Aumentar a resiliência e melhorar a experiência do cliente usando a engenharia do caos no AWS

AWS Recomendações

AWS Recomendações: Aumentar a resiliência e melhorar a experiência do cliente usando a engenharia do caos no AWS

Table of Contents

Aumentar a resiliência e melhorar a experiência do cliente usando a engenharia do caos no AWS

Laurent Domb, tecnólogo chefe de finanças federais da Amazon Web Services

Abril de 2025 (histórico do documento)

A engenharia do caos é a disciplina de fazer experiências em um aplicativo para criar confiança na capacidade da sua organização e do aplicativo de suportar condições turbulentas na produção. É uma abordagem proativa à resiliência, com o objetivo de verificar se seu aplicativo e sua organização são capazes de absorver, se adaptar e, eventualmente, se recuperar das deficiências do serviço, introduzindo falhas controladas entre pessoas, processos e tecnologias. A intenção também é identificar e eliminar pontos fracos antes que eles possam causar interrupções ou outras interrupções na produção.

Na Amazon, entendemos que a falha é inevitável em sistemas distribuídos, a ponto de funcionar apesar da presença de falhas ser um modo normal de operação. Como as interações entre os serviços estão fadadas a falhar, você precisa entender como seus serviços reagem durante vários modos de falha e criar serviços que sejam resilientes às principais vulnerabilidades, como falhas de dependência, tempestades de novas tentativas, zonas de disponibilidade prejudicadas e esgotamento dos recursos do host.

Vamos dar o exemplo de uma nova tempestade. Uma falha localizada em um cliente pode afetar vários serviços de forma significativa. Isso é comumente chamado de efeito borboleta. Uma tempestade de novas tentativas é uma manifestação do efeito borboleta, em que uma dependência com falha faz com que clientes e clientes desses clientes tentem novamente a operação que falhou, levando a um crescimento exponencial no tráfego. Os serviços ficam sobrecarregados porque precisam responder ao tráfego regular, além de repetir o tráfego, enquanto lidam com uma degradação no desempenho.

A engenharia do caos surgiu como uma resposta à crescente complexidade dos sistemas distribuídos. É uma abordagem multidisciplinar que combina princípios da teoria do caos, pensamento sistêmico e engenharia para projetar e gerenciar sistemas complexos que são resilientes a eventos e comportamentos inesperados. Em sua essência, a engenharia do caos se preocupa em entender e gerenciar o comportamento de sistemas complexos sob condições de incerteza e imprevisibilidade. Ele reconhece que as abordagens tradicionais de engenharia, que dependem da previsão e controle de resultados, geralmente são insuficientes para lidar com a

natureza complexa e dinâmica dos sistemas distribuídos. À medida que esses sistemas crescem, eles geralmente excedem o escopo de compreensão de qualquer indivíduo.

A engenharia do caos fornece conceitos, técnicas e ferramentas para injetar falhas intencionalmente nos sistemas e descobrir pontos fracos antes que eles se manifestem na produção. Essa abordagem proativa permite que as organizações criem a confiança de que seus sistemas funcionarão sob condições estressantes. Embora a engenharia do caos ainda seja uma prática em evolução, ela representa uma mudança fundamental para projetar, gerenciar e operar sistemas de computação modernos para serem resilientes diante da crescente complexidade e interconexão.

As seções a seguir deste guia discutem os benefícios da engenharia do caos, explicam como conduzir experimentos de engenharia do caos e descrevem as abordagens que você pode adotar para implementar a engenharia do caos em grande escala em sua organização. Também estão incluídos exemplos de documentos de planejamento e resultados de experimentos que você pode usar como modelos para seus experimentos de engenharia do caos.

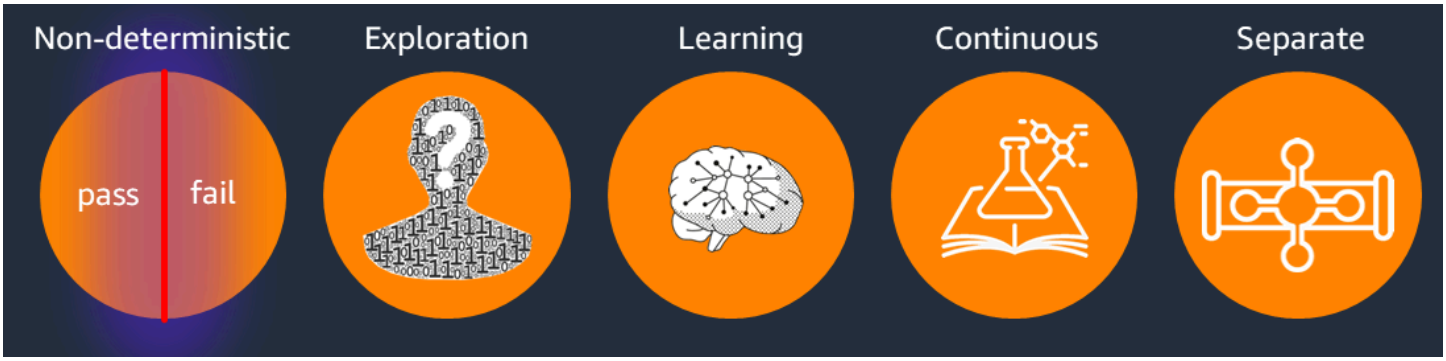
A próxima seção explora como as características da engenharia do caos diferem dos testes de resiliência tradicionais, como testes unitários, de fumaça ou de integração.

Visão geral do

Comparando os testes de resiliência com a engenharia do caos

O teste de resiliência é determinístico. Ou seja, ele valida características conhecidas sobre mecanismos de resiliência, como disjuntores, novas tentativas, failovers ou fallbacks, que foram implementados em seu aplicativo. Ele confirma como esses componentes do aplicativo absorvem interrupções controladas com mínimo ou nenhum impacto sobre o usuário. Portanto, o teste de resiliência se concentra na validação de modos de falha conhecidos que são injetados nos componentes do aplicativo com o objetivo de produzir pass/fail resultados. Você deve executar testes de resiliência continuamente como uma etapa em seu pipeline para garantir que você não introduza regressões em sua postura de resiliência. Nos testes de resiliência, você geralmente não executa testes em componentes reais, mas simula APIs que simulam um determinado componente. Essa abordagem permite testes consistentes e reproduzíveis de cenários de falha em um ambiente controlado, tornando-a adequada para testes automatizados de integração e regressão de tubulações.

Em contraste, a engenharia do caos não é determinística. Ou seja, ele é baseado em hipóteses e verifica seu modelo mental sobre como o aplicativo e suas dependências (pessoas, processos e tecnologia) absorvem, se adaptam e, eventualmente, se recuperam de modos de falha imprevistos. Portanto, a engenharia do caos se concentra na verificação de ponta a ponta de modos de falha desconhecidos, com o objetivo de detectar defeitos precocemente e corrigi-los antes que se transformem em eventos de grande escala. A engenharia do caos promove o aprendizado contínuo e deve ser praticada por meio de um pipeline de caos separado ou de experimentos ad-hoc que permitem que você execute vários experimentos a qualquer momento sem bloquear a produtividade do desenvolvedor na implantação do código.



O processo de engenharia do caos geralmente começa com um dia de jogo do caos, que é um evento dedicado em que as equipes injetam intencionalmente falhas ou falhas controladas em seus aplicativos. O dia do jogo é progressivo: começa em ambientes de nível inferior (como desenvolvimento ou teste) e avança gradualmente para ambientes de nível superior (como preparação e pré-produção) à medida que aumenta a confiança. Ao percorrer sistematicamente esses ambientes, as equipes podem verificar se seus sistemas toleram adequadamente as falhas injetadas antes de chegarem à produção. Essa progressão metódica garante que, quando os experimentos de caos forem conduzidos em ambientes de produção, as equipes tenham adquirido uma confiança substancial nas capacidades de resiliência de seus sistemas. O processo do dia do jogo é uma abordagem proativa para identificar pontos fracos e vulnerabilidades na arquitetura e nas práticas operacionais de um aplicativo, ao mesmo tempo em que elimina o estresse do aprendizado durante uma interrupção inesperada na produção.

O valor da engenharia do caos

Sistemas complexos são onipresentes no mundo atual. Eles desempenham um papel fundamental em muitos aspectos de nossas vidas, dos mercados financeiros aos cuidados de saúde. Esperamos que esses sistemas estejam sempre operacionais. No entanto, sistemas complexos geralmente são vulneráveis a eventos e comportamentos inesperados que podem ter consequências significativas. As organizações precisam planejar a disrupção em vez de se perguntar se isso acontecerá. Eles podem fazer isso aplicando testes de cenários em seus serviços comerciais críticos ou de missão crítica. É aqui que a engenharia do caos entra em ação.

A engenharia do caos oferece uma abordagem para gerenciar sistemas complexos que podem ajudar a mitigar riscos e melhorar a resiliência. O processo de preparação para experimentos de caos exige que as equipes desenvolvam hipóteses sobre o comportamento de seus sistemas, o que aprofunda sua compreensão de como os sistemas são construídos e como operam. Essa preparação geralmente revela lacunas mentais, percepções arquitetônicas e conhecimento

operacional que, de outra forma, poderiam permanecer desconhecidos. Ao aprofundar a compreensão de como sistemas complexos reagem às falhas, a engenharia do caos promove maior transparência e responsabilidade no design e gerenciamento do sistema. Quanto mais frequentemente sua organização pratica a engenharia do caos, mais bem preparada ela se torna operacionalmente. A engenharia do caos ajuda você a estabelecer as melhores práticas para projetar aplicativos resilientes que possam sobreviver a falhas de componentes com o mínimo ou nenhum impacto no usuário. Isso garante que os aplicativos essenciais operem dentro dos níveis de serviço esperados e da tolerância ao impacto, ao mesmo tempo em que aprimora continuamente o conhecimento da equipe sobre seus próprios sistemas.

Preparando-se para condições adversas

Ao criar AWS, você usa diferentes tipos de serviços, incluindo serviços zonais, como o Amazon Elastic Compute Cloud (Amazon EC2), serviços regionais, como o Amazon Simple Storage Service (Amazon S3), serviços globais, como (IAM), serviços de software AWS Identity and Access Management como serviço (SaaS) de terceiros e serviços locais. Cada tipo de serviço expõe diferentes domínios de falha que você precisa considerar. Como você se prepara para eventos autoinfligidos ou causados por terceiros sobre os quais sua organização não tem controle?

Para ajudar a entender como seu aplicativo pode responder a condições adversas, você pode usar AWS Fault Injection Service (AWS FIS). AWS FIS é um serviço totalmente gerenciado para executar experimentos de injeção de falhas de forma controlada. Você pode usar esse serviço para injetar cenários AWS fornecidos, como interrupções de energia na zona de disponibilidade e problemas de conectividade entre regiões, ou criar seus próprios experimentos agrupando uma ampla variedade de ações de falha fornecidas pelo serviço. AWS FIS permite que suas equipes pratiquem e aprendam continuamente como seus aplicativos reagiriam a falhas comuns e corrigiriam defeitos à medida que os detectassem.

Praticando engenharia de caos controlada

Os princípios fundamentais dos experimentos de caos controlado são:

- Comece com um ambiente semelhante ao seu ambiente de produção.
- Estabeleça uma hipótese e interrompa as condições para seu experimento.
- Comece devagar.
- Exerça controle sobre seus experimentos de caos.

- Defina o escopo do impacto.
- Conheça a linha de base do seu serviço.
- Agende experimentos.
- Corrija primeiro e depois experimente.
- Monitore o experimento de perto.
- Aprenda com seus resultados.
- Priorize as descobertas, corrija e verifique.
- Propague os aprendizados em toda a sua organização.

Para escalar com sucesso a engenharia do caos, você deve implementar experimentos de caos de forma controlada. Ao usar AWS FIS, você pode criar condições de parada usando os CloudWatchalarmes da Amazon. Você pode incorporar essas condições em um modelo de experimento para garantir que os experimentos sejam interrompidos se estiverem fora dos limites e revertidos ao último estado conhecido. AWS FIS também fornece alavancas de segurança. Quando você aciona essas alavancas, AWS FIS interrompe e reverte todos os experimentos em execução na conta do Região da AWS, incluindo experimentos com várias contas, e impede o início de novos experimentos. Isso evita a injeção de falhas durante determinados períodos de tempo, como horários comerciais, eventos de vendas ou lançamentos de produtos, ou em resposta a alarmes de integridade de aplicativos. A alavanca de segurança permanece engatada até ser desengatada manualmente.

Ao conduzir um experimento de caos, você deve definir salvaguardas para evitar efeitos colaterais indesejáveis no ambiente, especialmente se houver a possibilidade de o experimento afetar os aplicativos que estão em produção. Ao planejar o experimento, antecipe quaisquer efeitos adversos que ele possa ter em outras aplicações no ambiente. Por exemplo, outros aplicativos podem receber mensagens errôneas do aplicativo que faz parte do experimento, enfrentar altos volumes de solicitações ou enfrentar contenção de recursos se compartilharem infraestrutura. Documente esses riscos e resolva quaisquer problemas conhecidos ou inaceitáveis antes de realizar o experimento.

Começando com a engenharia do caos

Antes de realizar um experimento, recomendamos que você implemente alguns itens essenciais para aproveitar ao máximo suas práticas de engenharia do caos. Esses itens essenciais incluem:

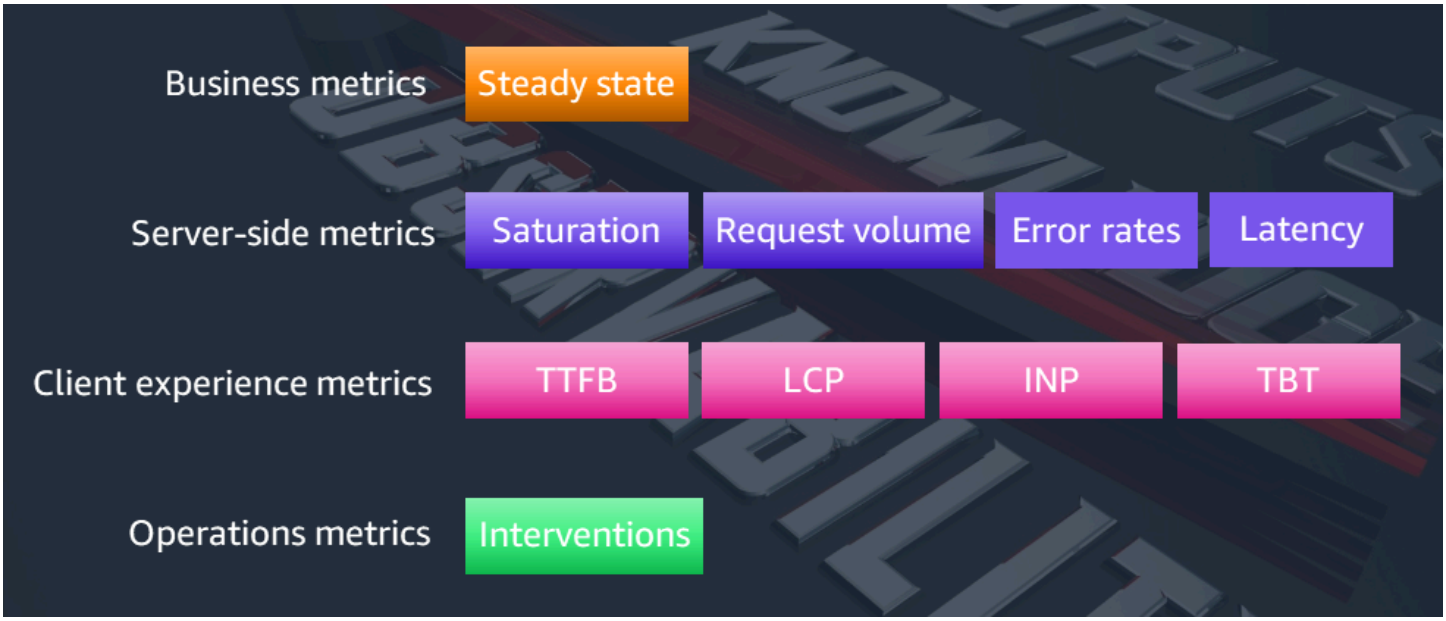
- Observabilidade (métricas, registro, rastreamento de solicitações)
- Uma lista de eventos ou falhas do mundo real que você gostaria de explorar
- Patrocínio de resiliência organizacional por meio da adesão da liderança
- Priorização de descobertas críticas, com base no impacto potencial nos negócios, em relação aos novos recursos que são descobertos ao realizar experimentos de caos

Observabilidade para experimentos de caos

A observabilidade, que inclui métricas, registros e rastreamento de solicitações, desempenha um papel fundamental na engenharia do caos. Você vai querer entender as métricas de negócios, métricas do lado do servidor, métricas de experiência do cliente e métricas de operações ao realizar um experimento. Sem a observabilidade, você não conseguirá definir um comportamento estável ou criar um experimento significativo para verificar se sua hipótese sobre seu aplicativo é verdadeira.

Metrics

O diagrama a seguir mostra os tipos de métricas que você pode monitorar para experimentos de caos em diferentes tipos de aplicativos.



- Métricas de negócios — O estado estável indica a operação normal do seu sistema e é definido pelas métricas do seu negócio. Ela pode ser representada por transações por segundo (TPS), fluxos de cliques por segundo, pedidos por segundo ou uma medida similar. Seu aplicativo exibe um estado estável quando está operando conforme o esperado. Portanto, verifique se o aplicativo está íntegro antes de executar experimentos. O estado estável não significa necessariamente que não haverá impacto no aplicativo quando ocorrer uma falha, pois uma porcentagem de falhas pode estar dentro de limites aceitáveis. O estado estacionário é sua linha de base. Por exemplo, o estado estável de um sistema de pagamentos pode ser definido como o processamento de 300 TPS com uma taxa de sucesso de 99% e tempo de ida e volta de 500 ms. Visualmente, pense no estado estacionário como um eletrocardiograma (EKG). Se o estado estável do seu sistema flutuar repentinamente, você sabe que há um problema com seu serviço.
- Server-side métricas — Para entender o desempenho de seus recursos durante o experimento, você precisa de insights sobre o desempenho deles antes, durante e depois do experimento. Para medir o impacto de seus recursos AWS, você pode usar a Amazon CloudWatch. CloudWatch é um serviço que monitora aplicativos, responde às mudanças de desempenho, otimiza o uso de recursos e fornece informações sobre a integridade operacional. Durante seus experimentos, você desejará capturar métricas do lado do servidor, como saturação, volumes de solicitações, taxas de erro e latência.
- Métricas de experiência do cliente — Ativado AWS, você pode capturar métricas reais do usuário usando o CloudWatchRUM para simular solicitações do usuário por meio de ferramentas como Locust, Grafana k6, Selenium ou Puppeteer. Métricas reais de usuários são cruciais para organizações que realizam experimentos de engenharia do caos. Ao monitorar como os usuários

reais são afetados durante um experimento, as equipes podem ter uma visão precisa de como as falhas e interrupções afetarão os clientes na produção. As principais métricas de experiência do cliente são Time to First Byte (TTFB), Largest Contentful Paint (LCP), Interaction to Next Paint (INP) e Total Blocking Time (TBT).

- Métricas de operações — As intervenções medem o sucesso na mitigação de falhas de forma automatizada – por exemplo, a latência bem-sucedida da solicitação do cliente durante a reinicialização de pods, tarefas ou instâncias do EC2 com mecanismos como controlador de replicação ou escalabilidade automática. Ser capaz de intervir automaticamente durante uma falha se correlaciona diretamente com uma boa experiência do usuário. Entender se há alguma mudança nesses mecanismos de mitigação ao longo do tempo é crucial. Ao definir métricas para mitigações automatizadas bem-sucedidas e fracassadas, você cria diretrizes que ajudam a identificar possíveis regressões em todo o sistema.

Registro em log

O registro centralizado é fundamental para entender os estados dos componentes do seu aplicativo antes, durante e depois de um experimento de caos. Recomendamos que você colete registros de todos os componentes do seu aplicativo para criar uma visão consolidada do que cada componente estava fazendo no momento em que o experimento foi injetado. Isso fornece uma imagem clara do fluxo do experimento de ponta a ponta.

Rastreamento de solicitação

O rastreamento de solicitações permite que você observe o fluxo de qualquer solicitação única nos componentes do seu aplicativo para obter uma compreensão abrangente do impacto que a falha injetada tem no sistema e em suas dependências. Ao rastrear as solicitações, você pode ver como a falha se propaga por meio de diferentes serviços e componentes, para que você possa avaliar melhor o escopo da interrupção. Para rastrear suas solicitações AWS, você pode usar AWS X-Ray.

Cenários de falha para injetar em experimentos de caos

O objetivo de injetar falhas comuns em seu aplicativo é entender como o aplicativo reage a esses eventos inesperados, para que você possa criar mecanismos de mitigação e tornar seu sistema resiliente a essas falhas. Além disso, você deve usar a engenharia do caos para reproduzir cenários históricos de falhas para verificar se seus mecanismos de mitigação ainda estão funcionando conforme o esperado e não se alteraram com o tempo.

Considere os eventos a seguir ao planejar seus experimentos de engenharia do caos.

Modo de falha	Description
Deficiência do servidor	Reinicie instâncias do EC2, exclua pods do Kubernetes ou tarefas do Amazon Elastic Container Service (Amazon ECS) para entender como seu aplicativo reage a essas falhas.
Erros de API	Injete falhas nas AWS suas próprias APIs de serviço para entender o comportamento do aplicativo.
Problemas de rede	Introduza latência ou congestionamento ou bloqueie conexões para imitar problemas de rede do mundo real.
AWS Diminuição da zona de disponibilidade	Repita a deficiência de uma zona de disponibilidade inteira para verificar a recuperação em todas as zonas.
Região da AWS comprometimento da conectividade	Repita uma falha na rede Regiões da AWS para verificar como os recursos na região secundária reagem a esse evento.
Falhas no banco de	Faça o failover de réplicas de banco de dados ou corrompa dados, ou torne as instâncias do banco de dados inacessíveis, para entender o impacto em suas estratégias de aplicativo e recuperação.
Pausa no banco de dados e na replicação do Amazon S3	Pause a replicação do banco de dados ou do Amazon Simple Storage Service (Amazon S3) entre zonas de disponibilidade ou Regiões da AWS para entender o impacto do aplicativo downstream.

Modo de falha	Description
Degradação do armazenamento	Pause I/O, remova volumes do Amazon Elastic Block Store (Amazon EBS) ou exclua arquivos para verificar a durabilidade e a recuperação dos dados.
Diminuição da dependência	Reduza ou diminua o desempenho dos serviços downstream ou upstream dos quais você depende, incluindo serviços de terceiros , para entender o fluxo e o impacto de ponta a ponta para seus clientes.
Surtos de tráfego	Gere picos no tráfego de usuários para testar os recursos de escalabilidade automática e veja como o tempo de inicialização a frio pode afetar o estado geral do aplicativo.
Esgotamento de recursos	Maximize a CPU, a memória e o espaço em disco para verificar a degradação normal do seu aplicativo.
Falhas em cascata	Inicie falhas primárias que se espalham em cascata para aplicativos e componentes posteriores.
Implantações incorretas	Implemente alterações ou configurações problemáticas para verificar os mecanismos de reversão.

Patrocínio de resiliência organizacional

A engenharia do caos fornece mais valor quando aplicada em toda a organização. Recomendamos que você trabalhe com um patrocinador executivo que possa ajudar a definir metas de resiliência em toda a organização, eliminar o medo, a incerteza e a dúvida sobre o domínio e iniciar o processo de transformação para tornar a resiliência responsabilidade de todos.

Para apoiar o caso comercial da criação de uma prática de engenharia do caos, associe esforços de engenharia do caos aos seus projetos comerciais críticos. Fazer da resiliência um ativo e impulsionador da aceleração ajudará você a monitorar o sucesso ao longo do tempo. Comece com uma contagem de incidentes críticos por mês ou por trimestre, o tempo médio de recuperação e o impacto que esses incidentes causaram em seus clientes e organização. Estabeleça uma meta com suas equipes para reduzir o número de incidentes em um período de 6 a 12 meses, à medida que melhorias são feitas em suas pilhas de aplicativos em resposta aos experimentos de engenharia do caos.

Avalie se os incidentes são altamente repetitivos. Por exemplo, digamos que um certificado TLS expirado leve ao tempo de inatividade porque os clientes não conseguem estabelecer uma conexão confiável. Se vários incidentes ocorrerem em um ano devido à expiração de vários certificados TLS, você pode realizar um experimento de expiração de um certificado TLS e verificar se suas equipes recebem alertas ou são capazes de mitigar automaticamente esses problemas. Isso ajudará a garantir que você se torne resiliente a essas falhas.

Para acompanhar o progresso na engenharia do caos ao longo do tempo, capture as seguintes métricas para ajudar a destacar o valor da engenharia do caos em todo o ciclo de vida de um aplicativo:

- Taxa de incidentes reduzida — Acompanhe o número de incidentes de produção ao longo do tempo e correlacione esse número com a adoção da engenharia do caos. A expectativa é que a taxa de incidentes graves diminua.
- Tempo médio de resolução aprimorado (MTTR) — Calcule o tempo médio necessário para resolver incidentes e acompanhe esses dados para ver se eles melhoram com a engenharia do caos ao longo do tempo.
- Maior disponibilidade de aplicativos — use métricas de nível de serviço para mostrar melhorias na disponibilidade à medida que a resiliência dos aplicativos aumenta por meio de experimentos caóticos.
- Menor tempo de lançamento no mercado — A engenharia do caos pode fornecer a confiança necessária para lançar ofertas inovadoras com mais rapidez, porque você sabe que seus aplicativos são resilientes. Acompanhe os aumentos na velocidade de liberação do produto.
- Redução de custos operacionais — Quantifique se indicadores como ruído de alerta, carga operacional e esforço manual para gerenciar aplicativos diminuem com a implementação de práticas caóticas.

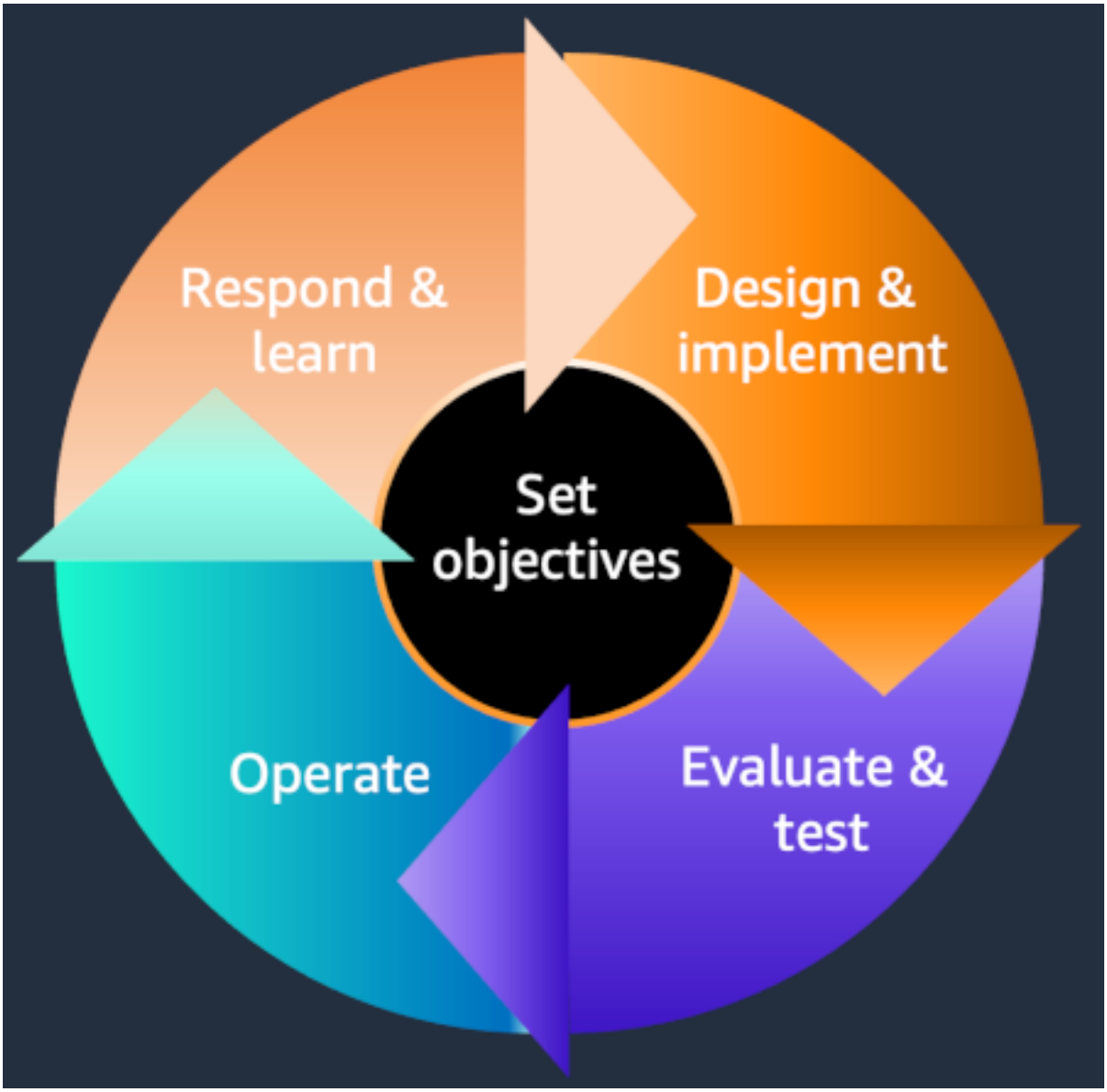
- Aumentar a confiança — Faça uma pesquisa com desenvolvedores, engenheiros de confiabilidade de sites (SREs) e outras equipes técnicas para avaliar se a engenharia do caos aumentou sua confiança na resiliência de aplicativos. As percepções importam.
- Experiências aprimoradas do cliente – Conecte a engenharia do caos a resultados positivos para os clientes, como menos interrupções no serviço, reversões e interrupções.

Priorizando a remediação

Ao realizar experimentos de caos, é provável que você identifique áreas de melhoria nas quais o aplicativo não funciona conforme o esperado. A correção desses itens se tornará um trabalho em sua lista de pendências que deverá ser priorizado junto com outros trabalhos, como o desenvolvimento de recursos. Recomendamos que você reserve um tempo para esses aprimoramentos para evitar futuras falhas. Considere priorizar essas tarefas de aprendizado e remediação com base no nível de impacto que elas podem causar. As descobertas que afetam diretamente a resiliência ou a segurança do seu aplicativo devem ter prioridade sobre os novos recursos, para evitar o impacto no cliente. Se a equipe tiver dificuldade em priorizar o trabalho de remediação em vez do desenvolvimento de recursos, considere entrar em contato com seu patrocinador executivo para garantir que as prioridades sejam definidas com base na tolerância ao risco comercial.

Implementando engenharia de caos em AWS

A engenharia do caos faz parte do estágio de avaliação e teste do ciclo de vida da AWS resiliência, conforme ilustrado no diagrama a seguir. Os aplicativos distribuídos não operam isoladamente de outros aplicativos ou clientes, portanto, recomendamos que você revise todo o ciclo de vida da resiliência. A mudança é constante em aplicativos distribuídos à medida que a rede evolui, os aplicativos upstream e downstream passam por mudanças e o uso do cliente muda com o tempo.



Para entender como essas mudanças em seu aplicativo podem afetar sua resiliência, faça da engenharia do caos uma parte de suas operações diárias. Você pode implementar experimentos de caos de diferentes maneiras:

- Ad hoc — Você pode realizar experimentos de caos como experimentos únicos para abordar um problema ou questão específica.
- Dias de jogo do caos — Esses são eventos estruturados e recorrentes projetados para verificar a confiabilidade e a resiliência de um aplicativo. O objetivo de um dia de jogo do caos é identificar possíveis problemas ou deficiências de resiliência entre pessoas, processos e tecnologias e praticar os processos e procedimentos para identificar, mitigar e responder a incidentes.
- Chaos pipeline — Integração e entrega contínuas (CI/CD) consistem em criar novos recursos e implantá-los com segurança em todos os ambientes. Para implementar experimentos de engenharia do caos, crie um pipeline do caos separado do seu CI/CD pipeline. Para entender o porquê, vamos supor que você queira adicionar um único experimento de engenharia de caos ao seu CI/CD pipeline que injete uma perda crescente de pacotes para componentes posteriores. Esse experimento é executado 3 vezes e leva 5 minutos para ser concluído a cada vez. A perda de pacotes aumenta de 10 por cento para 20 por cento a 30 por cento a cada execução, e o experimento leva 15 minutos no geral para ser concluído. Se você tiver 100 implantações paralelas, precisará esperar 1500 minutos para que um único experimento seja concluído. Se você tiver 10 experimentos para realizar, o impacto para seus desenvolvedores seria insuportável. Em grande escala, a engenharia do caos precisa de seu próprio pipeline que permita que você execute experimentos paralelamente ao seu processo de ciclo de vida de desenvolvimento de software (SDLC).
- Implantações do Canary — Os Canários fornecem um ambiente de teste para experimentos de caos. Ao direcionar uma pequena porcentagem do tráfego para um serviço canário ou usar métodos como espelhamento de tráfego ou repetição, você pode verificar novas alterações na infraestrutura ou no código sem impacto em seu sistema de produção estável. Você pode realizar experimentos contra o canário e injetar falhas conforme necessário, pois pode limitar o escopo do impacto para o usuário final.
- Experimentos programados — Você pode programar experimentos para verificar mecanismos de recuperação previsíveis para seu aplicativo. Use experimentos programados para reproduzir eventos comumente conhecidos e capturar como seus sistemas podem se recuperar de eventos como o encerramento de uma instância do EC2 por trás de um grupo de escalabilidade automática, a remoção de um pod do Kubernetes ou a exclusão de uma tarefa do Amazon ECS.

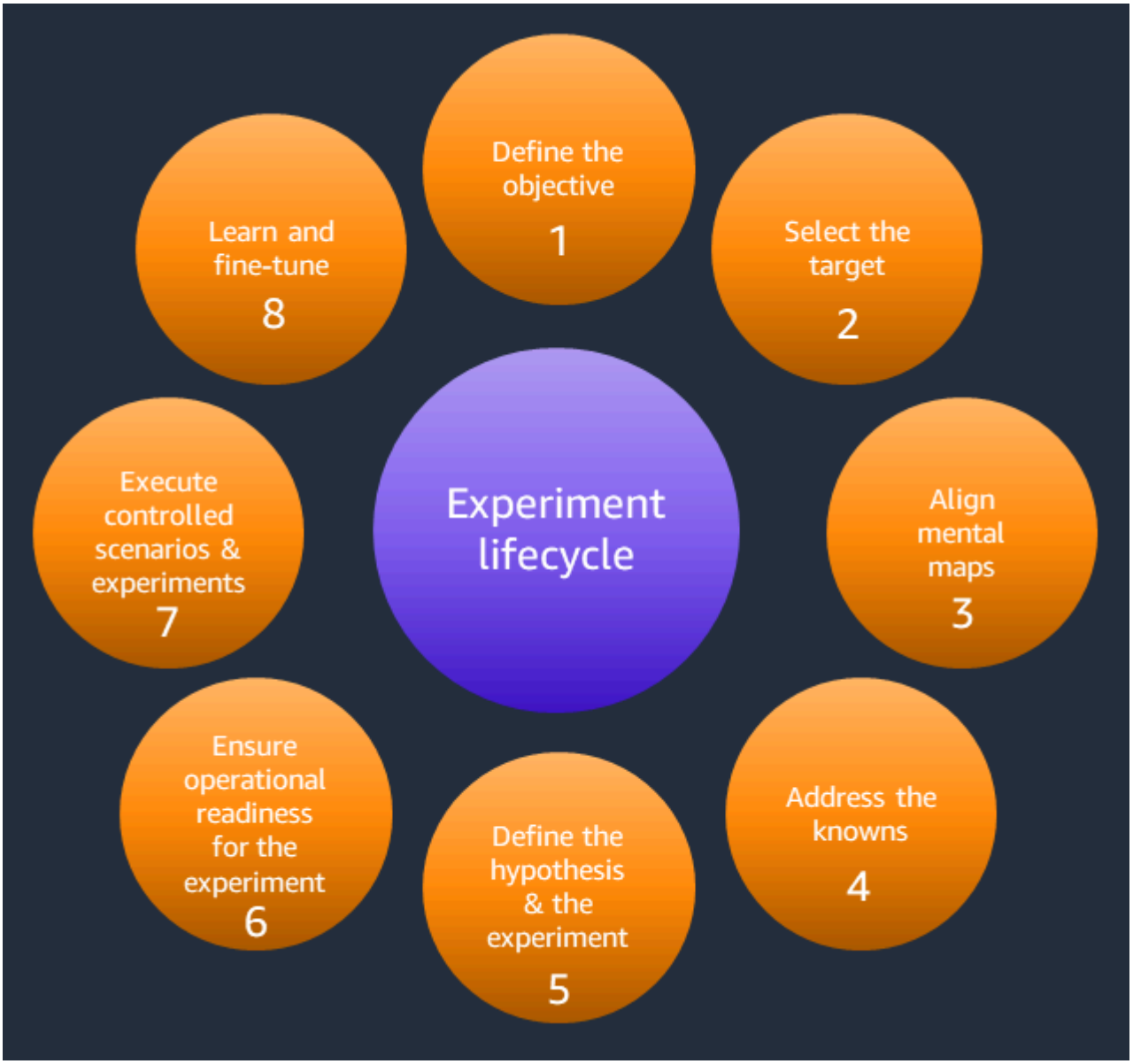
Ciclo de vida contínuo do experimento de engenharia do caos

Conforme discutido na seção anterior, você pode implementar experimentos de engenharia do caos de maneiras diferentes. Em todos os casos, a chave para criar um experimento de caos significativo é entender a aplicação, os incidentes históricos e as remediações implementadas, além de entender claramente as áreas a serem investigadas, como resiliência ou segurança. Seu conhecimento sobre o aplicativo ajuda você a formular uma hipótese sobre as possíveis fraquezas do aplicativo e a entender como ele detectará, remediará e se recuperará quando a falha for injetada.

O ciclo de vida do experimento de caos inclui estas etapas:

1. Defina o objetivo do experimento.
2. Selecione o aplicativo de destino.
3. Alinhe os mapas mentais.
4. Resolva os problemas conhecidos com seu aplicativo.
5. Defina a hipótese e o experimento.
6. Garanta a prontidão operacional para o experimento.
7. Execute cenários e experimentos controlados.
8. Aprenda e ajuste o experimento.

Essas etapas são ilustradas no diagrama e discutidas nas seções a seguir.



Defina objetivos e estabeleça expectativas

Antes de cada experimento, certifique-se de que seus objetivos e expectativas sejam específicos, mensuráveis, alcançáveis, relevantes e com limite de tempo. Defina claramente o seguinte:

- Identifique possíveis falhas ou fraquezas em sistemas e serviços para entender como elas podem afetar os usuários. Isso inclui identificar possíveis modos de falha, como falhas no servidor, falhas na rede ou bugs de software, e avaliar seu impacto potencial no desempenho geral e na confiabilidade do sistema.

- Quantifique o impacto das falhas definindo indicadores-chave de risco (KRIs) em seus sistemas e serviços. Isso inclui medir o efeito das falhas quando métricas como latência, taxa de transferência e taxas de erro se desviam de seu estado estável. Ao entender o impacto desses desvios, você pode priorizar os esforços para mitigar as falhas com base nos riscos comerciais.
- Desenvolva e verifique estratégias para mitigar ou prevenir falhas. Isso inclui identificar possíveis soluções, como redundância, correção de erros ou estratégias alternativas, e testar sua eficácia em um ambiente controlado. Ao verificar essas estratégias, você pode garantir que é eficaz na prevenção ou mitigação de falhas e pode implantá-las em seus sistemas de produção com confiança.
- Melhore os processos de resposta a incidentes e recuperação de desastres. Ao repetir falhas em um ambiente controlado, você pode testar os processos de resposta a incidentes, identificar possíveis gargalos ou lacunas e refinar os procedimentos de recuperação de desastres. Isso ajuda a garantir que você esteja preparado para responder com rapidez e eficácia no caso de falhas inesperadas.

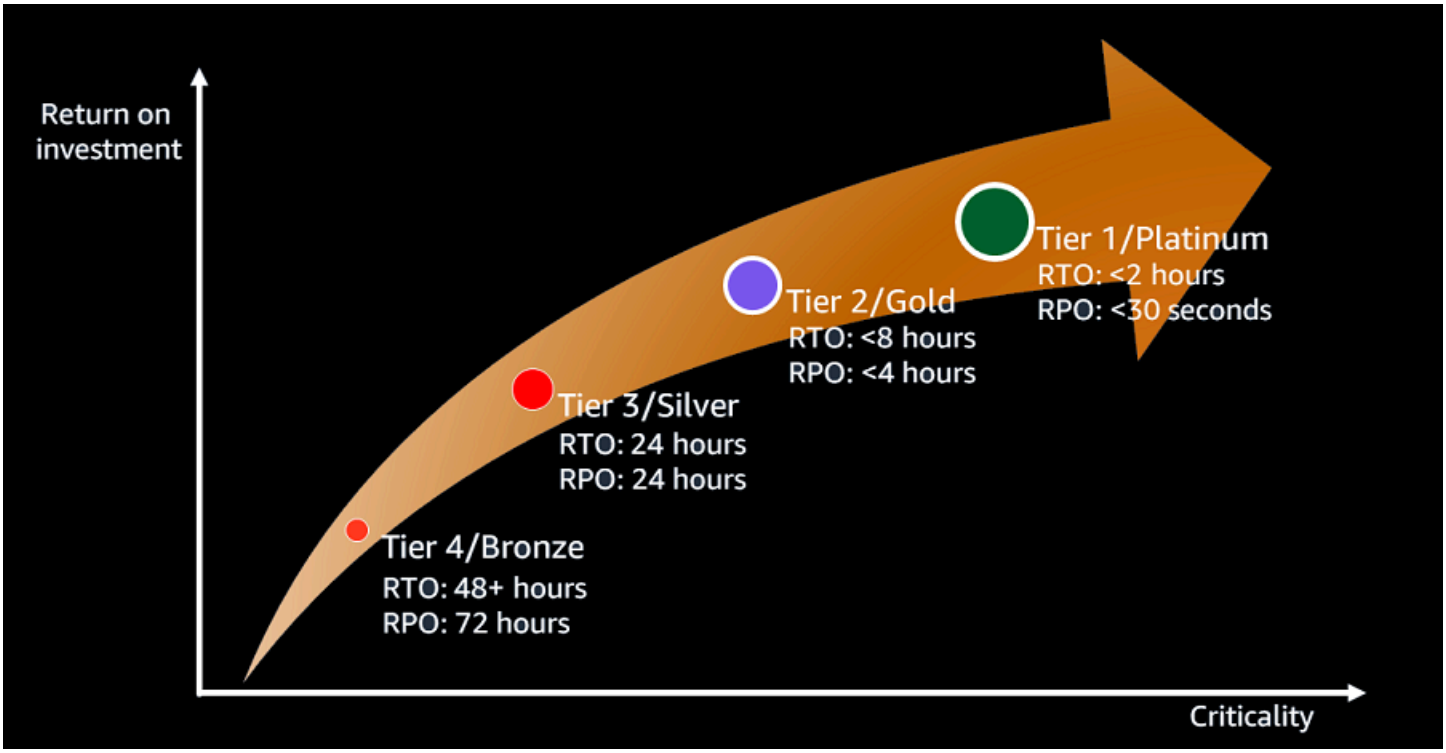
Selecione o aplicativo de destino

A engenharia do caos é uma técnica poderosa, mas requer uma priorização cuidadosa para maximizar o valor. Ao decidir onde concentrar os esforços de engenharia do caos, comece considerando os serviços essenciais de sua empresa. Peça às suas equipes que percorram os estágios do ciclo de vida de desenvolvimento de software e comecem a injetar falhas nos ambientes de teste primeiro. Business-critical os aplicativos estão diretamente vinculados à receita, à experiência do cliente e às principais operações. Experimentos de caos nesses serviços podem revelar vulnerabilidades que podem afetar gravemente a organização, e potencialmente mercados inteiros, se não forem resolvidas. Por exemplo, concentre-se primeiro nos serviços voltados para o cliente, como sistemas de negociação ou sistemas de pedidos. Priorizar esses serviços centrais fornece a maior proteção por investimento de tempo.

Depois de serviços essenciais, analise os componentes fundamentais, como bancos de dados, filas de mensagens, redes e APIs de serviços compartilhados. Eles podem ser usados como componentes ou serviços compartilhados em toda a organização, portanto, sua falha causará problemas generalizados. Confirmar a resiliência dos serviços de infraestrutura fornece a confiança de que eles não prejudicarão os aplicativos dependentes acima deles. Por exemplo, um experimento de engenharia do caos que derruba um cluster Kafka revela muito sobre a tolerância a falhas de aplicativos posteriores. Embora a infraestrutura do sistema não esteja diretamente voltada para o cliente, ela é o principal alvo da engenharia do caos.

Não se esqueça de mapear as lacunas mentais de pessoas, processos, informações sobre instalações e dependências de terceiros, pois elas podem causar grandes interrupções se não estiverem alinhadas com os objetivos de tolerância ao impacto da sua organização. Para obter mais informações sobre como medir o ROI da engenharia do caos, leia Quantificando o ROI da engenharia do caos no documento estratégico Investir na engenharia do caos como uma necessidade estratégica.

O diagrama a seguir mostra o retorno do investimento para realizar experimentos de caos em diferentes níveis de serviços.



Alinhe mapas mentais (descoberta de aplicativos)

Ao realizar experimentos ad-hoc ou dias de jogos, você iniciará o processo de descoberta do aplicativo realizando uma sessão de quadro branco que se concentra em mapear os detalhes do seu aplicativo. (Se você executar os experimentos no pipeline do caos, você já terá alinhado esse mapa mental, definindo o aplicativo de destino.) Uma boa abordagem para entender as lacunas mentais é fazer com que o membro mais jovem da equipe desenhe primeiro um diagrama do aplicativo e, em seguida, peça a mais funcionários seniores que o adicionem progressivamente. Isso revelará quaisquer lacunas na compreensão em todos os níveis de experiência.

O diagrama deve descrever as dependências diretas a montante e a jusante do aplicativo, bem como quaisquer integrações críticas de terceiros. Certifique-se de que haja alinhamento no fluxo esperado de uma solicitação por meio do aplicativo. Mapeie os principais fluxos de trabalho e jornadas do usuário para obter clareza sobre como os clientes usam o aplicativo. Considere usar um diagrama de sequência para capturar essas informações.

Após essa sessão colaborativa, a equipe deve ter um modelo mental compartilhado do aplicativo, suas dependências críticas e seus recursos de monitoramento, além de uma compreensão dos riscos de tomar uma decisão informada de prosseguir ou cancelar um possível experimento de caos.

Resolva os problemas conhecidos com seu aplicativo

Os experimentos de engenharia de caos são projetados para revelar defeitos de forma proativa em uma aplicação. Ao injetar falhas, como aumentos de latência, reinicializações de servidores ou deficiências de energia na zona de disponibilidade, você pode verificar a capacidade do seu aplicativo de tolerar interrupções realistas. No entanto, esse processo pressupõe um nível subjacente de estabilidade e integridade no aplicativo de destino. Executar experimentos de caos em um aplicativo já problemático corre o risco de mascarar problemas mais profundos.

Antes de empreender a engenharia do caos, as equipes devem resolver quaisquer defeitos, bugs e problemas de desempenho conhecidos em seus aplicativos.

Defina a hipótese e o experimento

Incidentes anteriores que causaram interrupções em seu aplicativo ou em outros aplicativos em sua organização podem servir como excelentes fontes para ideias de experimentos caóticos. Por exemplo, as interrupções anteriores foram provocadas por erros de configuração ou falta de padrões de resiliência? Analisar histórias de incidentes e repetir as causas dessas falhas no mundo real por meio de experimentos de caos é uma forma eficaz de desenvolver resiliência contra problemas semelhantes no futuro.

Outra fonte valiosa de conceitos experimentais pode vir diretamente dos engenheiros, arquitetos e operadores que estão mais familiarizados com um aplicativo. Permitir que os membros da equipe enviem cenários hipotéticos de falha que eles acreditam que poderiam interromper significativamente o aplicativo permite que você colete ideias com base no conhecimento interno. A equipe de aplicativos pode então avaliar quais desses cenários propostos podem ter o maior impacto potencial ou expor os maiores riscos desconhecidos. Visar experimentos de caos para cenários de alto risco e menos compreendidos pode gerar aprendizados importantes e evitar problemas no futuro.

Uma terceira fonte de ideias vem da modelagem de resiliência para antecipar as condições que levariam às perdas comerciais identificadas. Alguns exercícios de modelagem de resiliência têm uma abordagem baseada em componentes para construir um modelo de resiliência, enquanto outros têm uma abordagem baseada em sistemas. Uma abordagem baseada em componentes faz a pergunta: “O que acontece quando o componente x está sob carga extrema ou falha?” A equipe que desenvolve o modelo de resiliência então especula o efeito desse cenário na aplicação mais ampla e identifica os controles preventivos e de monitoramento atualmente em vigor para detectar e mitigar os efeitos do cenário. Como alternativa, uma abordagem baseada em sistemas segue um processo de cima para baixo para destacar um estado indesejável do aplicativo — como “A loja virtual está mostrando níveis de estoque obsoletos” — e convida a equipe de aplicativos a prever quais condições fariam com que o aplicativo se comportasse dessa maneira.

Garanta a prontidão operacional para o experimento

Você precisa de indicadores quantificáveis para medir o impacto de condições adversas no aplicativo e em seu comportamento, conforme descrito anteriormente na seção sobre observabilidade. A capacidade de medir o comportamento do aplicativo permite determinar se as condições adversas afetaram o aplicativo e em que magnitude.

A melhor maneira de entender se há um impacto em seu aplicativo é medir seu estado estável. O estado estável mede a aparência da operação normal e normalmente se alinha aos indicadores de negócios e de experiência do cliente de um determinado aplicativo. Antes de passar para a próxima etapa, certifique-se de ter a observabilidade pronta para entender o impacto e os mecanismos de reversão, caso o experimento não ocorra conforme o esperado.

Execute experimentos e cenários controlados

Em AWS, não recomendamos realizar um experimento inicial de caos em um aplicativo que está em produção. O objetivo de um experimento de caos é aprender algo novo sobre como o aplicativo se comporta em condições estressantes. O comportamento do aplicativo pode ser imprevisível durante o experimento, portanto, realizar um experimento pela primeira vez na produção pode ter consequências impactantes para o cliente. Portanto, você deve sempre realizar um experimento inicial de caos em um ambiente de nível inferior que tenha um potencial mínimo de afetar usuários do mundo real e, em seguida, percorrer seus ambientes depois de verificar e ter certeza de que seu aplicativo pode absorver, se adaptar e se recuperar das ações injetadas.

Planeje cada experimento minuciosamente usando um documento que capture os principais detalhes, semelhante ao documento de planejamento do experimento fornecido no apêndice. Alguns dos campos críticos a serem incluídos são a definição de estado estacionário, a hipótese e o método de injeção de falhas. O planejamento, a execução e a análise de um experimento de caos podem ser abordados em um único artefato.

Depois de finalizar o plano escrito para o experimento, prepare qualquer código necessário para injetar as interrupções planejadas descritas no documento.

Para capturar o impacto potencial durante o experimento, certifique-se de que os mecanismos de observabilidade estejam em vigor. Se você ainda não tem uma forma automatizada de capturar os resultados do experimento, como relatórios do AWS FIS experimento, identifique os membros da equipe que farão anotações durante a execução, capturarão capturas de tela dos painéis e conduzirão a equipe durante o experimento.

Aprenda e ajuste

Depois de cada experimento, reúna-se como uma equipe para analisar e refletir sobre o experimento do caos. Faça um esforço consciente para manter uma mentalidade irrepreensível. Seu objetivo deve ser ter um diálogo aberto e construtivo que se concentre inteiramente em obter o máximo de aprendizado, sem atribuir culpas.

Comece revisando a definição e a hipótese do estado estacionário para o experimento. O aplicativo se comportou conforme o esperado? Houve alguma surpresa que invalidou as suposições? Discuta as observações de como o aplicativo reagiu durante o experimento, tanto boas quanto ruins. Os dados coletados — métricas, registros, capturas de tela etc. — devem contar exatamente o que aconteceu.

Aborde essa análise de dados com curiosidade em vez de julgamento e identifique áreas em que melhorias podem ser feitas no design, na documentação, no monitoramento ou em outros recursos do aplicativo com base nos aprendizados. Esses itens de ação são capturados como projetos de acompanhamento para tornar o aplicativo mais resiliente.

Por meio dessa abordagem irrepreensível, você pode ter conversas francas sobre o que deu errado e como corrigi-lo. Assuma a intenção positiva de todos os envolvidos e confie que eles estavam trabalhando para obter bons resultados. Seu objetivo comum é o crescimento e a progressão organizacional por meio do aprendizado e da adaptação contínuos. As análises de experimentos do Chaos, conduzidas de forma construtiva e irrepreensível, fornecem um espaço seguro para sua

equipe obter informações valiosas que tornam seus aplicativos e sua organização mais confiáveis e resilientes a longo prazo. O foco permanece nos aprendizados, não nas pessoas. Para divulgar o aprendizado entre suas equipes, publique o relatório do resultado do experimento em um local central e divulgue as descobertas para que outras pessoas possam aprender com elas.

Dimensionando a engenharia do caos em toda a sua organização

À medida que sua organização adota a engenharia do caos, padronizá-la e implementá-la apresentará desafios. Nos estágios iniciais de maturidade, é provável que equipes diferentes usem ferramentas e variações diferentes do processo de engenharia do caos descrito nas seções anteriores. Ao mesmo tempo, algumas equipes podem não priorizar ou adotar a engenharia do caos, apesar de seus benefícios potenciais. As seções a seguir fornecem orientação sobre como superar esses desafios.

No geral, sua abordagem à engenharia do caos deve ser projetada para encontrar um equilíbrio entre liderança centralizada e participação descentralizada. Esse equilíbrio ajuda a garantir que a engenharia do caos seja integrada ao processo de desenvolvimento e que os aprendizados sejam compartilhados em toda a organização.

Estabelecendo uma prática de engenharia do caos

Padronizar a prática da engenharia do caos pode acelerar sua adoção. Compartilhar os aprendizados dos experimentos entre as equipes pode aumentar o retorno dos investimentos em engenharia do caos.

Crie um centro de excelência centralizado ou reúna um grupo de especialistas no assunto, como parte de sua prática de engenharia do caos. Como uma função pequena e centralizada, essa equipe pode atuar em equipes de desenvolvimento de software, infraestrutura, segurança e negócios e manter os padrões usados por essas equipes. Para simplificar, o centro de excelência é chamado de equipe de prática centralizada, e os grupos que aplicam a engenharia do caos são chamados de equipes praticantes no restante deste guia.

Papel da equipe de prática centralizada

A equipe de prática centralizada é responsável por desenvolver e implementar práticas de engenharia do caos em toda a organização. Eles trabalham em estreita colaboração com as equipes de prática para orientá-las na concepção e condução de experimentos e garantir que os experimentos sejam valiosos para os negócios. A equipe de prática centralizada também fornece orientação e suporte às equipes de desenvolvimento, infraestrutura e segurança para ajudá-las a integrar a engenharia do caos em seus processos de desenvolvimento.

As principais responsabilidades de uma equipe centralizada de prática de engenharia do caos incluem o seguinte:

- Capacitação — Uma função centralizada de engenharia do caos atua como facilitadora para introduzir a prática da engenharia do caos por meio de dias de jogos e workshops. Eles orientam as equipes no processo de engenharia do caos, incluindo a seleção de cenários de falha, a definição de hipóteses e a produção de relatórios para serem compartilhados com a organização em geral. A equipe de prática centralizada deve possuir materiais de treinamento e trabalhar para aprimorar as equipes praticantes no uso da engenharia do caos.
- Consultoria — A equipe de prática centralizada também pode atuar em uma função consultiva para supervisionar os experimentos conduzidos pelas equipes de prática. Sua experiência e conhecimento podem garantir que os experimentos agreguem valor aos negócios e sejam conduzidos de maneira segura. Da mesma forma, a equipe pode supervisionar a execução e o resumo de um experimento para orientar pessoas que são novas na engenharia do caos.
- Marketing e rastreamento de valor — Comunicar o valor comercial da engenharia do caos é fundamental para o sucesso desse programa. Cada equipe que participa de experimentos de engenharia do caos deve coletar dados dos experimentos em toda a empresa e demonstrar o valor do investimento da organização na engenharia do caos. Isso inclui quantificar e comemorar o número de incidentes que foram evitados durante cada experimento, o tempo de inatividade que teria ocorrido se o experimento tivesse falhado e o impacto geral nos negócios se os cenários de falha tivessem ocorrido na produção. Ao reunir e centralizar esses dados de todas as equipes e disponibilizá-los em toda a organização, a equipe de prática centralizada pode rastrear e influenciar o valor derivado da adoção da engenharia do caos em toda a organização.
- Padrões — A equipe de prática centralizada deve possuir e manter o processo de realização de experimentos caóticos, os modelos para planejar e relatar os experimentos e as ferramentas usadas para conduzir os experimentos.

  A equipe central deve possuir e gerenciar modelos de planejamento de experimentos, modelos de relatórios de experimentos, documentação de processos e materiais de capacitação. A documentação de melhores práticas e os materiais de capacitação fornecem orientação às equipes praticantes sobre tópicos como as grades de proteção que elas podem usar para limitar o impacto de um experimento, quando conduzir um experimento na produção e como desenvolver o uso da engenharia do caos ao longo do tempo. Para exemplos de modelos e saídas, consulte o apêndice.

A equipe de prática centralizada também deve ser proprietária do processo de realização de um experimento, incluindo comunicações e escalonamento, e quando e como se comunicar com outras equipes da organização antes ou durante um experimento. O processo também deve descrever quando são necessárias grades de proteção.

A equipe de prática centralizada também deve selecionar e possuir as principais ferramentas para conduzir experimentos de caos (por exemplo, ferramentas como AWS FIS). A seleção e implementação de ferramentas complementares, como ferramentas de geração de carga, devem ser deixadas para as equipes praticantes decidirem. As equipes praticantes devem ser capazes de adaptar o processo geral e as ferramentas para melhor atender às suas necessidades.

Papel das equipes praticantes

A equipe centralizada é responsável por conduzir a estratégia geral de engenharia do caos, enquanto as equipes praticantes participam do processo e são responsáveis pelo desenvolvimento e execução dos experimentos. Isso ajuda a garantir que os experimentos sejam relevantes para cada produto ou serviço específico e que os aprendizados sejam acionáveis e possam ser aplicados para melhorar a confiabilidade e a resiliência do produto. A equipe de prática centralizada atua como mentora e proprietária dos padrões e processos de engenharia do caos da organização. No entanto, para evitar que a equipe centralizada se torne um gargalo, as equipes praticantes individuais precisarão aprender com a prática central para realizar experiências de caos por si mesmas.

Estabelecendo uma comunidade de prática

Além de criar uma equipe centralizada, recomendamos que você estabeleça uma comunidade informal de profissionais interessados na engenharia do caos. Essa comunidade fornece uma plataforma para compartilhar conhecimento, melhores práticas e experiências entre as equipes de prática e a organização em geral.

A comunidade de prática pode ser operada pela equipe centralizada de prática de engenharia do caos, mas qualquer pessoa dentro da organização pode se tornar membro da comunidade. A equipe centralizada pode aproveitar a comunidade de prática para transmitir atualizações e obter aprendizados e coletar feedback de equipes praticantes que estão usando os padrões e os processos gerenciados pela equipe centralizada. A comunidade atuará como um ciclo de feedback para informar a equipe centralizada sobre a eficácia das práticas de engenharia do caos em todas

as equipes praticantes. A equipe de prática centralizada pode então ajustar sua documentação e artefatos de suporte para melhor apoiar as equipes de produto.

Incorporando a engenharia do caos à sua resiliência operacional

Um experimento de caos é um investimento da sua empresa para evitar incidentes na produção. Será necessário determinar onde a empresa pode obter o maior retorno sobre esse investimento. A organização pode trabalhar com a equipe centralizada de prática de engenharia do caos para atualizar seus padrões e determinar quais produtos são essenciais o suficiente para exigir a experimentação do caos.

Processo de desenvolvimento de sistemas

A engenharia do caos e os experimentos do caos devem ser realizados repetidamente como parte do ciclo de vida de um aplicativo. Da mesma forma que as equipes realizam regularmente testes de recuperação de desastres, elas devem conduzir experimentos de caos e dias de jogos de forma contínua e periódica ao longo do ano. Essa abordagem melhora a forma como uma organização antecipa, observa e responde aos incidentes.

Conclusão

A disciplina da engenharia do caos evoluiu muito na última década. Ele foi adotado em vários setores e ajudou as organizações a criar serviços resilientes e aumentar a satisfação do cliente. (Para exemplos de como as organizações implementaram essas práticas, consulte Chaos Engineering Stories.) A engenharia do caos está permitindo que as organizações reduzam os riscos de seus aplicativos essenciais injetando falhas controladas em todos os níveis de sua pilha de aplicativos, incluindo serviços de provedores de nuvem. Ter a capacidade de impactar toda uma pilha de aplicativos de forma controlada permite resiliência contínua, melhoria na excelência operacional, observabilidade e arquitetura orientada à recuperação sem o estresse das interrupções na produção. Essa abordagem também leva a melhores práticas de teste de resiliência em toda a organização. Para começar a adotar a engenharia do caos, organize um dia de jogo do caos ou um workshop para mostrar o valor que a engenharia do caos pode oferecer à sua organização.

Recursos

AWS FIS implementações do modelo de falha:

- Use AWS Fault Injection Service para demonstrar a resiliência de aplicativos multirregionais e multi-AZ (AWS postagem no blog)
- AWS O Fault Injection Simulator oferece suporte a experimentos de engenharia de caos nos Amazon EKS Pods (AWS postagem do blog)
- Anunciando novos recursos do AWS Fault Injection Simulator para cargas de trabalho do Amazon ECS (publicação no blog)AWS
- Melhore a resiliência do aplicativo com as métricas de volume do Amazon EBS e o AWS Fault Injection Simulator (AWS publicação no blog)
- Engenharia do caos aproveitando o AWS Fault Injection Simulator em um AWS ambiente com várias contas (AWS postagem no blog)
- Experimentos de caos no Amazon RDS usando o AWS Fault Injection Simulator (AWS postagem no blog)
- Criação de aplicativos resilientes sem servidor usando engenharia de caos (AWS postagem no blog)
- Use o FIS para interromper uma instância spot (workshop)AWS
- Automatizando a engenharia do caos em seus canais de entrega (AWS publicação na comunidade)

Outros recursos:

- Investir na engenharia do caos como uma necessidade estratégica
- Histórias de engenharia do caos
- CloudWatchDocumentação da Amazon
- Documentação do Amazon CloudWatch RUM
- Documentação do AWS X-Ray
- Documentação do AWS FIS

Apêndice: Documentos de amostra

Os exemplos de documentos fornecidos nesta seção usam um site fictício de adoção de animais de estimação (PetSite) como aplicativo alvo para um experimento de caos.

- Documento de planejamento de experimentos
- Documento do resultado do experimento

Documento de planejamento de experimentos

Estado estacionário

Nome do processo	Site de adoção de animais de estimação
Arquitetura física	(Link para o diagrama de arquitetura.)
Arquitetura lógica	(Link para o diagrama lógico.)
Definir estado estacionário	O tempo médio de carregamento da página, medido usando o Largest Contentful Paint (LCP), para o site de adoção de animais de estimação é de 2,5 segundos ou menos com uma latência de 99 percentis (P99) de 4,0 segundos ou menos com uma linha de base de 5.000 usuários simultâneos.
Métricas de estado estacionário	Métrica LCP capturada em toda a base de usuários e métricas douradas (latência, taxa de transferência, taxas de erro, saturação).
Observabilidade em estado estacionário	O LCP será capturado pelo navegador do usuário, enviado para a Amazon CloudWatch e inspecionado com CloudWatch RUM. Em um período de 60 segundos, a média e o tempo de LCP do P99 serão agregados para todas as solicitações nesse período. Top-level as

Nome do processo	Site de adoção de animais de estimação
	métricas douradas são capturadas usando CloudWatch.
Processo para atingir o estado estacionário	O Grafana K6 será usado para criar uma carga que simula os níveis normais de tráfego de produção de aproximadamente 5 mil usuários simultâneos.

Requisitos de observabilidade

As equipes devem ser capazes de ver o seguinte:

- Estado estacionário: o que será observado para verificar se a aplicação está em condições normais?
- Condição de falha: como a condição de falha aparecerá no painel? Por exemplo:
  - Alarmes que devem ser acionados
  - Registros que devem ser gerados
- Impacto da falha: o que deve ser observado para visualizar os componentes que devem ser afetados (escopo do impacto)?
- Recuperação: Como a recuperação será visualizada e medida para capturar o MTTR?
- Depuração: detalhes da solução de problemas em falhas de experimentos.

A tabela a seguir fornece sugestões e exemplos para um gráfico de requisitos de observabilidade. Você deve definir o que deve ser observado com base em seu experimento específico.

O que precisa ser observado	Link para a ferramenta de observabilidade	O que está sendo observado
Fonte de entrada	Painel de controle Grafana K6	• Contagem contínua de contêineres • Solicitações por segundo

O que precisa ser observado	Link para a ferramenta de observabilidade	O que está sendo observado
Integridade geral do aplicativo	• CloudWatch Painel de adoção de animais de estimação • Painel de experiência do usuário (RUM) para adoção de animais de estimação	• Contagem de nós saudáveis do Amazon EKS • Utilização da CPU do nó Amazon EKS
Saúde do fluxo de trabalho	CloudWatch Painel de adoção de animais de estimação	Tempo de LCP, métricas douradas
Rastreamentos	X-Ray Painel de adoção de animais de estimação	• Latência da solicitação • Request count • Contagem de falhas
Logs	CloudWatch Registros de adoção de animais	Quaisquer erros encontrados pelos pods serão enviados para o CloudWatch Logs.

Definição do experimento

Nome do experimento	Stress de CPU no Amazon EKS PetSite pod
Código-fonte do experimento	(Link para o repositório de origem do experimento.)
Descrição do experimento	Esse experimento explora como um aumento no uso da CPU do pod de PetSite aplicativ os afetaria a experiência geral do cliente. Ao injetar estresse da CPU em cada PetSite pod em execução, poderemos entender se

Nome do experimento	Stress de CPU no Amazon EKS PetSite pod
	há impacto nos clientes e a extensão desse impacto.
Requisitos ou parâmetros do experimento	Carga de aplicação: média de produção Seletor de etiquetas para pods: `app=petsite`
Duração do experimento	10 minutos
Environment	Ambiente de teste Alpha
Recursos alvo do experimento	PetSite cápsulas de aplicação
Linha de base do experimento que é introduzida por meio da ferramenta de geração de carga	• 54% das solicitações têm um LCP de <2,5 segundos. • 46% das solicitações têm um LCP de <4 segundos. • Nenhum erro é observado.
Condição de recuo	Nenhum

Hipótese

E se	Impacto	Recuperação
O que aconteceria com o estado estável se os pods de PetSite aplicativos experimentassem ou causassem mais de 60% de utilização da CPU por 10 minutos sob o tráfego normal em nível de produção?	Os tempos de LCP permanecerão abaixo de 2,5 segundos para P50 de usuários com P99 de 4,0 segundos ou menos. O consumidor deve ser capaz de carregar a página de PetSite destino.	Detecção: • O estresse da CPU será detectado pelos alarmes configurados em CloudWatch.

E se	Impacto	Recuperação
		• As métricas de LCP também gerarão alarmes para a degradação da experiência do usuário. Self-healing: • A natureza distribuída da arquitetura de microsser viços significa que muitas instâncias de pods estão sendo executadas em várias zonas de disponibi lidade. • O plano de controle do cluster EKS afastará o tráfego dos pods afetados e lançará novos pods nos nós de trabalho. Recuperação: Quando a utilização da CPU voltar ao normal, o LCP deverá se recuperar automatic amente.

Processo experimental

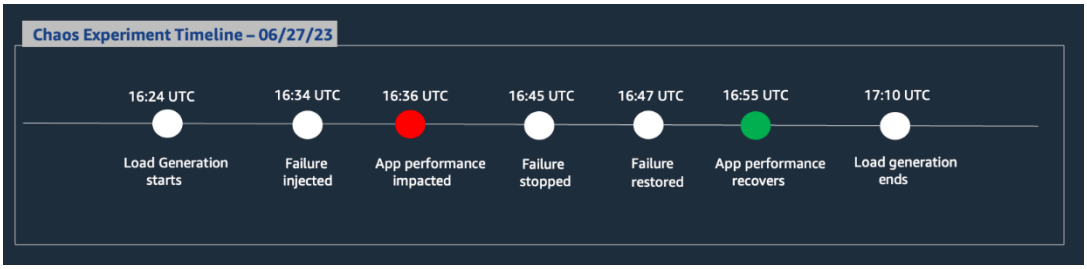
Adapte o seguinte exemplo de processo passo a passo ao seu experimento específico:

1. Valide o acesso e a funcionalidade de todos os AWS X-Ray painéis CloudWatch, Amazon e CloudWatch RUM.

2. Valide a integridade do ambiente de aplicativos:
    a. Confirme se o cluster EKS está íntegro usando o CloudWatch painel.
    b. Visite a implantação do aplicativo do site de teste de adoção de animais de estimação no URL de exemplo.
3. Inicie uma carga para atingir o estado estacionário:
    a. Confirme se o gerador de carga está funcionando e enviando 5000 solicitações por segundo.
    b. Aguarde 5 minutos para que o aplicativo atinja seu estado estável.
    c. Confirme o estado estável do aplicativo usando o painel do CloudWatch RUM.
4. Iniciar uma falha (experimento):
    a. Abra o AWS FIS console.
    b. Faça o experimento de adoção de animais de estimação e estresse.
    c. Confirme se o experimento está sendo executado no console.
5. Observe o impacto da falha em seu aplicativo:
    a. Capture capturas de tela do CloudWatch RUM e dos CloudWatch painéis e anote quaisquer pontos de dados anômalos.
    b. Depois que o experimento for concluído AWS FIS, capture capturas de tela adicionais para registrar se o aplicativo retorna ao estado estacionário na ausência de estresse e observe quaisquer anomalias nos pontos de dados.
    c. Se o estado estacionário não for retomado, tome medidas para recuperar o aplicativo e registre as etapas realizadas.
6. Valide se o ambiente voltou ao normal:
    - Analise todas as métricas de negócios, experiência do usuário, aplicativos e infraestrutura para verificar se o sistema voltou a um estado conhecido. Capture capturas de tela do painel, se for útil.

Cronograma do experimento

Certifique-se de capturar o cronograma do experimento de ponta a ponta, começando com a geração de carga, a injeção da falha, a observação do impacto e a recuperação do aplicativo, e terminando quando você interrompe a geração de carga. Isso é ilustrado no exemplo a seguir.



Resultados do experimento

ID de execução do experimento	Resultados do experimento
`PET-ADOPT-EXP-23`	(Link para os resultados do experimento.)

Defeitos identificados

- O cluster Kubernetes não detectou o comprometimento da CPU dos PetSite pods, por isso não agendou implantações adicionais.
- Não houve aumento nas taxas de erro 4XX ou 5XX como resultado desse experimento.
- Precisamos ajustar a verificação de integridade do pod para considerar o impacto no LCP quando há restrições de recursos.

Documento do resultado do experimento

Configuração

Documente as configurações específicas do experimento. Por exemplo:

- Conjunto de geração de carga para simular 5 mil usuários emitindo um total de 85 solicitações por segundo.

Pré-requisitos

- Verificou se o site de adoção de animais de estimação estava funcionando no ambiente de teste alfa.

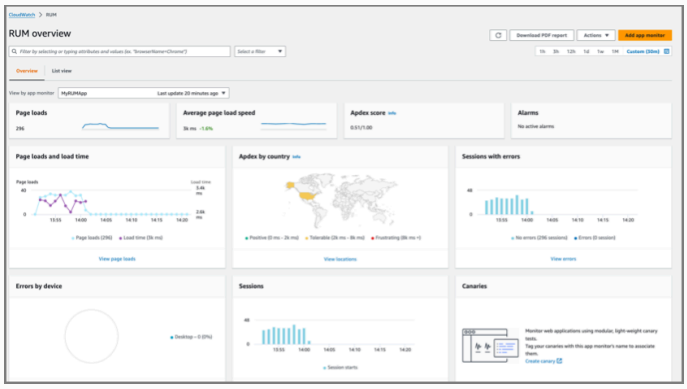
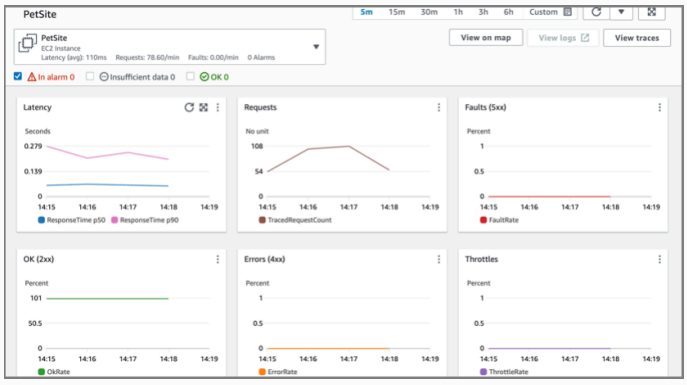
- Verifiquei se o modelo do experimento foi configurado para aplicar estresse de CPU aos pods de PetSite aplicativos que estão sendo executados no cluster EKS. Os pods de aplicativos foram identificados pelo rótulo Kubernetes. `app=petsite`
- Foi confirmado que a carga está sendo executada e gerando 85 solicitações por segundo.

Estado estacionário

Documente as etapas tomadas para alcançar o estado estável e como você o verificou. Por exemplo:

Para a implantação de teste do local de adoção de animais de estimação, uma carga de 85 RPS está sendo gerada para simular o estado estacionário. O CloudWatch RUM e os CloudWatch painéis foram revisados para verificar se todas as métricas de negócios e aplicativos estavam dentro dos intervalos normais antes da execução do experimento.

Dados de observabilidade:

Esperados	Observado
• O LCP é inferior a 4 segundos para P99 de solicitações. • A latência de resposta é inferior a 500 ms. • Não há erros 4XX ou 5XX.	 

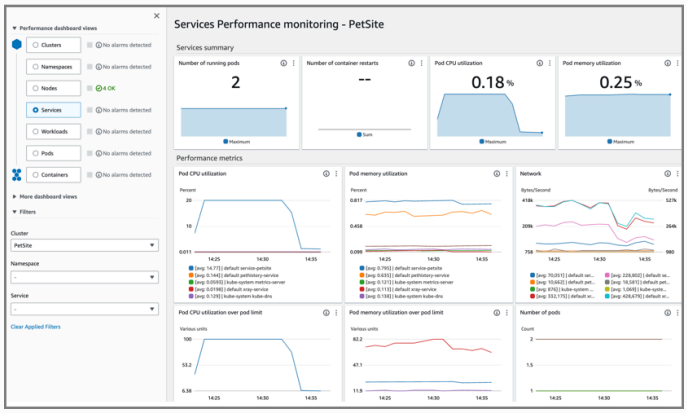
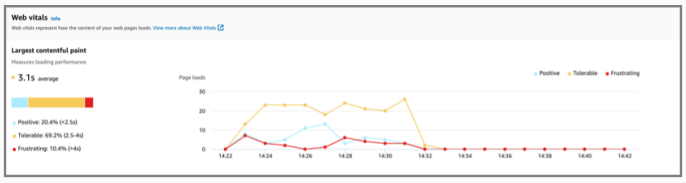
Injeção de falhas

AWS FIS foi usado para injetar falhas usando o modelo do experimento (forneça o link). O experimento foi configurado para ser executado por 10 minutos, e uma reversão foi configurada se os nós de trabalho sofressem estresse na CPU acima de 60%.

Observação de falhas

O CloudWatch RUM e os CloudWatch painéis foram revisados para rastrear o estado estável do aplicativo (definido usando métricas de LCP). As capturas de tela foram capturadas na tabela a seguir.

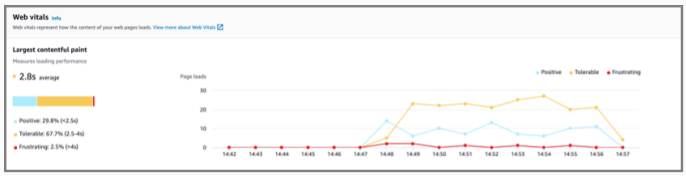
Dados de observabilidade:

Esperados	Observado
• O LCP deve permanecer abaixo de 4 segundos para o P99. • O tempo de resposta deve permanecer abaixo de 500 ms. • Nenhum erro 4XX ou 5XX deve ser encontrado.	 

Recuperação

Depois que o estresse for removido (o AWS FIS experimento foi concluído e o estresse da CPU foi removido dos pods), o aplicativo deve retomar seu estado estável normal. Nenhuma intervenção manual deve ser necessária.

Dados de observabilidade:

Esperados	Observado (captura de tela)
O LCP P99 deve estar abaixo de 4 segundos, com a média abaixo de 2,5 segundos.	

Histórico do documento

A tabela a seguir descreve alterações significativas feitas neste guia. Se desejar receber notificações sobre futuras atualizações, inscreva-se em um feed RSS.

Alteração	Descrição	Data
Publicação inicial	—	04 de abril de 2025

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.