi i casi.
> Per ulteriori informazioni sui metodi di provisioning, consulta. the section called “Metodi di approvvigionamento”

Modelli di servizio

Sono disponibili versioni principali e secondarie dei modelli di servizio. Quando si utilizza la console, si seleziona la versione `Recommended` principale e secondaria più recente del modello di servizio. Quando si utilizza AWS CLI e si specifica solo la versione principale del modello di servizio, si specifica implicitamente la versione `Recommended` secondaria più recente.

Di seguito viene descritta la differenza tra le versioni principali e secondarie del modello e il relativo utilizzo.

- Le nuove versioni di un modello diventano disponibili non `Recommended` appena vengono approvate da un membro del team della piattaforma. Ciò significa che i nuovi servizi vengono creati utilizzando quella versione e ti viene richiesto di aggiornare i servizi esistenti alla nuova versione.
- Tramite AWS Proton, il team della piattaforma può aggiornare automaticamente le istanze del servizio a una nuova versione secondaria di un modello di servizio. Le versioni secondarie devono essere compatibili con le versioni precedenti.
- Poiché le versioni principali richiedono l'immissione di nuovi input come parte del processo di aggiornamento, è necessario aggiornare il servizio a una versione principale del relativo modello di servizio. Le versioni principali non sono retrocompatibili.

Creazione di un servizio

Le procedure seguenti mostrano come utilizzare la AWS Proton console o AWS CLI creare un servizio con o senza una pipeline di servizi.

AWS Management Console

Crea un servizio come illustrato nei seguenti passaggi della console.

1. Nella AWS Proton console, scegli Servizi.
2. Selezionare Create service (Crea servizio).
3. Nella pagina Scegli un modello di servizio, seleziona un modello e scegli Configura.

   Se non desideri utilizzare una pipeline abilitata, scegli un modello contrassegnato con Esclude la pipeline per il tuo servizio.
4. Nella pagina Configura servizio, nella sezione Impostazioni del servizio, inserisci un nome di servizio.
5. (Facoltativo) Inserisci una descrizione del servizio.
6. Nella sezione Impostazioni del repository del servizio:

   a. Per CodeStar Connessione, scegli la tua connessione dall'elenco.

   b. Per Repository ID, scegli il nome del tuo repository di codice sorgente dall'elenco.

   c. Per Branch name, scegli il nome del ramo del repository del codice sorgente dall'elenco.
7. (Facoltativo) Nella sezione Tag, scegli Aggiungi nuovo tag e inserisci una chiave e un valore per creare un tag gestito dal cliente.
8. Scegli Next (Successivo).

9. Nella pagina Configura impostazioni personalizzate, nella sezione Istanze di servizio, nella sezione Nuova istanza. È necessario inserire valori per i `required` parametri. È possibile inserire valori per i `optional` parametri o utilizzare i valori predefiniti, se disponibili.
10. Nella sezione Pipeline input, è necessario inserire i valori per i parametri. `required` È possibile inserire valori per i `optional` parametri o utilizzare i valori predefiniti, se disponibili.
11. Scegliete Avanti e controllate i dati inseriti.
12. Scegli Create (Crea).

    Visualizza i dettagli e lo stato del servizio, nonché i tag AWS gestiti e i tag gestiti dai clienti per il tuo servizio.
13. Nel riquadro di navigazione, scegli Servizi.

    Una nuova pagina mostra un elenco dei tuoi servizi insieme allo stato e ad altri dettagli del servizio.

AWS CLI

Quando si utilizza AWS CLI, si specificano gli input del servizio in un `spec` file in formato YAML`.aws-proton/service.yaml`, che si trova nella directory del codice sorgente.

È possibile utilizzare il `get-service-template-minor-version` comando CLI per visualizzare i parametri obbligatori e facoltativi dello schema per i quali si forniscono i valori nel file delle specifiche.

Se desideri utilizzare un modello di servizio che lo abbia`pipelineProvisioning: "CUSTOMER_MANAGED"`, non includi la `pipeline:` sezione nelle specifiche e non `-repository-connection-arn` includa né `-branch-name` i parametri nel comando. `-repository-id` `create-service`

Crea un servizio con una pipeline di servizi come illustrato nei seguenti passaggi della CLI.

1. Imposta il ruolo di servizio per la pipeline come mostrato nel seguente comando di esempio CLI.

   Comando:

   ```
   $ aws proton update-account-settings \
   ```

```
        --pipeline-service-role-arn "arn:aws:iam::123456789012:role/AWS
 ProtonServiceRole"
```

2. L'elenco seguente mostra una specifica di esempio, basata sullo schema del modello di servizio, che include la pipeline di servizio e gli input dell'istanza.

   Specifiche:

```
proton: ServiceSpec

pipeline:
  my_sample_pipeline_required_input: "hello"
  my_sample_pipeline_optional_input: "bye"

instances:
  - name: "acme-network-dev"
    environment: "ENV_NAME"
    spec:
      my_sample_service_instance_required_input: "hi"
      my_sample_service_instance_optional_input: "ho"
```

   Crea un servizio con una pipeline come mostrato nel seguente esempio di comando e risposta CLI.

   Comando:

```
$ aws proton create-service \
        --name "MySimpleService" \
        --branch-name "mainline" \
        --template-major-version "1" \
        --template-name "fargate-service" \
        --repository-connection-arn "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" \
        --repository-id "myorg/myapp" \
        --spec "file://spec.yaml"
```

   Risposta:

```
{
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService",
        "createdAt": "2020-11-18T19:50:27.460000+00:00",
```

```
        "lastModifiedAt": "2020-11-18T19:50:27.460000+00:00",
        "name": "MySimpleService",
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "repositoryId": "myorg/myapp",
        "status": "CREATE_IN_PROGRESS",
        "templateName": "fargate-service"
    }
}
```

Crea un servizio senza una pipeline di servizi come mostrato nel seguente esempio di comando e risposta CLI.

Di seguito viene illustrata una specifica di esempio che non include gli input della pipeline di servizio.

Specifiche:

```
proton: ServiceSpec

instances:
  - name: "acme-network-dev"
    environment: "ENV_NAME"
    spec:
      my_sample_service_instance_required_input: "hi"
      my_sample_service_instance_optional_input: "ho"
```

Per creare un servizio senza una pipeline di servizi fornita, fornisci il percorso a **`spec.yaml`** e non includi i parametri del repository, come mostrato nel seguente esempio di comando e risposta della CLI.

Comando:

```
$ aws proton create-service \
        --name "MySimpleServiceNoPipeline" \
        --template-major-version "1" \
        --template-name "fargate-service" \
        --spec "file://spec-no-pipeline.yaml"
```

Risposta:

```
{
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/
MySimpleServiceNoPipeline",
        "createdAt": "2020-11-18T19:50:27.460000+00:00",
        "lastModifiedAt": "2020-11-18T19:50:27.460000+00:00",
        "name": "MySimpleServiceNoPipeline",
        "status": "CREATE_IN_PROGRESS",
        "templateName": "fargate-service-no-pipeline"
    }
}
```

Visualizza i dati del servizio

È possibile visualizzare ed elencare i dati di dettaglio del servizio utilizzando la AWS Proton console o il AWS CLI.

AWS Management Console

Elencare e visualizzare i dettagli del servizio utilizzando la AWS Proton console, come illustrato nei passaggi seguenti.

1. Per visualizzare un elenco dei tuoi servizi, scegli Servizi nel riquadro di navigazione.
2. Per visualizzare i dati di dettaglio, scegli il nome di un servizio.

   Visualizza i dati di dettaglio del servizio.

AWS CLI

Visualizza i dettagli di un servizio con una pipeline di servizi come mostrato nel seguente esempio di comando e risposta CLI.

Comando:

```
$ aws proton get-service \
    --name "simple-svc"
```

Risposta:

```
{
```

```
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc",
        "branchName": "mainline",
        "createdAt": "2020-11-28T22:40:50.512000+00:00",
        "lastModifiedAt": "2020-11-28T22:44:51.207000+00:00",
        "name": "simple-svc",
        "pipeline": {
            "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/
pipeline/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "createdAt": "2020-11-28T22:40:50.512000+00:00",
            "deploymentStatus": "SUCCEEDED",
            "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",
            "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",
            "spec": "proton: ServiceSpec\npipeline:\n
 my_sample_pipeline_required_input: hello\n  my_sample_pipeline_optional_input:
 bye\ninstances:\n- name: instance-svc-simple\n  environment: my-simple-
env\n  spec:\n    my_sample_service_instance_required_input: hi\n
 my_sample_service_instance_optional_input: ho\n",
            "templateMajorVersion": "1",
            "templateMinorVersion": "1",
            "templateName": "svc-simple"
        },
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
        "repositoryId": "myorg/myapp",
        "spec": "proton: ServiceSpec\npipeline:\n
 my_sample_pipeline_required_input: hello\n  my_sample_pipeline_optional_input:
 bye\ninstances:\n- name: instance-svc-simple\n  environment: my-simple-
env\n  spec:\n    my_sample_service_instance_required_input: hi\n
 my_sample_service_instance_optional_input: ho\n",
        "status": "ACTIVE",
        "templateName": "svc-simple"
    }
}
```


Visualizza i dettagli di un servizio senza una pipeline di servizi, come mostrato nel seguente esempio di comando e risposta CLI.

Comando:

```
$ aws proton get-service \
    --name "simple-svc-no-pipeline"
```

Risposta:

```
{
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc-without-
pipeline",
        "createdAt": "2020-11-28T22:40:50.512000+00:00",
        "lastModifiedAt": "2020-11-28T22:44:51.207000+00:00",
        "name": "simple-svc-without-pipeline",
        "spec": "proton: ServiceSpec\ninstances:\n- name: instance-svc-simple\n
 environment: my-simple-env\n  spec:\n    my_sample_service_instance_required_input:
 hi\n    my_sample_service_instance_optional_input: ho\n",
        "status": "ACTIVE",
        "templateName": "svc-simple-no-pipeline"
    }
}
```

Modifica un servizio

È possibile apportare le seguenti modifiche a un AWS Proton servizio.

- Modifica la descrizione del servizio.
- Modifica un servizio aggiungendo e rimuovendo istanze del servizio.

Modifica la descrizione del servizio

È possibile utilizzare la console o AWS CLI modificare la descrizione di un servizio.

AWS Management Console

Modifica un servizio utilizzando la console come descritto nei passaggi seguenti.

Nell'elenco dei servizi.

1. Nella AWS Proton console, scegli Servizi.
2. Nell'elenco dei servizi, scegli il pulsante di opzione a sinistra del servizio che desideri aggiornare.
3. Scegli Modifica.
4. Nella pagina Configura servizio, compila il modulo e scegli Avanti.

5. Nella pagina Configura impostazioni personalizzate, scegli Avanti.
6. Controlla le modifiche e scegli Salva modifiche.

Nella pagina dei dettagli del servizio.

1. Nella AWS Proton console, scegli Servizi.
2. Nell'elenco dei servizi, scegli il nome del servizio che desideri modificare.
3. Nella pagina dei dettagli del servizio, scegli Modifica.
4. Nella pagina Configura servizio, compila il modulo e scegli Avanti.
5. Nella pagina Configura impostazioni personalizzate, compila il modulo e scegli Avanti.
6. Controlla le modifiche e scegli Salva modifiche.

AWS CLI

Modifica una descrizione come mostrato nel seguente esempio di comando e risposta CLI.

Comando:

```
$ aws proton update-service \
    --name "MySimpleService" \
    --description "Edit by updating description"
```

Risposta:

```
{
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService",
        "branchName": "main",
        "createdAt": "2021-03-12T22:39:42.318000+00:00",
        "description": "Edit by updating description",
        "lastModifiedAt": "2021-03-12T22:44:21.975000+00:00",
        "name": "MySimpleService",
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "repositoryId": "my-repository/myorg-myapp",
        "status": "ACTIVE",
        "templateName": "fargate-service"
    }
```

```
}
```

Modifica un servizio per aggiungere o rimuovere istanze del servizio

Per un AWS Proton servizio, è possibile aggiungere o eliminare istanze di servizio inviando una specifica modificata. Affinché la richiesta abbia esito positivo, devono essere soddisfatte le seguenti condizioni:

- Il servizio e la pipeline non sono già stati modificati o eliminati quando invii la richiesta di modifica.
- Le specifiche modificate non includono modifiche che modificano la pipeline del servizio o modifiche alle istanze di servizio esistenti che non devono essere eliminate.
- La specifica modificata non rimuove alcuna istanza di servizio esistente a cui è collegato un componente. Per eliminare un'istanza di servizio di questo tipo, è necessario prima aggiornare il componente per scollegarlo dalla relativa istanza di servizio. Per ulteriori informazioni sui componenti, vedereComponenti.

Le istanze con eliminazione non riuscita sono istanze di servizio nello stato. `DELETE_FAILED` Quando richiedi una modifica al servizio, AWS Proton tenta di rimuovere al posto tuo le istanze che non sono riuscite a eliminare, come parte del processo di modifica. Se una delle istanze del servizio non è stata eliminata, potrebbero esserci ancora risorse associate alle istanze, anche se non sono visibili dalla console o. AWS CLI Controlla le risorse dell'infrastruttura dell'istanza che non è riuscita a eliminare e puliscile in modo che AWS Proton possa rimuoverle per conto tuo.

Per la quota di istanze di servizio per un servizio, consulta. AWS Proton quote È inoltre necessario mantenere almeno un'istanza di servizio per il servizio dopo la sua creazione. Durante il processo di aggiornamento, AWS Proton effettua un conteggio delle istanze del servizio esistenti e delle istanze da aggiungere o rimuovere. Le istanze con eliminazione non riuscita sono incluse in questo conteggio e devi tenerne conto quando modifichi le tue. `spec`

Utilizza la console o AWS CLI per aggiungere o rimuovere istanze di servizio

AWS Management Console

Modifica il servizio per aggiungere o rimuovere istanze di servizio utilizzando la console.

Nella console AWS Proton

1. Nel riquadro di navigazione, scegli Servizi.

2. Seleziona il servizio che desideri modificare.
3. Scegli Modifica.
4. (Facoltativo) Nella pagina Configura servizio, modifica il nome o la descrizione del servizio, quindi scegli Avanti.
5. Nella pagina Configura impostazioni personalizzate, scegli Elimina per eliminare un'istanza del servizio e scegli Aggiungi nuova istanza per aggiungere un'istanza del servizio e compila il modulo.
6. Scegli Next (Successivo).
7. Controlla l'aggiornamento e scegli Salva modifiche.
8. Una modalità ti chiede di verificare l'eliminazione delle istanze del servizio. Segui le istruzioni e scegli Sì, elimina.
9. Nella pagina dei dettagli del servizio, visualizza i dettagli sullo stato del servizio.

AWS CLI

Aggiungi ed elimina le istanze del servizio con una modifica**`spec`**, come mostrato nei seguenti comandi e risposte di AWS CLI esempio.

Quando si utilizza la CLI, è `spec` necessario escludere le istanze del servizio da eliminare e includere sia le istanze del servizio da aggiungere sia le istanze del servizio esistenti che non sono state contrassegnate per l'eliminazione.

L'elenco seguente mostra l'esempio `spec` prima della modifica e un elenco delle istanze del servizio distribuite dalla specifica. Questa specifica è stata utilizzata nell'esempio precedente per modificare la descrizione di un servizio.

Specifiche:

```
proton: ServiceSpec

pipeline:
  my_sample_pipeline_optional_input: "abc"
  my_sample_pipeline_required_input: "123"

instances:
  - name: "my-instance"
    environment: "simple-env"
```

```
    spec:
      my_sample_service_instance_optional_input: "def"
      my_sample_service_instance_required_input: "456"
  - name: "my-other-instance"
    environment: "simple-env"
    spec:
      my_sample_service_instance_required_input: "789"
```

Il `list-service-instances` comando e la risposta CLI di esempio seguenti mostrano le istanze attive prima di aggiungere o eliminare un'istanza di servizio.

Comando:

```
$ aws proton list-service-instances \
    --service-name "MySimpleService"
```

Risposta:

```
{
    "serviceInstances": [
        {
            "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService/
service-instance/my-other-instance",
            "createdAt": "2021-03-12T22:39:42.318000+00:00",
            "deploymentStatus": "SUCCEEDED",
            "environmentName": "simple-env",
            "lastDeploymentAttemptedAt": "2021-03-12T22:39:43.109000+00:00",
            "lastDeploymentSucceededAt": "2021-03-12T22:39:43.109000+00:00",
            "name": "my-other-instance",
            "serviceName": "example-svc",
            "templateMajorVersion": "1",
            "templateMinorVersion": "0",
            "templateName": "fargate-service"
        },
        {
            "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService/
service-instance/my-instance",
            "createdAt": "2021-03-12T22:39:42.318000+00:00",
            "deploymentStatus": "SUCCEEDED",
            "environmentName": "simple-env",
            "lastDeploymentAttemptedAt": "2021-03-12T22:39:43.160000+00:00",
            "lastDeploymentSucceededAt": "2021-03-12T22:39:43.160000+00:00",
```

```
            "name": "my-instance",
            "serviceName": "example-svc",
            "serviceTemplateArn": "arn:aws:proton:region-id:123456789012:service-
template/fargate-service",
            "templateMajorVersion": "1",
            "templateMinorVersion": "0",
            "templateName": "fargate-service"
        }
    ]
}
```

L'elenco seguente mostra l'esempio modificato `spec` utilizzato per eliminare e aggiungere un'istanza. L'istanza esistente denominata `my-instance` viene rimossa e `yet-another-instance` viene aggiunta una nuova istanza denominata.

Specifiche:

```
proton: ServiceSpec

pipeline:
  my_sample_pipeline_optional_input: "abc"
  my_sample_pipeline_required_input: "123"

instances:
  - name: "my-other-instance"
    environment: "simple-env"
    spec:
      my_sample_service_instance_required_input: "789"
  - name: "yet-another-instance"
    environment: "simple-env"
    spec:
      my_sample_service_instance_required_input: "789"
```

È possibile utilizzare `"${Proton::CURRENT_VAL}"` per indicare i valori dei parametri da preservare rispetto all'originale`spec`, se i valori esistono in. `spec` `get-service`Da utilizzare per visualizzare l'originale `spec` di un servizio, come descritto inVisualizza i dati del servizio.

L'elenco seguente mostra come è possibile assicurarsi che `spec` non vengano incluse le modifiche `"${Proton::CURRENT_VAL}"` ai valori dei parametri affinché le istanze dei servizi esistenti rimangano invariate.

Specifiche:

```
proton: ServiceSpec

pipeline:
  my_sample_pipeline_optional_input: "${Proton::CURRENT_VAL}"
  my_sample_pipeline_required_input: "${Proton::CURRENT_VAL}"

instances:
  - name: "my-other-instance"
    environment: "simple-env"
    spec:
      my_sample_service_instance_required_input: "${Proton::CURRENT_VAL}"
  - name: "yet-another-instance"
    environment: "simple-env"
    spec:
      my_sample_service_instance_required_input: "789"
```

L'elenco successivo mostra il comando e la risposta CLI per modificare il servizio.

Comando:

```
$ aws proton update-service
    --name "MySimpleService" \
    --description "Edit by adding and deleting a service instance" \
    --spec "file://spec.yaml"
```

Risposta:

```
{
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService",
        "branchName": "main",
        "createdAt": "2021-03-12T22:39:42.318000+00:00",
        "description": "Edit by adding and deleting a service instance",
        "lastModifiedAt": "2021-03-12T22:55:48.169000+00:00",
        "name": "MySimpleService",
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "repositoryId": "my-repository/myorg-myapp",
        "status": "UPDATE_IN_PROGRESS",
        "templateName": "fargate-service"
    }
}
```

Il `list-service-instances` comando e la risposta seguenti confermano che l'istanza esistente denominata `my-instance` viene rimossa e `yet-another-instance` viene aggiunta una nuova istanza denominata.

Comando:

```
$ aws proton list-service-instances \
    --service-name "MySimpleService"
```

Risposta:

```
{
    "serviceInstances": [
        {
            "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService/
service-instance/yet-another-instance",
            "createdAt": "2021-03-12T22:39:42.318000+00:00",
            "deploymentStatus": "SUCCEEDED",
            "environmentName": "simple-env",
            "lastDeploymentAttemptedAt": "2021-03-12T22:56:01.565000+00:00",
            "lastDeploymentSucceededAt": "2021-03-12T22:56:01.565000+00:00",
            "name": "yet-another-instance",
            "serviceName": "MySimpleService",
            "templateMajorVersion": "1",
            "templateMinorVersion": "0",
            "templateName": "fargate-service"
        },
        {
            "arn": "arn:aws:proton:region-id:123456789012:service/MySimpleService/
service-instance/my-other-instance",
            "createdAt": "2021-03-12T22:39:42.318000+00:00",
            "deploymentStatus": "SUCCEEDED",
            "environmentName": "simple-env",
            "lastDeploymentAttemptedAt": "2021-03-12T22:39:43.109000+00:00",
            "lastDeploymentSucceededAt": "2021-03-12T22:39:43.109000+00:00",
            "name": "my-other-instance",
            "serviceName": "MySimpleService",
            "templateMajorVersion": "1",
            "templateMinorVersion": "0",
            "templateName": "fargate-service"
        }
    ]
```

```
}
```

Cosa succede quando si aggiungono o rimuovono istanze di servizio

Dopo aver inviato una modifica al servizio per eliminare e aggiungere istanze del servizio, AWS Proton esegue le seguenti azioni.

- Imposta il servizio su. `UPDATE_IN_PROGRESS`
- Se il servizio ha una pipeline, ne imposta lo stato su `IN_PROGRESS` e blocca le azioni della pipeline.
- Imposta tutte le istanze del servizio su cui eliminare. `DELETE_IN_PROGRESS`
- Blocca le azioni di servizio.
- Blocca le azioni sulle istanze di servizio contrassegnate per l'eliminazione.
- Crea nuove istanze di servizio.
- Elimina le istanze elencate per l'eliminazione.
- Tentativi di rimozione delle istanze non riusciti.
- Una volta completate le aggiunte e le eliminazioni, riorganizza la pipeline del servizio (se presente), imposta il servizio e abilita le azioni di servizio e pipeline. `ACTIVE`

AWS Proton tenta di rimediare alle modalità di errore come segue.

- Se una o più istanze di servizio non vengono create, AWS Proton tenta di annullare il provisioning di tutte le istanze di servizio appena create e ripristina lo stato precedente. `spec` Non elimina alcuna istanza di servizio e non modifica in alcun modo la pipeline.
- Se una o più istanze del servizio non vengono eliminate, AWS Proton riorganizza il provisioning della pipeline senza le istanze eliminate. `spec`Viene aggiornato per includere le istanze aggiunte e per escludere le istanze contrassegnate per l'eliminazione.
- Se la pipeline non riesce a eseguire il provisioning, non viene tentato un rollback e sia il servizio che la pipeline riflettono uno stato di aggiornamento non riuscito.

Etichettatura e modifiche al servizio

Quando si aggiungono istanze di servizio come parte della modifica del servizio, i tag AWS gestiti si propagano e vengono creati automaticamente per le nuove istanze e le risorse assegnate. Se

crei nuovi tag, questi tag vengono applicati solo alle nuove istanze. I tag esistenti gestiti dai clienti del servizio si propagano anche alle nuove istanze. Per ulteriori informazioni, consulta AWS Proton risorse e etichettatura.

Eliminazione di un servizio

È possibile eliminare un AWS Proton servizio, con le relative istanze e pipeline, utilizzando la console o il AWS Proton . AWS CLI

Non è possibile eliminare un servizio a cui è associata un'istanza di servizio con un componente collegato. Per eliminare un servizio di questo tipo, è necessario innanzitutto aggiornare tutti i componenti collegati per scollegarli dalle relative istanze del servizio. Per ulteriori informazioni sui componenti, vedere. Componenti

AWS Management Console

Eliminare un servizio utilizzando la console come descritto nei passaggi seguenti.

Nella pagina dei dettagli del servizio.

1. Nella AWS Proton console, scegli Servizi.
2. Nell'elenco dei servizi, scegli il nome del servizio che desideri eliminare.
3. Nella pagina dei dettagli del servizio, scegli Azioni, quindi Elimina.
4. Una finestra modale richiede di confermare l'azione di eliminazione.
5. Segui le istruzioni e scegli Sì, elimina.

AWS CLI

Eliminare un servizio come illustrato nel seguente esempio di comando e risposta CLI.

Comando:

```
$ aws proton delete-service \
    --name "simple-svc"
```

Risposta:

```
{
    "service": {
```

```
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc",
        "branchName": "mainline",
        "createdAt": "2020-11-28T22:40:50.512000+00:00",
        "description": "Edit by updating description",
        "lastModifiedAt": "2020-11-29T00:30:39.248000+00:00",
        "name": "simple-svc",
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "repositoryId": "myorg/myapp",
        "status": "DELETE_IN_PROGRESS",
        "templateName": "fargate-service"
    }
}
```

Visualizza i dati dell'istanza del servizio

Scopri come visualizzare i dati di dettaglio delle istanze di AWS Proton servizio. Puoi utilizzare la console o il AWS CLI.

Un'istanza del servizio appartiene a un servizio. È possibile creare o eliminare un'istanza solo nell'ambito delle azioni di modifica, creazione ed eliminazione del servizio. Per informazioni su come aggiungere e rimuovere istanze da un servizio, consultaModifica un servizio.

AWS Management Console

Elenca e visualizza i dettagli delle istanze di servizio utilizzando la AWS Proton console, come illustrato nei passaggi seguenti.

1. Per visualizzare un elenco delle istanze del servizio, scegli Istanze di servizi nel riquadro di navigazione.
2. Per visualizzare i dati di dettaglio, scegli il nome di un'istanza del servizio.

   Visualizza i dati di dettaglio dell'istanza di servizio.

AWS CLI

Elenca e visualizza i dettagli dell'istanza del servizio come mostrato nei seguenti comandi e risposte di esempio CLI.

Comando:

```
$ aws proton list-service-instances
```

Risposta:

```
{
    "serviceInstances": [
        {
            "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/
service-instance/instance-one",
            "createdAt": "2020-11-28T22:40:50.512000+00:00",
            "deploymentStatus": "SUCCEEDED",
            "environmentArn": "arn:aws:proton:region-id:123456789012:environment/
simple-env",
            "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",
            "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",
            "name": "instance-one",
            "serviceName": "simple-svc",
            "templateMajorVersion": "1",
            "templateMinorVersion": "0",
            "templateName": "fargate-service"
        }
    ]
}
```

Comando:

```
$ aws proton get-service-instance \
    --name "instance-one" \
    --service-name "simple-svc"
```

Risposta:

```
{
    "serviceInstance": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-
instance/instance-one",
        "createdAt": "2020-11-28T22:40:50.512000+00:00",
        "deploymentStatus": "SUCCEEDED",
        "environmentName": "simple-env",
        "lastDeploymentAttemptedAt": "2020-11-28T22:40:50.512000+00:00",
        "lastDeploymentSucceededAt": "2020-11-28T22:40:50.512000+00:00",
```

```
        "name": "instance-one",
        "serviceName": "simple-svc",
        "spec": "proton: ServiceSpec\npipeline:\n
 my_sample_pipeline_optional_input: hello world\n
 my_sample_pipeline_required_input: pipeline up\ninstances:\n- name: instance-one\n
 environment: my-simple-env\n  spec:\n    my_sample_service_instance_optional_input:
 Ola\n    my_sample_service_instance_required_input: Ciao\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
        "templateName": "svc-simple"
    }
}
```

Aggiornare un'istanza di servizio

Scopri come aggiornare un'istanza del AWS Proton servizio e annullare l'aggiornamento.

Un'istanza di servizio appartiene a un servizio. È possibile creare o eliminare un'istanza solo nell'ambito delle azioni di modifica, creazione ed eliminazione del servizio. Per informazioni su come aggiungere e rimuovere istanze da un servizio, consultaModifica un servizio.

Esistono quattro modalità per aggiornare un'istanza del servizio, come descritto nell'elenco seguente. Quando si utilizza AWS CLI, il `deployment-type` campo definisce la modalità. Quando si utilizza la console, queste modalità sono mappate alle azioni Modifica e Aggiornamento all'ultima versione secondaria e Aggiornamento all'ultima versione principale, disponibili in Azioni nella pagina di dettaglio dell'istanza del servizio.

`NONE`

In questa modalità, non si verifica una distribuzione. Vengono aggiornati solo i parametri dei metadati richiesti.

`CURRENT_VERSION`

In questa modalità, l'istanza del servizio viene distribuita e aggiornata con le nuove specifiche fornite dall'utente. Vengono aggiornati solo i parametri richiesti. Non includere i parametri della versione minore o principale quando lo usi`deployment-type`.

`MINOR_VERSION`

In questa modalità, l'istanza del servizio viene distribuita e aggiornata con la versione secondaria pubblicata, consigliata (più recente) della versione principale corrente, utilizzata per impostazione predefinita. È inoltre possibile specificare una versione secondaria diversa della versione principale corrente in uso.

`MAJOR_VERSION`

In questa modalità, per impostazione predefinita, l'istanza del servizio viene distribuita e aggiornata con la versione principale e secondaria pubblicata, consigliata (più recente) del modello corrente. È inoltre possibile specificare una versione principale diversa che sia superiore alla versione principale in uso e una versione secondaria (opzionale).

È possibile tentare di annullare la distribuzione di un aggiornamento di un'istanza di servizio, se lo `deploymentStatus` è`IN_PROGRESS`. AWS Proton tenta di annullare la distribuzione. L'annullamento riuscito non è garantito.

Quando annulli la distribuzione di un aggiornamento, AWS Proton tenta di annullare la distribuzione come indicato nei passaggi seguenti.

- Imposta lo stato di distribuzione su`CANCELLING`.
- Interrompe il processo di distribuzione ed elimina tutte le nuove risorse create dalla distribuzione quando`IN_PROGRESS`.
- Imposta lo stato di distribuzione su`CANCELLED`.
- Riporta lo stato della risorsa allo stato precedente all'avvio della distribuzione.

Per ulteriori informazioni sull'annullamento della distribuzione di un'istanza di servizio, consulta l'AWS Proton API CancelServiceInstanceDeploymentReference.

Usa la console o AWS CLI per effettuare aggiornamenti o annullare le distribuzioni degli aggiornamenti.

AWS Management Console

Aggiorna un'istanza del servizio utilizzando la console seguendo questi passaggi.

1. Nella AWS Proton console, scegli Istanze di servizio nel riquadro di navigazione.
2. Nell'elenco delle istanze del servizio, scegli il nome dell'istanza di servizio che desideri aggiornare.

3. Scegli Azioni, quindi scegli una delle opzioni di aggiornamento, Modifica per aggiornare le specifiche o Azioni e quindi Aggiorna all'ultima versione secondaria o Aggiorna all'ultima versione principale.
4. Compila ogni modulo e scegli Avanti fino a raggiungere la pagina di revisione.
5. Controlla le modifiche e scegli Aggiorna.

AWS CLI

Aggiorna un'istanza del servizio a una nuova versione secondaria, come mostrato nei comandi e nelle risposte di esempio della CLI.

Quando si aggiorna l'istanza del servizio con un'istanza modificata`spec`, è possibile utilizzarla `"${Proton::CURRENT_VAL}"` per indicare quali valori dei parametri preservare rispetto all'originale`spec`, se i valori esistono in. `spec` `get-service`Da utilizzare per visualizzare l'originale `spec` di un'istanza di servizio, come descritto inVisualizza i dati del servizio.

L'esempio seguente mostra come è possibile utilizzare `"${Proton::CURRENT_VAL}"` in un`spec`.

Spec:

```
proton: ServiceSpec

pipeline:
  my_sample_pipeline_optional_input: "${Proton::CURRENT_VAL}"
  my_sample_pipeline_required_input: "${Proton::CURRENT_VAL}"

instances:
  - name: "my-instance"
    environment: "simple-env"
    spec:
      my_sample_service_instance_optional_input: "${Proton::CURRENT_VAL}"
      my_sample_service_instance_required_input: "${Proton::CURRENT_VAL}"
  - name: "my-other-instance"
    environment: "simple-env"
    spec:
      my_sample_service_instance_required_input: "789"
```

Comando: aggiornare

```
$ aws proton update-service-instance \
```

```
    --name "instance-one" \
    --service-name "simple-svc" \
    --spec "file://service-spec.yaml" \
    --template-major-version "1" \
    --template-minor-version "1" \
    --deployment-type "MINOR_VERSION"
```

Risposta:

```
{
    "serviceInstance": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-
instance/instance-one",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "deploymentStatus": "IN_PROGRESS",
        "environmentName": "arn:aws:proton:region-id:123456789012:environment/
simple-env",
        "lastDeploymentAttemptedAt": "2021-04-02T21:38:00.823000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T21:29:59.962000+00:00",
        "name": "instance-one",
        "serviceName": "simple-svc",
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
        "templateName": "svc-simple"
    }
}
```

Comando: per ottenere e confermare lo stato

```
$ aws proton get-service-instance \
    --name "instance-one" \
    --service-name "simple-svc"
```

Risposta:

```
{
    "serviceInstance": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-
instance/instance-one",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "deploymentStatus": "SUCCEEDED",
        "environmentName": "simple-env",
```

```
        "lastDeploymentAttemptedAt": "2021-04-02T21:38:00.823000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T21:38:00.823000+00:00",
        "name": "instance-one",
        "serviceName": "simple-svc",
        "spec": "proton: ServiceSpec\n\npipeline:\n
 my_sample_pipeline_optional_input: \"abc\"\n  my_sample_pipeline_required_input:
 \"123\"\n\ninstances:\n  - name: \"instance-one\"\n    environment: \"simple-
env\"\n    spec:\n      my_sample_service_instance_optional_input: \"def\"\n
      my_sample_service_instance_required_input: \"456\"\n  - name: \"my-
other-instance\"\n    environment: \"kls-simple-env\"\n    spec:\n
 my_sample_service_instance_required_input: \"789\"\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "svc-simple"
    }
}
```

AWS Management Console

Annulla la distribuzione di un'istanza di servizio utilizzando la console, come illustrato nei passaggi seguenti.

1. Nella AWS Proton console, scegli Istanze di servizio nel riquadro di navigazione.
2. Nell'elenco delle istanze del servizio, scegli il nome dell'istanza del servizio con l'aggiornamento di distribuzione che desideri annullare.
3. Se lo stato di distribuzione dell'aggiornamento è In corso, nella pagina dei dettagli dell'istanza di servizio, scegli Azioni e quindi Annulla distribuzione.
4. Un modale ti chiede di confermare l'annullamento. Scegli Annulla distribuzione.
5. Lo stato di distribuzione degli aggiornamenti è impostato su Annullamento e quindi su Annullato per completare l'annullamento.

AWS CLI

Annulla l'aggiornamento della distribuzione dell'istanza del servizio IN_PROGRESS alla nuova versione secondaria 2, come mostrato nei seguenti comandi e risposte di esempio CLI.

Nel modello utilizzato per questo esempio è inclusa una condizione di attesa in modo che l'annullamento inizi prima del completamento della distribuzione dell'aggiornamento.

Comando: annullare

```
$ aws proton cancel-service-instance-deployment \
    --service-instance-name "instance-one" \
    --service-name "simple-svc"
```

Risposta:

```
{
    "serviceInstance": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-
instance/instance-one",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "deploymentStatus": "CANCELLING",
        "environmentName": "simple-env",
        "lastDeploymentAttemptedAt": "2021-04-02T21:45:15.406000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T21:38:00.823000+00:00",
        "name": "instance-one",
        "serviceName": "simple-svc",
        "spec": "proton: ServiceSpec\npipeline:\n
 my_sample_pipeline_optional_input: abc\n  my_sample_pipeline_required_input:
 '123'\ninstances:\n- name: my-instance\n  environment: MySimpleEnv
\n  spec:\n    my_sample_service_instance_optional_input: def\n
 my_sample_service_instance_required_input: '456'\n- name: my-other-instance\n
 environment: MySimpleEnv\n  spec:\n    my_sample_service_instance_required_input:
 '789'\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "svc-simple"
    }
}
```

Comando: per ottenere e confermare lo stato

```
$ aws proton get-service-instance \
    --name "instance-one" \
    --service-name "simple-svc"
```

Risposta:

```
{
```

```
    "serviceInstance": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/service-
instance/instance-one",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "deploymentStatus": "CANCELLED",
        "deploymentStatusMessage": "User initiated cancellation.",
        "environmentName": "simple-env",
        "lastDeploymentAttemptedAt": "2021-04-02T21:45:15.406000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T21:38:00.823000+00:00",
        "name": "instance-one",
        "serviceName": "simple-svc",
        "spec": "proton: ServiceSpec\n\npipeline:\n
 my_sample_pipeline_optional_input: \"abc\"\n  my_sample_pipeline_required_input:
 \"123\"\n\ninstances:\n  - name: \"instance-one\"\n    environment: \"simple-
env\"\n    spec:\n      my_sample_service_instance_optional_input: \"def\"\n
      my_sample_service_instance_required_input: \"456\"\n  - name: \"my-
other-instance\"\n    environment: \"kls-simple-env\"\n    spec:\n
 my_sample_service_instance_required_input: \"789\"\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "svc-simple"
    }
}
```


Aggiorna una pipeline di servizi

Scopri come aggiornare una pipeline AWS Proton di servizi e annullare l'aggiornamento.

Una pipeline di servizi appartiene a un servizio. È possibile creare o eliminare una pipeline solo nel contesto delle azioni di creazione ed eliminazione del servizio.

Esistono quattro modalità per aggiornare una pipeline di servizi, come descritto nell'elenco seguente. Quando si utilizza AWS CLI, il `deployment-type` campo definisce la modalità. Quando si utilizza la console, queste modalità vengono mappate alla pipeline Modifica e all'aggiornamento alla versione consigliata.

`NONE`

In questa modalità, non si verifica una distribuzione. Vengono aggiornati solo i parametri dei metadati richiesti.

`CURRENT_VERSION`

In questa modalità, la pipeline di servizi viene distribuita e aggiornata con le nuove specifiche fornite dall'utente. Vengono aggiornati solo i parametri richiesti. Non includere i parametri della versione minore o principale quando lo usi`deployment-type`.

`MINOR_VERSION`

In questa modalità, la pipeline del servizio viene distribuita e aggiornata con la versione secondaria pubblicata, consigliata (più recente) della versione principale corrente, utilizzata per impostazione predefinita. È inoltre possibile specificare una versione secondaria diversa della versione principale corrente in uso.

`MAJOR_VERSION`

In questa modalità, per impostazione predefinita, la pipeline di servizi viene distribuita e aggiornata con la versione principale e secondaria pubblicata, consigliata (più recente) del modello corrente. È inoltre possibile specificare una versione principale diversa che sia superiore alla versione principale in uso e una versione secondaria (opzionale).

È possibile tentare di annullare la distribuzione di un aggiornamento della pipeline di servizio, se lo `deploymentStatus` è`IN_PROGRESS`. AWS Proton tenta di annullare la distribuzione. L'annullamento riuscito non è garantito.

Quando annulli la distribuzione di un aggiornamento, AWS Proton tenta di annullare la distribuzione come indicato nei passaggi seguenti.

- Imposta lo stato di distribuzione su`CANCELLING`.
- Interrompe il processo di distribuzione ed elimina tutte le nuove risorse create dalla distribuzione quando`IN_PROGRESS`.
- Imposta lo stato di distribuzione su`CANCELLED`.
- Riporta lo stato della risorsa allo stato precedente all'avvio della distribuzione.

Per ulteriori informazioni sull'annullamento della distribuzione di una pipeline di servizi, consulta l'AWS Proton API CancelServicePipelineDeploymentReference.

Utilizza la console o AWS CLI per effettuare aggiornamenti o annullare le distribuzioni degli aggiornamenti.

AWS Management Console

Aggiorna una pipeline di servizi utilizzando la console come descritto nei passaggi seguenti.

1. Nella AWS Proton console, scegli Servizi.
2. Nell'elenco dei servizi, scegli il nome del servizio per cui desideri aggiornare la pipeline.
3. Nella pagina dei dettagli del servizio sono presenti due schede, Overview e Pipeline. Scegliete Pipeline.
4. Se desideri aggiornare le specifiche, scegli Modifica pipeline e compila ogni modulo e scegli Avanti fino a completare il modulo finale, quindi scegli Aggiorna pipeline.

   Se desideri eseguire l'aggiornamento a una nuova versione e c'è un'icona informativa che indica che una nuova versione è disponibile in Pipeline template, scegli il nome della nuova versione del modello.

   a. Scegli Aggiorna alla versione consigliata.
   b. Compila ogni modulo e scegli Avanti fino a completare il modulo finale e scegli Aggiorna.

AWS CLI

Aggiorna una pipeline di servizi a una nuova versione secondaria, come mostrato nei seguenti comandi e risposte di esempio CLI.

Quando si aggiorna la pipeline di servizio con una versione modificata`spec`, è possibile utilizzare `"${Proton::CURRENT_VAL}"` per indicare quali valori dei parametri preservare rispetto all'originale`spec`, se i valori esistono in. `spec` `get-service`Da utilizzare per visualizzare l'originale `spec` di una pipeline di servizi, come descritto in. Visualizza i dati del servizio

L'esempio seguente mostra come è possibile utilizzare `"${Proton::CURRENT_VAL}"` in un`spec`.

Spec:

```
proton: ServiceSpec

pipeline:
  my_sample_pipeline_optional_input: "${Proton::CURRENT_VAL}"
  my_sample_pipeline_required_input: "${Proton::CURRENT_VAL}"
```

```
instances:
  - name: "my-instance"
    environment: "simple-env"
    spec:
      my_sample_service_instance_optional_input: "${Proton::CURRENT_VAL}"
      my_sample_service_instance_required_input: "${Proton::CURRENT_VAL}"
  - name: "my-other-instance"
    environment: "simple-env"
    spec:
      my_sample_service_instance_required_input: "789"
```

Comando: aggiornare

```
$ aws proton update-service-pipeline \
    --service-name "simple-svc" \
    --spec "file://service-spec.yaml" \
    --template-major-version "1" \
    --template-minor-version "1" \
    --deployment-type "MINOR_VERSION"
```

Risposta:

```
{
    "pipeline": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/pipeline/
a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "deploymentStatus": "IN_PROGRESS",
        "lastDeploymentAttemptedAt": "2021-04-02T21:39:28.991000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T21:29:59.962000+00:00",
        "spec": "proton: ServiceSpec\n\npipeline:\n
 my_sample_pipeline_optional_input: \"abc\"\n  my_sample_pipeline_required_input:
 \"123\"\n\ninstances:\n  - name: \"my-instance\"\n    environment: \"MySimpleEnv
\"\n    spec:\n      my_sample_service_instance_optional_input: \"def
\"\n      my_sample_service_instance_required_input: \"456\"\n  - name:
 \"my-other-instance\"\n    environment: \"MySimpleEnv\"\n    spec:\n
 my_sample_service_instance_required_input: \"789\"\n",
        "templateMajorVersion": "1",
        "templateMinorVersion": "0",
        "templateName": "svc-simple"
    }
}
```

Comando: per ottenere e confermare lo stato

```
$ aws proton get-service \
    --name "simple-svc"
```

Risposta:


```
{
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc",
        "branchName": "main",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "lastModifiedAt": "2021-04-02T21:30:54.364000+00:00",
        "name": "simple-svc",
        "pipeline": {
            "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/
pipeline",
            "createdAt": "2021-04-02T21:29:59.962000+00:00",
            "deploymentStatus": "SUCCEEDED",
            "lastDeploymentAttemptedAt": "2021-04-02T21:39:28.991000+00:00",
            "lastDeploymentSucceededAt": "2021-04-02T21:39:28.991000+00:00",
            "spec": "proton: ServiceSpec\n\npipeline:\n
 my_sample_pipeline_optional_input: \"abc\"\n  my_sample_pipeline_required_input:
 \"123\"\n\ninstances:\n  - name: \"instance-one\"\n    environment: \"simple-
env\"\n    spec:\n      my_sample_service_instance_optional_input: \"def
\"\n      my_sample_service_instance_required_input: \"456\"\n  - name:
 \"my-other-instance\"\n    environment: \"simple-env\"\n    spec:\n
 my_sample_service_instance_required_input: \"789\"\n",
            "templateMajorVersion": "1",
            "templateMinorVersion": "1",
            "templateName": "svc-simple"
        },
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "repositoryId": "repo-name/myorg-myapp",
        "spec": "proton: ServiceSpec\n\npipeline:\n
 my_sample_pipeline_optional_input: \"abc\"\n  my_sample_pipeline_required_input:
 \"123\"\n\ninstances:\n  - name: \"instance-one\"\n    environment: \"simple-
env\"\n    spec:\n      my_sample_service_instance_optional_input: \"def
\"\n      my_sample_service_instance_required_input: \"456\"\n  - name:
 \"my-other-instance\"\n    environment: \"simple-env\"\n    spec:\n
 my_sample_service_instance_required_input: \"789\"\n",
        "status": "ACTIVE",
```

```
        "templateName": "svc-simple"
    }
}
```

AWS Management Console

Annulla l'implementazione di una pipeline di servizi utilizzando la console, come illustrato nei passaggi seguenti.

1. Nella AWS Proton console, scegli Servizi nel riquadro di navigazione.
2. Nell'elenco dei servizi, scegli il nome del servizio che include la pipeline con l'aggiornamento di distribuzione che desideri annullare.
3. Nella pagina dei dettagli del servizio, scegli la scheda Pipeline.
4. Se lo stato della distribuzione degli aggiornamenti è In corso, nella pagina dei dettagli della pipeline del servizio, scegli Annulla distribuzione.
5. Un modale ti chiede di confermare l'annullamento. Scegli Annulla distribuzione.
6. Lo stato di distribuzione degli aggiornamenti è impostato su Annullamento e quindi su Annullato per completare l'annullamento.

AWS CLI

Annulla un aggiornamento della distribuzione della pipeline del servizio IN_PROGRESS alla versione secondaria 2, come mostrato nei seguenti comandi e risposte di esempio CLI.

Nel modello utilizzato per questo esempio è inclusa una condizione di attesa in modo che l'annullamento inizi prima del completamento della distribuzione dell'aggiornamento.

Comando: annullare

```
$ aws proton cancel-service-pipeline-deployment \
    --service-name "simple-svc"
```

Risposta:

```
{
    "pipeline": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/pipeline",
```

```
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "deploymentStatus": "CANCELLING",
        "lastDeploymentAttemptedAt": "2021-04-02T22:02:45.095000+00:00",
        "lastDeploymentSucceededAt": "2021-04-02T21:39:28.991000+00:00",
        "templateMajorVersion": "1",
        "templateMinorVersion": "1",
        "templateName": "svc-simple"
    }
}
```

Comando: per ottenere e confermare lo stato

```
$ aws proton get-service \
    --name "simple-svc"
```

Risposta:

```
{
    "service": {
        "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc",
        "branchName": "main",
        "createdAt": "2021-04-02T21:29:59.962000+00:00",
        "lastModifiedAt": "2021-04-02T21:30:54.364000+00:00",
        "name": "simple-svc",
        "pipeline": {
            "arn": "arn:aws:proton:region-id:123456789012:service/simple-svc/
pipeline",
            "createdAt": "2021-04-02T21:29:59.962000+00:00",
            "deploymentStatus": "CANCELLED",
            "deploymentStatusMessage": "User initiated cancellation.",
            "lastDeploymentAttemptedAt": "2021-04-02T22:02:45.095000+00:00",
            "lastDeploymentSucceededAt": "2021-04-02T21:39:28.991000+00:00",
            "spec": "proton: ServiceSpec\n\npipeline:\n
 my_sample_pipeline_optional_input: \"abc\"\n  my_sample_pipeline_required_input:
 \"123\"\n\ninstances:\n  - name: \"instance-one\"\n    environment: \"simple-
env\"\n    spec:\n      my_sample_service_instance_optional_input: \"def
\"\n      my_sample_service_instance_required_input: \"456\"\n  - name:
 \"my-other-instance\"\n    environment: \"simple-env\"\n    spec:\n
 my_sample_service_instance_required_input: \"789\"\n",
            "templateMajorVersion": "1",
            "templateMinorVersion": "1",
            "templateName": "svc-simple"
        },
```

```
        "repositoryConnectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
        "repositoryId": "repo-name/myorg-myapp",
        "spec": "proton: ServiceSpec\n\npipeline:\n
 my_sample_pipeline_optional_input: \"abc\"\n  my_sample_pipeline_required_input:
 \"123\"\n\ninstances:\n  - name: \"instance-one\"\n    environment: \"simple-
env\"\n    spec:\n      my_sample_service_instance_optional_input: \"def
\"\n      my_sample_service_instance_required_input: \"456\"\n  - name:
 \"my-other-instance\"\n    environment: \"simple-env\"\n    spec:\n
 my_sample_service_instance_required_input: \"789\"\n",
        "status": "ACTIVE",
        "templateName": "svc-simple"
    }
}
```

AWS Proton componenti

I componenti sono un tipo di risorsa. AWS Proton Aggiungono flessibilità ai modelli di servizio. I componenti forniscono ai team della piattaforma un meccanismo per estendere i modelli di infrastruttura di base e definire misure di protezione che consentono agli sviluppatori di gestire aspetti della loro infrastruttura applicativa.

In, AWS Proton gli amministratori definiscono l'infrastruttura standard da utilizzare tra i team di sviluppo e le applicazioni. Tuttavia, i team di sviluppo potrebbero dover includere risorse aggiuntive per i loro casi d'uso specifici, come le code di Amazon Simple Queue Service (Amazon SQS) o le tabelle Amazon DynamoDB. Queste risorse specifiche dell'applicazione potrebbero cambiare frequentemente, in particolare durante lo sviluppo iniziale delle applicazioni. Mantenere queste frequenti modifiche nei modelli creati dagli amministratori potrebbe essere difficile da gestire e scalare: gli amministratori avrebbero bisogno di mantenere molti più modelli senza un reale valore aggiunto da parte degli amministratori. Anche l'alternativa, che consente agli sviluppatori di applicazioni di creare modelli per le proprie applicazioni, non è ideale, perché impedisce agli amministratori di standardizzare i principali componenti dell'architettura, come le attività. AWS Fargate È qui che entrano in gioco i componenti.

Con un componente, uno sviluppatore può aggiungere risorse supplementari alla propria applicazione, oltre a quanto definito dagli amministratori nei modelli di ambiente e servizio. Lo sviluppatore collega quindi il componente a un'istanza del servizio. AWS Proton fornisce le risorse dell'infrastruttura definite dal componente proprio come fornisce le risorse per ambienti e istanze di servizio.

Un componente può leggere gli input dell'istanza di servizio e fornire output all'istanza di servizio, per un'esperienza completamente integrata. Ad esempio, se il componente aggiunge un bucket Amazon Simple Storage Service (Amazon S3) per l'utilizzo da parte di un'istanza di servizio, il modello del componente può tenere conto dei nomi delle istanze dell'ambiente e del servizio per denominare il bucket. Quando AWS Proton esegue il rendering del modello di servizio per fornire un'istanza di servizio, l'istanza del servizio può fare riferimento al bucket e utilizzarlo.

I componenti AWS Proton attualmente supportati sono componenti definiti direttamente. Il file Infrastructure as Code (IaC) che definisce l'infrastruttura del componente viene passato direttamente all' AWS Proton API o alla console. È diverso da un ambiente o un servizio, in cui si definisce IaC in un pacchetto modello e si registra il pacchetto come risorsa modello, quindi si utilizza una risorsa modello per creare l'ambiente o il servizio.

> **Note**
>
> I componenti definiti direttamente consentono agli sviluppatori di definire un'infrastruttura aggiuntiva e di eseguirne il provisioning. AWS Proton esegue il provisioning di tutti i componenti definiti direttamente in esecuzione nello stesso ambiente utilizzando lo stesso ruolo AWS Identity and Access Management (IAM).

Un amministratore può controllare ciò che gli sviluppatori possono fare con i componenti in due modi:

- Fonti di componenti supportate: un amministratore può consentire il collegamento di componenti alle istanze di servizio in base a una proprietà delle versioni dei modelli di AWS Proton servizio. Per impostazione predefinita, gli sviluppatori non possono collegare componenti alle istanze del servizio.

  Per ulteriori informazioni su questa proprietà, consulta il supportedComponentSourcesparametro dell'azione CreateServiceTemplateVersionAPI nell'AWS Proton API Reference.

  > **Note**
  >
  > Quando si utilizza la sincronizzazione dei modelli di servizio, AWS Proton crea le versioni dei modelli di servizio in modo implicito quando si eseguono modifiche a un pacchetto di modelli di servizio in un repository. In questo caso, anziché specificare le origini dei componenti supportate durante la creazione della versione del modello di servizio, si specifica questa proprietà in un file associato a ogni versione principale del modello di servizio. Per ulteriori informazioni, consulta the section called “Modelli di servizio di sincronizzazione”.

- Ruoli dei componenti: un amministratore può assegnare un ruolo componente a un ambiente. AWS Proton assume questo ruolo quando effettua il provisioning dell'infrastruttura definita da un componente definito direttamente nell'ambiente. Pertanto, il ruolo del componente delimita l'infrastruttura che gli sviluppatori possono aggiungere utilizzando componenti definiti direttamente nell'ambiente. In assenza del ruolo del componente, gli sviluppatori non possono creare componenti definiti direttamente nell'ambiente.

  Per ulteriori informazioni sull'assegnazione del ruolo di un componente, consulta il componentRoleArnparametro dell'azione CreateEnvironmentAPI nell'AWS Proton API Reference.

> **Note**
>
> I ruoli dei componenti non vengono utilizzati negli Provisioning autogestito ambienti.

Argomenti

- Come si confrontano i componenti con altre AWS Proton risorse?
- Componenti nella console AWS Proton
- Componenti nell' AWS Proton API e AWS CLI
- Domande frequenti sui componenti
- Stati dei componenti
- Infrastruttura dei componenti come file di codice
- AWS CloudFormation Esempio di componente

Come si confrontano i componenti con altre AWS Proton risorse?

Per molti aspetti, i componenti sono simili ad altre AWS Proton risorse. La loro infrastruttura è definita in un file modello IAc, creato in formato AWS CloudFormation YAML o Terraform HCL. AWS Proton può fornire l'infrastruttura dei componenti utilizzando il provisioning gestito o AWS il provisioning autogestito.

I componenti sono tuttavia diversi dalle altre AWS Proton risorse per alcuni aspetti:

- Stato distaccato: i componenti sono progettati per essere collegati alle istanze di servizio e per estenderne l'infrastruttura, ma possono anche trovarsi in uno stato distaccato, in cui non sono collegati a nessuna istanza di servizio. Per ulteriori informazioni sugli stati dei componenti, vedere. the section called “Stati dei componenti”
- Nessuno schema: i componenti non hanno uno schema associato come i pacchetti di modelli. Gli input dei componenti sono definiti da un servizio. Un componente può consumare input quando è collegato a un'istanza di servizio.
- Nessun componente gestito dal cliente: fornisce AWS Proton sempre l'infrastruttura dei componenti per te. Non esiste una versione dei componenti basata sull'utilizzo delle risorse proprie. Per ulteriori informazioni sugli ambienti gestiti dai clienti, consulta. the section called “Crea”

- Nessuna risorsa modello: ai componenti definiti direttamente non è associata una risorsa modello simile ai modelli di ambiente e servizio. Fornisci un file modello IAc direttamente al componente. Analogamente, fornite direttamente un manifesto che definisce il linguaggio dei modelli e il motore di rendering per il provisioning dell'infrastruttura del componente. La creazione del file modello e del manifesto è simile alla creazione di un pacchetto di modelli. Tuttavia, con i componenti definiti direttamente, non è necessario archiviare i file IAc come pacchetti in posizioni particolari e non si crea una risorsa modello all'interno AWS Proton di file IaC esterni.
- Nessun provisioning CodeBuild basato: non è possibile effettuare il provisioning di componenti definiti direttamente utilizzando uno script di provisioning personalizzato, noto come provisioning basato. CodeBuild Per ulteriori informazioni, consulta the section called “CodeBuild approvvigionamento”.

Componenti nella console AWS Proton

Usa la AWS Proton console per creare, aggiornare, visualizzare e utilizzare AWS Proton i componenti.

Le seguenti pagine della console sono relative ai componenti. Includiamo collegamenti diretti alle pagine di console di primo livello.

- Componenti: visualizza l'elenco dei componenti del tuo AWS account. Puoi creare nuovi componenti e aggiornare o eliminare i componenti esistenti. Scegliete il nome di un componente dall'elenco per visualizzarne la pagina dei dettagli.

  Elenchi simili sono disponibili anche nelle pagine dei dettagli dell'ambiente e dei dettagli dell'istanza del servizio. Questi elenchi mostrano solo i componenti associati alla risorsa visualizzata. Quando si crea un componente da uno di questi elenchi, AWS Proton preseleziona l'ambiente associato nella pagina Crea componente.
- Dettagli del componente: per visualizzare la pagina dei dettagli del componente, scegliete il nome di un componente nell'elenco Componenti.

  Nella pagina dei dettagli, visualizza i dettagli e lo stato del componente e aggiorna o elimina il componente. Visualizza e gestisci gli elenchi di output (ad esempio, risorse assegnate ARNs), AWS CloudFormation stack assegnati e tag assegnati.
- Crea componente: crea un componente. Immettete il nome e la descrizione del componente, scegliete le risorse associate, specificate il file IAc di origine del componente e assegnate i tag.

- Aggiorna componente: per aggiornare un componente, selezionalo nell'elenco Componenti e quindi, nel menu Azioni, scegli Aggiorna componente. In alternativa, nelle pagine dei dettagli del componente, scegli Aggiorna.

  Puoi aggiornare la maggior parte dei dettagli del componente. Non è possibile aggiornare il nome del componente. E puoi scegliere se ridistribuire o meno il componente dopo un aggiornamento riuscito.
- Configurazione dell'ambiente: quando si crea o si aggiorna un ambiente, è possibile specificare il ruolo del componente. Questo ruolo controlla la capacità di eseguire componenti definiti direttamente nell'ambiente e fornisce le autorizzazioni per il loro provisioning.
- Crea una nuova versione del modello di servizio: quando si crea una versione del modello di servizio, è possibile specificare le origini dei componenti supportate per la versione del modello. Ciò controlla la capacità di collegare componenti alle istanze di servizio dei servizi basati su questa versione del modello.

Componenti nell' AWS Proton API e AWS CLI

Utilizza l' AWS Proton API o il AWS CLI per creare, aggiornare, visualizzare e utilizzare AWS Proton i componenti.

Le seguenti azioni API gestiscono direttamente le risorse AWS Proton dei componenti.

- CreateComponent— Creare un AWS Proton componente.
- DeleteComponent— Eliminare un AWS Proton componente.
- GetComponent— Ottieni dati dettagliati per un componente.
- ListComponentOutputs— Ottieni un elenco degli output di Infrastructure as Code (IaC) dei componenti.
- ListComponentProvisionedResources— Elenca le risorse fornite per un componente con dettagli.
- ListComponents— Elenca i componenti con dati di riepilogo. È possibile filtrare l'elenco dei risultati per ambiente, servizio o una singola istanza del servizio.

Le seguenti azioni API di altre AWS Proton risorse hanno alcune funzionalità relative ai componenti.

- CreateEnvironment, UpdateEnvironment— `componentRoleArn` Da utilizzare per specificare l'Amazon Resource Name (ARN) del ruolo del servizio IAM AWS Proton utilizzato per il

provisioning di componenti definiti direttamente in questo ambiente. Determina l'ambito dell'infrastruttura che un componente definito direttamente può fornire.

- CreateServiceTemplateVersion— `supportedComponentSources` Da utilizzare per specificare le sorgenti dei componenti supportate. I componenti con fonti supportate possono essere collegati alle istanze di servizio in base a questa versione del modello di servizio.

Domande frequenti sui componenti

Qual è il ciclo di vita di un componente?

I componenti possono trovarsi in uno stato collegato o distaccato. Sono progettati per essere collegati a un'istanza di servizio e migliorarne l'infrastruttura per la maggior parte del tempo. I componenti scollegati si trovano in uno stato transitorio che consente di eliminare un componente o collegarlo a un'altra istanza del servizio in modo controllato e sicuro. Per ulteriori informazioni, consulta the section called “Stati dei componenti”.

Perché non posso eliminare i componenti allegati?

Soluzione: per eliminare un componente allegato, aggiorna il componente per scollegarlo dall'istanza del servizio, convalida la stabilità dell'istanza del servizio e quindi elimina il componente.

Perché è necessario? I componenti collegati forniscono l'infrastruttura aggiuntiva di cui l'applicazione ha bisogno per eseguire le sue funzioni di runtime. L'istanza del servizio potrebbe utilizzare gli output dei componenti per rilevare e utilizzare le risorse di questa infrastruttura. L'eliminazione del componente, con conseguente rimozione delle relative risorse infrastrutturali, potrebbe causare interruzioni dell'istanza di servizio collegata.

Come misura di sicurezza aggiuntiva, AWS Proton richiede l'aggiornamento del componente e lo scollegamento dalla relativa istanza di servizio prima di poterlo eliminare. È quindi possibile convalidare l'istanza del servizio per assicurarsi che continui a essere distribuita e a funzionare correttamente. Se rilevi un problema, puoi ricollegare rapidamente il componente all'istanza del servizio, quindi lavorare per risolvere il problema. Quando sei sicuro che la tua istanza di servizio sia libera da qualsiasi dipendenza dal componente, puoi eliminare il componente in tutta sicurezza.

Perché non posso modificare direttamente l'istanza del servizio collegata a un componente?

Soluzione: per modificare l'allegato, aggiorna il componente per scollegarlo dall'istanza del servizio, convalida la stabilità del componente e dell'istanza del servizio, quindi collega il componente alla nuova istanza di servizio.

Perché è necessario? Un componente è progettato per essere collegato a un'istanza di servizio. Il componente potrebbe utilizzare gli input dell'istanza di servizio per la denominazione e la configurazione delle risorse dell'infrastruttura. La modifica dell'istanza del servizio collegata potrebbe causare interruzioni per il componente (oltre alla possibile interruzione dell'istanza del servizio), come descritto nelle domande frequenti precedenti, Perché non posso eliminare i componenti allegati?). Ad esempio, potrebbe causare la ridenominazione e possibilmente anche la sostituzione delle risorse definite nel modello IaC del componente.

Come misura di sicurezza aggiuntiva, AWS Proton richiede l'aggiornamento del componente e lo scollegamento dalla relativa istanza di servizio prima di poterlo collegare a un'altra istanza del servizio. È quindi possibile convalidare la stabilità sia del componente che dell'istanza del servizio prima di collegare il componente alla nuova istanza di servizio.

Stati dei componenti

AWS Proton i componenti possono trovarsi in due stati fondamentalmente diversi:

- Allegato: il componente è collegato a un'istanza del servizio. Definisce l'infrastruttura che supporta la funzionalità di runtime dell'istanza di servizio. Il componente estende l'infrastruttura definita nei modelli di ambiente e servizio con un'infrastruttura definita dallo sviluppatore.

  Un componente tipico rimane nello stato collegato per la maggior parte della parte utile del suo ciclo di vita.
- Distaccato: il componente è associato a un AWS Proton ambiente e non è collegato a nessuna istanza di servizio nell'ambiente.

  Si tratta di uno stato transitorio per estendere la durata di un componente oltre una singola istanza di servizio.

La tabella seguente fornisce un confronto di primo livello tra i diversi stati dei componenti.

	Collegato	Distaccato
Scopo principale dello Stato	Estendere l'infrastruttura di un'istanza di servizio.	Per mantenere l'infrastruttura del componente tra gli allegati delle istanze di servizio.
Associato a	Un'istanza di servizio e un ambiente	Un ambiente

	Collegato	Distaccato
Proprietà specifiche chiave	• Nome servizio • Nome dell'istanza del servizio • Spec	• Nome ambiente
Può essere eliminato	✗ No	✓ Sì
Può essere aggiornato a un'altra istanza del servizio	✗ No	✓ Sì
Può leggere gli input	✓ Sì	✗ No

Lo scopo principale di un componente è quello di essere collegato a un'istanza di servizio ed estenderne l'infrastruttura con risorse aggiuntive. Un componente collegato può leggere gli input dall'istanza del servizio in base alle specifiche. Non è possibile eliminare direttamente il componente o collegarlo a un'altra istanza del servizio. Non puoi nemmeno eliminare la relativa istanza di servizio o il servizio e l'ambiente correlati. Per eseguire una di queste operazioni, aggiorna prima il componente per scollegarlo dall'istanza del servizio.

Per mantenere l'infrastruttura del componente oltre la durata di una singola istanza di servizio, è necessario aggiornare il componente e scollegarlo dalla relativa istanza di servizio rimuovendo i nomi del servizio e delle istanze di servizio. Questo stato distaccato è uno stato transitorio. Il componente non ha ingressi. La sua infrastruttura rimane predisposta e puoi aggiornarla. È possibile eliminare le risorse a cui il componente era associato al momento del collegamento (istanza di servizio, servizio). È possibile eliminare il componente o aggiornarlo per collegarlo nuovamente a un'istanza del servizio.

Infrastruttura dei componenti come file di codice

I file IAc (Component Infrastructure as Code) sono simili a quelli di altre AWS Proton risorse. Scopri qui alcuni dettagli specifici dei componenti. Per informazioni complete sulla creazione di file IAc per AWS Proton, vedere. Creazione di modelli e pacchetti

Utilizzo dei parametri con i componenti

Lo spazio AWS Proton dei nomi dei parametri include alcuni parametri a cui un file IAc di servizio può fare riferimento per ottenere il nome e gli output di un componente associato. Lo spazio dei nomi include anche parametri a cui un file IAc del componente può fare riferimento per ottenere input, output e valori delle risorse dall'ambiente, dal servizio e dall'istanza del servizio a cui il componente è associato.

Un componente non dispone di input propri, ma riceve gli input dall'istanza di servizio a cui è collegato. Un componente può anche leggere gli output dell'ambiente.

Per ulteriori informazioni sull'utilizzo dei parametri nei file IAc del componente e del servizio associato, vedere. the section called “Parametri del componente CloudFormation IAc” Per informazioni generali sui AWS Proton parametri e un riferimento completo allo spazio dei nomi dei parametri, vedere. the section called “Parametri”

Creazione di file IAc robusti

In qualità di amministratore, quando create una versione del modello di servizio, potete decidere se consentire alle istanze di servizio create a partire dalla versione del modello di avere componenti allegati. Vedi il supportedComponentSourcesparametro dell'azione CreateServiceTemplateVersionAPI nell'AWS Proton API Reference. Tuttavia, per tutte le future istanze di servizio, la persona che crea l'istanza decide se allegarvi o meno un componente e (nel caso di componenti definiti direttamente) l'autore del componente IaC è in genere una persona diversa, uno sviluppatore che utilizza il modello di servizio. Pertanto, non è possibile garantire che un componente venga collegato a un'istanza di servizio. Inoltre, non è possibile garantire l'esistenza di nomi di output di componenti specifici o la validità e la sicurezza dei valori di questi output.

AWS Proton e la sintassi Jinja vi aiutano a risolvere questi problemi e a creare solidi modelli di servizio che eseguono il rendering senza errori nei seguenti modi:

- AWS Proton filtri parametrici: quando si fa riferimento alle proprietà di output dei componenti, è possibile utilizzare filtri parametrici, ossia modificatori che convalidano, filtrano e formattano i valori dei parametri. Per maggiori informazioni ed esempi, consulta the section called “CloudFormation filtri parametrici”.
- Proprietà singola predefinita: quando si fa riferimento a una singola risorsa o proprietà di output di un componente, è possibile garantire che il rendering del modello di servizio non abbia esito negativo utilizzando il `default` filtro, con o senza un valore predefinito. Se il componente o

uno specifico parametro di output a cui ti riferisci non esiste, viene invece renderizzato il valore predefinito (o una stringa vuota, se non hai specificato un valore predefinito) e il rendering ha esito positivo. Per ulteriori informazioni, consulta the section called “Fornire valori predefiniti”.

Esempi:

- `{{ service_instance.components.default.name | default("") }}`
- `{{ service_instance.components.default.outputs.my-output | default("17") }}`

> **Note**
>
> Non confondete la `.default` parte dello spazio dei nomi, che designa i componenti definiti direttamente, con il `default` filtro, che fornisce un valore predefinito quando la proprietà referenziata non esiste.

- Riferimento all'intero oggetto: quando si fa riferimento all'intero componente o alla raccolta degli output di un componente, AWS Proton restituisce un oggetto vuoto e quindi garantisce che il rendering del `{}` modello di servizio non abbia esito negativo. Non è necessario utilizzare alcun filtro. Assicurati di inserire il riferimento in un contesto che può accettare un oggetto vuoto o usa una `{{ if .. }}` condizione per verificare la presenza di un oggetto vuoto.

  Esempi:

  - `{{ service_instance.components.default }}`
  - `{{ service_instance.components.default.outputs }}`

AWS CloudFormation Esempio di componente

Ecco un esempio completo di un componente definito AWS Proton direttamente e di come utilizzarlo in un AWS Proton servizio. Il componente fornisce un bucket Amazon Simple Storage Service (Amazon S3) e la relativa politica di accesso. L'istanza del servizio può fare riferimento a questo bucket e utilizzarlo. Il nome del bucket si basa sui nomi dell'ambiente, del servizio, dell'istanza del servizio e del componente, il che significa che il bucket è associato a un'istanza specifica del modello di componente che estende un'istanza di servizio specifica. Gli sviluppatori possono creare più componenti basati su questo modello di componenti, per fornire bucket Amazon S3 per diverse istanze di servizio ed esigenze funzionali.

L'esempio riguarda la creazione dei vari file Required AWS CloudFormation Infrastructure as Code (IAc) e la creazione di un ruolo richiesto AWS Identity and Access Management (IAM). L'esempio raggruppa i passaggi in base ai ruoli delle persone proprietarie.

Passaggi per amministratori

Per consentire agli sviluppatori di utilizzare i componenti con un servizio

1. Crea un ruolo AWS Identity and Access Management (IAM) che limiti le risorse che possono essere fornite dai componenti definiti direttamente in esecuzione nel tuo ambiente. AWS Proton assume questo ruolo in un secondo momento per fornire componenti definiti direttamente nell'ambiente.

   Per questo esempio, utilizza la seguente politica:

   Example ruolo del componente definito direttamente

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:CancelUpdateStack",
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:DescribeStacks",
        "cloudformation:ContinueUpdateRollback",
        "cloudformation:DetectStackResourceDrift",
        "cloudformation:DescribeStackResourceDrifts",
        "cloudformation:DescribeStackEvents",
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:UpdateStack",
        "cloudformation:DescribeChangeSet",
        "cloudformation:ExecuteChangeSet",
        "cloudformation:ListChangeSets",
        "cloudformation:ListStackResources"
      ],
      "Resource": "arn:aws:cloudformation:*:123456789012:stack/AWSProton-*"
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:DeleteBucket",
        "s3:GetBucket",
        "iam:CreatePolicy",
        "iam:DeletePolicy",
        "iam:GetPolicy",
        "iam:ListPolicyVersions",
        "iam:DeletePolicyVersion"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "cloudformation.amazonaws.com"
        }
      }
    }
  ]
}
```

2. Fornisci il ruolo creato nel passaggio precedente quando crei o aggiorni l'ambiente. Nella AWS Proton console, specifica un ruolo Componente nella pagina Configura ambiente. Se stai utilizzando l' AWS Proton API o AWS CLI, specifica le azioni `componentRoleArn` delle CreateEnvironmento dell'UpdateEnvironmentAPI.
3. Crea un modello di servizio che faccia riferimento a un componente definito direttamente collegato all'istanza del servizio.

   L'esempio mostra come scrivere un modello di servizio affidabile che non si interrompa se un componente non è collegato all'istanza del servizio.

   Example file CloudFormation IAc di servizio che utilizza un componente

```
# service/instance_infrastructure/cloudformation.yaml

Resources:
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      TaskRoleArn: !Ref TaskRole
      ContainerDefinitions:
        - Name: '{{service_instance.name}}'
```

```
          # ...
          {% if service_instance.components.default.outputs | length > 0 %}
          Environment:
            {{ service_instance.components.default.outputs |
                proton_cfn_ecs_task_definition_formatted_env_vars }}
          {% endif %}

  # ...

  TaskRole:
    Type: AWS::IAM::Role
    Properties:
      # ...
      ManagedPolicyArns:
        - !Ref BaseTaskRoleManagedPolicy
        {{ service_instance.components.default.outputs
            | proton_cfn_iam_policy_arns }}

  # Basic permissions for the task
  BaseTaskRoleManagedPolicy:
    Type: AWS::IAM::ManagedPolicy
    Properties:
      # ...
```

4. Crea una nuova versione secondaria del modello di servizio che dichiari come supportati i componenti definiti direttamente.

   - Pacchetto di modelli in Amazon S3: nella console, quando AWS Proton crei una versione del modello di servizio, per Origini di componenti supportate, scegli Definito direttamente. Se utilizzi l' AWS Proton API o AWS CLI, specifica `DIRECTLY_DEFINED` nel `supportedComponentSources` parametro CreateServiceTemplateVersiono nelle azioni UpdateServiceTemplateVersionAPI.
   - Sincronizzazione dei modelli: esegui una modifica `supported_component_sources:` nell'archivio dei pacchetti di modelli di servizio, dove specifichi `DIRECTLY_DEFINED` come elemento del `.template-registration.yaml` file nella directory delle versioni principali. Per ulteriori informazioni su questo file, consulta the section called “Modelli di servizio di sincronizzazione”.

5. Pubblica la nuova versione secondaria del modello di servizio. Per ulteriori informazioni, consulta the section called “Pubblicare”.

6. Assicurati di consentire il `proton:CreateComponent` ruolo IAM degli sviluppatori che utilizzano questo modello di servizio.

Passaggi per gli sviluppatori

Per utilizzare un componente definito direttamente con un'istanza di servizio

1. Crea un servizio che utilizzi la versione del modello di servizio creata dall'amministratore con il supporto dei componenti. In alternativa, aggiorna una delle istanze del servizio esistenti per utilizzare la versione più recente del modello.
2. Scrivi un file modello IAc componente che fornisce un bucket Amazon S3 e una politica di accesso correlata ed espone queste risorse come output.

   Example file IAc componente CloudFormation

```
# cloudformation.yaml

# A component that defines an S3 bucket and a policy for accessing the bucket.
Resources:
  S3Bucket:
    Type: 'AWS::S3::Bucket'
    Properties:
      BucketName: '{{environment.name}}-{{service.name}}-{{service_instance.name}}-
{{component.name}}'
  S3BucketAccessPolicy:
    Type: AWS::IAM::ManagedPolicy
    Properties:
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Action:
              - 's3:Get*'
              - 's3:List*'
              - 's3:PutObject'
            Resource: !GetAtt S3Bucket.Arn
Outputs:
  BucketName:
    Description: "Bucket to access"
    Value: !GetAtt S3Bucket.Arn
  BucketAccessPolicyArn:
```

```
    Value: !Ref S3BucketAccessPolicy
```

3. Se stai usando l' AWS Proton API o AWS CLI, scrivi un file manifest per il componente.

   Example manifesto del componente definito direttamente

   ```
   infrastructure:
     templates:
       - file: "cloudformation.yaml"
         rendering_engine: jinja
         template_language: cloudformation
   ```

4. Crea un componente definito direttamente. AWS Proton assume il ruolo del componente definito dall'amministratore per il provisioning del componente.

   Nella AWS Proton console, nella pagina Componenti, scegli Crea componente. Per le impostazioni dei componenti, inserisci un nome del componente e una descrizione opzionale del componente. Per Collegamento del componente, scegli Collega il componente a un'istanza del servizio. Seleziona l'ambiente, il servizio e l'istanza del servizio. Per Component source, scegliete AWS CloudFormation, quindi scegliete il file IAc del componente.

   > **Note**
   >
   > Non è necessario fornire un manifesto: la console ne crea uno automaticamente.

   Se stai utilizzando l' AWS Proton API oppure AWS CLI, utilizza l'azione CreateComponentAPI. Imposta un componente `name` e facoltativo`description`. Set `environmentName``serviceName`, e`serviceInstanceName`. Imposta `templateSource` e `manifest` accedi ai percorsi dei file che hai creato.

   > **Note**
   >
   > La specificazione di un nome di ambiente è facoltativa quando si specificano i nomi del servizio e delle istanze del servizio. La combinazione di questi due elementi è univoca nell' AWS account e AWS Proton può determinare l'ambiente a partire dall'istanza del servizio.

5. Aggiorna l'istanza del servizio per ridistribuirla. AWS Proton utilizza gli output del componente nel modello di istanza del servizio reso, per consentire all'applicazione di utilizzare il bucket Amazon S3 fornito dal componente.

Usare i repository git con AWS Proton

AWS Proton utilizza i repository git per una varietà di scopi. L'elenco seguente classifica i tipi di repository associati alle risorse. AWS Proton Per AWS Proton le funzionalità che si connettono ripetutamente al repository per inviare o estrarre contenuti dal repository, è necessario registrare un collegamento al repository nel proprio account. AWS Proton AWS Un link al repository è un insieme di proprietà che è AWS Proton possibile utilizzare quando si connette a un repository. AWS Proton attualmente supporta GitHub GitHub Enterprise e. BitBucket

Archivi per sviluppatori

Repository di codice: un repository utilizzato dagli sviluppatori per archiviare il codice dell'applicazione. Utilizzato per la distribuzione del codice. AWS Proton non interagisce direttamente con questo repository. Quando uno sviluppatore fornisce un servizio che include una pipeline, fornisce il nome del repository e il ramo da cui leggere il codice dell'applicazione. AWS Proton trasmette queste informazioni alla pipeline che fornisce.

Per ulteriori informazioni, consulta the section called “Crea”.

Archivi per amministratori

Archivio di modelli: un repository in cui gli amministratori archiviano i pacchetti di modelli. AWS Proton Utilizzato per la sincronizzazione dei modelli. Quando un amministratore crea un modello in AWS Proton, può puntare a un archivio di modelli e AWS Proton mantenere il nuovo modello sincronizzato con esso. Quando l'amministratore aggiorna il pacchetto di modelli nell'archivio, crea AWS Proton automaticamente una nuova versione del modello. Collega un archivio di modelli a AWS Proton prima di poterlo utilizzare per la sincronizzazione.

Per ulteriori informazioni, consulta the section called “Configurazioni di sincronizzazione dei modelli”.

> **Note**
>
> Non è necessario un archivio di modelli se continui a caricare i modelli su Amazon Simple Storage Service (Amazon S3) e richiami la gestione dei modelli APIs per creare nuovi AWS Proton modelli o versioni di modelli.

Archivi di provisioning gestiti automaticamente

Archivio dell'infrastruttura: un repository che ospita modelli di infrastruttura renderizzati. Utilizzato per il provisioning autogestito dell'infrastruttura di risorse. Quando un amministratore crea un ambiente per il provisioning autogestito, fornisce un archivio. AWS Proton invia richieste pull (PRs) a questo repository per creare l'infrastruttura per l'ambiente e per qualsiasi istanza di servizio distribuita nell'ambiente. Collega un repository dell'infrastruttura a AWS Proton prima di poterlo utilizzare per il provisioning dell'infrastruttura autogestito.

Pipeline repository: un repository utilizzato per creare pipeline. Utilizzato per il provisioning autogestito delle pipeline. L'utilizzo di un repository aggiuntivo per il provisioning delle pipeline consente di AWS Proton archiviare le configurazioni delle pipeline indipendentemente da qualsiasi ambiente o servizio individuale. È sufficiente fornire un unico repository di pipeline per tutti i servizi di provisioning autogestiti. Collega un repository di pipeline a AWS Proton prima di poterlo utilizzare per il provisioning di pipeline autogestito.

Per ulteriori informazioni, consulta the section called “AWS-approvvigionamento gestito”.

Argomenti

- Crea un link al tuo repository
- Visualizza i dati del repository collegato
- Eliminare il collegamento a un repository

Crea un link al tuo repository

Puoi creare un link al tuo repository utilizzando la console o la CLI. Quando crei un link al repository, AWS Proton crea automaticamente un ruolo collegato al servizio.

AWS Management Console

Crea un link al tuo repository come illustrato nei seguenti passaggi della console.

1. Nella AWS Proton console, scegli Repository.
2. Scegli Create repository (Crea repository).
3. Nella pagina Collega nuovo repository, nella sezione Dettagli del repository:

    a. Scegli il fornitore del tuo repository.

b. Scegli una delle tue connessioni esistenti. Se non ne hai una, scegli Aggiungi una nuova CodeStar connessione per creare una connessione, quindi torna alla AWS Proton console, aggiorna l'elenco delle connessioni e scegli la nuova connessione.

c. Scegli tra gli archivi di codice sorgente collegati.

4. [opzionale] Nella sezione Tag, scegli Aggiungi nuovo tag una o più volte e inserisci le coppie Chiave e Valore.
5. Scegli Create repository (Crea repository).
6. Visualizza i dati di dettaglio del tuo repository collegato.

AWS CLI

Crea e registra un link al tuo repository.

Esegui il comando seguente:

```
$ aws proton create-repository \
    --name myrepos/environments \
    --connection-arn "arn:aws:codestar-connections:region-
id:123456789012:connection/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111" \
    --provider "GITHUB" \
    --encryption-key "arn:aws:kms:region-id:123456789012:key/bPxRfiCYEXAMPLEKEY" \
    --tags key=mytag1,value=value1 key=mytag2,value=value2
```

Gli ultimi due parametri, --encryption-key e--tags, sono opzionali.

Risposta:

```
{
    "repository": {
        "arn": "arn:aws:proton:region-id:123456789012:repository/github:myrepos/
environments",
        "connectionArn": "arn:aws:codestar-connections:region-
id:123456789012:connection/2ad03b28-a7c4-EXAMPLE11111",
        "encryptionKey": "arn:aws:kms:region-id:123456789012:key/
bPxRfiCYEXAMPLEKEY",
        "name": "myrepos/environments",
        "provider": "GITHUB"
    }
}
```

Dopo aver creato un link al repository, è possibile visualizzare un elenco di AWS tag gestiti dai clienti, come illustrato nel comando di esempio seguente. AWS Proton genera automaticamente tag AWS gestiti per te. Puoi anche modificare e creare tag gestiti dai clienti utilizzando AWS CLI. Per ulteriori informazioni, consulta AWS Proton risorse e etichettatura.

Comando:

```
$ aws proton list-tags-for-resource \
    --resource-arn "arn:aws:proton:region-id:123456789012:repository/github:myrepos/
environments"
```

Visualizza i dati del repository collegato

È possibile elencare e visualizzare i dettagli del repository collegato utilizzando la console o il. AWS CLI Per i link ai repository utilizzati per sincronizzare i repository git con AWS Proton, puoi recuperare la definizione e lo stato di sincronizzazione del repository utilizzando il. AWS CLI

AWS Management Console

Elenca e visualizza i dettagli del repository collegato utilizzando la console.AWS Proton

1. Per elencare i repository collegati, scegli Repository nel riquadro di navigazione.
2. Per visualizzare i dati di dettaglio, scegli il nome di un repository.

AWS CLI

Elenca i repository collegati.

Esegui il comando seguente:

```
$ aws proton list-repositories
```

Risposta:

```
{
    "repositories": [
        {
            "arn": "arn:aws:proton:region-id:123456789012:repository/github:myrepos/
templates",
            "name": "myrepos/templates",
```

```
            "provider": "GITHUB"
        },
        {
            "arn": "arn:aws:proton:region-id:123456789012:repository/github:myrepos/
environments",
            "name": "myrepos/environments",
            "provider": "GITHUB"
        }
    ]
}
```

Visualizza i dettagli di un repository collegato.

Esegui il comando seguente:

```
$ aws proton get-repository \
    --name myrepos/templates \
    --provider "GITHUB"
```

Risposta:

```
{
    "repository": {
        "arn": "arn:aws:proton:region-id:123456789012:repository/github:myrepos/
templates",
        "name": "myrepos/templates",
        "provider": "GITHUB"
    }
}
```

Elenca i tuoi repository sincronizzati.

L'esempio seguente elenca i repository configurati per la sincronizzazione dei modelli.

Esegui il comando seguente:

```
$ aws proton list-repository-sync-definitions \
    --branch "main" \
    --repository-name myrepos/templates \
    --repository-provider "GITHUB" \
    --sync-type "TEMPLATE_SYNC"
```

Visualizza lo stato di sincronizzazione del repository.

L'esempio seguente recupera lo stato di sincronizzazione di un repository di sincronizzazione dei modelli.

Esegui il comando seguente:

```
$ aws proton get-repository-sync-status \
    --branch "main" \
    --repository-name myrepos/templates \
    --repository-provider "GITHUB" \
    --sync-type "TEMPLATE_SYNC"
```

Risposta:

```
{
    "latestSync": {
        "events": [
            {
                "event": "Clone started",
                "time": "2021-11-21T00:26:35.883000+00:00",
                "type": "CLONE_STARTED"
            },
            {
                "event": "Updated configuration",
                "time": "2021-11-21T00:26:41.894000+00:00",
                "type": "CONFIG_UPDATED"
            },
            {
                "event": "Starting syncs for commit 62c03ff86eEXAMPLE1111111",
                "externalId": "62c03ff86eEXAMPLE1111111",
                "time": "2021-11-21T00:26:44.861000+00:00",
                "type": "STARTING_SYNC"
            }
        ],
        "startedAt": "2021-11-21T00:26:29.728000+00:00",
        "status": "SUCCEEDED"
    }
}
```

Eliminare il collegamento a un repository

È possibile eliminare un collegamento al repository utilizzando la console o il. AWS CLI

> **Note**
>
> L'eliminazione di un link al repository rimuove solo il link AWS Proton registrato presente nel tuo account. AWS Non elimina alcuna informazione dal tuo repository.

AWS Management Console

Elimina un link al repository utilizzando la console.

Nella pagina dei dettagli del repository.

1. Nella AWS Proton console, scegli Repository.
2. Nell'elenco dei repository, scegli il pulsante di opzione a sinistra del repository che desideri eliminare.
3. Scegliere Delete (Elimina).
4. Una finestra modale richiede di confermare l'azione Elimina.
5. Segui le istruzioni e scegli Sì, elimina.

AWS CLI

Elimina un link al repository.

Esegui il comando seguente:

```
$ aws proton delete-repository \
    --name myrepos/templates \
    --provider"GITHUB"
```

Risposta:

```
{
    "repository": {
        "arn": "arn:aws:proton:region-id:123456789012:repository/github:myrepos/
templates",
        "name": "myrepos/templates",
        "provider": "GITHUB"
    }
}
```

Monitoraggio AWS Proton

Il monitoraggio è un elemento importante per mantenere l'affidabilità, la disponibilità e le prestazioni delle AWS Proton AWS soluzioni esistenti. La sezione seguente descrive gli strumenti di monitoraggio che è possibile utilizzare con AWS Proton.

Automatizza con AWS Proton EventBridge

Puoi monitorare AWS Proton gli eventi in Amazon EventBridge. EventBridge fornisce un flusso di dati in tempo reale dalle tue applicazioni, applicazioni software-as-a-service (SaaS) e. Servizi AWSÈ possibile configurare gli eventi per rispondere alle modifiche dello stato AWS delle risorse. EventBridge indirizza quindi questi dati a servizi di destinazione come Amazon AWS Lambda Simple Notification Service. Questi eventi sono gli stessi che compaiono in Amazon CloudWatch Events. CloudWatch Events fornisce un flusso quasi in tempo reale di eventi di sistema che descrivono i cambiamenti nelle AWS risorse. Per ulteriori informazioni, consulta What Is Amazon EventBridge? nella Amazon EventBridge User Guide.

Utilizzalo EventBridge per ricevere notifiche sui cambiamenti di stato nei flussi di lavoro di AWS Proton approvvigionamento.

Event types (Tipi di evento)

Gli eventi sono composti da regole che includono un modello di evento e obiettivi. Si configura una regola scegliendo il modello di evento e gli oggetti di destinazione:

Modello di evento

> Ogni regola è espressa come modello di evento con l'origine e il tipo di eventi da monitorare e gli obiettivi degli eventi. Per monitorare gli eventi, crei una regola con il servizio che stai monitorando come fonte dell'evento. Ad esempio, puoi creare una regola con un pattern di eventi che viene utilizzato AWS Proton come fonte di eventi per attivare una regola in caso di modifiche nello stato di distribuzione.

Destinazioni

> La regola riceve un servizio selezionato come destinazione dell'evento. È possibile configurare un servizio di destinazione per inviare notifiche, acquisire informazioni sullo stato, intraprendere azioni correttive, avviare eventi o intraprendere altre azioni.

Gli oggetti evento contengono campi standard di ID, account, tipo di dettaglio Regione AWS, origine, versione, risorsa e ora (opzionale). Il campo dettaglio è un oggetto annidato contenente campi personalizzati per l'evento.

AWS Proton gli eventi vengono emessi con la massima diligenza possibile. Per «consegna con il massimo impegno» si intende che il servizio tenta di inviare tutti gli eventi a EventBridge, ma in alcuni rari casi un evento potrebbe non essere consegnato.

Per ogni AWS Proton risorsa che può emettere eventi, la tabella seguente elenca il valore del tipo di dettaglio, i campi di dettaglio e (se disponibile) un riferimento a un elenco di valori per i campi `status` e `previousStatus` dettaglio. Quando una risorsa viene eliminata, il valore del campo di `status` dettaglio è. `DELETED`

Risorsa	Valore del tipo di dettaglio	Campi di dettaglio

`EnvironmentTemplate`	AWS Proton Modifica dello stato del modello di ambiente	`name`
		`status`
		`previousStatus`

`EnvironmentTemplateVersion`	AWS Proton Modifica dello stato della versione del modello di ambiente	`name`
		`majorVersion`
		`minorVersion`
		`status`
		`previousStatus`
		valori di stato

`ServiceTemplate`	AWS Proton Modifica dello stato del modello di servizio	`name` `status`

Risorsa	Valore del tipo di dettaglio	Campi di dettaglio
		`previousStatus`

`ServiceTemplateVersion`	AWS Proton Modifica dello stato della versione del modello di servizio	`name`
		`majorVersion`
		`minorVersion`
		`status`
		`previousStatus`
		valori di stato

`Environment`	AWS Proton Modifica dello stato dell'ambiente	`name`
		`status`
		`previousStatus`

`Service`	AWS Proton Modifica dello stato del servizio	`name`
		`status`
		`previousStatus`
		valori di stato

Risorsa	Valore del tipo di dettaglio	Campi di dettaglio

`ServiceInstance`	AWS Proton Modifica dello stato dell'istanza di servizio	`name`
		`serviceName`
		`status`
		`previousStatus`

`ServicePipeline`	AWS Proton Modifica dello stato della pipeline di servizio	`serviceName`
		`status`
		`previousStatus`

`EnvironmentAccountConnection`	AWS Proton Modifica dello stato di connessione dell'account di ambiente	`id`
		`status`
		`previousStatus`
		valori di stato

`Component`	AWS Proton Modifica dello stato del componente	`name`
		`status`
		`previousStatus`

AWS Proton esempi di eventi

Gli esempi seguenti mostrano i modi in cui AWS Proton è possibile inviare eventi a EventBridge.

Modello di servizio

```
{
    "source": "aws.proton",
    "detail-type": ["AWS Proton Service Template Status Change"],
    "time": "2021-03-22T23:21:40.734Z",
    "resources": ["arn:aws:proton:region_id:123456789012:service-template/sample-
service-template-name"],
    "detail": {
        "name": "sample-service-template-name",
        "status": "PUBLISHED",
        "previousStatus": "DRAFT"
    }
}
```

Versione del modello di servizio

```
{
    "source": "aws.proton",
    "detail-type": ["AWS Proton Service Template Version Status Change"],
    "time": "2021-03-22T23:21:40.734Z",
    "resources": ["arn:aws:proton:region_id:123456789012:service-template/sample-
service-template-name:1.0"],
    "detail": {
        "name": "sample-service-template-name",
        "majorVersion": "1",
        "minorVersion": "0",
        "status": "REGISTRATION_FAILED",
        "previousStatus": "REGISTRATION_IN_PROGRESS"
    }
}
```

Ambiente

```
{
    "source": "aws.proton",
    "detail-type": ["AWS Proton Environment Status Change"],
    "time": "2021-03-22T23:21:40.734Z",
    "resources": ["arn:aws:proton:region_id:123456789012:environment/sample-
environment"],
    "detail": {
        "name": "sample-environment",
        "status": "DELETE_FAILED",
        "previousStatus": "DELETE_IN_PROGRESS"
```

```
    }
}
```

EventBridgeTutorial: invia avvisi di Amazon Simple Notification Service per le modifiche allo stato del AWS Proton servizio

In questo tutorial, utilizzi una regola di evento AWS Proton preconfigurata che registra le modifiche allo stato del tuo servizio. AWS Proton EventBridgeinvia le modifiche di stato a un argomento di Amazon SNS. Ti iscrivi all'argomento e Amazon SNS ti invia e-mail di modifica dello stato del tuo AWS Proton servizio.

Prerequisiti

Hai già un AWS Proton servizio con uno `Active` stato. Come parte di questo tutorial, puoi aggiungere istanze di servizio a questo servizio e quindi eliminare le istanze.

Se devi creare un AWS Proton servizio, consulta. Nozioni di base Per ulteriori informazioni, consultare AWS Proton quote e the section called “Modificare”.

Fase 1: Creazione e sottoscrizione a un argomento Amazon SNS

Crea un argomento Amazon SNS che funga da obiettivo dell'evento per la regola evento che crei nella fase 2.

Creare un argomento Amazon SNS.

1. Accedi e apri la console Amazon SNS.
2. Nel riquadro di navigazione, scegli Argomenti, Crea argomento.
3. Nella pagina Crea argomento:
    a. Scegli Type Standard.
    b. Per Nome, inserisci **`tutorial-service-status-change`** e scegli Crea argomento.
4. Nella pagina dei tutorial-service-status-changedettagli, scegli Crea abbonamento.
5. Nella pagina Crea abbonamento:
    a. Per Protocollo, scegli E-mail.
    b. In Endpoint, inserire un indirizzo e-mail a cui si ha accesso correntemente, quindi scegliere Crea sottoscrizione.

6. Controllare l'account e-mail e attendere di ricevere una e-mail di conferma della sottoscrizione. Quando lo ricevi, aprilo e scegli Conferma abbonamento.

Fase 2: Registrazione di una regola di evento

Registra una regola di evento che registri le modifiche allo stato del tuo AWS Proton servizio. Per ulteriori informazioni, consulta Prerequisiti.

Crea una regola di evento.

1. Apri la EventBridge console Amazon.
2. Nel riquadro di navigazione, seleziona Events (Eventi), Rules (Regole).
3. Nella pagina Regole, nella sezione Regole, scegli Crea regola.
4. Nella pagina Crea regola:

    a. Nella sezione Nome e descrizione, in Nome, immettere**`tutorial-rule`**.

    b. Nella sezione Definisci modello, scegli Schema di eventi.

        i. In Event matching pattern (Modello di corrispondenza eventi), scegli Pre-defined by service (Predefinito per servizio).

        ii. Per Service provider (Provider di servizi), selezionare AWS.

        iii. Per Service Name (Nome del servizio), selezionare AWS Proton.

        iv. Per Tipo di evento, scegli Modifica dello stato del AWS Proton servizio.

        Il pattern Event viene visualizzato in un editor di testo.

        v. Apri la AWS Proton console.

        vi. Nel riquadro di navigazione, scegli Servizi.

        vii. Nella pagina Servizi, scegli il nome del tuo AWS Proton servizio.

        viii. Nella pagina dei dettagli del servizio, copia il servizio Amazon Resource Name (ARN).

        ix. Torna alla EventBridge console e alla regola del tutorial e scegli Modifica nell'editor di testo.

        x. Nell'editor di testo, per`"resources":`, inserisci l'ARN del servizio che hai copiato nel passaggio viii.

```
{
    "source": ["aws.proton"],
```

```
    "detail-type": ["AWS Proton Service Status Change"],
    "resources": ["arn:aws:proton:region-id:123456789012:service/your-service"]
}
```

xi. Salva lo schema dell'evento.

c. Nella sezione Seleziona obiettivi:

i. In Target (Destinazione), scegli SNS topic (Argomento SNS).

ii. Per Argomento, scegli tutorial-service-status-change.

d. Scegli Create (Crea).

Fase 3: Verifica la regola del tuo evento

Verifica che la regola dell'evento funzioni aggiungendo un'istanza al AWS Proton servizio.

1. Passa alla AWS Proton console.
2. Nel riquadro di navigazione, scegli Servizi.
3. Nella pagina Servizi, scegli il nome del tuo servizio.
4. Nella pagina dei dettagli del servizio, scegli Modifica.
5. Nella pagina Configura servizio, scegli Avanti.
6. Nella pagina Configura impostazioni personalizzate, nella sezione Istanze di servizio, scegli Aggiungi nuova istanza.
7. Completa il modulo per la tua nuova istanza:

   a. Inserisci un nome per la tua nuova istanza.

   b. Seleziona gli stessi ambienti compatibili che hai scelto per le tue istanze esistenti.

   c. Inserisci i valori per gli input richiesti.

   d. Scegli Next (Successivo).
8. Controlla i tuoi input e scegli Aggiorna.
9. Dopo aver verificato lo stato del servizio`Active`, controlla la tua e-mail per verificare di aver ricevuto AWS notifiche che forniscono aggiornamenti sullo stato.

```
{
    "version": "0",
    "id": "af76c382-2b3c-7a0a-cf01-936dff228276",
```

```
    "detail-type": "AWS Proton Service Status Change",
    "source": "aws.proton",
    "account": "123456789012",
    "time": "2021-06-29T20:40:16Z",
    "region": "region-id",
    "resources": ["arn:aws:proton:region-id:123456789012:service/your-service"],
    "detail": {
        "previousStatus": "ACTIVE",
        "status": "UPDATE_IN_PROGRESS",
        "name": "your-service"
    }
}
```

```
{
    "version": "0",
    "id": "87131e29-ad95-bda2-cd30-0ce825dfb0cd",
    "detail-type": "AWS Proton Service Status Change",
    "source": "aws.proton",
    "account": "123456789012",
    "time": "2021-06-29T20:42:27Z",
    "region": "region-id",
    "resources": ["arn:aws:proton:region-id:123456789012:service/your-service"],
    "detail": {
        "previousStatus": "UPDATE_IN_PROGRESS",
        "status": "ACTIVE",
        "name": "your-service"
    }
}
```

Fase 4: pulizia

Elimina l'argomento e l'abbonamento ad Amazon SNS ed elimina la EventBridge regola.

Elimina l'argomento e l'abbonamento ad Amazon SNS.

1. Accedi alla console Amazon SNS.
2. Nel pannello di navigazione, scegli Abbonamenti.
3. Nella pagina Abbonamenti, scegli l'abbonamento che hai sottoscritto all'argomento denominato, **`tutorial-service-status-change`** quindi scegli Elimina.
4. Nel pannello di navigazione, scegli Argomenti.

5. Nella pagina Argomenti, scegli l'argomento denominato`tutorial-service-status-change`, quindi scegli Elimina.
6. Una finestra modale richiede di verificare l'eliminazione. Segui le istruzioni e scegli Elimina.

Elimina la tua EventBridge regola.

1. Accedi alla EventBridge console Amazon.
2. Nel riquadro di navigazione, seleziona Events (Eventi), Rules (Regole).
3. Nella pagina Regole, scegli la regola denominata`tutorial-rule`, quindi scegli Elimina.
4. Un modale richiede di verificare l'eliminazione. Scegliere Delete (Elimina).

Eliminare l'istanza di servizio aggiunta.

1. Passare alla console AWS Proton.
2. Nel riquadro di navigazione, scegli Servizi.
3. Nella pagina Servizi, scegli il nome del servizio.
4. Nella pagina dei dettagli del servizio, scegli Modifica, quindi Avanti.
5. Nella pagina Configura impostazioni personalizzate, nella sezione Istanze di servizio, scegli Elimina per l'istanza di servizio che hai creato come parte di questo tutorial, quindi scegli Avanti.
6. Controlla i tuoi input e scegli Aggiorna.
7. Un modale ti chiede di verificare l'eliminazione. Segui le istruzioni e scegli Sì, elimina.

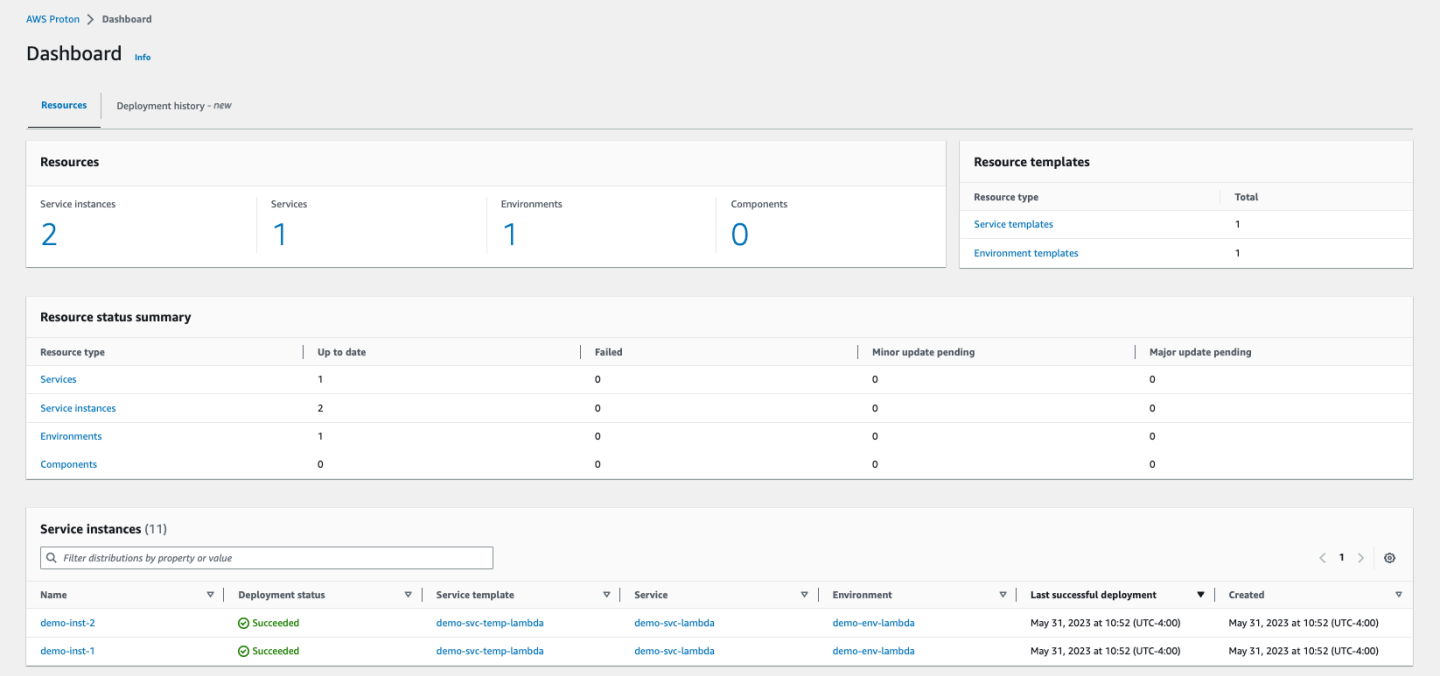
Mantieni l'infrastruttura aggiornata con la AWS Proton dashboard

La AWS Proton dashboard fornisce un riepilogo delle AWS Proton risorse presenti nel tuo AWS account, con particolare attenzione all'inattività, ossia all'aggiornamento delle risorse distribuite. Una risorsa distribuita è aggiornata quando utilizza la versione consigliata del modello associato. Una risorsa out-of-date distribuita potrebbe richiedere un aggiornamento della versione principale o secondaria del modello.

Visualizza la dashboard nella console AWS Proton

Per visualizzare la AWS Proton dashboard, apri la AWS Proton console e quindi, nel riquadro di navigazione, scegli Dashboard.

Risorse

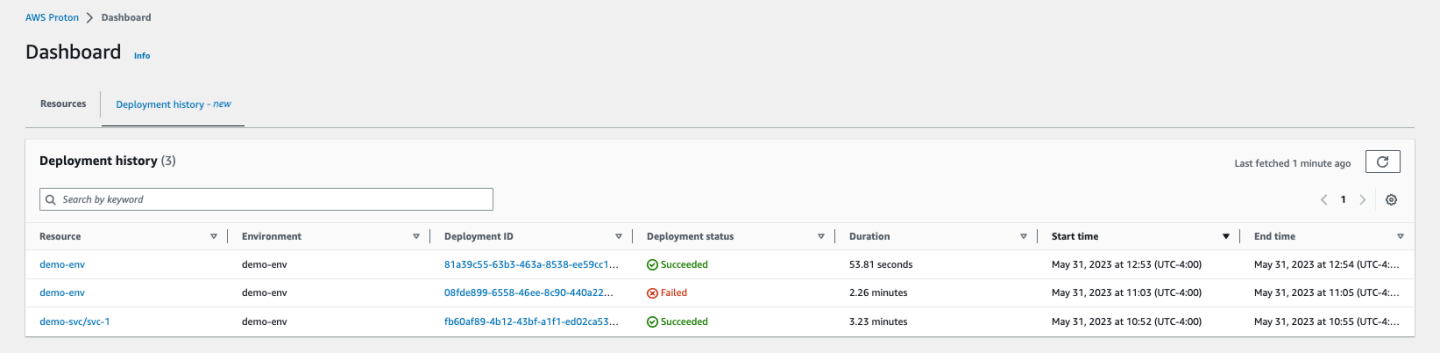


La prima scheda della dashboard mostra il conteggio di tutte le risorse del tuo account. La scheda delle risorse mostra il numero di istanze di servizio, servizi, ambienti e componenti, nonché i modelli di risorse. Inoltre, suddivide il numero di risorse per ogni tipo di risorsa distribuita in base allo stato delle risorse di quel tipo. Una tabella di istanze di servizio mostra i dettagli di ciascuna istanza di servizio: lo stato di distribuzione, AWS Proton le risorse a cui è associata, gli aggiornamenti disponibili e alcuni timestamp.

È possibile filtrare l'elenco delle istanze del servizio in base a qualsiasi proprietà della tabella. Ad esempio, è possibile filtrare per visualizzare le istanze di servizio con distribuzioni entro una finestra temporale specifica o le istanze di servizio che non sono aggiornate rispetto ai consigli sulle versioni principali o secondarie.

Scegli il nome di un'istanza di servizio per accedere alla pagina dei dettagli dell'istanza di servizio, in cui puoi intervenire per apportare gli aggiornamenti di versione appropriati. Scegli un altro nome di AWS Proton risorsa per accedere alla relativa pagina di dettaglio oppure scegli un tipo di risorsa per accedere al rispettivo elenco di risorse.

Cronologia delle distribuzioni



La scheda della cronologia delle distribuzioni consente di visualizzare i dettagli sulle distribuzioni. Nella tabella della cronologia delle distribuzioni, puoi tenere traccia dello stato della distribuzione, nonché dell'ambiente e dell'ID di distribuzione. Puoi scegliere il nome della risorsa o l'ID di distribuzione per visualizzare ancora più dettagli, come un messaggio sullo stato della distribuzione e gli output delle risorse. La tabella consente inoltre di filtrare in base a qualsiasi proprietà della tabella.

Sicurezza in AWS Proton

La sicurezza del cloud AWS è la massima priorità. In qualità di AWS cliente, puoi beneficiare di data center e architetture di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra te e te. AWS Il modello di responsabilità condivisa descrive questo aspetto come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gira Servizi AWS su Cloud AWS. AWS fornisce inoltre servizi che è possibile utilizzare in modo sicuro. I revisori esterni testano e verificano regolarmente l'efficacia della nostra sicurezza nell'ambito dei AWS Programmi di AWS conformità dei Programmi di conformità dei di . Per maggiori informazioni sui programmi di conformità applicabili AWS Proton, consulta Servizi AWS la sezione Scope by Compliance Program Servizi AWS .
- Sicurezza nel cloud: la tua responsabilità è determinata dal Servizio AWS materiale che utilizzi. Sei anche responsabile di altri fattori, tra cui la riservatezza dei dati, i requisiti della tua azienda e le leggi e normative vigenti.

Questa documentazione aiuta a capire come applicare il modello di responsabilità condivisa durante l'utilizzo AWS Proton. I seguenti argomenti illustrano come configurare l' AWS Proton per soddisfare gli obiettivi di sicurezza e conformità. Imparerai anche a utilizzarne altri Servizi AWS che ti aiutano a monitorare e proteggere AWS Proton le tue risorse.

Argomenti

- Identity and Access Management per AWS Proton
- Analisi della configurazione e delle vulnerabilità in AWS Proton
- Protezione dei dati in AWS Proton
- Sicurezza dell'infrastruttura in AWS Proton
- Registrazione e monitoraggio AWS Proton
- Resilienza in AWS Proton
- Best practice di sicurezza per AWS Proton
- Prevenzione del confused deputy tra servizi
- CodeBuild fornitura di supporto Amazon VPC personalizzato

Identity and Access Management per AWS Proton

AWS Identity and Access Management (IAM) è un software Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso alle AWS risorse. Gli amministratori IAM controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (disporre delle autorizzazioni) a utilizzare le risorse. AWS Proton IAM è uno Servizio AWS strumento che puoi utilizzare senza costi aggiuntivi.

Argomenti

- Destinatari
- Autenticazione con identità
- Gestione dell'accesso con policy
- Come AWS Proton funziona con IAM
- Esempi di policy per AWS Proton
- AWS politiche gestite per AWS Proton
- Utilizzo di ruoli collegati ai servizi per AWS Proton
- Risoluzione dei problemi di AWS Proton identità e accesso

Destinatari

Il modo in cui usi AWS Identity and Access Management (IAM) varia a seconda del lavoro che AWS Proton svolgi.

Utente del servizio: se utilizzi il AWS Proton servizio per svolgere il tuo lavoro, l'amministratore ti fornisce le credenziali e le autorizzazioni necessarie. Man mano che utilizzi più AWS Proton funzionalità per svolgere il tuo lavoro, potresti aver bisogno di autorizzazioni aggiuntive. La comprensione della gestione dell'accesso ti consente di richiedere le autorizzazioni corrette all'amministratore. Se non riesci ad accedere a una funzionalità di AWS Proton, consulta Risoluzione dei problemi di AWS Proton identità e accesso.

Amministratore del servizio: se sei responsabile delle AWS Proton risorse della tua azienda, probabilmente hai pieno accesso a AWS Proton. È tuo compito determinare a quali AWS Proton funzionalità e risorse devono accedere gli utenti del servizio. Devi inviare le richieste

all'amministratore IAM per cambiare le autorizzazioni degli utenti del servizio. Esamina le informazioni contenute in questa pagina per comprendere i concetti di base relativi a IAM. Per saperne di più su come la tua azienda può utilizzare IAM con AWS Proton, consultaCome AWS Proton funziona con IAM.

Amministratore IAM: un amministratore IAM potrebbe essere interessato a ottenere dei dettagli su come scrivere policy per gestire l'accesso a AWS Proton. Per visualizzare esempi di policy AWS Proton basate sull'identità che puoi utilizzare in IAM, consulta. Esempi di policy basate sull'identità per AWS Proton

Autenticazione con identità

L'autenticazione è il modo in cui accedi AWS utilizzando le tue credenziali di identità. Devi essere autenticato (aver effettuato l' Utente root dell'account AWS accesso AWS) come utente IAM o assumendo un ruolo IAM.

Puoi accedere AWS come identità federata utilizzando le credenziali fornite tramite una fonte di identità. AWS IAM Identity Center Gli utenti (IAM Identity Center), l'autenticazione Single Sign-On della tua azienda e le tue credenziali di Google o Facebook sono esempi di identità federate. Se accedi come identità federata, l'amministratore ha configurato in precedenza la federazione delle identità utilizzando i ruoli IAM. Quando accedi AWS utilizzando la federazione, assumi indirettamente un ruolo.

A seconda del tipo di utente, puoi accedere al AWS Management Console o al portale di AWS accesso. Per ulteriori informazioni sull'accesso a AWS, vedi Come accedere al tuo Account AWS nella Guida per l'Accedi ad AWS utente.

Se accedi a AWS livello di codice, AWS fornisce un kit di sviluppo software (SDK) e un'interfaccia a riga di comando (CLI) per firmare crittograficamente le tue richieste utilizzando le tue credenziali. Se non utilizzi AWS strumenti, devi firmare tu stesso le richieste. Per ulteriori informazioni sul metodo consigliato per la firma delle richieste, consulta Signature Version 4 AWS per le richieste API nella Guida per l'utente IAM.

A prescindere dal metodo di autenticazione utilizzato, potrebbe essere necessario specificare ulteriori informazioni sulla sicurezza. Ad esempio, ti AWS consiglia di utilizzare l'autenticazione a più fattori (MFA) per aumentare la sicurezza del tuo account. Per ulteriori informazioni, consulta Autenticazione a più fattori nella Guida per l'utente di AWS IAM Identity Center e Utilizzo dell'autenticazione a più fattori (MFA)AWS in IAM nella Guida per l'utente IAM.

Account AWS utente root

Quando si crea un account Account AWS, si inizia con un'identità di accesso che ha accesso completo a tutte Servizi AWS le risorse dell'account. Questa identità è denominata utente Account AWS root ed è accessibile effettuando l'accesso con l'indirizzo e-mail e la password utilizzati per creare l'account. Si consiglia vivamente di non utilizzare l'utente root per le attività quotidiane. Conserva le credenziali dell'utente root e utilizzale per eseguire le operazioni che solo l'utente root può eseguire. Per un elenco completo delle attività che richiedono l'accesso come utente root, consulta la sezione Attività che richiedono le credenziali dell'utente root nella Guida per l'utente IAM.

Identità federata

Come procedura consigliata, richiedi agli utenti umani, compresi gli utenti che richiedono l'accesso come amministratore, di utilizzare la federazione con un provider di identità per accedere Servizi AWS utilizzando credenziali temporanee.

Un'identità federata è un utente dell'elenco utenti aziendale, di un provider di identità Web AWS Directory Service, della directory Identity Center o di qualsiasi utente che accede utilizzando le Servizi AWS credenziali fornite tramite un'origine di identità. Quando le identità federate accedono Account AWS, assumono ruoli e i ruoli forniscono credenziali temporanee.

Per la gestione centralizzata degli accessi, consigliamo di utilizzare AWS IAM Identity Center. Puoi creare utenti e gruppi in IAM Identity Center oppure puoi connetterti e sincronizzarti con un set di utenti e gruppi nella tua fonte di identità per utilizzarli su tutte le tue applicazioni. Account AWS Per ulteriori informazioni su IAM Identity Center, consulta Cos'è IAM Identity Center? nella Guida per l'utente di AWS IAM Identity Center .

Utenti e gruppi IAM

Un utente IAM è un'identità interna Account AWS che dispone di autorizzazioni specifiche per una singola persona o applicazione. Ove possibile, consigliamo di fare affidamento a credenziali temporanee invece di creare utenti IAM con credenziali a lungo termine come le password e le chiavi di accesso. Tuttavia, se si hanno casi d'uso specifici che richiedono credenziali a lungo termine con utenti IAM, si consiglia di ruotare le chiavi di accesso. Per ulteriori informazioni, consulta la pagina Rotazione periodica delle chiavi di accesso per casi d'uso che richiedono credenziali a lungo termine nella Guida per l'utente IAM.

Un gruppo IAM è un'identità che specifica un insieme di utenti IAM. Non è possibile eseguire l'accesso come gruppo. È possibile utilizzare gruppi per specificare le autorizzazioni per più utenti

alla volta. I gruppi semplificano la gestione delle autorizzazioni per set di utenti di grandi dimensioni. Ad esempio, potresti avere un gruppo denominato IAMAdminse concedere a quel gruppo le autorizzazioni per amministrare le risorse IAM.

Gli utenti sono diversi dai ruoli. Un utente è associato in modo univoco a una persona o un'applicazione, mentre un ruolo è destinato a essere assunto da chiunque ne abbia bisogno. Gli utenti dispongono di credenziali a lungo termine permanenti, mentre i ruoli forniscono credenziali temporanee. Per ulteriori informazioni, consulta Casi d'uso per utenti IAM nella Guida per l'utente IAM.

Ruoli IAM

Un ruolo IAM è un'identità interna all'utente Account AWS che dispone di autorizzazioni specifiche. È simile a un utente IAM, ma non è associato a una persona specifica. Per assumere temporaneamente un ruolo IAM in AWS Management Console, puoi passare da un ruolo utente a un ruolo IAM (console). Puoi assumere un ruolo chiamando un'operazione AWS CLI o AWS API o utilizzando un URL personalizzato. Per ulteriori informazioni sui metodi per l'utilizzo dei ruoli, consulta Utilizzo di ruoli IAM nella Guida per l'utente IAM.

I ruoli IAM con credenziali temporanee sono utili nelle seguenti situazioni:

- Accesso utente federato: per assegnare le autorizzazioni a una identità federata, è possibile creare un ruolo e definire le autorizzazioni per il ruolo. Quando un'identità federata viene autenticata, l'identità viene associata al ruolo e ottiene le autorizzazioni da esso definite. Per ulteriori informazioni sulla federazione dei ruoli, consulta Create a role for a third-party identity provider (federation) nella Guida per l'utente IAM. Se utilizzi IAM Identity Center, configura un set di autorizzazioni. IAM Identity Center mette in correlazione il set di autorizzazioni con un ruolo in IAM per controllare a cosa possono accedere le identità dopo l'autenticazione. Per informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center .
- Autorizzazioni utente IAM temporanee: un utente IAM o un ruolo può assumere un ruolo IAM per ottenere temporaneamente autorizzazioni diverse per un'attività specifica.
- Accesso multi-account: è possibile utilizzare un ruolo IAM per permettere a un utente (un principale affidabile) con un account diverso di accedere alle risorse nell'account. I ruoli sono lo strumento principale per concedere l'accesso multi-account. Tuttavia, con alcuni Servizi AWS, è possibile allegare una policy direttamente a una risorsa (anziché utilizzare un ruolo come proxy). Per informazioni sulle differenze tra ruoli e policy basate su risorse per l'accesso multi-account, consulta Accesso a risorse multi-account in IAM nella Guida per l'utente IAM.

- Accesso a più servizi: alcuni Servizi AWS utilizzano le funzionalità di altri Servizi AWS. Ad esempio, quando effettui una chiamata in un servizio, è normale che quel servizio esegua applicazioni in Amazon EC2 o archivi oggetti in Amazon S3. Un servizio può eseguire questa operazione utilizzando le autorizzazioni dell'entità chiamante, utilizzando un ruolo di servizio o utilizzando un ruolo collegato al servizio.
  - Sessioni di accesso inoltrato (FAS): quando utilizzi un utente o un ruolo IAM per eseguire azioni AWS, sei considerato un principale. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra operazione in un servizio diverso. FAS utilizza le autorizzazioni del principale che chiama an Servizio AWS, combinate con la richiesta Servizio AWS per effettuare richieste ai servizi downstream. Le richieste FAS vengono effettuate solo quando un servizio riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse per essere completata. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le azioni. Per i dettagli delle policy relative alle richieste FAS, consulta Forward access sessions.
  - Ruolo di servizio: un ruolo di servizio è un ruolo IAM che un servizio assume per eseguire operazioni per tuo conto. Un amministratore IAM può creare, modificare ed eliminare un ruolo di servizio dall'interno di IAM. Per ulteriori informazioni, consulta la sezione Create a role to delegate permissions to an Servizio AWS nella Guida per l'utente IAM.
  - Ruolo collegato al servizio: un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un. Servizio AWS Il servizio può assumere il ruolo per eseguire un'azione per tuo conto. I ruoli collegati al servizio vengono visualizzati nel tuo account Account AWS e sono di proprietà del servizio. Un amministratore IAM può visualizzare le autorizzazioni per i ruoli collegati ai servizi, ma non modificarle.
- Applicazioni in esecuzione su Amazon EC2: puoi utilizzare un ruolo IAM per gestire le credenziali temporanee per le applicazioni in esecuzione su un' EC2 istanza e che AWS CLI effettuano richieste AWS API. Questa soluzione è preferibile alla memorizzazione delle chiavi di accesso all'interno dell' EC2 istanza. Per assegnare un AWS ruolo a un' EC2 istanza e renderlo disponibile per tutte le sue applicazioni, create un profilo di istanza collegato all'istanza. Un profilo di istanza contiene il ruolo e consente ai programmi in esecuzione sull' EC2 istanza di ottenere credenziali temporanee. Per ulteriori informazioni, consulta Utilizzare un ruolo IAM per concedere le autorizzazioni alle applicazioni in esecuzione su EC2 istanze Amazon nella IAM User Guide.

Gestione dell'accesso con policy

Puoi controllare l'accesso AWS creando policy e collegandole a AWS identità o risorse. Una policy è un oggetto AWS che, se associato a un'identità o a una risorsa, ne definisce le autorizzazioni.

AWS valuta queste politiche quando un principale (utente, utente root o sessione di ruolo) effettua una richiesta. Le autorizzazioni nelle policy determinano l'approvazione o il rifiuto della richiesta. La maggior parte delle politiche viene archiviata AWS come documenti JSON. Per ulteriori informazioni sulla struttura e sui contenuti dei documenti delle policy JSON, consulta Panoramica delle policy JSON nella Guida per l'utente IAM.

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire operazioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Per concedere agli utenti l'autorizzazione a eseguire operazioni sulle risorse di cui hanno bisogno, un amministratore IAM può creare policy IAM. L'amministratore può quindi aggiungere le policy IAM ai ruoli e gli utenti possono assumere i ruoli.

Le policy IAM definiscono le autorizzazioni relative a un'operazione, a prescindere dal metodo utilizzato per eseguirla. Ad esempio, supponiamo di disporre di una policy che consente l'operazione `iam:GetRole`. Un utente con tale policy può ottenere informazioni sul ruolo dall' AWS Management Console AWS CLI, dall'o dall' AWS API.

Policy basate sull'identità

Le policy basate su identità sono documenti di policy di autorizzazione JSON che è possibile allegare a un'identità (utente, gruppo di utenti o ruolo IAM). Tali policy definiscono le operazioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una policy basata su identità, consulta Definizione di autorizzazioni personalizzate IAM con policy gestite dal cliente nella Guida per l'utente IAM.

Le policy basate su identità possono essere ulteriormente classificate come policy inline o policy gestite. Le policy inline sono integrate direttamente in un singolo utente, gruppo o ruolo. Le politiche gestite sono politiche autonome che puoi allegare a più utenti, gruppi e ruoli nel tuo Account AWS. Le politiche gestite includono politiche AWS gestite e politiche gestite dai clienti. Per informazioni su come scegliere tra una policy gestita o una policy inline, consulta Scelta fra policy gestite e policy inline nella Guida per l'utente IAM.

Policy basate sulle risorse

Le policy basate su risorse sono documenti di policy JSON che è possibile collegare a una risorsa. Esempi di policy basate sulle risorse sono le policy di attendibilità dei ruoli IAM e le policy dei bucket Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi

possono utilizzarli per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le operazioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o. Servizi AWS

Le policy basate sulle risorse sono policy inline che si trovano in tale servizio. Non puoi utilizzare le policy AWS gestite di IAM in una policy basata sulle risorse.

Liste di controllo degli accessi () ACLs

Le liste di controllo degli accessi (ACLs) controllano quali principali (membri dell'account, utenti o ruoli) dispongono delle autorizzazioni per accedere a una risorsa. ACLs sono simili alle politiche basate sulle risorse, sebbene non utilizzino il formato del documento di policy JSON.

Amazon S3 e Amazon VPC sono esempi di servizi che supportano. AWS WAF ACLs Per ulteriori informazioni ACLs, consulta la panoramica della lista di controllo degli accessi (ACL) nella Amazon Simple Storage Service Developer Guide.

Altri tipi di policy

AWS supporta tipi di policy aggiuntivi e meno comuni. Questi tipi di policy possono impostare il numero massimo di autorizzazioni concesse dai tipi di policy più comuni.

- Limiti delle autorizzazioni: un limite delle autorizzazioni è una funzionalità avanzata nella quale si imposta il numero massimo di autorizzazioni che una policy basata su identità può concedere a un'entità IAM (utente o ruolo IAM). È possibile impostare un limite delle autorizzazioni per un'entità. Le autorizzazioni risultanti sono l'intersezione delle policy basate su identità dell'entità e i relativi limiti delle autorizzazioni. Le policy basate su risorse che specificano l'utente o il ruolo nel campo `Principal`sono condizionate dal limite delle autorizzazioni. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni sui limiti delle autorizzazioni, consulta Limiti delle autorizzazioni per le entità IAM nella Guida per l'utente IAM.
- Politiche di controllo del servizio (SCPs): SCPs sono politiche JSON che specificano le autorizzazioni massime per un'organizzazione o un'unità organizzativa (OU) in. AWS Organizations AWS Organizations è un servizio per il raggruppamento e la gestione centralizzata di più di proprietà dell' Account AWS azienda. Se abiliti tutte le funzionalità di un'organizzazione, puoi applicare le politiche di controllo del servizio (SCPs) a uno o tutti i tuoi account. L'SCP limita le autorizzazioni per le entità presenti negli account dei membri, inclusa ciascuna di esse. Utente root dell'account AWS Per ulteriori informazioni su Organizations and SCPs, consulta le politiche di controllo dei servizi nella Guida AWS Organizations per l'utente.

- Politiche di controllo delle risorse (RCPs): RCPs sono politiche JSON che puoi utilizzare per impostare le autorizzazioni massime disponibili per le risorse nei tuoi account senza aggiornare le politiche IAM allegate a ciascuna risorsa di tua proprietà. L'RCP limita le autorizzazioni per le risorse negli account dei membri e può influire sulle autorizzazioni effettive per le identità, incluse le Utente root dell'account AWS, indipendentemente dal fatto che appartengano o meno all'organizzazione. Per ulteriori informazioni su Organizations e RCPs, incluso un elenco di Servizi AWS tale supporto RCPs, vedere Resource control policies (RCPs) nella Guida per l'AWS Organizations utente.
- Policy di sessione: le policy di sessione sono policy avanzate che vengono trasmesse come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un utente federato. Le autorizzazioni della sessione risultante sono l'intersezione delle policy basate su identità del ruolo o dell'utente e le policy di sessione. Le autorizzazioni possono anche provenire da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni, consulta Policy di sessione nella Guida per l'utente IAM.

Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per scoprire come si AWS determina se consentire o meno una richiesta quando sono coinvolti più tipi di policy, consulta la logica di valutazione delle policy nella IAM User Guide.

Come AWS Proton funziona con IAM

Prima di utilizzare IAM per gestire l'accesso a AWS Proton, scopri con quali funzionalità IAM è disponibile l'uso AWS Proton.

Funzionalità IAM che puoi utilizzare con AWS Proton

Funzionalità IAM	AWS Proton supporto
Policy basate su identità	Sì
Policy basate su risorse	No
Azioni di policy	Sì
Risorse relative alle policy	Sì

Funzionalità IAM	AWS Proton supporto
Chiavi di condizione delle policy	Sì
ACLs	No
ABAC (tag nelle policy)	Sì
Credenziali temporanee	Sì
Autorizzazioni del principale	Sì
Ruoli di servizio	Sì
Ruoli collegati al servizio	Sì

Per avere una panoramica generale di come AWS Proton e altri Servizi AWS utilizzi la maggior parte delle funzionalità IAM, consulta la sezione dedicata alla Servizi AWS compatibilità con IAM nella IAM User Guide.

Politiche basate sull'identità per AWS Proton

Supporta le policy basate su identità: sì

Le policy basate su identità sono documenti di policy di autorizzazione JSON che è possibile allegare a un'identità (utente, gruppo di utenti o ruolo IAM). Tali policy definiscono le operazioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una policy basata su identità, consulta Definizione di autorizzazioni personalizzate IAM con policy gestite dal cliente nella Guida per l'utente IAM.

Con le policy basate su identità di IAM, è possibile specificare quali operazioni e risorse sono consentite o respinte, nonché le condizioni in base alle quali le operazioni sono consentite o respinte. Non è possibile specificare l'entità principale in una policy basata sull'identità perché si applica all'utente o al ruolo a cui è associato. Per informazioni su tutti gli elementi utilizzabili in una policy JSON, consulta Guida di riferimento agli elementi delle policy JSON IAM nella Guida per l'utente di IAM.

Esempi di politiche basate sull'identità per AWS Proton

Per visualizzare esempi di politiche basate sull' AWS Proton identità, vedere. Esempi di policy basate sull'identità per AWS Proton

Politiche basate sulle risorse all'interno AWS Proton

Supporta le policy basate su risorse: no

Le policy basate su risorse sono documenti di policy JSON che è possibile collegare a una risorsa. Esempi di policy basate sulle risorse sono le policy di attendibilità dei ruoli IAM e le policy dei bucket Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarli per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le operazioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o. Servizi AWS

Per consentire l'accesso multi-account, puoi specificare un intero account o entità IAM in un altro account come principale in una policy basata sulle risorse. L'aggiunta di un principale multi-account a una policy basata sulle risorse rappresenta solo una parte della relazione di trust. Quando il principale e la risorsa sono diversi Account AWS, un amministratore IAM dell'account affidabile deve inoltre concedere all'entità principale (utente o ruolo) l'autorizzazione ad accedere alla risorsa. L'autorizzazione viene concessa collegando all'entità una policy basata sull'identità. Tuttavia, se una policy basata su risorse concede l'accesso a un principale nello stesso account, non sono richieste ulteriori policy basate su identità. Per ulteriori informazioni, consulta Accesso a risorse multi-account in IAM nella Guida per l'utente IAM.

Azioni politiche per AWS Proton

Supporta le operazioni di policy: si

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento `Action`di una policy JSON descrive le operazioni che è possibile utilizzare per consentire o negare l'accesso a un criterio. Le azioni politiche in genere hanno lo stesso nome dell'operazione AWS API associata. Ci sono alcune eccezioni, ad esempio le operazioni di sola autorizzazione che non hanno un'operazione API corrispondente. Esistono anche alcune operazioni che richiedono più operazioni in una policy. Queste operazioni aggiuntive sono denominate operazioni dipendenti.

Includi le operazioni in una policy per concedere le autorizzazioni a eseguire l'operazione associata.

Per visualizzare un elenco di AWS Proton azioni, vedere Azioni definite da AWS Proton nel Service Authorization Reference.

Le azioni politiche in AWS Proton uso utilizzano il seguente prefisso prima dell'azione:

```
proton
```

Per specificare più operazioni in una sola istruzione, occorre separarle con la virgola.

```
"Action": [
      "proton:action1",
      "proton:action2"
         ]
```

È possibile specificare più azioni tramite caratteri jolly (*). Ad esempio, per specificare tutte le azioni che iniziano con la parola `List`, includi la seguente azione:

```
"Action": "proton:List*"
```

Per visualizzare esempi di politiche AWS Proton basate sull'identità, vedere. Esempi di policy basate sull'identità per AWS Proton

Risorse politiche per AWS Proton

Supporta le risorse di policy: sì

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento JSON `Resource`della policy specifica l'oggetto o gli oggetti ai quali si applica l'operazione. Le istruzioni devono includere un elemento `Resource`o un elemento `NotResource`. Come best practice, specifica una risorsa utilizzando il suo nome della risorsa Amazon (ARN). È possibile eseguire questa operazione per operazioni che supportano un tipo di risorsa specifico, note come autorizzazioni a livello di risorsa.

Per le azioni che non supportano le autorizzazioni a livello di risorsa, ad esempio le operazioni di elenco, utilizza un carattere jolly (*) per indicare che l'istruzione si applica a tutte le risorse.

```
"Resource": "*"
```

Per visualizzare un elenco dei tipi di AWS Proton risorse e relativi ARNs, vedere Resources defined by AWS Proton nel Service Authorization Reference. Per informazioni sulle operazioni con cui è possibile specificare l'ARN di ogni risorsa, consulta la sezione Operazioni definite da AWS Proton.

Per visualizzare esempi di politiche AWS Proton basate sull'identità, vedere. Esempi di policy basate sull'identità per AWS Proton

Chiavi relative alle condizioni delle politiche per AWS Proton

Supporta le chiavi di condizione delle policy specifiche del servizio: sì

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento `Condition`(o blocco `Condition`) consente di specificare le condizioni in cui un'istruzione è in vigore. L'elemento `Condition`è facoltativo. È possibile compilare espressioni condizionali che utilizzano operatori di condizione, ad esempio uguale a o minore di, per soddisfare la condizione nella policy con i valori nella richiesta.

Se specifichi più elementi `Condition`in un'istruzione o più chiavi in un singolo elemento `Condition`, questi vengono valutati da AWS utilizzando un'operazione `AND`logica. Se si specificano più valori per una singola chiave di condizione, AWS valuta la condizione utilizzando un'operazione logica. `OR` Tutte le condizioni devono essere soddisfatte prima che le autorizzazioni dell'istruzione vengano concesse.

È possibile anche utilizzare variabili segnaposto quando specifichi le condizioni. Ad esempio, è possibile autorizzare un utente IAM ad accedere a una risorsa solo se è stata taggata con il relativo nome utente IAM. Per ulteriori informazioni, consulta Elementi delle policy IAM: variabili e tag nella Guida per l'utente di IAM.

AWS supporta chiavi di condizione globali e chiavi di condizione specifiche del servizio. Per visualizzare tutte le chiavi di condizione AWS globali, consulta le chiavi di contesto delle condizioni AWS globali nella Guida per l'utente IAM.

Per visualizzare un elenco di chiavi di AWS Proton condizione, consulta Condition keys for AWS Proton nel Service Authorization Reference. Per sapere con quali azioni e risorse puoi utilizzare una chiave di condizione, vedi Azioni definite da AWS Proton.

Per visualizzare un esempio di condition-key-based politica per limitare l'accesso a una risorsa, consultaEsempi di politiche basate su chiavi condizionali per AWS Proton.

Liste di controllo degli accessi (ACLs) in AWS Proton

Supporti ACLs: no

Le liste di controllo degli accessi (ACLs) controllano quali principali (membri dell'account, utenti o ruoli) dispongono delle autorizzazioni per accedere a una risorsa. ACLs sono simili alle politiche basate sulle risorse, sebbene non utilizzino il formato del documento di policy JSON.

Le liste di controllo degli accessi (ACLs) sono elenchi di assegnatari che è possibile allegare alle risorse. Essi concedono le autorizzazioni di accesso alla risorsa a cui sono associati.

Controllo degli accessi basato sugli attributi (ABAC) con AWS Proton

Supporta ABAC (tag nelle policy): sì

Il controllo dell'accesso basato su attributi (ABAC) è una strategia di autorizzazione che definisce le autorizzazioni in base agli attributi. In AWS, questi attributi sono chiamati tag. Puoi allegare tag a entità IAM (utenti o ruoli) e a molte AWS risorse. L'assegnazione di tag alle entità e alle risorse è il primo passaggio di ABAC. In seguito, vengono progettate policy ABAC per consentire operazioni quando il tag dell'entità principale corrisponde al tag sulla risorsa a cui si sta provando ad accedere.

La strategia ABAC è utile in ambienti soggetti a una rapida crescita e aiuta in situazioni in cui la gestione delle policy diventa impegnativa.

Per controllare l'accesso basato su tag, fornisci informazioni sui tag nell'elemento condizione di una policy utilizzando le chiavi di condizione `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`o `aws:TagKeys`.

Se un servizio supporta tutte e tre le chiavi di condizione per ogni tipo di risorsa, il valore per il servizio è Yes (Sì). Se un servizio supporta tutte e tre le chiavi di condizione solo per alcuni tipi di risorsa, allora il valore sarà Parziale.

Per ulteriori informazioni su ABAC, consulta Definizione delle autorizzazioni con autorizzazione ABAC nella Guida per l'utente IAM. Per visualizzare un tutorial con i passaggi per l'impostazione di ABAC, consulta Utilizzo del controllo degli accessi basato su attributi (ABAC) nella Guida per l'utente di IAM.

Per ulteriori informazioni sull'etichettatura AWS Proton delle risorse, consultaAWS Proton risorse e etichettatura.

Utilizzo di credenziali temporanee con AWS Proton

Supporta le credenziali temporanee: sì

Alcune Servizi AWS non funzionano quando si accede utilizzando credenziali temporanee. Per ulteriori informazioni, incluse quelle che Servizi AWS funzionano con credenziali temporanee, consulta la sezione relativa alla Servizi AWS compatibilità con IAM nella IAM User Guide.

Stai utilizzando credenziali temporanee se accedi AWS Management Console utilizzando qualsiasi metodo tranne nome utente e password. Ad esempio, quando accedi AWS utilizzando il link Single Sign-On (SSO) della tua azienda, tale processo crea automaticamente credenziali temporanee. Le credenziali temporanee vengono create in automatico anche quando accedi alla console come utente e poi cambi ruolo. Per ulteriori informazioni sullo scambio dei ruoli, consulta Passaggio da un ruolo utente a un ruolo IAM (console) nella Guida per l'utente IAM.

È possibile creare manualmente credenziali temporanee utilizzando l'API or. AWS CLI AWS È quindi possibile utilizzare tali credenziali temporanee per accedere. AWS AWS consiglia di generare dinamicamente credenziali temporanee anziché utilizzare chiavi di accesso a lungo termine. Per ulteriori informazioni, consulta Credenziali di sicurezza provvisorie in IAM.

Autorizzazioni principali multiservizio per AWS Proton

Supporta l'inoltro delle sessioni di accesso (FAS): sì

Quando utilizzi un utente o un ruolo IAM per eseguire azioni AWS, sei considerato un principale. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra operazione in un servizio diverso. FAS utilizza le autorizzazioni del principale che chiama an Servizio AWS, in combinazione con la richiesta Servizio AWS per effettuare richieste ai servizi downstream. Le richieste FAS vengono effettuate solo quando un servizio riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse per essere completata. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le azioni. Per i dettagli delle policy relative alle richieste FAS, consulta Forward access sessions.

Ruoli di servizio per AWS Proton

Supporta i ruoli di servizio: sì

Un ruolo di servizio è un ruolo IAM che un servizio assume per eseguire operazioni per tuo conto. Un amministratore IAM può creare, modificare ed eliminare un ruolo di servizio dall'interno di IAM. Per ulteriori informazioni, consulta la sezione Create a role to delegate permissions to an Servizio AWS nella Guida per l'utente IAM.

Per ulteriori informazioni, consulta AWS Proton Esempi di policy relative ai ruoli di servizio IAM.

> **Warning**
>
> La modifica delle autorizzazioni per un ruolo di servizio potrebbe compromettere AWS Proton la funzionalità. Modifica i ruoli di servizio solo quando viene AWS Proton fornita una guida in tal senso.

Ruoli collegati ai servizi per AWS Proton

Supporta ruoli collegati ai servizi: Sì

Un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un. Servizio AWS Il servizio può assumere il ruolo per eseguire un'azione per tuo conto. I ruoli collegati al servizio vengono visualizzati in Account AWS e sono di proprietà del servizio. Un amministratore IAM può visualizzare le autorizzazioni per i ruoli collegati ai servizi, ma non modificarle.

Per ulteriori informazioni, consulta Utilizzo di ruoli collegati ai servizi per AWS Proton.

Esempi di policy per AWS Proton

Trova esempi di policy AWS Proton IAM nelle seguenti sezioni.

Argomenti

- Esempi di policy basate sull'identità per AWS Proton
- AWS Proton Esempi di policy relative ai ruoli di servizio IAM
- Esempi di politiche basate su chiavi condizionali per AWS Proton

Esempi di policy basate sull'identità per AWS Proton

Per impostazione predefinita, gli utenti e i ruoli non dispongono dell'autorizzazione per creare o modificare risorse AWS Proton . Inoltre, non possono eseguire attività utilizzando AWS Management Console, AWS Command Line Interface (AWS CLI) o l'API. AWS Per concedere agli utenti l'autorizzazione a eseguire operazioni sulle risorse di cui hanno bisogno, un amministratore IAM può creare policy IAM. L'amministratore può quindi aggiungere le policy IAM ai ruoli e gli utenti possono assumere i ruoli.

Per informazioni su come creare una policy basata su identità IAM utilizzando questi documenti di policy JSON di esempio, consulta Creazione di policy IAM (console) nella Guida per l'utente IAM.

Per informazioni dettagliate sulle azioni e sui tipi di risorse definiti da AWS Proton, incluso il formato di ARNs per ogni tipo di risorsa, consulta Azioni, risorse e chiavi di condizione AWS Proton nel Service Authorization Reference.

Argomenti

- Best practice per le policy
- Collegamenti a esempi di policy basate sull'identità per AWS Proton

Best practice per le policy

Le politiche basate sull'identità determinano se qualcuno può creare, accedere o eliminare AWS Proton risorse nel tuo account. Queste azioni possono comportare costi aggiuntivi per l' Account AWS. Quando crei o modifichi policy basate su identità, segui queste linee guida e raccomandazioni:

- Inizia con le policy AWS gestite e passa alle autorizzazioni con privilegi minimi: per iniziare a concedere autorizzazioni a utenti e carichi di lavoro, utilizza le politiche gestite che concedono le autorizzazioni per molti casi d'uso comuni.AWS Sono disponibili nel tuo. Account AWS Ti consigliamo di ridurre ulteriormente le autorizzazioni definendo politiche gestite dai AWS clienti specifiche per i tuoi casi d'uso. Per ulteriori informazioni, consulta Policy gestite da AWSo Policy gestite da AWS per le funzioni dei processi nella Guida per l'utente IAM.
- Applica le autorizzazioni con privilegio minimo: quando imposti le autorizzazioni con le policy IAM, concedi solo le autorizzazioni richieste per eseguire un'attività. È possibile farlo definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come autorizzazioni con privilegi minimi. Per ulteriori informazioni sull'utilizzo di IAM per applicare le autorizzazioni, consulta Policy e autorizzazioni in IAM nella Guida per l'utente IAM.
- Condizioni d'uso nelle policy IAM per limitare ulteriormente l'accesso: per limitare l'accesso a operazioni e risorse è possibile aggiungere una condizione alle tue policy. Ad esempio, è possibile scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzando SSL. Puoi anche utilizzare le condizioni per concedere l'accesso alle azioni del servizio se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio AWS CloudFormation. Per ulteriori informazioni, consulta la sezione Elementi delle policy JSON di IAM: condizione nella Guida per l'utente IAM.
- Utilizzo di IAM Access Analyzer per convalidare le policy IAM e garantire autorizzazioni sicure e funzionali: IAM Access Analyzer convalida le policy nuove ed esistenti in modo che aderiscano alla

sintassi della policy IAM (JSON) e alle best practice di IAM. IAM Access Analyzer offre oltre 100 controlli delle policy e consigli utili per creare policy sicure e funzionali. Per ulteriori informazioni, consulta Convalida delle policy per il Sistema di analisi degli accessi IAM nella Guida per l'utente IAM.

- Richiedi l'autenticazione a più fattori (MFA): se hai uno scenario che richiede utenti IAM o un utente root nel Account AWS tuo, attiva l'MFA per una maggiore sicurezza. Per richiedere la MFA quando vengono chiamate le operazioni API, aggiungi le condizioni MFA alle policy. Per ulteriori informazioni, consulta Protezione dell'accesso API con MFA nella Guida per l'utente IAM.

Per maggiori informazioni sulle best practice in IAM, consulta Best practice di sicurezza in IAM nella Guida per l'utente di IAM.

Collegamenti a esempi di policy basate sull'identità per AWS Proton

Collegamenti ad esempi di politiche basate sull'identità per AWS Proton

- AWS politiche gestite per AWS Proton
- AWS Proton Esempi di policy relative ai ruoli di servizio IAM
- Esempi di politiche basate su chiavi condizionali per AWS Proton

AWS Proton Esempi di policy relative ai ruoli di servizio IAM

Gli amministratori possiedono e gestiscono le risorse AWS Proton create dall'ambiente e dai modelli di servizio. Associano ruoli di servizio IAM al proprio account che consentono AWS Proton di creare risorse per loro conto. Gli amministratori forniscono i ruoli e AWS Key Management Service le chiavi IAM per le risorse che vengono successivamente possedute e gestite dagli sviluppatori quando AWS Proton distribuiscono la loro applicazione come AWS Proton servizio in un AWS Proton ambiente. Per ulteriori informazioni sulla AWS KMS crittografia dei dati, consulta. Protezione dei dati in AWS Proton

Un ruolo di servizio è un ruolo di Amazon Web Services (IAM) che consente di AWS Proton effettuare chiamate alle risorse per tuo conto. Se specifichi un ruolo del servizio, AWS Proton utilizza le credenziali del ruolo. Utilizza un ruolo di servizio per specificare in modo esplicito le azioni che è AWS Proton possibile eseguire.

Puoi creare il ruolo del servizio e le sue policy di autorizzazione con il servizio IAM. Per ulteriori informazioni sulla creazione di un ruolo di servizio, consulta Creating a role to delegate permissions to an AWS service nella IAM User Guide.

AWS Proton ruolo di servizio per il provisioning utilizzando AWS CloudFormation

In qualità di membro del team della piattaforma, in qualità di amministratore puoi creare un ruolo di AWS Proton servizio e assegnarlo AWS Proton quando crei un ambiente come ruolo di CloudFormation servizio dell'ambiente (il `protonServiceRoleArn` parametro dell'azione dell'CreateEnvironmentAPI). Questo ruolo consente di AWS Proton effettuare chiamate API ad altri servizi per conto dell'utente quando l'ambiente o una qualsiasi delle istanze di servizio in esso eseguite utilizzano il provisioning AWS gestito e AWS CloudFormation il provisioning dell'infrastruttura.

Ti consigliamo di utilizzare i seguenti ruoli IAM e la policy di fiducia per il tuo AWS Proton ruolo di servizio. Quando utilizzi la AWS Proton console per creare un ambiente e scegli di creare un nuovo ruolo, questa è la politica che si AWS Proton aggiunge al ruolo di servizio che crea per te. Quando definisci l'ambito delle autorizzazioni relative a questa politica, tieni presente che AWS Proton non funziona in caso di `Access Denied` errori.

> **Important**
>
> Tieni presente che le politiche mostrate negli esempi seguenti concedono i privilegi di amministratore a chiunque possa registrare un modello nel tuo account. Poiché non sappiamo quali risorse definirai nei tuoi AWS Proton modelli, queste politiche dispongono di autorizzazioni ampie. Ti consigliamo di limitare le autorizzazioni alle risorse specifiche che verranno distribuite nei tuoi ambienti.

AWS Proton esempio di politica relativa al ruolo di servizio per AWS CloudFormation

*123456789012*Sostituiscilo con il tuo Account AWS ID.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:CancelUpdateStack",
        "cloudformation:ContinueUpdateRollback",
        "cloudformation:CreateChangeSet",
        "cloudformation:CreateStack",
        "cloudformation:DeleteChangeSet",
        "cloudformation:DeleteStack",
```

```
        "cloudformation:DescribeChangeSet",
        "cloudformation:DescribeStackDriftDetectionStatus",
        "cloudformation:DescribeStackEvents",
        "cloudformation:DescribeStackResourceDrifts",
        "cloudformation:DescribeStacks",
        "cloudformation:DetectStackResourceDrift",
        "cloudformation:ExecuteChangeSet",
        "cloudformation:ListChangeSets",
        "cloudformation:ListStackResources",
        "cloudformation:UpdateStack"
      ],
      "Resource": "arn:aws:cloudformation:*:123456789012:stack/AWSProton-*"
    },
    {
      "Effect": "Allow",
      "NotAction": [
        "organizations:*",
        "account:*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": [
            "cloudformation.amazonaws.com"
          ]
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "organizations:DescribeOrganization",
        "account:ListRegions"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": [
            "cloudformation.amazonaws.com"
          ]
        }
      }
    }
  ]
```

```
}
```

AWS Proton politica di fiducia del servizio

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ServiceTrustRelationshipWithConfusedDeputyPrevention",
    "Effect": "Allow",
    "Principal": {
      "Service": "proton.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws::proton:*:123456789012:environment/*"
      }
    }
  }
}
```

Politica relativa al ruolo del servizio AWS di provisioning gestito e dettagliata

Di seguito è riportato un esempio di policy relativa ai ruoli di AWS Proton servizio con un ambito specifico che puoi utilizzare se hai bisogno solo di AWS Proton servizi per il provisioning delle risorse S3.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:CancelUpdateStack",
        "cloudformation:ContinueUpdateRollback",
        "cloudformation:CreateChangeSet",
        "cloudformation:CreateStack",
        "cloudformation:DeleteChangeSet",
        "cloudformation:DeleteStack",
        "cloudformation:DescribeChangeSet",
```

```
        "cloudformation:DescribeStackDriftDetectionStatus",
        "cloudformation:DescribeStackEvents",
        "cloudformation:DescribeStackResourceDrifts",
        "cloudformation:DescribeStacks",
        "cloudformation:DetectStackResourceDrift",
        "cloudformation:ExecuteChangeSet",
        "cloudformation:ListChangeSets",
        "cloudformation:ListStackResources",
        "cloudformation:UpdateStack"
      ],
      "Resource": "arn:aws:cloudformation:*:123456789012:stack/AWSProton-*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:*"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": [
            "cloudformation.amazonaws.com"
          ]
        }
      }
    }
  ]
}
```

AWS Proton ruolo di servizio per il provisioning CodeBuild

In qualità di membro del team della piattaforma, in qualità di amministratore puoi creare un ruolo di AWS Proton servizio e assegnarlo AWS Proton quando crei un ambiente come ruolo di CodeBuild servizio dell'ambiente (il `codebuildRoleArn` parametro dell'azione CreateEnvironmentAPI). Questo ruolo consente di AWS Proton effettuare chiamate API ad altri servizi per conto dell'utente quando l'ambiente o una qualsiasi delle istanze di servizio in esso eseguite utilizzano il CodeBuild provisioning per fornire l'infrastruttura.

Quando utilizzi la AWS Proton console per creare un ambiente e scegli di creare un nuovo ruolo, AWS Proton aggiunge una policy con privilegi di amministratore al ruolo di servizio che crea per te. Quando create il vostro ruolo e limitate le autorizzazioni, tenete presente che AWS Proton non funziona in `Access Denied` caso di errori.

> **Important**
>
> Tieni presente che le politiche associate ai AWS Proton ruoli che crea per te concedono i privilegi di amministratore a chiunque possa registrare un modello nel tuo account. Poiché non sappiamo quali risorse definirai nei tuoi AWS Proton modelli, queste politiche dispongono di autorizzazioni ampie. Ti consigliamo di limitare le autorizzazioni alle risorse specifiche che verranno distribuite nei tuoi ambienti.

AWS Proton esempio di politica relativa al ruolo di servizio per CodeBuild

L'esempio seguente fornisce le autorizzazioni CodeBuild per il provisioning di risorse utilizzando. AWS Cloud Development Kit (AWS CDK)

*123456789012*Sostituiscilo con il tuo Account AWS ID.

```
{
 "Version": "2012-10-17",
 "Statement": [
   {
     "Action": [
       "logs:CreateLogStream",
       "logs:CreateLogGroup",
       "logs:PutLogEvents"
     ],
     "Resource": [
       "arn:aws:logs:us-east-1:123456789012:log-group:/aws/codebuild/AWSProton-
Shell-*",
       "arn:aws:logs:us-east-1:123456789012:log-group:/aws/codebuild/AWSProton-
Shell-*:*"
     ],
     "Effect": "Allow"
   },
   {
     "Action": "proton:NotifyResourceDeploymentStatusChange",
     "Resource": "arn:aws:proton:us-east-1:123456789012:*",
     "Effect": "Allow"
   },
   {
     "Action": "sts:AssumeRole",
     "Resource": [
       "arn:aws:iam::123456789012:role/cdk-*-deploy-role-*",
```

```
        "arn:aws:iam::123456789012:role/cdk-*-file-publishing-role-*"
      ],
      "Effect": "Allow"
    }
  ]
}
```

AWS Proton CodeBuild politica di fiducia

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "CodeBuildTrustRelationshipWithConfusedDeputyPrevention",
    "Effect": "Allow",
    "Principal": {
      "Service": "codebuild.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws::proton:*:123456789012:environment/*"
      }
    }
  }
}
```

AWS Proton ruoli del servizio di pipeline

Per fornire le pipeline di servizi, sono AWS Proton necessarie le autorizzazioni per effettuare chiamate API ad altri servizi. I ruoli di servizio richiesti sono simili ai ruoli di servizio forniti quando si creano ambienti. Tuttavia, i ruoli per la creazione di pipeline sono condivisi tra tutti i servizi del tuo AWS account e li fornisci come impostazioni dell'account nella console o tramite l'azione UpdateAccountSettingsAPI.

Quando utilizzi la AWS Proton console per aggiornare le impostazioni dell'account e scegli di creare un nuovo ruolo per i ruoli AWS CloudFormation o per i ruoli di CodeBuild servizio, le politiche che si AWS Proton aggiungono ai ruoli di servizio che crea per te sono le stesse descritte nelle sezioni precedenti AWS-ruolo di provisioning gestito eCodeBuild ruolo di approvvigionamento. Quando

definisci l'ambito delle autorizzazioni relative a questa politica, tieni presente che AWS Proton non funziona in caso di `Access Denied` errori.

> **Important**
>
> Tieni presente che le politiche di esempio nelle sezioni precedenti concedono i privilegi di amministratore a chiunque possa registrare un modello nel tuo account. Poiché non sappiamo quali risorse definirai nei tuoi AWS Proton modelli, queste politiche hanno autorizzazioni ampie. Ti consigliamo di limitare le autorizzazioni alle risorse specifiche che verranno distribuite nelle tue pipeline.

AWS Proton ruolo del componente

In qualità di membro del team della piattaforma, in qualità di amministratore puoi creare un ruolo di AWS Proton servizio e assegnarlo al AWS Proton momento della creazione di un ambiente come ruolo CloudFormation componente dell'ambiente (il `componentRoleArn` parametro dell'azione CreateEnvironmentAPI). Questo ruolo definisce l'infrastruttura che i componenti definiti direttamente possono fornire. Per ulteriori informazioni sui componenti, vedereComponenti.

La seguente policy di esempio supporta la creazione di un componente definito direttamente che fornisce un bucket Amazon Simple Storage Service (Amazon S3) e una policy di accesso correlata.

AWS Proton esempio di politica relativa al ruolo dei componenti

*123456789012*Sostituiscilo con il tuo Account AWS ID.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:CancelUpdateStack",
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:DescribeStacks",
        "cloudformation:ContinueUpdateRollback",
        "cloudformation:DetectStackResourceDrift",
        "cloudformation:DescribeStackResourceDrifts",
        "cloudformation:DescribeStackEvents",
```

```
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:UpdateStack",
        "cloudformation:DescribeChangeSet",
        "cloudformation:ExecuteChangeSet",
        "cloudformation:ListChangeSets",
        "cloudformation:ListStackResources"
      ],
      "Resource": "arn:aws:cloudformation:*:123456789012:stack/AWSProton-*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:CreateBucket",
        "s3:DeleteBucket",
        "s3:GetBucket",
        "iam:CreatePolicy",
        "iam:DeletePolicy",
        "iam:GetPolicy",
        "iam:ListPolicyVersions",
        "iam:DeletePolicyVersion"
      ],
      "Resource": "*",
      "Condition": {
        "ForAnyValue:StringEquals": {
          "aws:CalledVia": "cloudformation.amazonaws.com"
        }
      }
    }
  ]
}
```

AWS Proton politica di attendibilità dei componenti

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ServiceTrustRelationshipWithConfusedDeputyPrevention",
    "Effect": "Allow",
    "Principal": {
      "Service": "proton.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
```

```
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws::proton:*:123456789012:environment/*"
      }
    }
  }
}
```

Esempi di politiche basate su chiavi condizionali per AWS Proton

Il seguente esempio di policy IAM nega l'accesso alle AWS Proton azioni che corrispondono ai modelli specificati nel blocco. `Condition` Tieni presente che queste chiavi di condizione sono supportate solo dalle azioni elencate in Actions, resources and condition keys for AWS Proton. Per gestire le autorizzazioni su altre azioni, ad esempio`DeleteEnvironmentTemplate`, è necessario utilizzare il controllo dell'accesso a livello di risorsa.

Esempio di politica che nega le azioni del modello su un AWS Proton modello specifico:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": ["proton:*"],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "proton:EnvironmentTemplate":
 ["arn:aws:proton:region_id:123456789012:environment-template/my-environment-template"]
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": ["proton:*"],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "proton:ServiceTemplate":
 ["arn:aws:proton:region_id:123456789012:service-template/my-service-template"]
```

```
                }
            }
        }
    ]
}
```

Nella seguente politica di esempio, la prima istruzione a livello di risorsa nega l'accesso alle azioni del AWS Proton modello, diverse da quelle `ListServiceTemplates` che corrispondono al modello di servizio elencato nel blocco. `Resource` La seconda istruzione nega l'accesso alle AWS Proton azioni che corrispondono al modello elencato nel blocco. `Condition`

Esempio di politica che nega le AWS Proton azioni che corrispondono a un modello specifico:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "proton:*"
            ],
            "Resource": "arn:aws:region_id:123456789012:service-template/my-service-
template"
        },
        {
            "Effect": "Deny",
            "Action": [
                "proton:*"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "proton:ServiceTemplate": [
                        "arn:aws:proton:region_id:123456789012:service-template/my-
service-template"
                    ]
                }
            }
        }
    ]
}
```

L'ultimo esempio di policy consente agli sviluppatori AWS Proton azioni che corrispondono allo specifico modello di servizio elencato nel `Condition` blocco.

Esempio di politica per consentire azioni di AWS Proton sviluppo che corrispondono a un modello specifico:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "proton:ListServiceTemplates",
                "proton:ListServiceTemplateVersions",
                "proton:ListServices",
                "proton:ListServiceInstances",
                "proton:ListEnvironments",
                "proton:GetServiceTemplate",
                "proton:GetServiceTemplateVersion",
                "proton:GetService",
                "proton:GetServiceInstance",
                "proton:GetEnvironment",
                "proton:CreateService",
                "proton:UpdateService",
                "proton:UpdateServiceInstance",
                "proton:UpdateServicePipeline",
                "proton:DeleteService",
                "codestar-connections:ListConnections"
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "proton:ServiceTemplate":
 "arn:aws:proton:region_id:123456789012:service-template/my-service-template"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "codestar-connections:PassConnection"
            ],
            "Resource": "arn:aws:codestar-connections:*:*:connection/*",
```

```
            "Condition": {
                "StringEquals": {
                    "codestar-connections:PassedToService": "proton.amazonaws.com"
                }
            }
        }

    ]
}
```

AWS politiche gestite per AWS Proton

Per aggiungere autorizzazioni a utenti, gruppi e ruoli, è più facile utilizzare le politiche AWS gestite che scrivere le politiche da soli. Creare policy gestite dal cliente IAM per fornire al tuo team solo le autorizzazioni di cui ha bisogno richiede tempo e competenza. Per iniziare rapidamente, puoi utilizzare le nostre politiche AWS gestite. Queste policy coprono i casi d'uso comuni e sono disponibili nel tuo Account AWS. Per ulteriori informazioni sulle policy AWS gestite, consulta le policy AWS gestite nella IAM User Guide.

Servizi AWS mantenere e aggiornare le politiche AWS gestite. Non è possibile modificare le autorizzazioni nelle politiche AWS gestite. I servizi occasionalmente aggiungono altre autorizzazioni a una policy gestita da AWS per supportare nuove funzionalità. Questo tipo di aggiornamento interessa tutte le identità (utenti, gruppi e ruoli) a cui è collegata la policy. È più probabile che i servizi aggiornino una policy gestita da AWS quando viene avviata una nuova funzionalità o quando diventano disponibili nuove operazioni. I servizi non rimuovono le autorizzazioni da una policy AWS gestita, quindi gli aggiornamenti delle policy non comprometteranno le autorizzazioni esistenti.

Inoltre, AWS supporta politiche gestite per le funzioni lavorative che si estendono su più servizi. Ad esempio, la policy ReadOnlyAccess AWS gestita fornisce l'accesso in sola lettura a tutte le Servizi AWS risorse. Quando un servizio avvia una nuova funzionalità, AWS aggiunge autorizzazioni di sola lettura per nuove operazioni e risorse. Per l'elenco e la descrizione delle policy di funzione dei processi, consulta la sezione Policy gestite da AWS per funzioni di processi nella Guida per l'utente di IAM.

AWS Proton fornisce politiche IAM gestite e relazioni di fiducia che è possibile associare a utenti, gruppi o ruoli che consentono diversi livelli di controllo sulle risorse e sulle operazioni delle API. Puoi applicare queste policy direttamente oppure utilizzarle come punto di partenza per la creazione di tue policy.

La seguente relazione di trust viene utilizzata per ciascuna delle policy AWS Proton gestite.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ExampleTrustRelationshipWithProtonConfusedDeputyPrevention",
    "Effect": "Allow",
    "Principal": {
      "Service": "proton.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      },
      "ArnLike": {
        "aws:SourceArn": "arn:aws::proton:*:123456789012:environment/*"
      }
    }
  }
}
```

AWS politica gestita: AWSProton FullAccess

Puoi collegarti `AWSProtonFullAccess` alle tue entità IAM. AWS Proton associa inoltre questa policy a un ruolo di servizio che consente di AWS Proton eseguire azioni per tuo conto.

Questa politica concede autorizzazioni amministrative che consentono l'accesso completo alle AWS Proton azioni e l'accesso limitato ad altre azioni di AWS servizio da cui AWS Proton dipende.

La policy include i seguenti namespace di azioni chiave:

- `proton`— Consente agli amministratori l'accesso completo a. AWS Proton APIs
- `iam`— Consente agli amministratori di passare ruoli a. AWS Proton Ciò è necessario per AWS Proton poter effettuare chiamate API ad altri servizi per conto dell'amministratore.
- `kms`— Consente agli amministratori di aggiungere una concessione a una chiave gestita dal cliente.
- `codeconnections`— Consente agli amministratori di elencare e trasmettere codici di connessione in modo che possano essere utilizzati da. AWS Proton

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ProtonPermissions",
            "Effect": "Allow",
            "Action": [
                "proton:*",
                "codestar-connections:ListConnections",
                "kms:ListAliases",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CreateGrantPermissions",
            "Effect": "Allow",
            "Action": [
                "kms:CreateGrant"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": "proton.*.amazonaws.com"
                }
            }
        },
        {
            "Sid": "PassRolePermissions",
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "proton.amazonaws.com"
                }
            }
        },
```

```
        {
            "Sid": "CreateServiceLinkedRolePermissions",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
sync.proton.amazonaws.com/AWSServiceRoleForProtonSync",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": "sync.proton.amazonaws.com"
                }
            }
        },
        {
            "Sid": "CodeStarConnectionsPermissions",
            "Effect": "Allow",
            "Action": [
                "codestar-connections:PassConnection"
            ],
            "Resource": [
                "arn:aws:codestar-connections:*:*:connection/*",
                "arn:aws:codeconnections:*:*:connection/*"
            ],
            "Condition": {
                "StringEquals": {
                    "codestar-connections:PassedToService": "proton.amazonaws.com"
                }
            }
        },
        {
            "Sid": "CodeConnectionsPermissions",
            "Effect": "Allow",
            "Action": [
                "codeconnections:PassConnection"
            ],
            "Resource": [
                "arn:aws:codestar-connections:*:*:connection/*",
                "arn:aws:codeconnections:*:*:connection/*"
            ],
            "Condition": {
                "StringEquals": {
                    "codeconnections:PassedToService": "proton.amazonaws.com"
                }
            }
        }
```

```
    ]
}
```

AWS politica gestita: AWSProton DeveloperAccess

Puoi collegarti `AWSProtonDeveloperAccess` alle tue entità IAM. AWS Proton associa inoltre questa policy a un ruolo di servizio che consente di AWS Proton eseguire azioni per tuo conto.

Questa politica concede autorizzazioni che consentono un accesso limitato alle AWS Proton azioni e ad altre AWS azioni che AWS Proton dipendono da. L'ambito di queste autorizzazioni è progettato per supportare il ruolo di uno sviluppatore che crea e distribuisce servizi. AWS Proton

Questa politica non fornisce l'accesso alla creazione, all'eliminazione e all'aggiornamento del AWS Proton modello e dell'ambiente. APIs Se gli sviluppatori necessitano di autorizzazioni ancora più limitate di quelle fornite da questa politica, consigliamo di creare una politica personalizzata con un ambito ristretto in modo da concedere il minimo privilegio.

La policy include i seguenti namespace di azioni chiave:

- `proton`— Consente ai collaboratori di accedere a un set limitato di. AWS Proton APIs
- `codeconnections`— Consente ai collaboratori di elencare e trasmettere connessioni di codice in modo che possano essere utilizzate da. AWS Proton

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ProtonPermissions",
            "Effect": "Allow",
            "Action": [
                "codecommit:ListRepositories",
                "codepipeline:GetPipeline",
                "codepipeline:GetPipelineExecution",
                "codepipeline:GetPipelineState",
                "codepipeline:ListPipelineExecutions",
                "codepipeline:ListPipelines",
                "codestar-connections:ListConnections",
```

```
                "codestar-connections:UseConnection",
                "proton:CancelServiceInstanceDeployment",
                "proton:CancelServicePipelineDeployment",
                "proton:CreateService",
                "proton:DeleteService",
                "proton:GetAccountRoles",
                "proton:GetAccountSettings",
                "proton:GetEnvironment",
                "proton:GetEnvironmentAccountConnection",
                "proton:GetEnvironmentTemplate",
                "proton:GetEnvironmentTemplateMajorVersion",
                "proton:GetEnvironmentTemplateMinorVersion",
                "proton:GetEnvironmentTemplateVersion",
                "proton:GetRepository",
                "proton:GetRepositorySyncStatus",
                "proton:GetResourcesSummary",
                "proton:GetService",
                "proton:GetServiceInstance",
                "proton:GetServiceTemplate",
                "proton:GetServiceTemplateMajorVersion",
                "proton:GetServiceTemplateMinorVersion",
                "proton:GetServiceTemplateVersion",
                "proton:GetTemplateSyncConfig",
                "proton:GetTemplateSyncStatus",
                "proton:ListEnvironmentAccountConnections",
                "proton:ListEnvironmentOutputs",
                "proton:ListEnvironmentProvisionedResources",
                "proton:ListEnvironments",
                "proton:ListEnvironmentTemplateMajorVersions",
                "proton:ListEnvironmentTemplateMinorVersions",
                "proton:ListEnvironmentTemplates",
                "proton:ListEnvironmentTemplateVersions",
                "proton:ListRepositories",
                "proton:ListRepositorySyncDefinitions",
                "proton:ListServiceInstanceOutputs",
                "proton:ListServiceInstanceProvisionedResources",
                "proton:ListServiceInstances",
                "proton:ListServicePipelineOutputs",
                "proton:ListServicePipelineProvisionedResources",
                "proton:ListServices",
                "proton:ListServiceTemplateMajorVersions",
                "proton:ListServiceTemplateMinorVersions",
                "proton:ListServiceTemplates",
                "proton:ListServiceTemplateVersions",
```

```
                "proton:ListTagsForResource",
                "proton:UpdateService",
                "proton:UpdateServiceInstance",
                "proton:UpdateServicePipeline",
                "s3:ListAllMyBuckets",
                "s3:ListBucket"
            ],
            "Resource": "*"
        },
        {
            "Sid": "CodeStarConnectionsPermissions",
            "Effect": "Allow",
            "Action": "codestar-connections:PassConnection",
            "Resource": [
                "arn:aws:codestar-connections:*:*:connection/*",
                "arn:aws:codeconnections:*:*:connection/*"
            ],
            "Condition": {
                "StringEquals": {
                    "codestar-connections:PassedToService": "proton.amazonaws.com"
                }
            }
        },
        {
            "Sid": "CodeConnectionsPermissions",
            "Effect": "Allow",
            "Action": "codeconnections:PassConnection",
            "Resource": [
                "arn:aws:codestar-connections:*:*:connection/*",
                "arn:aws:codeconnections:*:*:connection/*"
            ],
            "Condition": {
                "StringEquals": {
                    "codeconnections:PassedToService": "proton.amazonaws.com"
                }
            }
        }
    ]
}
```

AWS politica gestita: AWSProton ReadOnlyAccess

Puoi collegarti `AWSProtonReadOnlyAccess` alle tue entità IAM. AWS Proton associa inoltre questa policy a un ruolo di servizio che consente di AWS Proton eseguire azioni per tuo conto.

Questa politica concede autorizzazioni che consentono l'accesso in sola lettura alle AWS Proton azioni e l'accesso limitato in sola lettura ad altre azioni di servizio da cui dipende. AWS AWS Proton

La policy include i seguenti namespace di azioni chiave:

- `proton`— Consente ai collaboratori l'accesso in sola lettura a. AWS Proton APIs

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codepipeline:ListPipelineExecutions",
        "codepipeline:ListPipelines",
        "codepipeline:GetPipeline",
        "codepipeline:GetPipelineState",
        "codepipeline:GetPipelineExecution",
        "proton:GetAccountRoles",
        "proton:GetAccountSettings",
        "proton:GetEnvironment",
        "proton:GetEnvironmentAccountConnection",
        "proton:GetEnvironmentTemplate",
        "proton:GetEnvironmentTemplateMajorVersion",
        "proton:GetEnvironmentTemplateMinorVersion",
        "proton:GetEnvironmentTemplateVersion",
        "proton:GetRepository",
        "proton:GetRepositorySyncStatus",
        "proton:GetResourcesSummary",
        "proton:GetService",
        "proton:GetServiceInstance",
        "proton:GetServiceTemplate",
        "proton:GetServiceTemplateMajorVersion",
        "proton:GetServiceTemplateMinorVersion",
```

```
        "proton:GetServiceTemplateVersion",
        "proton:GetTemplateSyncConfig",
        "proton:GetTemplateSyncStatus",
        "proton:ListEnvironmentAccountConnections",
        "proton:ListEnvironmentOutputs",
        "proton:ListEnvironmentProvisionedResources",
        "proton:ListEnvironments",
        "proton:ListEnvironmentTemplateMajorVersions",
        "proton:ListEnvironmentTemplateMinorVersions",
        "proton:ListEnvironmentTemplates",
        "proton:ListEnvironmentTemplateVersions",
        "proton:ListRepositories",
        "proton:ListRepositorySyncDefinitions",
        "proton:ListServiceInstanceOutputs",
        "proton:ListServiceInstanceProvisionedResources",
        "proton:ListServiceInstances",
        "proton:ListServicePipelineOutputs",
        "proton:ListServicePipelineProvisionedResources",
        "proton:ListServices",
        "proton:ListServiceTemplateMajorVersions",
        "proton:ListServiceTemplateMinorVersions",
        "proton:ListServiceTemplates",
        "proton:ListServiceTemplateVersions",
        "proton:ListTagsForResource"
      ],
      "Resource": "*"
    }
  ]
}
```

AWS politica gestita: AWSProton SyncServiceRolePolicy

AWS Proton allega questa politica al ruolo AWSServiceRoleForProtonSynccollegato al servizio che consente di AWS Proton eseguire la sincronizzazione dei modelli.

Questa politica concede autorizzazioni che consentono un accesso limitato alle AWS Proton azioni e ad altre azioni di AWS servizio da cui dipende. AWS Proton

La policy include i seguenti namespace di azioni chiave:

- `proton`— Consente un accesso limitato alla AWS Proton sincronizzazione a. AWS Proton APIs
- `codeconnections`— Consente un accesso limitato alla AWS Proton sincronizzazione a CodeConnections APIs.

AWS politica gestita: AWSProton CodeBuildProvisioningBasicAccess

Le autorizzazioni CodeBuild devono eseguire una build per AWS Proton CodeBuild Provisioning. Puoi collegarti `AWSProtonCodeBuildProvisioningBasicAccess` al tuo ruolo di CodeBuild Provisioning.

Questa politica concede le autorizzazioni minime per il funzionamento del AWS Proton CodeBuild Provisioning. Concede autorizzazioni che consentono CodeBuild di generare registri di compilazione. Concede inoltre l'autorizzazione a Proton di rendere disponibili agli utenti gli output Infrastructure as Code (IaC). AWS Proton Non fornisce le autorizzazioni necessarie agli strumenti IaC per gestire l'infrastruttura.

La policy include i seguenti namespace di azioni chiave:

- `logs`- Consente di generare registri CodeBuild di compilazione. Senza questa autorizzazione, non CodeBuild riuscirà ad avviarsi.
- `proton`- Consente a un comando CodeBuild Provisioning di richiedere `aws proton notify-resource-deployment-status-change` l'aggiornamento degli output IaaC per una determinata risorsa. AWS Proton

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": [
    "logs:CreateLogStream",
    "logs:CreateLogGroup",
    "logs:PutLogEvents"
   ],
   "Resource": [
    "arn:aws:logs:*:*:log-group:/aws/codebuild/AWSProton-*"
   ]
  },
  {
   "Effect": "Allow",
   "Action": "proton:NotifyResourceDeploymentStatusChange",
```

```
    "Resource": "arn:aws:proton:*:*:*"
  }
 ]
}
```

AWS politica gestita: AWSProton CodeBuildProvisioningServiceRolePolicy

AWS Proton associa questa policy al ruolo AWSServiceRoleForProtonCodeBuildProvisioningcollegato al servizio che consente di AWS Proton eseguire CodeBuild il provisioning basato.

Questa politica concede autorizzazioni che consentono un accesso limitato alle azioni di servizio che dipendono da AWS . AWS Proton

La policy include i seguenti namespace di azioni chiave:

- `cloudformation`— Consente un accesso limitato al provisioning AWS Proton CodeBuild basato su. AWS CloudFormation APIs
- `codebuild`— Consente un accesso limitato al provisioning AWS Proton CodeBuild basato su. CodeBuild APIs
- `iam`— Consente agli amministratori di trasferire ruoli a. AWS Proton Ciò è necessario per AWS Proton poter effettuare chiamate API ad altri servizi per conto dell'amministratore.
- `servicequotas`— Consente di AWS Proton controllare il limite di CodeBuild compilazione simultanea, che garantisce una corretta coda di compilazione.

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "CloudFormationPermissions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:DeleteStack",
```

```
        "cloudformation:UpdateStack",
        "cloudformation:DescribeStacks",
        "cloudformation:DescribeStackEvents",
        "cloudformation:ListStackResources",
        "cloudformation:TagResource",
        "cloudformation:UntagResource"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stack/AWSProton-CodeBuild-*"
      ]
    },
    {
      "Sid": "CodeBuildPermissions",
      "Effect": "Allow",
      "Action": [
        "codebuild:CreateProject",
        "codebuild:DeleteProject",
        "codebuild:UpdateProject",
        "codebuild:StartBuild",
        "codebuild:StopBuild",
        "codebuild:RetryBuild",
        "codebuild:BatchGetBuilds",
        "codebuild:BatchGetProjects"
      ],
      "Resource": "arn:aws:codebuild:*:*:project/AWSProton*"
    },
    {
      "Sid": "PassRolePermissions",
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "*",
      "Condition": {
        "StringEqualsIfExists": {
          "iam:PassedToService": "codebuild.amazonaws.com"
        }
      }
    },
    {
      "Sid": "ServiceQuotaPermissions",
      "Effect": "Allow",
      "Action": [
          "servicequotas:GetServiceQuota"
      ],
      "Resource": "*"
```

```
    }
  ]
}
```

AWS politica gestita: AWSProton ServiceGitSyncServiceRolePolicy

AWS Proton allega questa policy al ruolo AWSServiceRoleForProtonServiceSynccollegato al servizio che consente di AWS Proton eseguire la sincronizzazione del servizio.

Questa politica concede autorizzazioni che consentono un accesso limitato alle AWS Proton azioni e ad altre azioni di AWS servizio da cui dipende. AWS Proton

La policy include i seguenti namespace di azioni chiave:

- `proton`— Consente un accesso limitato alla AWS Proton sincronizzazione a. AWS Proton APIs

AWS Proton aggiornamenti alle politiche AWS gestite

Visualizza i dettagli sugli aggiornamenti delle politiche AWS gestite AWS Proton da quando questo servizio ha iniziato a tenere traccia di queste modifiche. Per ricevere avvisi automatici sulle modifiche a questa pagina, iscriviti al feed RSS nella pagina della cronologia dei AWS Proton documenti.

Modifica	Descrizione	Data
AWSProtonCodeBuild ProvisioningServiceRolePolicy: aggiornamento a una policy esistente	La policy gestita per il ruolo collegato al servizio, che consente di AWS Proton eseguire il provisioning CodeBuild basato, ora concede le autorizzazioni per richiamare le azioni e l'API. CloudFormation `TagResource` `UntagResource` Queste autorizzazioni sono necessarie per eseguire operazioni di tagging sulle risorse.	15 giugno 2024

Modifica	Descrizione	Data
AWSProtonFullAccess: aggiornamento a una policy esistente	La politica gestita per il ruolo collegato al servizio di utilizzar e Git sync con i repositor y Git è stata aggiornata per le risorse con entrambi i prefissi di servizio. Per ulteriori informazioni, consulta Using service-linked roles for AWS CodeConnections and Managed Policy.	25 aprile 2024
AWSProtonDeveloperAccess: aggiornamento a una policy esistente	La politica gestita per il ruolo collegato al servizio di utilizzar e Git sync con i repositor y Git è stata aggiornata per le risorse con entrambi i prefissi di servizio. Per ulteriori informazioni, consulta Using service-linked roles for AWS CodeConnections and Managed Policy.	25 aprile 2024
AWSProtonSyncServiceRolePolicy: aggiornamento a una policy esistente	La politica gestita per il ruolo collegato al servizio di utilizzar e Git sync con i repositor y Git è stata aggiornata per le risorse con entrambi i prefissi di servizio. Per ulteriori informazioni, consulta Using service-linked roles for AWS CodeConnections and Managed Policy.	25 aprile 2024

Modifica	Descrizione	Data
AWSProtonCodeBuildProvisioningServiceRolePolicy: aggiornamento a una policy esistente	AWS Proton ha aggiornato questa policy per aggiungere autorizzazioni per garantire che gli account abbiano il limite di CodeBuild compilazione simultanea necessario per utilizzare Provisioning. CodeBuild	12 maggio 2023
AWSProtonServiceGitSyncServiceRolePolicy: nuova policy	AWS Proton ha aggiunto una nuova politica per consentire di eseguire la sincronizzazione AWS Proton del servizio. La policy viene utilizzata nel ruolo collegato al AWSServiceRoleForProtonServiceSyncservizio.	31 marzo 2023
AWSProtonDeveloperAccess: aggiornamento a una policy esistente	AWS Proton ha aggiunto una nuova `GetResourcesSummary` azione che consente di visualizzare un riepilogo dei modelli, delle risorse dei modelli distribuite e delle risorse non aggiornate.	18 novembre 2022
AWSProtonReadOnlyAccess: aggiornamento a una policy esistente	AWS Proton ha aggiunto una nuova `GetResourcesSummary` azione che consente di visualizzare un riepilogo dei modelli, delle risorse dei modelli distribuite e delle risorse non aggiornate.	18 novembre 2022

Modifica	Descrizione	Data
AWSProtonCodeBuildProvisioningBasicAccess: nuova policy	AWS Proton ha aggiunto una nuova politica che fornisce CodeBuild le autorizzazioni necessarie per eseguire una build for AWS Proton CodeBuild Provisioning.	16 novembre 2022
AWSProtonSyncServiceRolePolicy: nuova policy	AWS Proton ha aggiunto una nuova politica per consentire di AWS Proton eseguire operazioni relative al provisioning CodeBuild basato. La policy viene utilizzata nel ruolo collegato al AWSServiceRoleForProtonCodeBuildProvisioningservizio.	02 settembre 2022
AWSProtonFullAccess: aggiornamento a una policy esistente	AWS Proton ha aggiornato questa politica per fornire l'accesso a nuove operazioni AWS Proton API e per risolvere i problemi di autorizzazione per alcune operazioni AWS Proton della console.	30 marzo 2022
AWSProtonDeveloperAccess: aggiornamento a una policy esistente	AWS Proton aggiorna questa politica per fornire l'accesso a nuove operazioni AWS Proton API e per risolvere i problemi di autorizzazione per alcune operazioni AWS Proton della console.	30 marzo 2022

Modifica	Descrizione	Data
AWSProtonReadOnlyAccess: aggiornamento a una policy esistente	AWS Proton aggiorna questa politica per fornire l'accesso a nuove operazioni AWS Proton API e risolvere i problemi di autorizzazione per alcune operazioni AWS Proton della console.	30 marzo 2022
AWSProtonSyncServiceRolePolicy: nuova policy	AWS Proton ha aggiunto una nuova politica per consentire di AWS Proton eseguire operazioni relative alla sincronizzazione dei modelli. La policy viene utilizzata nel ruolo AWSServiceRoleForProtonSynccollegato al servizio.	23 novembre 2021
AWSProtonFullAccess: nuova policy	AWS Proton ha aggiunto una nuova politica per fornire l'accesso con ruolo amministrativo alle operazioni AWS Proton API e alla AWS Proton console.	09 giugno 2021
AWSProtonDeveloperAccess: nuova policy	AWS Proton ha aggiunto una nuova politica per fornire l'accesso del ruolo di sviluppatore alle operazioni AWS Proton API e alla AWS Proton console.	09 giugno 2021

Modifica	Descrizione	Data
AWSProtonReadOnlyAccess: nuova policy	AWS Proton ha aggiunto una nuova policy per fornire l'accesso in sola lettura alle operazioni AWS Proton API e alla AWS Proton console.	09 giugno 2021
AWS Proton ha iniziato a tenere traccia delle modifiche.	AWS Proton ha iniziato a tenere traccia delle modifiche per le sue politiche AWS gestite.	09 giugno 2021

Utilizzo di ruoli collegati ai servizi per AWS Proton

AWS Proton utilizza ruoli collegati ai servizi AWS Identity and Access Management (IAM). Un ruolo collegato ai servizi è un tipo unico di ruolo IAM a cui è collegato direttamente. AWS Proton I ruoli collegati ai servizi sono predefiniti AWS Proton e includono tutte le autorizzazioni richieste dal servizio per chiamare altri servizi per tuo conto. AWS

Argomenti

- Utilizzo dei ruoli per la sincronizzazione AWS Proton
- Utilizzo dei ruoli per CodeBuild il provisioning basato

Utilizzo dei ruoli per la sincronizzazione AWS Proton

AWS Proton utilizza ruoli collegati ai servizi AWS Identity and Access Management (IAM). Un ruolo collegato ai servizi è un tipo unico di ruolo IAM a cui è collegato direttamente. AWS Proton I ruoli collegati ai servizi sono predefiniti AWS Proton e includono tutte le autorizzazioni richieste dal servizio per chiamare altri servizi per tuo conto. AWS

Un ruolo collegato al servizio semplifica la configurazione AWS Proton perché non è necessario aggiungere manualmente le autorizzazioni necessarie. AWS Proton definisce le autorizzazioni dei ruoli collegati ai servizi e, se non diversamente definito, solo può assumerne i ruoli. AWS Proton Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere allegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. In questo modo proteggi AWS Proton le tue risorse perché non puoi rimuovere inavvertitamente l'autorizzazione ad accedere alle risorse.

Per informazioni su altri servizi che supportano i ruoli collegati ai servizi, consulta AWS i servizi che funzionano con IAM e cerca i servizi con Sì nella colonna Ruoli collegati ai servizi. Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato al servizio per tale servizio.

Autorizzazioni di ruolo collegate ai servizi per AWS Proton

AWS Proton utilizza due ruoli collegati ai servizi denominati e. AWSServiceRoleForProtonSyncAWSServiceRoleForProtonServiceSync

Ai fini dell'assunzione del ruolo, il ruolo collegato ai servizi AWSServiceRoleForProtonSync considera attendibili i seguenti servizi:

- `sync.proton.amazonaws.com`

La politica di autorizzazione dei ruoli denominata `AWSProtonSyncServiceRolePolicy` consente di AWS Proton completare le seguenti azioni sulle risorse specificate:

- Azione: creare, gestire e leggere AWS Proton modelli e versioni dei modelli
- Azione: utilizzare la connessione su CodeConnections

AWSProtonSyncServiceRolePolicy

Questa policy include le seguenti autorizzazioni:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SyncToProton",
            "Effect": "Allow",
            "Action": [
                "proton:UpdateServiceTemplateVersion",
                "proton:UpdateServiceTemplate",
                "proton:UpdateEnvironmentTemplateVersion",
                "proton:UpdateEnvironmentTemplate",
                "proton:GetServiceTemplateVersion",
```

```
                "proton:GetServiceTemplate",
                "proton:GetEnvironmentTemplateVersion",
                "proton:GetEnvironmentTemplate",
                "proton:DeleteServiceTemplateVersion",
                "proton:DeleteEnvironmentTemplateVersion",
                "proton:CreateServiceTemplateVersion",
                "proton:CreateServiceTemplate",
                "proton:CreateEnvironmentTemplateVersion",
                "proton:CreateEnvironmentTemplate",
                "proton:ListEnvironmentTemplateVersions",
                "proton:ListServiceTemplateVersions",
                "proton:CreateEnvironmentTemplateMajorVersion",
                "proton:CreateServiceTemplateMajorVersion"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AccessGitRepos",
            "Effect": "Allow",
            "Action": [
                "codestar-connections:UseConnection",
                "codeconnections:UseConnection"
             ],
            "Resource": [
                "arn:aws:codestar-connections:*:*:connection/*",
                "arn:aws:codeconnections:*:*:connection/*"
            ]
        }
    ]
}
```

Per ulteriori informazioni su questa policy, consulta [AWS politica gestita: AWSProton SyncServiceRolePolicy](#).

Ai fini dell'assunzione del ruolo, il ruolo collegato ai servizi AWSServiceRoleForProtonServiceSync considera attendibili i seguenti servizi:

- `service-sync.proton.amazonaws.com`

La politica di autorizzazione dei ruoli denominata `AWSProtonServiceGitSyncServiceRolePolicy` consente di AWS Proton completare le seguenti azioni sulle risorse specificate:

- Azione: creare, gestire e leggere su AWS Proton servizi e istanze di servizio

AWSProtonServiceGitSyncServiceRolePolicy

Questa policy include le seguenti autorizzazioni:

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "ProtonServiceSync",
   "Effect": "Allow",
   "Action": [
    "proton:GetService",
    "proton:UpdateService",
    "proton:UpdateServicePipeline",
    "proton:CreateServiceInstance",
    "proton:GetServiceInstance",
    "proton:UpdateServiceInstance",
    "proton:ListServiceInstances",
    "proton:GetComponent",
    "proton:CreateComponent",
    "proton:ListComponents",
    "proton:UpdateComponent",
    "proton:GetEnvironment",
    "proton:CreateEnvironment",
    "proton:ListEnvironments",
    "proton:UpdateEnvironment"
   ],
   "Resource": "*"
  }
 ]
}
```

Per ulteriori informazioni su questa policy, consulta AWS politica gestita: AWSProton ServiceGitSyncServiceRolePolicy.

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Creazione di un ruolo collegato ai servizi per AWS Proton

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando configuri un repository o un servizio per la sincronizzazione AWS Proton in AWS Management Console, l'o l' AWS API AWS CLI, AWS Proton crea automaticamente il ruolo collegato al servizio.

Se elimini questo ruolo collegato ai servizi, è possibile ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando configuri un repository o un servizio per la sincronizzazione AWS Proton, AWS Proton crea nuovamente il ruolo collegato al servizio.

Per ricreare il ruolo AWSServiceRoleForProtonSynccollegato al servizio, dovresti configurare un repository per la sincronizzazione e, per ricreare AWSServiceRoleForProtonServiceSync, dovresti configurare un servizio per la sincronizzazione.

Modifica di un ruolo collegato ai servizi per AWS Proton

AWS Proton non consente di modificare il ruolo collegato al servizio. AWSServiceRoleForProtonSync Dopo aver creato un ruolo collegato al servizio, non è possibile modificarne il nome, perché potrebbero farvi riferimento diverse entità. È possibile tuttavia modificarne la descrizione utilizzando IAM. Per ulteriori informazioni, consulta Modifica di un ruolo collegato ai servizi nella Guida per l'utente di IAM.

Eliminazione di un ruolo collegato ai servizi per AWS Proton

Non è necessario eliminare manualmente il ruolo AWSServiceRoleForProtonSync. Quando elimini tutti gli archivi AWS Proton collegati per la sincronizzazione del repository nell'API AWS Management Console, la o l' AWS API AWS CLI, AWS Proton pulisce le risorse ed elimina automaticamente il ruolo collegato al servizio.

Regioni supportate per i ruoli collegati ai servizi AWS Proton

AWS Proton supporta l'utilizzo di ruoli collegati al servizio in tutti i luoghi in cui il servizio è disponibile. Regioni AWS Per ulteriori informazioni, consulta Endpoint e quote AWS Proton nella Riferimenti generali di AWS.

Utilizzo dei ruoli per CodeBuild il provisioning basato

AWS Proton utilizza ruoli collegati ai servizi AWS Identity and Access Management (IAM). Un ruolo collegato ai servizi è un tipo unico di ruolo IAM a cui è collegato direttamente. AWS Proton I ruoli collegati ai servizi sono predefiniti AWS Proton e includono tutte le autorizzazioni richieste dal servizio per chiamare altri servizi per tuo conto. AWS

Un ruolo collegato al servizio semplifica la configurazione AWS Proton perché non è necessario aggiungere manualmente le autorizzazioni necessarie. AWS Proton definisce le autorizzazioni dei ruoli collegati ai servizi e, se non diversamente definito, solo può assumerne i ruoli. AWS Proton Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere allegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. In questo modo proteggi AWS Proton le tue risorse perché non puoi rimuovere inavvertitamente l'autorizzazione ad accedere alle risorse.

Per informazioni su altri servizi che supportano i ruoli collegati ai servizi, consulta AWS i servizi che funzionano con IAM e cerca i servizi con Sì nella colonna Ruoli collegati ai servizi. Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato al servizio per tale servizio.

Autorizzazioni di ruolo collegate ai servizi per AWS Proton

AWS Proton utilizza il ruolo collegato al servizio denominato AWSServiceRoleForProtonCodeBuildProvisioning: A Service Linked Role per il provisioning. AWS Proton CodeBuild

Ai fini dell'assunzione del ruolo, il ruolo collegato ai servizi AWSServiceRoleForProtonCodeBuildProvisioning considera attendibili i seguenti servizi:

- `codebuild.proton.amazonaws.com`

La politica di autorizzazione dei ruoli denominata `AWSProtonCodeBuildProvisioningServiceRolePolicy` consente di AWS Proton completare le seguenti azioni sulle risorse specificate:

- Azione: crea, gestisci e leggi su AWS CloudFormation pile e trasformazioni
- Azione: crea, gestisci e leggi CodeBuild progetti e build

AWSProtonCodeBuildProvisioningServiceRolePolicy

Questa policy include le seguenti autorizzazioni:

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

```
    {
      "Sid": "CloudFormationPermissions",
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:DeleteStack",
        "cloudformation:UpdateStack",
        "cloudformation:DescribeStacks",
        "cloudformation:DescribeStackEvents",
        "cloudformation:ListStackResources",
        "cloudformation:TagResource",
        "cloudformation:UntagResource"
      ],
      "Resource": [
        "arn:aws:cloudformation:*:*:stack/AWSProton-CodeBuild-*"
      ]
    },
    {
      "Sid": "CodeBuildPermissions",
      "Effect": "Allow",
      "Action": [
        "codebuild:CreateProject",
        "codebuild:DeleteProject",
        "codebuild:UpdateProject",
        "codebuild:StartBuild",
        "codebuild:StopBuild",
        "codebuild:RetryBuild",
        "codebuild:BatchGetBuilds",
        "codebuild:BatchGetProjects"
      ],
      "Resource": "arn:aws:codebuild:*:*:project/AWSProton*"
    },
    {
      "Sid": "PassRolePermissions",
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "*",
      "Condition": {
        "StringEqualsIfExists": {
          "iam:PassedToService": "codebuild.amazonaws.com"
        }
      }
```

```
    },
    {
      "Sid": "ServiceQuotaPermissions",
      "Effect": "Allow",
      "Action": [
          "servicequotas:GetServiceQuota"
      ],
      "Resource": "*"
    }
  ]
}
```

Per ulteriori informazioni su questa policy, consulta AWS politica gestita: AWSProton CodeBuildProvisioningServiceRolePolicy.

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Creazione di un ruolo collegato ai servizi per AWS Proton

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando crei un ambiente che utilizza il provisioning CodeBuild basato su AWS Management Console, AWS Proton nell'o nell' AWS API AWS CLI, AWS Proton crea automaticamente il ruolo collegato al servizio.

Se elimini questo ruolo collegato ai servizi, è possibile ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando crei un ambiente che utilizza il provisioning CodeBuild basato sul provisioning in AWS Proton, AWS Proton crea nuovamente il ruolo collegato al servizio.

Modifica di un ruolo collegato ai servizi per AWS Proton

AWS Proton non consente di modificare il ruolo collegato al AWSServiceRoleForProtonCodeBuildProvisioningservizio. Dopo aver creato un ruolo collegato al servizio, non potrai modificarne il nome perché varie entità potrebbero farvi riferimento. È possibile tuttavia modificarne la descrizione utilizzando IAM. Per ulteriori informazioni, consulta Modifica di un ruolo collegato ai servizi nella Guida per l'utente di IAM.

Eliminazione di un ruolo collegato ai servizi per AWS Proton

Se non è più necessario utilizzare una caratteristica o un servizio che richiede un ruolo collegato ai servizi, ti consigliamo di eliminare il ruolo. In questo modo non sarà più presente un'entità non

utilizzata che non viene monitorata e gestita attivamente. Tuttavia, è necessario eliminare tutti gli ambienti e i servizi (istanze e pipeline) che utilizzano il provisioning CodeBuild basato sul provisioning AWS Proton prima di poterlo eliminare manualmente.

Eliminazione manuale del ruolo collegato ai servizi

Utilizza la console IAM AWS CLI, o l' AWS API per eliminare il ruolo collegato al AWSServiceRoleForProtonCodeBuildProvisioningservizio. Per ulteriori informazioni, consulta Eliminazione del ruolo collegato al servizio nella Guida per l'utente di IAM.

Regioni supportate per i ruoli collegati ai servizi AWS Proton

AWS Proton supporta l'utilizzo di ruoli collegati al servizio in tutti i luoghi in Regioni AWS cui il servizio è disponibile. Per ulteriori informazioni, consulta Endpoint e quote AWS Proton nella Riferimenti generali di AWS.

Risoluzione dei problemi di AWS Proton identità e accesso

Utilizza le seguenti informazioni per aiutarti a diagnosticare e risolvere i problemi più comuni che potresti riscontrare quando lavori con un AWS Proton IAM.

Argomenti

- Non sono autorizzato a eseguire alcuna azione in AWS Proton
- Non sono autorizzato a eseguire iam: PassRole
- Voglio consentire a persone esterne a me di accedere Account AWS alle mie AWS Proton risorse

Non sono autorizzato a eseguire alcuna azione in AWS Proton

Se ti AWS Management Console dice che non sei autorizzato a eseguire un'azione, devi contattare l'amministratore per ricevere assistenza. L'amministratore è colui che ti ha fornito le credenziali di accesso.

L'errore di esempio seguente si verifica quando l'utente `mateojackson` IAM prova a utilizzare la console per visualizzare i dettagli relativi a una risorsa *`my-example-widget`* fittizia ma non dispone di autorizzazioni `proton:`*`GetWidget`* fittizie.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 proton:GetWidget on resource: my-example-widget
```

In questo caso, Mateo richiede al suo amministratore di aggiornare le policy per poter accedere alla risorsa *my-example-widget* utilizzando l'azione `proton:GetWidget`.

Non sono autorizzato a eseguire iam: PassRole

Se ricevi un errore che indica che non sei autorizzato a eseguire l'operazione `iam:PassRole`, le tue policy devono essere aggiornate per poter passare un ruolo a AWS Proton.

Alcuni Servizi AWS consentono di passare un ruolo esistente a quel servizio invece di creare un nuovo ruolo di servizio o un ruolo collegato al servizio. Per eseguire questa operazione, è necessario disporre delle autorizzazioni per trasmettere il ruolo al servizio.

L'errore di esempio seguente si verifica quando un utente IAM denominato `marymajor` cerca di utilizzare la console per eseguire un'operazione in AWS Proton. Tuttavia, l'operazione richiede che il servizio disponga delle autorizzazioni concesse da un ruolo di servizio. Mary non dispone delle autorizzazioni per passare il ruolo al servizio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In questo caso, le policy di Mary devono essere aggiornate per poter eseguire l'operazione `iam:PassRole`.

Se hai bisogno di aiuto, contatta il tuo AWS amministratore. L'amministratore è la persona che ti ha fornito le credenziali di accesso.

Voglio consentire a persone esterne a me di accedere Account AWS alle mie AWS Proton risorse

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle tue risorse. È possibile specificare chi è attendibile per l'assunzione del ruolo. Per i servizi che supportano politiche basate sulle risorse o liste di controllo degli accessi (ACLs), puoi utilizzare tali politiche per concedere alle persone l'accesso alle tue risorse.

Per ulteriori informazioni, consulta gli argomenti seguenti:

- Per sapere se AWS Proton supporta queste funzionalità, consulta. Come AWS Proton funziona con IAM

- Per scoprire come fornire l'accesso alle tue risorse attraverso Account AWS le risorse di tua proprietà, consulta Fornire l'accesso a un utente IAM in un altro Account AWS di tua proprietà nella IAM User Guide.
- Per scoprire come fornire l'accesso alle tue risorse a terze parti Account AWS, consulta Fornire l'accesso a soggetti Account AWS di proprietà di terze parti nella Guida per l'utente IAM.
- Per informazioni su come fornire l'accesso tramite la federazione delle identità, consulta Fornire l'accesso a utenti autenticati esternamente (Federazione delle identità) nella Guida per l'utente IAM.
- Per informazioni sulle differenze di utilizzo tra ruoli e policy basate su risorse per l'accesso multi-account, consulta Accesso a risorse multi-account in IAM nella Guida per l'utente IAM.

Analisi della configurazione e delle vulnerabilità in AWS Proton

AWS Proton non fornisce patch o aggiornamenti per il codice fornito dal cliente. I clienti sono responsabili dell'aggiornamento e dell'applicazione delle patch al proprio codice, incluso il codice sorgente dei servizi e delle applicazioni su cui sono in esecuzione AWS Proton e il codice fornito nei pacchetti di modelli di servizi e ambienti.

I clienti sono responsabili dell'aggiornamento e dell'applicazione di patch alle risorse dell'infrastruttura nei propri ambienti e servizi. AWS Proton non aggiornerà o correggerà automaticamente alcuna risorsa. I clienti devono consultare la documentazione relativa alle risorse della propria architettura per comprendere le rispettive politiche di applicazione delle patch.

Oltre a fornire gli aggiornamenti di ambiente e servizio richiesti dal cliente alle versioni minori dei modelli di servizio e ambiente, AWS Proton non fornisce patch o aggiornamenti alle risorse che i clienti definiscono nei modelli e nei pacchetti di modelli di servizio e ambiente.

Per ulteriori dettagli, consulta le seguenti risorse :

- Modello di responsabilità condivisa
- Amazon Web Services: panoramica dei processi di sicurezza

Protezione dei dati in AWS Proton

AWS Proton è conforme al modello di responsabilità AWS condivisa modello di di che include regolamenti e linee guida per la protezione dei dati. AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i Servizi AWS. AWS mantiene il controllo sui dati ospitati

su questa infrastruttura, compresi i controlli di configurazione di sicurezza per la gestione dei contenuti e dei dati personali dei clienti. AWS i clienti e i partner APN, che agiscono in qualità di titolari o incaricati del trattamento dei dati, sono responsabili di tutti i dati personali che inseriscono nel Cloud AWS

Ai fini della protezione dei dati, ti consigliamo di proteggere Account AWS le credenziali e di configurare account utente individuali con AWS Identity and Access Management (IAM), in modo che a ciascun utente vengano concesse solo le autorizzazioni necessarie per svolgere le proprie mansioni lavorative. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- Utilizza SSL/TLS per comunicare con le risorse. AWS È consigliabile TLS 1.2 o versioni successive.
- Configura l'API e la registrazione delle attività degli utenti con. AWS CloudTrail
- Utilizza soluzioni di AWS crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.

Ti consigliamo vivamente di non inserire mai informazioni identificative sensibili, come i numeri di conto dei tuoi clienti, in campi di testo in formato libero come il campo Nome. Ciò vale anche quando lavori AWS Proton o Servizi AWS utilizzi la console, l'API o. AWS CLI AWS SDKs Tutti i dati inseriti nei campi di testo in formato libero per gli identificatori delle risorse o elementi simili relativi alla gestione delle AWS risorse potrebbero essere raccolti per essere inclusi nei registri di diagnostica. Quando fornisci un URL a un server esterno, non includere informazioni sulle credenziali nell'URL per convalidare la tua richiesta a tale server.

Per ulteriori informazioni sulla protezione dei dati, consulta il post del blog AWS Modello di responsabilità condivisa e GDPR su AWS Security Blog.

Crittografia lato server inattiva

Se scegli di crittografare i dati sensibili nei tuoi pacchetti di modelli inattivi nel bucket S3 in cui sono archiviati i pacchetti di modelli, devi utilizzare una chiave SSE-S3 o SSE-KMS per consentire il recupero dei pacchetti di modelli in modo che possano essere allegati AWS Proton a un modello registrato. AWS Proton

Crittografia in transito

Tutte le comunicazioni da servizio a servizio sono crittografate in transito utilizzando SSL/TLS.

AWS Proton gestione delle chiavi di crittografia

All'interno AWS Proton, tutti i dati dei clienti vengono crittografati per impostazione predefinita utilizzando una chiave AWS Proton proprietaria. Se si fornisce una AWS KMS chiave di proprietà e gestita dal cliente, tutti i dati del cliente vengono crittografati utilizzando la chiave fornita dal cliente, come descritto nei paragrafi seguenti.

Quando si crea un AWS Proton modello, si specifica la chiave e si AWS Proton utilizzano le credenziali per creare una concessione che AWS Proton consente l'utilizzo della chiave.

Se ritiri manualmente la concessione oppure disabiliti o elimini la chiave specificata, non AWS Proton è in grado di leggere i dati che sono stati crittografati dalla chiave specificata e generati. `ValidationException`

AWS Proton contesto di crittografia

AWS Proton supporta le intestazioni del contesto di crittografia. Un contesto di crittografia è un set facoltativo di coppie chiave-valore che possono contenere ulteriori informazioni contestuali sui dati. Per informazioni generali sul contesto di crittografia, consulta Concetti di AWS Key Management Service - Contesto di crittografia nella Guida per gli sviluppatori di AWS Key Management Service .

Un contesto di crittografia è un insieme di coppie chiave-valore che contengono dati arbitrari non segreti. Quando si include un contesto di crittografia in una richiesta di crittografia dei dati, associa AWS KMS crittograficamente il contesto di crittografia ai dati crittografati. lo stesso contesto di crittografia sia necessario per decrittografare i dati.

I clienti possono utilizzare il contesto di crittografia per identificare l'uso della chiave gestita dal cliente nei registri e nei registri di controllo. Viene inoltre visualizzato in testo non crittografato nei log, ad esempio AWS CloudTrail Amazon Logs. CloudWatch

AWS Proton non utilizza alcun contesto di crittografia specificato dal cliente o specificato esternamente.

AWS Proton aggiunge il seguente contesto di crittografia.

```
{
  "aws:proton:template": "<proton-template-arn>",
  "aws:proton:resource": "<proton-resource-arn>"
}
```

Il primo contesto di crittografia identifica il AWS Proton modello a cui è associata la risorsa e funge anche da vincolo per le autorizzazioni e le concessioni delle chiavi gestite dal cliente.

Il secondo contesto di crittografia identifica la risorsa crittografata. AWS Proton

Gli esempi seguenti mostrano l'uso del contesto di AWS Proton crittografia.

Sviluppatore che crea un'istanza di servizio.

```
{
  "aws:proton:template": "arn:aws:proton:region_id:123456789012:service-template/my-
template",
  "aws:proton:resource": "arn:aws:proton:region_id:123456789012:service/my-service/
service-instance/my-service-instance"
}
```

Un amministratore che crea un modello.

```
{
  "aws:proton:template": "arn:aws:proton:region_id:123456789012:service-template/my-
template",
  "aws:proton:resource": "arn:aws:proton:region_id:123456789012:service-template/my-
template"
}
```

Sicurezza dell'infrastruttura in AWS Proton

In quanto servizio gestito, AWS Proton è protetto dalla sicurezza di rete AWS globale. Per informazioni sui servizi AWS di sicurezza e su come AWS protegge l'infrastruttura, consulta AWS Cloud Security. Per progettare il tuo AWS ambiente utilizzando le migliori pratiche per la sicurezza dell'infrastruttura, vedi Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Utilizzate chiamate API AWS pubblicate per accedere AWS Proton attraverso la rete. I client devono supportare quanto segue:

- Transport Layer Security (TLS). È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Suite di cifratura con Perfect Forward Secrecy (PFS), ad esempio Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associata a un principale IAM. O puoi utilizzare AWS Security Token Service (AWS STS) per generare credenziali di sicurezza temporanee per sottoscrivere le richieste.

Per migliorare l'isolamento della rete, è possibile utilizzare AWS PrivateLink quanto descritto nella sezione seguente.

AWS Proton e endpoint VPC di interfaccia ()AWS PrivateLink

Puoi stabilire una connessione privata tra il tuo VPC e creare un AWS Proton endpoint VPC di interfaccia. Gli endpoint di interfaccia sono alimentati da AWS PrivateLink, una tecnologia che consente di accedere in modo privato AWS Proton APIs senza un gateway Internet, un dispositivo NAT, una connessione VPN o una connessione. AWS Direct Connect Le istanze del tuo VPC non necessitano di indirizzi IP pubblici con cui comunicare. AWS Proton APIs Il traffico tra il tuo VPC e AWS Proton non esce dalla rete Amazon.

Ogni endpoint dell'interfaccia è rappresentato da una o più interfacce di rete elastiche nelle sottoreti.

Per ulteriori informazioni, consultare Endpoint VPC di interfaccia (AWS PrivateLink) nella Guida per l'utente di Amazon VPC.

Considerazioni sugli endpoint AWS Proton VPC

Prima di configurare un endpoint VPC di interfaccia per AWS Proton, assicurati di esaminare le proprietà e le limitazioni degli endpoint dell'interfaccia nella Amazon VPC User Guide.

AWS Proton supporta l'esecuzione di chiamate a tutte le sue azioni API dal tuo VPC.

Le policy degli endpoint VPC sono supportate per. AWS Proton Per impostazione predefinita, l'accesso completo a AWS Proton è consentito tramite l'endpoint. Per ulteriori informazioni, consulta Controllo degli accessi ai servizi con endpoint VPC nella Guida per l'utente di Amazon VPC.

Creazione di un endpoint VPC di interfaccia per AWS Proton

Puoi creare un endpoint VPC per il AWS Proton servizio utilizzando la console Amazon VPC o (). AWS Command Line Interface AWS CLI Per ulteriori informazioni, consulta Creazione di un endpoint dell'interfaccia nella Guida per l'utente di Amazon VPC.

Crea un endpoint VPC per AWS Proton utilizzare il seguente nome di servizio:

- com.amazonaws. *region*.protone

Se abiliti il DNS privato per l'endpoint, puoi effettuare richieste API AWS Proton utilizzando il nome DNS predefinito per la regione, ad esempio. `proton.region.amazonaws.com`

Per ulteriori informazioni, consulta Accesso a un servizio tramite un endpoint dell'interfaccia in Guida per l'utente di Amazon VPC.

Creazione di una policy per gli endpoint VPC per AWS Proton

È possibile allegare un criterio all'endpoint VPC che controlla l'accesso all' AWS Proton. La policy specifica le informazioni riportate di seguito:

- Il principale che può eseguire operazioni.
- Le azioni che possono essere eseguite.
- Le risorse sui cui si possono eseguire operazioni.

Per ulteriori informazioni, consulta Controllo degli accessi ai servizi con endpoint VPC in Guida per l'utente di Amazon VPC.

Esempio: policy degli endpoint VPC per le azioni AWS Proton

Di seguito è riportato un esempio di policy sugli endpoint per. AWS Proton Se associata a un endpoint, questa politica consente l'accesso alle AWS Proton azioni elencate per tutti i principali su tutte le risorse.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Principal": "*",
      "Action": [
        "proton:ListServiceTemplates",
        "proton:ListServiceTemplateMajorVersions",
        "proton:ListServiceTemplateMinorVersions",
        "proton:ListServices",
        "proton:ListServiceInstances",
        "proton:ListEnvironments",
        "proton:GetServiceTemplate",
        "proton:GetServiceTemplateMajorVersion",
        "proton:GetServiceTemplateMinorVersion",
        "proton:GetService",
```

```
        "proton:GetServiceInstance",
        "proton:GetEnvironment",
        "proton:CreateService",
        "proton:UpdateService",
        "proton:UpdateServiceInstance",
        "proton:UpdateServicePipeline",
        "proton:DeleteService"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

Registrazione e monitoraggio AWS Proton

Il monitoraggio è una parte importante per mantenere l'affidabilità, la disponibilità e le prestazioni delle altre AWS Proton soluzioni. AWS AWS fornisce i seguenti strumenti di monitoraggio per monitorare l'esecuzione delle istanze AWS Proton, segnalare quando qualcosa non va e intraprendere azioni automatiche laddove opportuno.

Al momento, di per AWS Proton sé non è integrato con Amazon CloudWatch Logs o AWS Trusted Advisor. Gli amministratori possono configurarli e utilizzarli CloudWatch per monitorarne altri Servizi AWS , come definito nei modelli di servizio e ambiente. AWS Proton è integrato con AWS CloudTrail.

- Amazon CloudWatch monitora AWS le tue risorse e le applicazioni su cui esegui AWS in tempo reale. Puoi raccogliere i parametri e tenerne traccia, creare pannelli di controllo personalizzati e impostare allarmi per inviare una notifica o intraprendere azioni quando un parametro specificato raggiunge una determinata soglia. Ad esempio, puoi tenere CloudWatch traccia dell'utilizzo della CPU o di altri parametri delle tue EC2 istanze Amazon e avviare automaticamente nuove istanze quando necessario. Per ulteriori informazioni, consulta la Amazon CloudWatch User Guide.
- Amazon CloudWatch Logs ti consente di monitorare, archiviare e accedere ai tuoi file di registro da EC2 istanze Amazon e altre fonti. CloudTrail CloudWatch I log possono monitorare le informazioni contenute nei file di registro e avvisarti quando vengono raggiunte determinate soglie. Puoi inoltre archiviare i dati del log in storage estremamente durevole. Per ulteriori informazioni, consulta la Amazon CloudWatch Logs User Guide.
- AWS CloudTrailacquisisce le chiamate API e gli eventi correlati effettuati da o per conto tuo Account AWS e invia i file di log a un bucket Amazon S3 da te specificato. Puoi identificare quali utenti e account hanno chiamato AWS, l'indirizzo IP di origine da cui sono state effettuate

le chiamate e quando sono avvenute le chiamate. Per ulteriori informazioni, consulta la AWS CloudTrail Guida per l'utente di .

- Amazon EventBridge è un servizio di bus eventi senza server che semplifica la connessione delle applicazioni con dati provenienti da una varietà di fonti. EventBridge fornisce un flusso di dati in tempo reale dalle tue applicazioni Software-as-a-Service (SaaS) Servizi AWS e indirizza tali dati verso destinazioni come Lambda. In questo modo puoi monitorare gli eventi che si verificano nei servizi e creare architetture basate su eventi. Per ulteriori informazioni, consulta Automatizza con AWS Proton EventBridge e la Guida per l'utente di EventBridge .

Resilienza in AWS Proton

L'infrastruttura AWS globale è costruita attorno a Regione AWS zone di disponibilità. Regione AWS forniscono più zone di disponibilità fisicamente separate e isolate, collegate con reti a bassa latenza, ad alto throughput e altamente ridondanti. Con le zone di disponibilità, puoi progettare e gestire applicazioni e database che eseguono automaticamente il failover tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili rispetto alle infrastrutture a data center singolo o multiplo tradizionali.

Per ulteriori informazioni su Regione AWS s e sulle zone di disponibilità, consulta Global Infrastructure.AWS

Oltre all'infrastruttura AWS globale, AWS Proton offre funzionalità che aiutano a supportare le esigenze di resilienza e backup dei dati.

AWS Proton backup

AWS Proton mantiene un backup di tutti i dati dei clienti. In caso di interruzione totale, questo backup può essere utilizzato per ripristinare AWS Proton i dati dei clienti da uno stato precedente valido.

Best practice di sicurezza per AWS Proton

AWS Proton fornisce funzionalità di sicurezza da considerare durante lo sviluppo e l'implementazione delle proprie politiche di sicurezza. Le seguenti best practice sono linee guida generali e non rappresentano una soluzione di sicurezza completa. Poiché queste best practice potrebbero non essere appropriate o sufficienti per l'ambiente, gestiscile come considerazioni utili anziché prescrizioni.

Argomenti

- Utilizza IAM per controllare gli accessi
- Non incorporate credenziali nei modelli e nei pacchetti di modelli
- Usa la crittografia per proteggere i dati sensibili
- Utilizzalo AWS CloudTrail per visualizzare e registrare le chiamate API

Utilizza IAM per controllare gli accessi

IAM è uno Servizio AWS strumento che puoi utilizzare per gestire gli utenti e le relative autorizzazioni. AWS Puoi utilizzare IAM con AWS Proton per specificare quali AWS Proton azioni possono eseguire amministratori e sviluppatori, come la gestione di modelli, ambienti o servizi. Puoi utilizzare i ruoli di servizio IAM per consentire AWS Proton di effettuare chiamate ad altri servizi per tuo conto.

Per ulteriori informazioni sui AWS Proton ruoli IAM, consultaIdentity and Access Management per AWS Proton.

Implementa l'accesso con privilegi minimi. Per ulteriori informazioni, consulta Politiche e autorizzazioni in IAM nella Guida per l'AWS Identity and Access Management utente.

Non incorporate credenziali nei modelli e nei pacchetti di modelli

Invece di incorporare informazioni sensibili nei AWS CloudFormation modelli e nei pacchetti di modelli, ti consigliamo di utilizzare riferimenti dinamici nel tuo modello di stack.

I riferimenti dinamici offrono un modo compatto e potente per fare riferimento a valori esterni archiviati e gestiti in altri servizi, come AWS Systems Manager Parameter Store o. AWS Secrets Manager Quando utilizzate un riferimento dinamico, CloudFormation recupera il valore del riferimento specificato quando necessario durante le operazioni di stack e change set e passa il valore alla risorsa appropriata. Tuttavia, CloudFormation non memorizza mai il valore di riferimento effettivo. Per ulteriori informazioni, vedere Uso dei riferimenti dinamici per specificare i valori dei modelli nella Guida per l'AWS CloudFormation utente.

AWS Secrets Manager ti aiuta a crittografare, archiviare e recuperare le credenziali per i database e altri servizi in modo sicuro. Il AWS Systems Manager Parameter Store fornisce uno storage gerarchico sicuro per la gestione dei dati di configurazione.

Per ulteriori informazioni sulla definizione dei parametri del modello, https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/parameters-section-structure.html consultate la Guida per l'AWS CloudFormation utente.

Usa la crittografia per proteggere i dati sensibili

All'interno AWS Proton, tutti i dati dei clienti sono crittografati per impostazione predefinita utilizzando una chiave AWS Proton proprietaria.

In qualità di membro del team della piattaforma, puoi fornire una chiave gestita dal cliente per AWS Proton crittografare e proteggere i tuoi dati sensibili. Crittografa i dati sensibili archiviati nel tuo bucket S3. Per ulteriori informazioni, consulta Protezione dei dati in AWS Proton.

Utilizzalo AWS CloudTrail per visualizzare e registrare le chiamate API

AWS CloudTrail tiene traccia di chiunque effettui chiamate API nel tuo Account AWS. Le chiamate API vengono registrate ogni volta che qualcuno utilizza l' AWS Proton API, la AWS Proton console o AWS Proton AWS CLI i comandi. Attivare la registrazione e specificare un bucket Amazon S3 in cui archiviare i log. In questo modo, se necessario, puoi verificare chi ha effettuato una determinata AWS Proton chiamata nel tuo account. Per ulteriori informazioni, consulta Registrazione e monitoraggio AWS Proton.

Prevenzione del confused deputy tra servizi

Il problema confused deputy è un problema di sicurezza in cui un'entità che non dispone dell'autorizzazione per eseguire un'azione può costringere un'entità maggiormente privilegiata a eseguire l'azione. Nel AWS, l'impersonificazione tra servizi può portare al confuso problema del vice. La rappresentazione tra servizi può verificarsi quando un servizio (il servizio chiamante) effettua una chiamata a un altro servizio (il servizio chiamato). Il servizio chiamante può essere manipolato per utilizzare le proprie autorizzazioni e agire sulle risorse di un altro cliente, a cui normalmente non avrebbe accesso. Per evitare che ciò accada, AWS mette a disposizione strumenti che consentono di proteggere i dati relativi a tutti i servizi con responsabili del servizio a cui è stato concesso l'accesso alle risorse del vostro account.

Ti consigliamo di utilizzare `aws:SourceArn`le chiavi di contesto della condizione `aws:SourceAccount`globale nelle politiche delle risorse per limitare le autorizzazioni che AWS Proton forniscono un altro servizio alla risorsa. Se il valore `aws:SourceArn` non contiene l'ID account, ad esempio un ARN di un bucket Amazon S3, è necessario utilizzare entrambe le chiavi di contesto delle condizioni globali per limitare le autorizzazioni. Se si utilizzano entrambe le chiavi di contesto delle condizioni globali e il valore `aws:SourceArn` contiene l'ID account, il valore `aws:SourceAccount` e l'account nel valore `aws:SourceArn` deve utilizzare lo stesso ID account nella stessa dichiarazione di policy. Utilizzare `aws:SourceArn` se si desidera consentire

l'associazione di una sola risorsa all'accesso tra servizi. Utilizza `aws:SourceAccount` se desideri consentire l'associazione di qualsiasi risorsa in tale account all'uso tra servizi.

Il valore di `aws:SourceArn` deve essere una risorsa che AWS Proton memorizza.

Il modo più efficace per proteggersi dal problema "confused deputy" è quello di usare la chiave di contesto della condizione globale `aws:SourceArn` con l'ARN completo della risorsa. Se non si conosce l'ARN completo della risorsa o si scelgono più risorse, è necessario utilizzare la chiave di contesto della condizione globale `aws:SourceArn` con caratteri jolly (`*`) per le parti sconosciute dell'ARN. Ad esempio, `arn:aws::proton:*:123456789012:environment/*`.

L'esempio seguente mostra come utilizzare le chiavi di contesto `aws:SourceArn` e `aws:SourceAccount` global condition AWS Proton per prevenire il confuso problema del vice.

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Sid": "ExampleProtonConfusedDeputyPreventionPolicy",
        "Effect": "Allow",
        "Principal": {"Service": "proton.amazonaws.com"},
        "Action": "sts:AssumeRole",
        "Condition": {
            "StringEquals": {
                "aws:SourceAccount": "123456789012"
            },
            "ArnLike": {
                "aws:SourceArn": "arn:aws::proton:*:123456789012:environment/*"
            }
        }
    }
}
```

CodeBuild fornitura di supporto Amazon VPC personalizzato

AWS Proton CodeBuild Il provisioning esegue comandi CLI arbitrari forniti dal cliente in un CodeBuild progetto situato nell'account Environment. AWS Proton Questi comandi in genere gestiscono le risorse utilizzando uno strumento Infrastructure as Code (IaC), come CDK. Se disponi di risorse in un Amazon VPC, CodeBuild potresti non essere in grado di accedervi. A tal fine, CodeBuild supporta la capacità di funzionare all'interno di uno specifico Amazon VPC. Alcuni esempi di casi d'uso includono:

- Recupera le dipendenze da archivi di artefatti interni ospitati autonomamente, ad esempio per `PyPI` Python, per Java e per Node.js `Maven` `npm`
- CodeBuild deve accedere a un server Jenkins in un particolare Amazon VPC per registrare una pipeline.
- Accedi agli oggetti in un bucket Amazon S3 configurato per consentire l'accesso solo tramite un endpoint Amazon VPC.
- Esegui i test di integrazione della tua build rispetto ai dati in un database Amazon RDS isolato su una sottorete privata.

Per ulteriori informazioni, consulta la CodeBuild documentazione relativa al VPC.

Se desideri che CodeBuild Provisioning venga eseguito in un VPC personalizzato AWS Proton , offre una soluzione semplice. Innanzitutto, è necessario aggiungere l'ID VPC, le sottoreti e i gruppi di sicurezza al modello di ambiente. Successivamente, inserisci questi valori nelle specifiche dell'ambiente. Ciò comporterà la creazione di un CodeBuild progetto per te destinato a un determinato VPC.

Aggiornamento del modello di ambiente

Schema

L'ID VPC, le sottoreti e i gruppi di sicurezza devono essere aggiunti allo schema del modello in modo che possano esistere nelle specifiche dell'ambiente.

Un esempio: `schema.yaml`

```
schema:
  format:
    openapi: "3.0.0"
  environment_input_type: "EnvironmentInputType"
  types:
    EnvironmentInputType:
      type: object
      properties:
        codebuild_vpc_id:
          type: string
        codebuild_subnets:
          type: array
          items:
```

```
            type: string
        codebuild_security_groups:
          type: array
          items:
            type: string
```

Ciò aggiunge tre nuove proprietà che verranno utilizzate dal manifesto:

- `codebuild_vpc_id`
- `codebuild_subnets`
- `codebuild_security_groups`

Manifesto

Per configurare le impostazioni di Amazon VPC in CodeBuild, nel manifesto del modello `project_properties` è disponibile una proprietà opzionale denominata. I contenuti di `project_properties` vengono aggiunti allo AWS CloudFormation stack che crea il CodeBuild progetto. In questo modo è possibile aggiungere non solo le proprietà di Amazon VPC, ma anche qualsiasi AWS CloudFormationCodeBuild CloudFormation proprietà supportata, ad esempio build timeout. Gli stessi dati forniti a `proton-inputs.json` sono resi disponibili per i valori di. `project_properties`

Aggiungi questa sezione al tuo`manifest.yaml`:

```
project_properties:
          VpcConfig:
            VpcId: "{{ environment.inputs.codebuild_vpc_id }}"
            Subnets: "{{ environment.inputs.codebuild_subnets }}"
            SecurityGroupIds: "{{ environment.inputs.codebuild_security_groups }}"
```

Di seguito è riportato come `manifest.yaml` potrebbe apparire il risultato:

```
infrastructure:
  templates:
    - rendering_engine: codebuild
      settings:
        image: aws/codebuild/amazonlinux2-x86_64-standard:4.0
        runtimes:
          nodejs: 16
        provision:
```

```
      - npm install
      - npm run build
      - npm run cdk bootstrap
      - npm run cdk deploy -- --require-approval never
    deprovision:
      - npm install
      - npm run build
      - npm run cdk destroy -- --force
    project_properties:
      VpcConfig:
        VpcId: "{{ environment.inputs.codebuild_vpc_id }}"
        Subnets: "{{ environment.inputs.codebuild_subnets }}"
        SecurityGroupIds: "{{ environment.inputs.codebuild_security_groups }}"
```

Creare l'ambiente

Quando crei un ambiente con il tuo modello abilitato per CodeBuild Provisioning VPC, devi fornire l'ID Amazon VPC, le sottoreti e i gruppi di sicurezza.

Per ottenere un elenco di tutti gli Amazon VPC IDs nella tua regione, esegui il seguente comando:

```
aws ec2 describe-vpcs
```

Per ottenere un elenco di tutte le sottoreti IDs, esegui:

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=vpc-id"
```

> **Important**
>
> Includi solo sottoreti private. CodeBuild fallirà se fornisci sottoreti pubbliche. Le sottoreti pubbliche hanno un percorso predefinito verso un Internet Gateway, mentre le sottoreti private no.

Esegui il comando seguente per ottenere il gruppo di sicurezza. IDs Questi IDs possono essere ottenuti anche tramite AWS Management Console:

```
aws ec2 describe-security-groups --filters "Name=vpc-id,Values=vpc-id"
```

I valori saranno simili a:

```
vpc-id: vpc-045ch35y28dec3a05
subnets:
  - subnet-04029a82e6ae46968
  - subnet-0f500a9294fc5f26a
 security-groups:
  - sg-03bc4c4ce32d67e8d
```

Garantire le CodeBuild autorizzazioni

Il supporto di Amazon VPC richiede determinate autorizzazioni, come la possibilità di creare un'interfaccia di rete elastica.

Se l'ambiente viene creato nella console, inserisci questi valori durante la procedura guidata di creazione dell'ambiente. Se desideri creare l'ambiente a livello di codice, il tuo è `spec.yaml` simile al seguente:

```
proton: EnvironmentSpec

spec:
  codebuild_vpc_id: vpc-045ch35y28dec3a05
  codebuild_subnets:
    - subnet-04029a82e6ae46968
    - subnet-0f500a9294fc5f26a
  codebuild_security_groups:
    - sg-03bc4c4ce32d67e8d
```

AWS Proton risorse e etichettatura

AWS Proton le risorse a cui viene assegnato un Amazon Resource Name (ARN) includono modelli di ambiente e relative versioni principali e secondarie, modelli di servizio e relative versioni principali e secondarie, ambienti, servizi, istanze di servizio, componenti e repository. Puoi etichettare queste risorse per organizzarle e identificarle. È possibile utilizzare i tag per suddividere le risorse in categorie in base allo scopo, al proprietario, all'ambiente o ad altri criteri. Per ulteriori informazioni, consulta Strategie di tagging di . Per tenere traccia e gestire le AWS Proton risorse, è possibile utilizzare le funzionalità di etichettatura descritte nelle sezioni seguenti.

AWS etichettatura

Puoi assegnare metadati alle tue AWS risorse sotto forma di tag. Ogni tag è composto da una chiave definita dal cliente e da un valore opzionale. Con i tag è possibile a gestire, identificare, organizzare, cercare e filtrare le risorse.

> **Important**
>
> Non aggiungere Informazioni personali di identificazione (PII) o altre informazioni riservate o sensibili nei tag. I tag sono accessibili a molti Servizi AWS, inclusa la fatturazione. I tag non sono destinati ad essere utilizzati per dati privati o sensibili.

Ogni tag è costituito da due parti.

- Una chiave del tag (ad esempio,`CostCenter`, `Environment`, o `Project`). Le chiavi dei tag prevedono una distinzione tra lettere maiuscole e minuscole.
- Un valore di tag (opzionale) (ad esempio, `111122223333` o`Production`). Analogamente alle chiavi dei tag, i valori dei tag prevedono una distinzione tra lettere maiuscole e minuscole.

I seguenti requisiti di base per la denominazione e l'utilizzo si applicano ai tag.

- Ogni risorsa può avere un massimo di 50 tag creati dall'utente.

> **Note**
>
> I tag creati dal sistema che iniziano con il `aws:` prefisso sono riservati AWS all'uso e non rientrano in questo limite. Non è possibile modificare o eliminare un tag che inizia con il prefisso `aws:`.

- Per ciascuna risorsa, ogni chiave del tag deve essere univoca e ogni chiave del tag può avere un solo valore.
- Il tag della chiave deve essere composto da un minimo di 1 e un massimo di 128 caratteri Unicode in UTF-8.
- Il valore del tag deve essere compreso tra un minimo di 1 e un massimo di 256 caratteri Unicode in UTF-8.
- I caratteri consentiti nei tag sono lettere, numeri, spazi rappresentabili in UTF-8 e i seguenti caratteri: * _.:/= + - @.

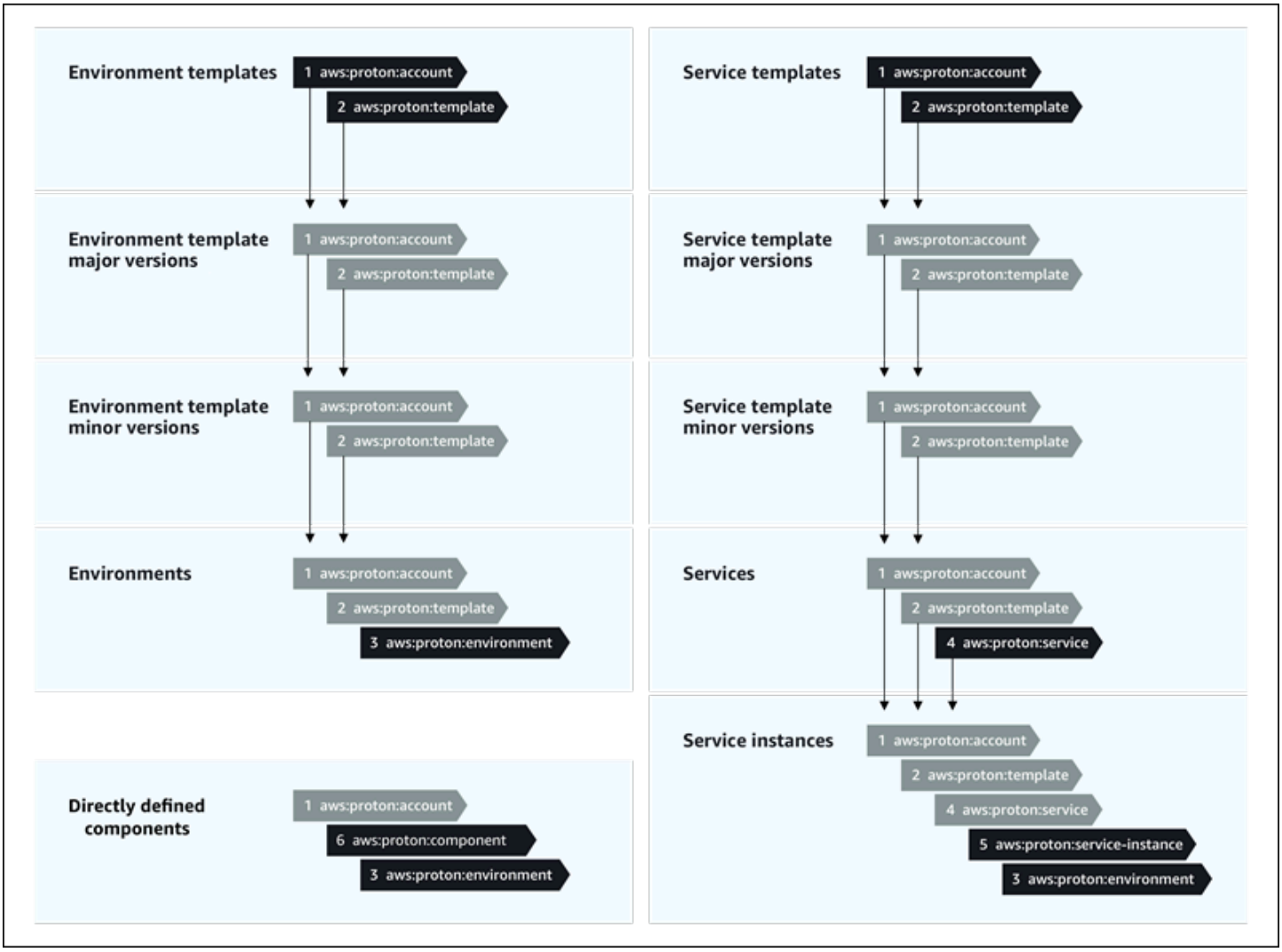
AWS Proton etichettatura

Con AWS Proton, puoi utilizzare sia i tag che crei sia i tag generati AWS Proton automaticamente per te.

AWS ProtonAWS tag gestiti

Quando si crea una AWS Proton risorsa, genera AWS Proton automaticamente tag AWS gestiti per la nuova risorsa, come illustrato nel diagramma seguente. AWS i tag gestiti si propagano successivamente ad altre AWS Proton risorse basate sulla nuova risorsa. Ad esempio, i tag gestiti di un modello di ambiente si propagano alle relative versioni e i tag gestiti di un servizio si propagano alle relative istanze del servizio.

> **Note**
>
> AWS i tag gestiti non vengono generati per le connessioni agli account di ambiente. Per ulteriori informazioni, consulta the section called “Connessioni all'account”.



Propagazione dei tag alle risorse assegnate

Se le risorse assegnate, come quelle definite nei modelli di servizio e ambiente, supportano il AWS tagging, i tag gestiti si propagano come tag AWS gestiti dal cliente alle risorse assegnate. Questi tag non si propagano a una risorsa fornita che non supporta il tagging. AWS

AWS Proton applica i tag alle risorse in base AWS Proton agli account, ai modelli registrati e agli ambienti distribuiti, nonché ai servizi e alle istanze di servizio, come descritto nella tabella seguente. Puoi utilizzare i tag AWS gestiti per visualizzare e gestire le tue AWS Proton risorse, ma non puoi modificarli.

AWS chiave tag gestita	Chiave diffusa gestita dal cliente	Descrizione
`aws:proton:account`	`proton:account`	L' AWS account che crea e distribuisce le risorse. AWS Proton
`aws:proton:template`	`proton:template`	L'ARN di un modello selezionato.
`aws:proton:environment`	`proton:environment`	L'ARN di un ambiente selezionato.
`aws:proton:service`	`proton:service`	L'ARN di un servizio selezionato.
`aws:proton:service-instance`	`proton:service-instance`	L'ARN di un'istanza di servizio selezionata.
`aws:proton:component`	`proton:component`	L'ARN di un componente selezionato.

Di seguito è riportato un esempio di tag AWS gestito per una AWS Proton risorsa.

```
"aws:proton:template" = "arn:aws:proton:region-id:account-id:environment-template/env-template"
```

Di seguito è riportato un esempio di tag gestito dal cliente applicato a una risorsa fornita che è stata propagata da un tag AWS gestito.

```
"proton:environment:database" = "arn:aws:proton:region-id:account-id:rds/env-db"
```

Con AWS-managed provisioning, AWS Proton applica i tag propagati direttamente alle risorse assegnate.

Con il provisioning autogestito, AWS Proton rende disponibili i tag propagati insieme ai file IaC renderizzati che invia nella pull request (PR) di provisioning. I tag sono forniti nella variabile della mappa delle stringhe. `proton_tags` Ti consigliamo di fare riferimento a questa variabile nella

configurazione di Terraform in `default_tags` cui includere i AWS Proton tag. Questo propaga i AWS Proton tag a tutte le risorse fornite.

L'esempio seguente mostra questo metodo di propagazione dei tag in un ambiente Terraform template.

Ecco la definizione della `proton_tags` variabile:

proton.environment.variables.tf:

```
variable "environment" {
  type = object({
    inputs = map(string)
    name = string
  })
}

variable "proton_tags" {
  type = map(string)
  default = null
}
```

Ecco come vengono assegnati i valori dei tag a questa variabile:

proton.auto.tfvars.json:

```
{
  "environment": {
    "name": "dev",
    "inputs": {
      "ssm_parameter_value": "MyNewParamValue"
    }
  }

  "proton_tags" : {
    "proton:account" : "123456789012",
    "proton:template" : "arn:aws:proton:us-east-1:123456789012:environment-template/
fargate-env",
    "proton:environment" : "arn:aws:proton:us-east-1:123456789012:environment/dev"
  }
}
```

Ed ecco come puoi aggiungere AWS Proton tag alla tua configurazione Terraform in modo che vengano aggiunti alle risorse fornite:

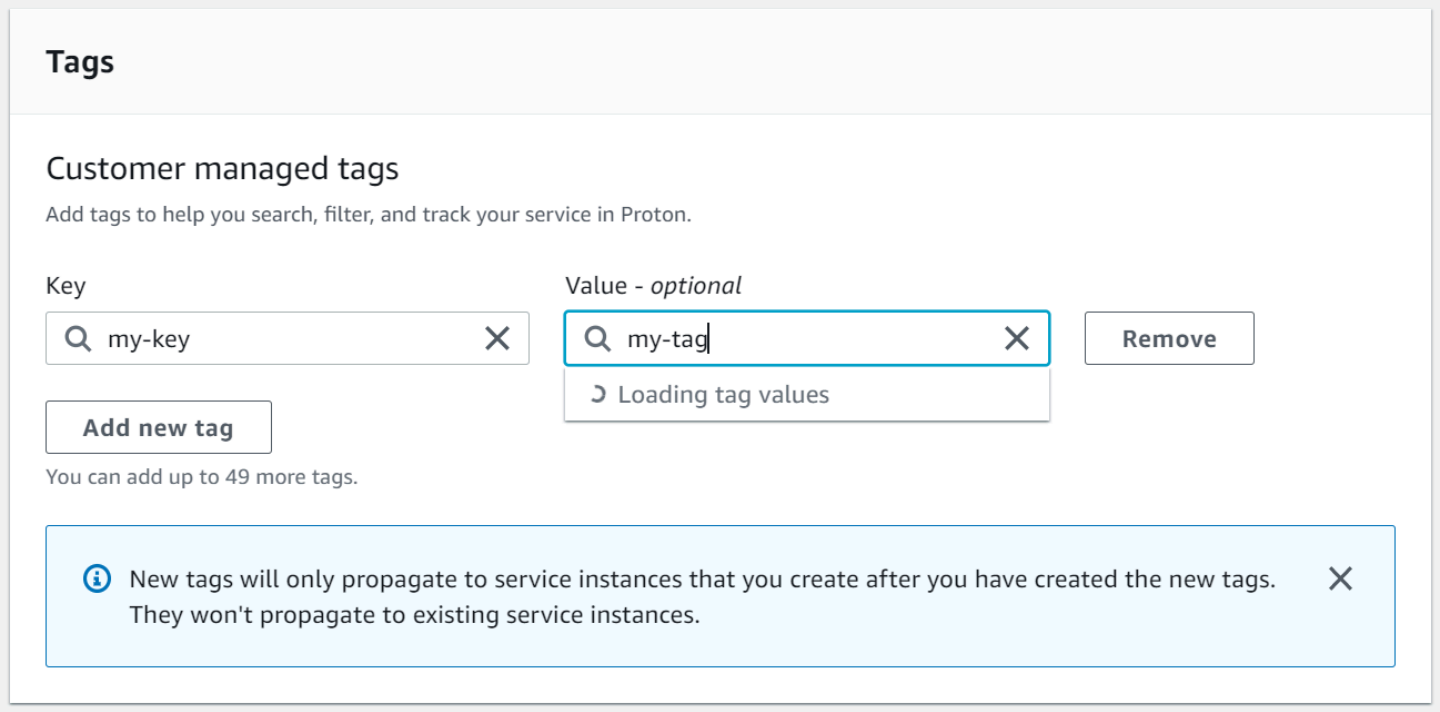
```
# Configure the AWS Provider
provider "aws" {
  region = var.aws_region
  default_tags {
    tags = var.proton_tags
  }
}
```

Tag gestiti dal cliente

Ogni AWS Proton risorsa ha una quota massima di 50 tag gestiti dai clienti. I tag gestiti dal cliente si propagano alle AWS Proton risorse secondarie allo stesso modo dei tag AWS gestiti, tranne per il fatto che non si propagano alle AWS Proton risorse esistenti o alle risorse assegnate. Se applichi un nuovo tag a una AWS Proton risorsa con risorse secondarie esistenti e desideri che le risorse secondarie esistenti vengano contrassegnate con il nuovo tag, devi etichettare ogni risorsa secondaria esistente manualmente, utilizzando la console o. AWS CLI

Crea tag utilizzando la console e la CLI

Quando crei una AWS Proton risorsa utilizzando la console, hai la possibilità di creare tag gestiti dai clienti nella prima o nella seconda pagina della procedura di creazione, come mostrato nella seguente istantanea della console. Scegli Aggiungi nuovo tag, inserisci la chiave e il valore e procedi.



Dopo aver creato una nuova risorsa utilizzando la AWS Proton console, puoi visualizzarne l'elenco dei tag AWS gestiti e gestiti dai clienti dalla pagina dei dettagli.

Crea o modifica un tag

1. Nella AWS Proton console, apri una pagina di dettaglio AWS Proton delle risorse in cui puoi visualizzare un elenco di tag.
2. Scegliere Gestisci tag.
3. Nella pagina Gestisci i tag, puoi visualizzare, creare, rimuovere e modificare i tag. Non puoi modificare i tag AWS gestiti elencati in alto. Tuttavia, puoi aggiungere e modificare i tag gestiti dal cliente modificando i campi, elencati dopo i tag AWS gestiti.

   Scegli Aggiungi nuovo tag per creare un nuovo tag.
4. Inserisci una chiave e un valore per il nuovo tag.
5. Per modificare un tag, inserisci un valore nel campo del valore del tag per una chiave selezionata.
6. Per eliminare un tag, scegli Rimuovi per un tag selezionato.
7. Una volta completate le modifiche, scegliete Salva modifiche.

Crea tag usando il AWS ProtonAWS CLI

È possibile visualizzare, creare, rimuovere e modificare i tag utilizzando AWS Proton AWS CLI.

È possibile creare o modificare un tag per una risorsa come illustrato nell'esempio seguente.

```
$ aws proton tag-resource \
    --resource-arn "arn:aws:proton:region-id:account-id:service-template/webservice" \
    --tags '[{"key":"mykey1","value":"myval1"},{"key":"mykey2","value":"myval2"}]'
```

È possibile rimuovere un tag per una risorsa come illustrato nell'esempio seguente.

```
$ aws proton untag-resource \
    --resource-arn "arn:aws:proton:region-id:account-id:service-template/webservice" \
    --tag-keys '["mykey1","mykey2"]'
```

È possibile elencare i tag per una risorsa come mostrato nell'esempio finale.

```
$ aws proton list-tags-for-resource \
    --resource-arn "arn:aws:proton:region-id:account-id:service-template/webservice"
```

Risoluzione dei problemi AWS Proton

Scopri come risolvere i problemi con. AWS Proton

Argomenti

- Errori di distribuzione che fanno riferimento AWS CloudFormation a parametri dinamici

Errori di distribuzione che fanno riferimento AWS CloudFormation a parametri dinamici

Se vedi errori di distribuzione che fanno riferimento alle tue variabili CloudFormation dinamiche, verifica che si tratti di Jinja Escape. Questi errori possono essere causati da un'interpretazione errata da parte di Jinja delle variabili dinamiche. La sintassi CloudFormation dinamica dei parametri è molto simile alla sintassi Jinja che usi con i tuoi parametri. AWS Proton

Esempio di sintassi CloudFormation dinamica delle variabili:

`'{{resolve:secretsmanager:MySecret:SecretString:password:EXAMPLE1-90ab-cdef-fedc-ba987EXAMPLE}}'`.

Esempio di AWS Proton sintassi Jinja del parametro:

`'{{ service_instance.environment.outputs.env-outputs }}'`.

Per evitare questi errori di interpretazione, Jinja evita i parametri CloudFormation dinamici come mostrato negli esempi seguenti.

Questo esempio è tratto dalla Guida per l' AWS CloudFormation utente. I AWS Secrets Manager segmenti secret-name e json-key possono essere utilizzati per recuperare le credenziali di accesso memorizzate nel segreto.

```
MyRDSInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBName: 'MyRDSInstance'
      AllocatedStorage: '20'
      DBInstanceClass: db.t2.micro
      Engine: mysql
```

```
      MasterUsername: '{{resolve:secretsmanager:MyRDSSecret:SecretString:username}}'
      MasterUserPassword:
 '{{resolve:secretsmanager:MyRDSSecret:SecretString:password}}'
```

Per sfuggire ai parametri dinamici è possibile utilizzare due metodi CloudFormation diversi:

- Racchiudi un blocco tra`{% raw %}` and `{% endraw %}`:

```
'{% raw %}'
MyRDSInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBName: 'MyRDSInstance'
      AllocatedStorage: '20'
      DBInstanceClass: db.t2.micro
      Engine: mysql
      MasterUsername: '{{resolve:secretsmanager:MyRDSSecret:SecretString:username}}'
      MasterUserPassword:
 '{{resolve:secretsmanager:MyRDSSecret:SecretString:password}}'
'{% endraw %}'
```

- Racchiudi un parametro tra: `"{{ }}"`

```
MyRDSInstance:
  Type: AWS::RDS::DBInstance
  Properties:
    DBName: 'MyRDSInstance'
      AllocatedStorage: '20'
      DBInstanceClass: db.t2.micro
      Engine: mysql
      MasterUsername:
 "{{ '{{resolve:secretsmanager:MyRDSSecret:SecretString:username}}' }}"
      MasterUserPassword:
 "{{ '{{resolve:secretsmanager:MyRDSSecret:SecretString:password}}' }}"
```

Per informazioni, vedi Jinja Escaping.

AWS Proton quote

La tabella seguente elenca le AWS Proton quote. Tutti i valori sono per AWS account, per AWS regione supportata.

Quota di risorse	Limite predefinito	Modificabile?
Dimensione massima del pacchetto di modelli	10 MB	✗ No
Dimensione massima del file manifesto del modello	2 MB	✗ No
Dimensione massima del file dello schema del modello	2 MB	✗ No
Dimensione massima di ogni file modello	2 MB	✗ No
Lunghezza massima di ogni nome di modello	100 caratteri	✗ No
Numero massimo di file CloudFormation modello per pacchetto	1	✗ No
Numero massimo di modelli registrati per account, modelli di servizio e ambiente combinati	1000	✓ Sì
Numero massimo di versioni per modello registrate per modello	1000	✓ Sì
Numero massimo di file per pacchetto di CodeBuild Provisioning	500	✗ No
Numero massimo di ambienti per account	1000	✓ Sì
Numero massimo di servizi per account	1000	✓ Sì
Numero massimo di istanze di servizio per servizio	20	✓ Sì
Numero massimo di componenti per account	1000	✓ Sì
Numero massimo di connessioni dell'account ambiente per account ambiente	1000	✓ Sì

Cronologia dei documenti

La tabella seguente descrive le modifiche importanti alla documentazione relativa all'ultima versione di AWS Proton e al feedback dei clienti. Per ricevere notifiche sugli aggiornamenti di questa documentazione, è possibile sottoscrivere un feed RSS.

- Versione API: 2020-07-20

Modifica	Descrizione	Data
Aggiornamento della policy gestita	`AWSProtonCodeBuildProvisioningServiceRolePolicy` la politica è stata aggiornata.	15 giugno 2024
Aggiornamento della policy gestita	`AWSProtonDeveloperAccess` la politica è stata aggiornata.	25 aprile 2024
Aggiornamento della policy gestita	`AWSProtonFullAccess` la politica è stata aggiornata.	25 aprile 2024
Aggiornamento della policy gestita	`AWSProtonSyncServiceRolePolicy` la politica è stata aggiornata.	25 aprile 2024
Aggiornamento della policy gestita	`AWSProtonCodeBuildProvisioningServiceRolePolicy` la politica è stata aggiornata.	12 maggio 2023
configurazioni di sincronizzazione dei servizi.	AWS Proton aggiunge il supporto per le configurazioni di sincronizzazione dei servizi.	31 marzo 2023

CodeBuild	AWS Proton aggiunge il supporto per il CodeBuild provisioning.	16 novembre 2022
Aggiornamento della policy gestita	È stata aggiunta una `AWSProtonCodeBuildProvisioningBasicAccess` politica che fornisce CodeBuild le autorizzazioni necessarie per eseguire una build for AWS Proton CodeBuild Provisioning.	11 novembre 2022
Propagazione dei tag Terraform	Aggiunta la propagazione dei tag Terraform al capitolo Tagging.	16 settembre 2022
Guida alla migrazione delle API	È stata rimossa la guida alla migrazione dell'API precedente a GA.	12 agosto 2022
AWS Proton oggetti	È stato aggiunto un argomento sugli AWS Proton oggetti e sulla loro relazione con altri oggetti AWS e di terze parti. Vedi AWS Proton oggetti.	29 luglio 2022
Chiarimenti sul repository collegato	È stato chiarito lo scopo degli archivi collegati (registrati) e il loro utilizzo in tutta la guida.	18 luglio 2022
Unisci le guide	Le due guide separate per l'amministratore e l'utente sono state unite in un'unica guida, la Guida per l'AWS Proton utente.	30 giugno 2022

Aggiornamento della policy gestita	Politiche gestite aggiornate per fornire l'accesso a nuove operazioni AWS Proton API e risolvere i problemi di autorizzazione per alcune operazioni AWS Proton della console. Vedi le politiche AWS gestite per AWS Proton.	20 giugno 2022
Guida introduttiva alla CLI	Aggiornato Inizia AWS CLI con un nuovo tutorial che utilizza la nuova libreria di modelli.	14 giugno 2022
Componenti definiti direttamente	È stato aggiunto il capitolo Componenti e sono state apportate le relative modifiche in tutta la guida.	1 giugno 2022
AWS Proton libreria di modelli	Aggiunto l'argomento relativo alla libreria di AWS Proton modelli.	6 maggio 2022
Disponibilità generale (GA) di Terraform	Il provisioning delle pull request è stato rinominato in provisioning autogestito. È stato aggiunto l'argomento sui metodi di provisioning.	23 marzo 2022
Etichettatura del repository	È stato aggiunto il supporto per l'etichettatura delle risorse del Repository. Vedi Creare un link al tuo repository.	23 marzo 2022
Aggiornamento della documentazione	È stata aggiunta la codifica della connessione all'account ambientale.	26 novembre 2021

Sincronizzazione dei modelli e anteprima di Terraform	Aggiunto il controllo automatico delle versioni dei modelli con la funzione di sincronizzazione dei modelli per la disponibilità generale e il provisioning delle pull request con Terraform in anteprima. La guida alla migrazione delle API è nuovamente disponibile.	24 novembre 2021
Aggiornamenti della documentazione	Sono stati aggiunti EventBridgetutorial, flusso di lavoro introduttivo, How AWS Proton works e miglioramenti alla sezione Template bundle.	17 settembre 2021
AWS Proton rilascio dei pannelli di aiuto della console	Pannelli di aiuto aggiunti alla console. L'eliminazione della versione del modello di console non elimina più le versioni precedenti. La guida alla migrazione delle API è stata rimossa.	8 settembre 2021
AWS Proton versione di disponibilità generale (GA)	Aggiunti ambienti tra account, EventBridge monitoraggio, chiavi di condizione IAM, supporto per l'idempotenza e aumento delle quote.	9 giugno 2021

Aggiungi ed elimina istanze di servizio per un servizio e utilizza l'infrastruttura esterna esistente per ambienti con AWS Proton	Questa versione di anteprima pubblica include aggiornamenti che consentono di aggiungere ed eliminare istanze di servizio da un servizio, utilizzare l'infrastruttura esterna esistente in un AWS Proton ambiente e annullare le distribuzioni di ambienti, istanze di servizio e pipeline. AWS Proton ora supporta. PrivateLink È stata aggiunta un'ulteriore convalida dell'eliminazione per evitare che una versione secondaria venga erroneamente eliminata mentre una risorsa la sta utilizzando.	27 aprile 2021
Contrassegnare con AWS Proton	La versione 2 di anteprima pubblica include l' AWS Proton etichettatura e la possibilità di avviare servizi senza una pipeline di servizi.	5 marzo 2021
Versione iniziale	La versione di anteprima pubblica è ora disponibile in aree geografiche selezionate.	1 dicembre 2020

AWS Glossario

Per la AWS terminologia più recente, consultate il AWS glossario nella sezione Reference.Glossario AWS

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.