Desenvolvimento e teste na Amazon Web Services

Novembro de 2012

Carlos Conde, Attila Narin

(Consulte <http://aws.amazon.com/whitepapers/>
para obter a versão mais recente deste documento)

Sumário

Resumo

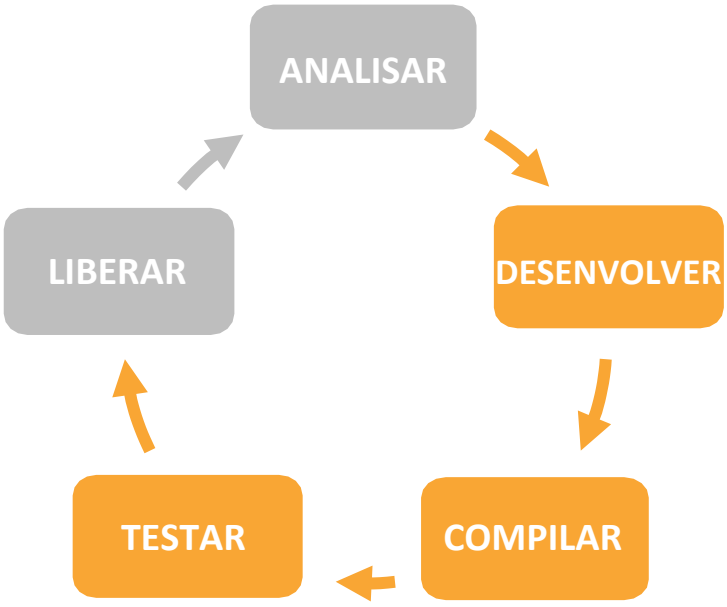
Este whitepaper descreve como a Amazon Web Services (AWS) agrega valor nas várias fases do ciclo de desenvolvimento de software, com foco específico em desenvolvimento e teste. Para a fase de desenvolvimento, ele mostra como usar a AWS para gerenciar o controle de versão, descreve as ferramentas de gerenciamento de projetos, o processo de compilação e ambientes hospedados na AWS e ilustra as melhores práticas. Para a fase de teste, ele descreve como gerenciar ambientes de teste e executar vários tipos de testes, incluindo testes de carga, testes de aceitação, testes de tolerância a falhas, etc. A AWS fornece vantagens exclusivas em cada um desses cenários e fases, permitindo que você escolha os mais apropriados para o seu projeto de desenvolvimento de software. O público-alvo pretendido para este documento é formado por gerentes de projetos, desenvolvedores, testadores, arquitetos de sistemas ou qualquer pessoa envolvida em atividades de produção de software.

Introdução

As organizações desenvolvem softwares por vários motivos, que variam desde as necessidades principais dos negócios — a organização é um fornecedor de software — até personalizar ou integrar o software. As organizações também criam diferentes tipos de software: aplicativos web, aplicativos independentes, agentes automatizados, etc. Em todos esses casos, as equipes de desenvolvimento são pressionadas a entregar um software de alta qualidade e o mais rápido possível, para reduzir o tempo de entrada no mercado ou tempo de produção.

Neste documento, “Desenvolvimento e teste” refere-se a várias ferramentas e práticas aplicadas na produção de software. Independentemente do tipo de software a ser desenvolvido, um conjunto adequado de práticas de desenvolvimento e teste é a chave para o sucesso. No entanto, produzir aplicativos não apenas requer engenheiros de software, mas também recursos de TI, que estão sujeitos a limitações como tempo, dinheiro e conhecimentos.

O ciclo de vida do software normalmente consiste nos seguintes elementos principais:



Este whitepaper abrange aspectos das fases de desenvolvimento, compilação e teste. Para cada uma dessas fases, diferentes tipos de infraestrutura de TI são necessários. É aqui que a AWS fornece vários benefícios para equipes de desenvolvimento de software. A AWS oferece acesso sob demanda para uma grande variedade de serviços de infraestrutura em nuvem, cobrando apenas pelos recursos que são usados. A AWS ajuda a eliminar a necessidade de um hardware dispendioso e os problemas administrativos que surgem ao possuí-lo e operá-lo. Possuir hardware e infraestrutura de TI geralmente envolve uma despesa de capital para um período de 3 a 5 anos em que a maioria das equipes de desenvolvimento e teste precisa de processamento ou de armazenamento por horas, dias, semanas ou meses. Essa diferença nos prazos pode causar atrito devido à dificuldade para as operações de TI atenderem solicitações simultâneas das equipes de projeto, até mesmo porque são limitados por um conjunto fixo de recursos. O resultado é que as equipes de projeto gastam muito tempo com justificativas, contratações e conservação de recursos. Esse tempo poderia ser gasto se concentrando no trabalho principal em mãos.

Ao provisionar apenas os recursos necessários para a duração das fases de desenvolvimento ou das execuções de testes, sua empresa pode obter economias importantes em comparação ao investimento inicial em hardware tradicional.[1] Com o nível adequado de granularidade, você pode alocar recursos de acordo com as necessidades e com o orçamento de cada projeto. Além desses benefícios econômicos, a AWS também oferece vantagens operacionais significativas, como a capacidade de configurar uma infraestrutura de desenvolvimento e teste em questão de minutos, em vez de semanas ou meses, e de expandir e reduzir a capacidade para fornecer os recursos de TI necessários, apenas quando forem necessários.

Este documento destaca algumas das melhores práticas e recomendações em torno de desenvolvimento e teste na AWS. Por exemplo, na fase de desenvolvimento, discutimos como configurar de forma segura e durável as ferramentas e os processos, como controle de versão, ambientes de colaboração e processos de compilação automatizados; na fase de teste, discutimos como configurar ambientes de teste no modo automatizado e como executar vários tipos de teste, incluindo testes lado a lado, testes de carga, testes de esforço, testes de resiliência e muito mais.

Fase de desenvolvimento

Independentemente do tamanho da equipe, do tipo de software em desenvolvimento ou da duração do projeto, as ferramentas de desenvolvimento são obrigatórias para racionalizar o processo, coordenar esforços e centralizar a produção. Como qualquer sistema de TI, as ferramentas de desenvolvimento exigem administração e manutenção adequadas. Operar essas ferramentas na AWS não apenas libera a equipe de desenvolvimento de tarefas de manutenção de baixo nível do sistema, como configuração de rede, configuração de hardware, etc., mas também facilita a execução de tarefas mais complexas. As seções a seguir descrevem como operar os principais componentes de ferramentas de desenvolvimento na AWS.

Repositório de código-fonte

O repositório de código-fonte é uma ferramenta chave para equipes de desenvolvimento. Como tal, ele precisa estar disponível, e os dados que contém (arquivos de origem no controle de versão) precisam ser armazenados de forma duradoura, com políticas de backup adequadas. Garantir essas duas características — disponibilidade e durabilidade — requer recursos, conhecimentos e investimento em tempo que normalmente não são a competência principal de uma equipe de desenvolvimento de software.

A criação de um repositório de código-fonte na AWS envolve criar uma instância do Amazon Elastic Compute Cloud[2] (Amazon EC2) e instalar remotamente nele o software de controle de versão. Em questão de minutos, os desenvolvedores podem criar instâncias do Amazon EC2, que são máquinas virtuais sobre as quais eles têm o

[1] Ou concluir campanhas de teste

[2] Consulte: http://aws.amazon.com/ec2/

controle completo. A variedade de diferentes sistemas operacionais e distribuições estão disponíveis como Amazon Machine Images (AMIs). Uma AMI é um modelo que contém uma configuração de software (sistema operacional, servidor de aplicativo e aplicativos) que você pode executar no Amazon EC2. Assim que você tiver instalado e configurado corretamente a instância do repositório de código-fonte, recomendamos que crie uma AMI dessa configuração para poder recriar rapidamente essa instância sem a necessidade de reinstalar e reconfigurar o software de controle de versão.

Você pode armazenar os dados do repositório separadamente do sistema do host para simplificar as operações de manutenção ou de migração. A Amazon Elastic Block Store[3] (Amazon EBS) fornece volumes de armazenamento fora da instância que perduram independentemente da vida de uma instância. Assim que criar um volume, você pode anexá-lo a uma instância do Amazon EC2 em execução. Dessa forma, um volume do Amazon EBS é provisionado e anexado à instância para armazenar os dados do repositório de controle de versão.

Você obtém durabilidade usando snapshots pontuais do volume do EBS que contém os dados do repositório. Os snapshots do EBS são armazenados no Amazon Simple Storage Service[4] (Amazon S3), um armazenamento de dados durável e escalável. Os objetos no Amazon S3 são armazenados de forma redundante em vários dispositivos em diversas instalações em uma região do Amazon S3. Em seguida, você pode usar esses snapshots como ponto inicial para novos volumes do Amazon EBS e pode proteger seus dados para durabilidade em longo prazo. Ao usar snapshots do EBS, você pode facilmente gerenciar backups do repositório de código-fonte. Em caso de falha, você pode recriar o volume de dados do repositório de snapshots armazenados de forma duradoura no Amazon S3 e recriar a instância do repositório de origem a partir de uma AMI.

Um endereço IP elástico fornece um endpoint estático para uma instância do Amazon EC2 e também pode ser usado em combinação com o DNS (por exemplo, atrás de um CNAME DNS). Isso ajuda as equipes acessarem seus serviços hospedados, como o repositório de código-fonte, de maneira consistente, mesmo se a infraestrutura for alterada abaixo, por exemplo, ao aumentar ou diminuir a escalabilidade, ou quando uma instância de substituição é encomendada.

À medida que o repositório de código-fonte cresce e requer mais capacidade de armazenamento, duas soluções estão disponíveis: provisionar e anexar volumes adicionais do Amazon EBS à instância do repositório ou provisionar um volume do Amazon EBS novo e maior (até 1 TB) com base em um snapshot recente dos dados do repositório existente. O novo volume então substituirá o volume existente, que você pode excluir. Em ambos os casos, os novos volumes do Amazon EBS são capacidade adicional que você cria sob demanda em questão de minutos.

Ferramentas de gerenciamento de projetos

Além do repositório de código-fonte, as equipes costumam usar ferramentas adicionais, como controle de problemas, rastreamento de projeto, código de análise de qualidade, colaboração, compartilhamento de conteúdo, etc. Na maior parte do tempo, essas ferramentas são fornecidas como aplicativos web. Como qualquer outro aplicativo web clássico, eles requerem um servidor para execução e também, frequentemente, um banco de dados relacional. Ambos os componentes podem ser instalados nas instâncias do Amazon EC2 seguindo procedimentos de implantação semelhantes aos usados em ambientes clássicos, com o banco de dados usando os volumes do Amazon EBS para armazenamento de dados.

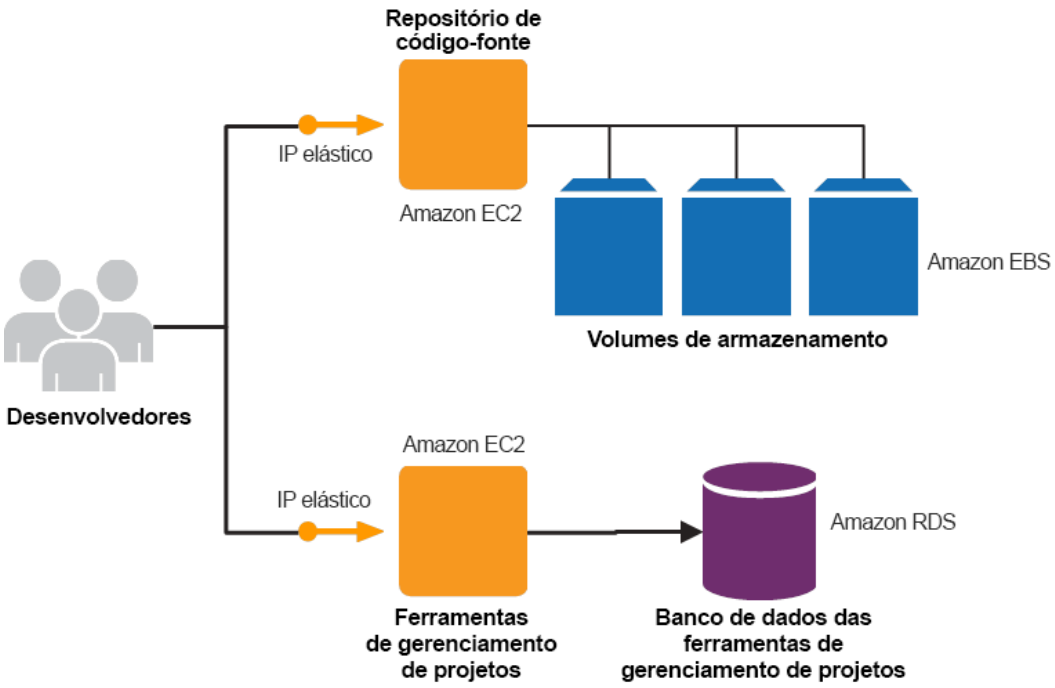
As ferramentas de gerenciamento de projetos têm as mesmas necessidades de repositórios de código-fonte: elas precisam estar disponíveis e os dados devem ser armazenados de forma duradoura. Embora você possa minimizar a perda de relatórios de análise de código ao recriá-los mediante a versão do repositório desejado, perder o projeto ou informações de rastreamento pode ter consequências mais graves. Você pode corrigir a disponibilidade do

[3] Consulte: http://aws.amazon.com/ebs/
[4] Consulte: http://aws.amazon.com/s3/

serviço de gerenciamento de projetos no aplicativo web usando AMIs para criar a substituição de instâncias do Amazon EC2 em caso de falha, da mesma forma que você faria para repositórios de código-fonte. Obter durabilidade adequada para um banco de dados exige mais esforço e mais atenção: mesmo usando volumes do Amazon EBS, os snapshots devem ser tirados em um sistema de arquivos congelado para ser consistente. Além disso, a restauração de um banco de dados pode exigir operações adicionais que não sejam a restauração de um volume a partir de um snapshot e a anexação a uma instância do Amazon EC2.

Para facilitar essa parte, o Amazon Relational Database Service (Amazon RDS) oferece uma maneira fácil de configurar, operar e escalar um banco de dados relacional na AWS. Ele fornece uma capacidade econômica e redimensionável enquanto gerencia tarefas demoradas de administração de banco de dados, liberando a equipe do projeto dessa responsabilidade. As instâncias de banco de dados do Amazon RDS (Instâncias de banco de dados) podem ser provisionadas em questão de minutos. Como opção, o Amazon RDS também garantirá que o software do banco de dados relacional permaneça atualizado com os patches mais recentes. O recurso de backup automatizado do Amazon RDS permite a recuperação point-in-time para instâncias de banco de dados, permitindo a restauração de uma instância de banco de dados em qualquer momento dentro do período de retenção de backup.

À medida que sua equipe de desenvolvimento cresce ou adiciona mais ferramentas à instância de gerenciamento de projetos, você pode precisar de capacidade extra para a instância do aplicativo web e a instância de banco de dados. Na AWS, escalar as instâncias verticalmente é uma operação fácil e simples. Basta criar um novo servidor do aplicativo web a partir da AMI em um tipo de instância do Amazon EC2 mais eficiente e substituir o servidor anterior. As instâncias de banco de dados do Amazon RDS podem escalar recursos de computação e de memória com alguns cliques no Console de Gerenciamento da AWS.



Observação: para uma implantação ainda mais rápida e fácil, muitas ferramentas de gerenciamento de projetos estão disponíveis a partir do AWS Marketplace[5] ou como Amazon Machine Images[6].

[5] Consulte: https://aws.amazon.com/marketplace/b/2649274011/
[6] Consulte: https://aws.amazon.com/amis

Ambientes de desenvolvimento sob demanda

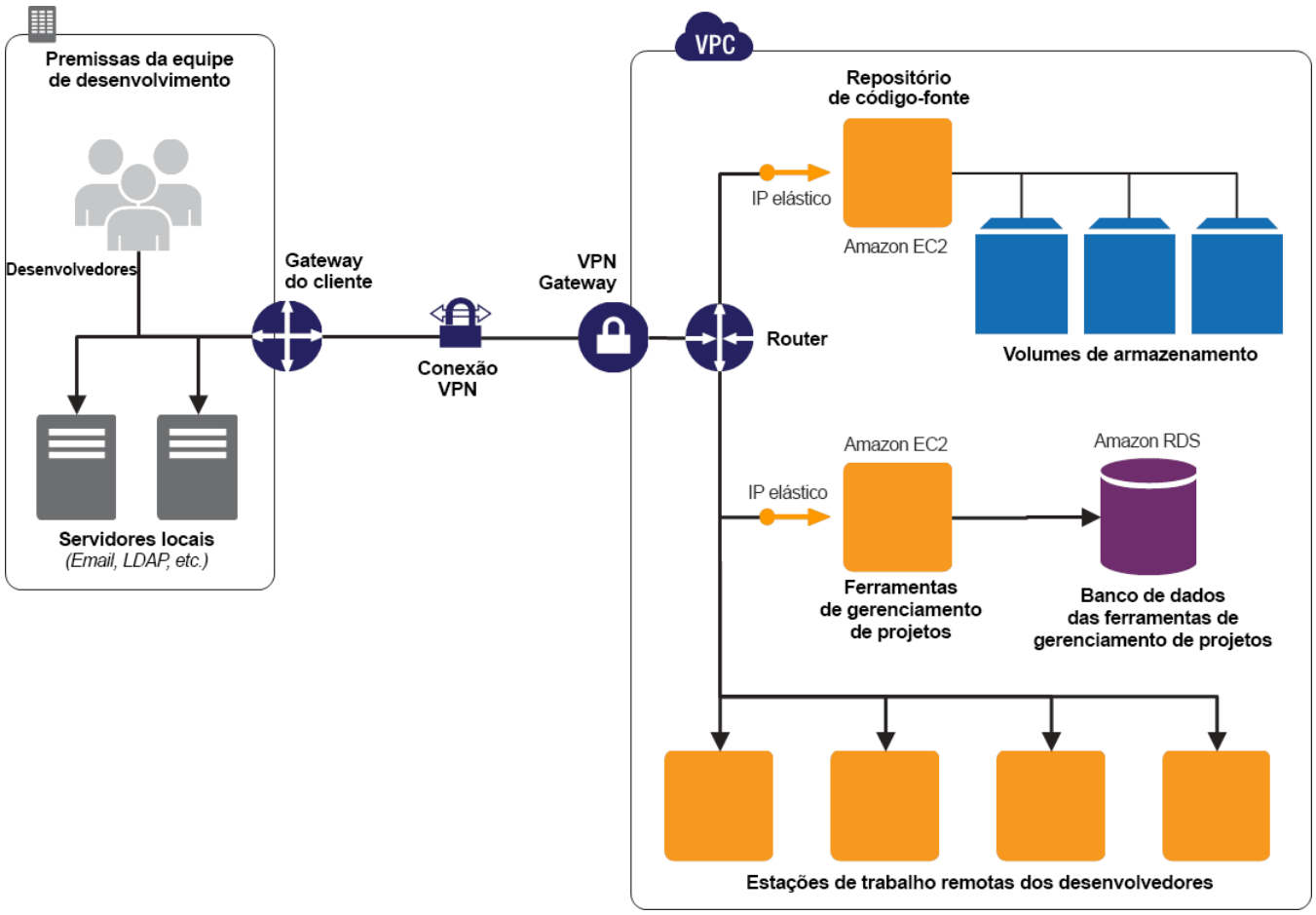
Os desenvolvedores usam, principalmente, seus notebooks ou desktops locais para executar seus ambientes de desenvolvimento. É geralmente onde o IDE está instalado, testes de unidade são executados, o código-fonte é verificado, etc. No entanto, também há alguns casos em que ambientes de desenvolvimento sob demanda hospedados na AWS são úteis.

Alguns projetos de desenvolvimento podem usar conjuntos de ferramentas especializadas onde seria complicado ou onde fossem necessários muitos recursos para instalá-los e mantê-los em máquinas locais, especialmente se essas ferramentas são usadas com pouca frequência. Em tais casos, você pode preparar e configurar ambientes de desenvolvimento com as ferramentas necessárias (ferramentas de desenvolvimento, controle de origem, pacotes de teste de unidade, IDEs, etc.) e, em seguida, agregá-los como AMI(s). Quando um ambiente é necessário, você pode facilmente iniciar o ambiente certo, instalá-lo e executá-lo rapidamente e com o mínimo de esforço. Em seguida, quando não precisar mais do ambiente, você pode encerrá-lo para liberar recursos. Isso também pode ser útil se você precisa alternar o contexto no meio da verificação de código e de um trabalho em andamento. Em vez de gerenciar ramificações ou lidar com verificações parciais, você pode criar um novo ambiente temporário.

Na AWS, você tem acesso a uma variedade de diferentes tipos de instância, alguns com configurações de hardware muito específicas. Se você desenvolve especificamente para uma determinada configuração, pode ser útil ter um ambiente de desenvolvimento na mesma plataforma em que o sistema também realizará a produção.

O conceito de desktops hospedados não é limitado a ambientes de desenvolvimento, mas pode ser aplicado a outros papéis ou outras funções. Para ambientes de trabalho mais complexos, o AWS CloudFormation facilita a configuração de coleções de recursos da AWS. Este tópico é discutido mais detalhadamente na seção Teste abaixo, no contexto de configuração de ambientes de teste. Em muitos casos, esses ambientes são configurados na Amazon Virtual Private Cloud (Amazon VPC), o que lhe permite estender sua rede local privada para a nuvem. Em seguida, você pode provisionar os ambientes de desenvolvimento como se estivessem na rede local, mas, em vez disso, eles são executados na AWS. Isso pode ser útil se esses ambientes exigem algum recurso local (como LDAP).

O diagrama a seguir mostra uma implantação em que ambientes de desenvolvimento são executados em instâncias do Amazon EC2 em uma Amazon VPC. Essas instâncias são acessadas remotamente a partir de uma rede corporativa por meio de uma conexão VPN segura.



Interrupção vs. encerramento de instâncias do Amazon EC2

Sempre que os ambientes de desenvolvimento não são usados, por exemplo, durante as horas em que você não está trabalhando, ou quando um determinado projeto está em espera, você pode facilmente fechá-los para economizar recursos e gastos. Há duas possibilidades: interromper as instâncias, que é quase equivalente à hibernação do sistema operacional; ou encerrá-las, que é quase equivalente a descartar o sistema operacional.

Quando você interromper uma instância (possível para as Amazon EBS−backed AMIs), os recursos de computação são liberados e não há mais cobranças por hora para a instância. O volume do Amazon EBS armazena o estado, e a próxima vez que você iniciar a instância, ela terá os dados de trabalho como antes de sua interrupção — isso é equivalente a reabrir a tampa de um notebook. (Observação: nenhum dado armazenado em drives temporários estará disponível depois de uma sequência de interrupção/início).

Quando você encerrar uma instância, o dispositivo raiz e quaisquer outros dispositivos conectados durante a execução da instância são automaticamente excluídos (a menos que o sinalizador DeleteOnTermination de um volume for definido como “false”), o que significa que os dados podem ser perdidos se não houver um backup ou snapshot disponível para os volumes excluídos. Uma instância encerrada não existe mais e precisa ser recriada a partir de uma AMI, se necessário. Em geral, você poderia encerrar a instância de um ambiente de desenvolvimento se todas as tarefas foram confirmadas e/ou o ambiente específico não for mais usado.

Integração com APIs da AWS e aprimoramentos IDE

Com a AWS, agora você pode codificar e controlar a infraestrutura de TI se a plataforma de destino do seu projeto for a AWS ou se o projeto for sobre a organização de recursos na AWS. Em tais casos, você pode usar os vários SDKs da AWS para integrar facilmente seus aplicativos com APIs da AWS, o que diminui a complexidade de codificação diretamente com base em uma interface de web service e lida com detalhes acerca de autenticação, novas tentativas, tratamento de erros, etc. As ferramentas do SDK da AWS estão disponíveis para várias linguagens: Java[7], .Net[8], PHP[9], Ruby[10] e para as plataformas móveis[11] Android e iOS.

A AWS também oferece ferramentas[12] como o AWS Toolkit for Visual Studio e o AWS Toolkit for Eclipse, facilitando sua interação com a AWS a partir de seus IDEs.

Fase de compilação

O processo de compilação de um aplicativo envolve várias etapas, incluindo a compilação, geração de recursos e empacotamento. Para aplicativos grandes, cada etapa envolve várias dependências, como a criação de bibliotecas internas o uso de aplicativos auxiliares, a geração de recursos em formatos diferentes, a geração de documentação, etc. Alguns projetos podem até mesmo exigir a criação de resultados para várias arquiteturas de CPU, plataformas ou sistemas operacionais. No final, o processo de compilação completo pode levar várias horas, o que afeta diretamente a agilidade da equipe de desenvolvimento de software. Esse impacto é ainda mais forte em equipes que adotam abordagens como a integração contínua[13] em que cada confirmação para o repositório de origem aciona uma compilação automatizada, seguida por pacotes de teste.

Compilações noturnas

Para atenuar esse problema, as equipes que trabalham em projetos com longos tempos de compilação muitas vezes adotam a “compilação noturna” (ou abordagem de compilação neutra[14]), ou quebram o projeto em subprojetos menores (ou até mesmo uma combinação de ambos). Realizar compilações noturnas envolve uma máquina de compilação que verifica o código-fonte mais recente do repositório e a compilação de resultados do projeto da noite para o dia. As equipes de desenvolvimento podem não compilar tantas versões quanto gostariam, e a compilação deve ser concluída em tempo para a realização de testes a iniciar no próximo dia. A divisão de um projeto em partes menores e mais gerenciáveis pode ser uma solução se cada subprojeto for compilado mais rápido de forma independente. No entanto, uma etapa de integração que combine todos os diferentes subprojetos muitas vezes ainda é necessária para a equipe acompanhar todo o projeto e para garantir que as diferentes partes continuam a funcionar bem em conjunto.

[7] Consulte: http://aws.amazon.com/sdkforjava/
[8] Consulte: http://aws.amazon.com/sdkfornet/
[9] Consulte: http://aws.amazon.com/sdkforphp/
[10] Consulte: http://aws.amazon.com/sdkforruby/
[11] Consulte: http://aws.amazon.com/mobile/
[12] Consulte: http://aws.amazon.com/developertools/
[13] Consulte: http://en.wikipedia.org/wiki/Continuous_integration
[14] Consulte: http://en.wikipedia.org/wiki/Nightly_build

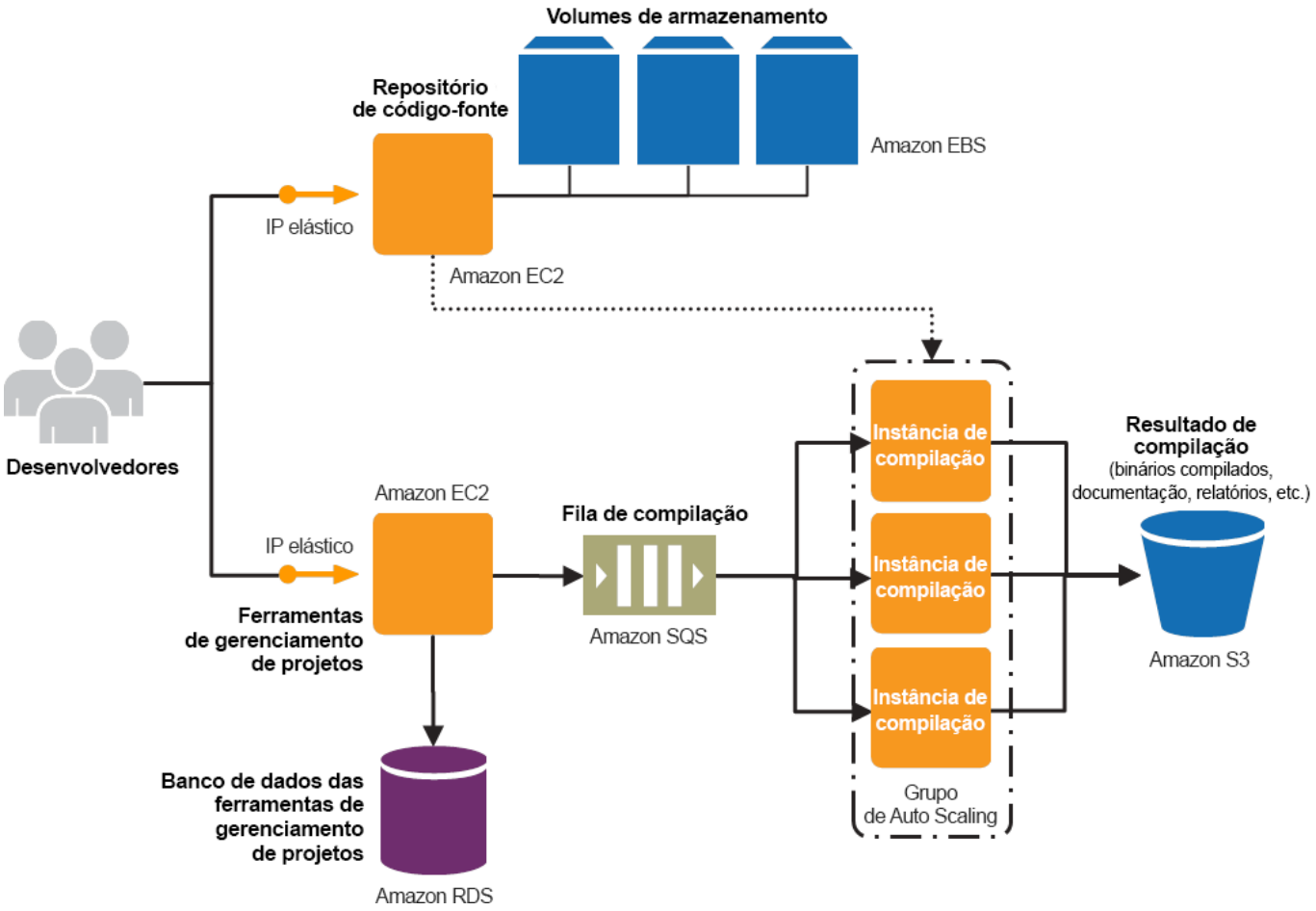
Compilações sob demanda

Uma solução mais prática é usar mais poder computacional para o processo de compilação. Em ambientes tradicionais em que o servidor de compilação é executado em hardware adquirido pela organização, essa opção pode não ser viável por limitações econômicas ou atrasos de provisionamento. Um servidor de compilação em execução em uma instância do Amazon EC2 pode ser expandido verticalmente em questão de minutos, como explicado nas seções anteriores, o que reduz o tempo de compilação, fornecendo mais capacidade de memória e de CPU quando for necessário.

Para equipes com várias compilações acionadas no mesmo dia, uma única instância do Amazon EC2 pode não ser capaz de produzir as compilações com rapidez suficiente. Uma solução seria aproveitar a natureza sob demanda e de pagamento conforme o uso do Amazon EC2 para usar várias instâncias de compilação (nós de operador). Cada vez que uma nova compilação é solicitada pela equipe de desenvolvimento ou acionada por uma nova confirmação para o repositório de código-fonte, o processo de compilação é distribuído para a frota de nós de operador. A distribuição de tarefas para nós de operador pode ser feita com uso de uma fila com todas as compilações a serem processadas. Os nós de operador escolhem a próxima compilação para processar sempre que estiverem livres. Para implementar esse sistema, o Amazon Simple Queue Service[15] (Amazon SQS) oferece um serviço de fila hospedada altamente escalável e confiável. O Amazon SQS facilita a criação de um fluxo de trabalho automatizado de compilação, trabalhando em estreita ligação com o Amazon EC2 e outros serviços de infraestrutura da AWS. Nesta configuração, os desenvolvedores podem confirmar o código para o repositório de código-fonte, que, por sua vez, envia uma mensagem de compilação para uma fila do Amazon SQS. Essa fila é consultada pelos nós de operador. Um nó de operador colocará uma mensagem e executará a compilação localmente de acordo com os parâmetros contidos na mensagem (por exemplo, a ramificação ou versão de origem para uso).

Você pode aprimorar ainda mais essa configuração ao ajustar dinamicamente o grupo de nós de operador que consome a fila. O Auto Scaling[16] é um serviço que facilita escalar o número ou nós de operador para cima ou para baixo automaticamente de acordo com condições predefinidas. Com o Auto Scaling, a capacidade dos nós de operador pode aumentar facilmente durante picos de demanda para manter a geração de compilação rápida, e diminuir automaticamente durante os períodos de baixa demanda para reduzir custos. Você pode definir as condições de escalabilidade com o Amazon CloudWatch[17], um serviço de monitoramento para recursos da nuvem AWS. Por exemplo, o Amazon CloudWatch pode monitorar o número de mensagens na fila de compilação e notificar o Auto Scaling de que é necessária mais ou menos capacidade, dependendo do número de mensagens na fila. O diagrama a seguir resume este cenário[18]:

[15] Consulte: http://aws.amazon.com/sqs/
[16] Consulte: http://aws.amazon.com/autoscaling/
[17] Consulte: http://aws.amazon.com/cloudwatch/
[18] Uma configuração mais genérica é descrita na arquitetura de referência “processamento em lote”. Consulte: http://d36cz9buwru1tt.cloudfront.net/architecturecenter/AWS_ac_ra_batch_03.pdf



Armazenamento e distribuição da saída da compilação

Cada vez que produzir uma compilação, você precisa armazenar a saída em algum lugar. O Amazon S3 é um serviço apropriado para isso. Inicialmente, a quantidade de dados a serem armazenados para um determinado projeto é pequena, mas ela cresce ao longo do tempo à medida que você produzir mais compilações. Aqui, o pagamento conforme o uso e as características de capacidade do Amazon S3 são especialmente atraentes. Quando não precisar mais da saída da compilação, você pode excluí-la ou usar as políticas de retenção de dados do Amazon S3 para fazê-lo.

Para distribuir a saída de compilação — por exemplo, para ser implantada no teste, na preparação ou na produção, ou para ser obtida por download para os clientes — a AWS oferece várias opções. Você pode distribuir pacotes de saída da compilação diretamente a partir do Amazon S3, tanto publicamente, como ao configurar políticas de bucket e/ou ACLs para restringir a distribuição. Outra opção é usar o Amazon CloudFront[19], um serviço da web para entrega de conteúdo que facilita a distribuição de pacotes para os usuários finais com baixa latência e alta velocidade de transferência de dados, melhorando, assim, a experiência do usuário final. Isso pode ser útil, por exemplo, quando um grande número de clientes está fazendo download de pacotes de instalação ou atualizações. O Amazon CloudFront oferece várias opções, por exemplo, para autorizar e/ou restringir o acesso, embora uma discussão completa esteja fora do escopo deste documento.

[19] Consulte: http://aws.amazon.com/cloudfront/

Fase de teste

Os testes são uma parte importante do desenvolvimento de software. Eles garantem a qualidade do software, mas, mais importante do que isso, eles ajudam a encontrar problemas no início da fase de desenvolvimento, reduzindo o custo de correções posteriores durante o projeto. Os testes vêm em muitas formas: testes de unidade, testes de performance, testes de aceitação do usuário, testes de integração, etc., e todos necessitam de recursos de TI para execução. Assim, as equipes de teste enfrentam os mesmos desafios que as equipes de desenvolvimento: a necessidade de ter um número suficiente de recursos de TI, mas apenas durante o período limitado em que o teste é executado. Além disso, ambientes de teste mudam com frequência e são diferentes de projeto a projeto; como consequência, podem exigir diferentes infraestruturas de TI, ou ter diferentes necessidades de capacidade.

As propostas de valor sob demanda e de pagamento conforme o uso da AWS são bem adaptadas para essas restrições. A AWS permite que suas equipes de teste eliminem a necessidade de um hardware dispendioso e os problemas administrativos que surgem ao possuí-lo e operá-lo. A AWS também oferece vantagens operacionais significativas para testadores: os ambientes de teste podem ser configurados em minutos, em vez de semanas ou meses, e uma variedade de recursos, incluindo diferentes tipos de instância, estão disponíveis para executar testes sempre que precisam ser executados.

Automatização de ambientes de teste

Há muitas ferramentas de software e estruturas disponíveis para automatizar o processo de execução de testes, mas, para executar esses testes, é necessária uma infraestrutura adequada. Isso envolve provisionar recursos de infraestrutura e inicializá-los com uma amostra de conjunto de dados, implantar o software para ser testado, organizar a execução do teste e coletar os resultados. O desafio aqui não é apenas ter recursos suficientes para implantar o aplicativo completo com todos os servidores ou serviços diferentes que ele possa exigir, mas também ser capaz de inicializar o ambiente de teste com o software certo e os dados corretos repetidamente. Os ambientes de teste devem ser idênticos entre as execuções de teste; caso contrário, é mais difícil comparar os resultados.

Outro benefício importante de executar testes na AWS é a capacidade de automatizá-los de várias formas. Você pode utilizar a AWS de forma programática usando as APIs da AWS ou as ferramentas da interface da linha de comando (CLI). As tarefas que exigem intervenção humana em ambientes clássicos (alocar um novo servidor, alocar e anexar o armazenamento, alocar um banco de dados, etc.) podem ser totalmente automatizadas na AWS. Para os testadores, projetar pacotes de testes na AWS significa poder automatizar um teste para a operação dos componentes, que tradicionalmente são dispositivos de hardware estático. A automação torna as equipes de teste mais eficientes, ao remover a tentativa de criar e inicializar ambientes de teste, e menos propensas a erros, ao limitar a intervenção humana durante a criação desses ambientes. Um ambiente de teste automatizado pode ser vinculado ao processo de compilação seguindo princípios de integração contínua.[20] Toda vez que uma compilação bem-sucedida é produzida, um ambiente de teste pode ser provisionado e testes automatizados executados nele.

As seções a seguir descrevem como provisionar automaticamente instâncias do Amazon EC2, bancos de dados e ambientes completos.

Instâncias de provisionamento

Você pode provisionar facilmente instâncias do Amazon EC2 a partir de AMIs[21]. Uma AMI encapsula o sistema operacional e qualquer outro software ou arquivos de configuração pré-instalados na instância. Quando você executar a instância, todos os aplicativos já estarão carregados na AMI e prontos para serem executados. Para obter

[20] Consulte: http://en.wikipedia.org/wiki/Continuous_integration
[21] Consulte: http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/launching-an-instance.html

informações sobre como criar AMIs, consulte a documentação do Amazon EC2[22]. O desafio com implantações baseadas em AMI é que cada vez que precisa atualizar o software, você deve criar uma nova AMI. Embora o processo de criação de uma nova AMI (e de exclusão de uma antiga) possa ser completamente automatizado, isso rapidamente impõe a necessidade de definir uma estratégia para gerenciar e manter várias versões de AMIs.

Uma abordagem alternativa é incluir apenas componentes na AMI que não são alterados com frequência (sistema operacional, plataforma de linguagem e bibliotecas de baixo nível, servidor de aplicativos, etc.). Componentes mais voláteis, como o aplicativo em desenvolvimento, são extraídos e implantados na instância no momento de execução. Para obter mais detalhes sobre como criar instâncias autoinicializadas, consulte o documento "Bootstrapping de aplicativos na AWS"[23].

Bancos de dados de provisionamento

Bancos de dados de teste podem ser implementados de maneira eficiente como instâncias de banco de dados do Amazon RDS. Suas equipes de teste podem instanciar um banco de dados totalmente operacional de maneira fácil e carregar um conjunto de dados de teste a partir de um snapshot. Para criar este conjunto de dados de teste, primeiro você deve provisionar uma instância do Amazon RDS. Depois de injetar o conjunto de dados, você deve criar um snapshot da instância[24]. Depois disso, sempre que precisar de um banco de dados de teste para um ambiente de teste, você pode criar facilmente um como uma instância do Amazon RDS a partir desse snapshot inicial[25]. Cada instância do Amazon RDS iniciada a partir do mesmo snapshot conterá o mesmo conjunto de dados, o que ajuda a garantir que seus testes serão consistentes.

Provisionamento de ambientes completos

Embora você possa criar ambientes de teste complexos que contenham várias instâncias usando as APIs da AWS, as ferramentas da linha de comando ou o Console de Gerenciamento da AWS, o AWS CloudFormation[26] facilita ainda mais para criar uma coleção de recursos relacionados da AWS e provisioná-los de forma organizada e previsível. O AWS CloudFormation usa modelos para criar e excluir um conjunto de recursos juntamente como uma única unidade (uma pilha). Um ambiente de teste completo em execução na AWS pode ser descrito em um modelo, que é um arquivo de texto em formato JSON. Como os modelos são apenas arquivos de texto, você pode editá-los e gerenciá-los no mesmo repositório de código-fonte que você usa para o seu projeto de desenvolvimento de software. Dessa forma, o modelo espelha o status do projeto, e ambientes de teste que correspondam a versões de origem mais antigas podem ser facilmente provisionados. Isso é especialmente útil ao lidar com erros de regressão: em apenas algumas etapas, você pode provisionar todo o ambiente de teste, que permite aos desenvolvedores e aos testadores simular um erro detectado em versões mais antigas do software. Os modelos do AWS CloudFormation também são compatíveis com parâmetros que podem ser usados para especificar uma determinada versão do software a ser carregado, os tamanhos de instância do Amazon EC2 para o ambiente de teste, o conjunto de dados a ser usado para os bancos de dados, etc.

Para obter mais informações sobre como criar e automatizar implantações na AWS usando o AWS CloudFormation, acesse: http://aws.amazon.com/cloudformation/aws-cloudformation-articles-and-tutorials/

[22] Consulte: http://docs.amazonwebservices.com/AWSEC2/latest/UserGuide/creating-an-ami.html
[23] Consulte: https://s3.amazonaws.com/cloudformation-examples/BoostrappingApplicationsWithAWSCloudFormation.pdf
[24] Consulte: http://docs.amazonwebservices.com/AmazonRDS/latest/UserGuide/USER_CreateSnapshot.html
[25] Consulte: http://docs.amazonwebservices.com/AmazonRDS/latest/UserGuide/USER_RestoreFromSnapshot.html
[26] Consulte: http://aws.amazon.com/cloudformation/

Testes de carga

Os testes de funcionalidade executados em ambientes controlados são ferramentas importantes para garantir a qualidade do software, mas eles fornecem poucas informações sobre como um aplicativo ou uma implantação completa atuará com uma carga pesada. Por exemplo, alguns sites são criados especificamente para fornecer um serviço por tempo limitado: venda de ingressos para eventos esportivos, vendas especiais (Black Friday), lançamentos de edição limitada, etc. Esses sites precisam ser desenvolvidos e projetados para executar com eficiência durante os períodos de maior utilização.

Em alguns casos, os requisitos do projeto definem claramente a métrica de performance mínima a ser atendida em condições de carga pesada (exemplo: resultados de pesquisa devem ser exibidos em menos de 100 ms para até 10.000 solicitações simultâneas) e testes de carga são exercidos para garantir que o sistema é capaz de sustentar a carga dentro desses limites. Em outros casos, não é possível ou prático especificar a carga que um sistema deve sustentar. Nesses casos, os testes de carga são realizados para medir o comportamento em condições de carga pesada. O objetivo é aumentar gradualmente a carga de um sistema para determinar o ponto em que a performance diminui de tal forma que o sistema não possa operar mais.

Os testes de carga simulam entradas pesadas que exercitam e pressionam um sistema. Dependendo do projeto, as entradas podem ser uma grande quantidade de solicitações que chegam simultaneamente, um grande conjunto de dados para processar, etc. Uma das principais dificuldades em testes de carga é ser capaz de gerar quantidades grandes o suficiente de entradas para levar o sistema testado até os seus limites. Normalmente, isso implica na necessidade de grandes quantidades de recursos de TI para implantar o sistema para testar e para gerar a entrada de teste, que requer mais infraestrutura. Como os testes de carga geralmente não são executados por mais de duas horas, o modelo de pagamento conforme o uso da AWS funciona muito bem para esse caso de uso.

Você também pode automatizar testes de carga usando as técnicas descritas na seção anterior, permitindo que os testadores os exerçam com mais frequência, a fim de garantir que cada alteração importante ao projeto não afete negativamente a performance do sistema e sua eficiência. Por outro lado, ao executar testes de carga automatizados, é fácil verificar se um novo algoritmo, camada de cache ou projeto da arquitetura é mais eficiente e beneficia o projeto.

> *Observação: para uma configuração rápida e fácil, as soluções e as ferramentas de teste também estão disponíveis a partir do AWS Marketplace: https://aws.amazon.com/marketplace/b/2649283011/*

Testes de carga de rede

Testar um aplicativo ou serviço quanto à carga de rede envolve enviar um grande número de solicitações para o sistema em teste. Há muitas soluções de software disponíveis para simular cenários de solicitação, mas usar várias instâncias do Amazon EC2 pode ser necessário para gerar tráfego suficiente. As instâncias do Amazon EC2 estão disponíveis sob demanda e são cobradas por hora, o que as torna ideais para cenários de testes de carga de rede. É importante ter em mente as características de diferentes tipos de instância: como uma regra geral, tipos de instância maiores fornecem mais capacidade de E/S de rede, o recurso principal consumido durante os testes de carga de rede.

Com a AWS, as equipes de teste também podem realizar testes de carga de rede em aplicativos que são executados fora da AWS. A disponibilidade de agentes de teste de carga espalhados por diferentes regiões da AWS permite que os testes de diferentes regiões geográficas, por exemplo, obtenham uma melhor compreensão da experiência do usuário final. Nesse cenário, faz sentido coletar informações de log das instâncias que simulam a carga. Esses logs contêm informações importantes, como os tempos de resposta do sistema testado. Ao executar os agentes de carga de diferentes regiões, o tempo de resposta do aplicativo testado pode ser medido para diferentes regiões geográficas. Isso pode ajudar você a compreender a experiência do usuário em todo o mundo. Como é possível encerrar instâncias do Amazon EC2 de teste de carga logo após o teste, você deve transferir os dados de log para o Amazon S3 para armazenamento e análise posterior.

Testes de carga para AWS

Realizar testes de carga em um aplicativo executado na AWS é útil para garantir que os recursos de elasticidade sejam implementados corretamente. Testar um sistema quanto à carga de rede é fundamental para garantir que as configurações para os front-ends da web, o Auto Scaling e o Elastic Load Balancing[27] estejam corretas. O Auto Scaling oferece muitos parâmetros[28] e pode usar várias condições definidas com o Amazon CloudWatch para escalar o número de instâncias de front-end para cima ou para baixo. Esses parâmetros e essas condições influenciam a velocidade com que um grupo de Auto Scaling adicionará ou removerá instâncias. Um tempo pós-provisionamento da instância do Amazon EC2 também pode afetar a capacidade de um aplicativo de expandir com rapidez suficiente. Após a inicialização do sistema operacional em execução nas instâncias do Amazon EC2, serviços adicionais, como servidores da web, servidores de aplicativos, caches de memória, serviços de middleware, etc., são inicializados. O tempo de inicialização desses diferentes serviços afeta o atraso de expansão, especialmente quando pacotes de software adicionais precisam ser baixados de um repositório. Os testes de carga fornecerão métricas valiosas sobre a rapidez com que a capacidade adicional pode ser adicionada em um determinado sistema.

O Auto Scaling não é usado apenas para sistemas de front-end. Você também pode usá-lo para escalar grupos de instâncias internas, como os consumidores que sondam uma fila do Amazon SQS[29] ou operadores e administradores que participam de um fluxo de trabalho do Amazon Simple Workflow Service (Amazon SWF)[30]. Em ambos os casos, realizar um teste de carga no sistema pode ajudar a garantir que você tenha implementado e configurado corretamente grupos de Auto Scaling ou outras técnicas de escalabilidade automatizadas, para tornar seu aplicativo final o mais econômico e escalável possível.

Otimização de custo com instâncias spot

A realização de testes de carga pode exigir muitas instâncias, especialmente quando empregar sistemas que são projetados para oferecer suporte a uma grande quantidade de carga. Embora você possa provisionar instâncias do Amazon EC2 sob demanda e descartá-las quando o teste for concluído enquanto paga apenas por hora, há uma maneira ainda mais econômica para executar esses testes usando instâncias spot do Amazon EC2. As instâncias spot[31] permitem aos clientes negociar a capacidade não usada do Amazon EC2. As instâncias são cobradas pelo preço spot, que é definido pelo Amazon EC2 e oscila em função da oferta e da demanda por capacidade das instâncias spot. Para usar instâncias spot, faça uma solicitação de instância spot, especificando o tipo de instância, a zona de disponibilidade desejada[32], o número de instâncias spot a serem executadas e o preço máximo a pagar por hora de instância. O histórico de preços spot dos últimos 90 dias está disponível por meio da API do Amazon EC2 e do Console de Gerenciamento da AWS[33]. Se o preço máximo de lance exceder o preço spot atual, a solicitação será cumprida e as instâncias serão iniciadas e executadas até que sejam encerradas, ou até que o preço spot exceda o preço máximo (o que ocorrer primeiro).

Para obter mais informações sobre instâncias spot, acesse: http://aws.amazon.com/ec2/spot-instances/. Um estudo de caso do uso de instâncias spot para testes de carga está disponível aqui: http://aws.amazon.com/solutions/case-studies/browsermob/.

[27] Leia o artigo técnico “Melhores práticas na avaliação do Elastic Load Balancing” disponível em http://aws.amazon.com/articles/1636185810492479
[28] Consulte a documentação do Auto Scaling: http://docs.amazonwebservices.com/AutoScaling/latest/DeveloperGuide/Welcome.html
[29] Consulte o artigo do blog: http://aws.typepad.com/aws/2011/07/additional-cloudwatch-metrics-for-amazon-sqs-and-amazon-sns.html
[30] Consulte o artigo do blog: http://aws.typepad.com/aws/amazon-simple-workflow-service/
[31] Consulte: http://aws.amazon.com/ec2/spot-instances/
[32] Uma zona de disponibilidade é um local distinto em uma região que é projetada para ser isolada das falhas de outras zonas de disponibilidade e fornece conectividade de rede de baixa latência e baixo custo para outras zonas de disponibilidade na mesma região.
[33] O preço spot atual está disponível aqui: http://aws.amazon.com/ec2/spot-instances/#6

Testes de aceitação do usuário

O objetivo dos testes de aceitação do usuário é apresentar a versão atual para uma equipe de teste que representa a base do usuário final para determinar se os requisitos e a especificação do projeto são atendidos. Quando os usuários podem testar o software antes, eles conseguem identificar vulnerabilidades conceituais que foram introduzidas durante a fase de análise ou esclarecer zonas cinzentas nos requisitos do projeto. Ao testar o software com mais frequência, os usuários podem identificar erros de implementação funcional e concepções erradas no fluxo do aplicativo ou na interface do usuário, reduzindo o custo e o impacto de corrigi-los. Falhas detectadas pelos testes de aceitação do usuário podem ser muito difíceis de detectar por outros meios. Quanto mais você conduzir testes de aceitação, melhor para o projeto, uma vez que os usuários finais fornecem feedback valioso para equipes de desenvolvimento à medida que os requisitos mudam.

No entanto, como qualquer outra prática de teste, testes de aceitação exigem recursos para executar o ambiente no qual o aplicativo a ser testado será implantado. Conforme descrito nas seções anteriores, a AWS fornece capacidade sob demanda conforme necessário de uma maneira econômica, que também é muito apropriada para testes de aceitação. Usando algumas das técnicas descritas acima, a AWS permite a automação completa do processo de provisionamento de novos ambientes de teste e a eliminação de ambientes não mais necessários. Os ambientes de teste podem ser fornecidos apenas para determinados momentos, continuamente a partir da versão mais recente do código-fonte ou para cada versão principal.

Ao implantar o ambiente de teste de aceitação no Amazon VPC, os usuários internos podem acessar de forma transparente o aplicativo a ser testado. Além disso, esse aplicativo também pode ser integrado a outros serviços de produção dentro da empresa, como o LDAP, servidores de e-mail, etc., oferecendo um ambiente de teste para os usuários finais que é ainda mais próximo do ambiente de produção real e final.

Testes lado a lado

Os testes lado a lado são um método usado para comparar um sistema de controle a um sistema de teste. O objetivo é avaliar se as alterações aplicadas ao sistema de teste melhoram uma métrica desejada em comparação ao sistema de controle. Você pode usar essa técnica para otimizar a performance de sistemas complexos em que vários parâmetros diferentes podem afetar a eficiência global. Saber qual parâmetro terá o efeito desejado não é sempre óbvio, especialmente quando vários componentes são usados em conjunto e influenciam a performance um do outro. Você também pode usar essa técnica quando introduzir alterações importantes em um projeto, como novos algoritmos, caches, diferentes mecanismos de banco de dados ou software de terceiros. Em tais casos, o objetivo é garantir que suas alterações tenham um impacto positivo sobre a performance global do sistema.

Assim que você tiver implantado os sistemas de teste e controle, envie a mesma entrada para ambos, usando técnicas de testes de carga ou entradas de teste simples. Por fim, reúna as métricas de performance e os logs de ambos os sistemas e compare-os para determinar se as alterações introduzidas no sistema de teste apresentam uma melhoria no sistema de controle.

Ao provisionar ambientes de teste completos sob demanda, você pode executar testes lado a lado com eficiência. Embora você possa fazer testes lado a lado sem provisionamento de ambiente automatizado, ao usar as técnicas de automação descritas acima, fica mais fácil executar esses testes sempre que forem necessários, aproveitando o modelo de pagamento conforme o uso da AWS. Por outro lado, com um hardware tradicional, pode não ser tão fácil executar vários ambientes de teste para diversos projetos simultaneamente.

Os testes lado a lado também são importantes de um ponto de vista de otimização de custos: ao comparar dois ambientes em diferentes contas da AWS, você pode facilmente encontrar taxas de custo/performance para comparar os dois ambientes. Ao testar continuamente as alterações na arquitetura para performance de custo, você pode otimizar suas arquiteturas para obter eficiência.

Testes de tolerância a falhas

Quando a AWS é o ambiente de produção de destino para o aplicativo que você desenvolveu, algumas práticas de teste específicas fornecem informações sobre como o sistema irá processar casos extremos, como falhas de componentes. A AWS oferece várias opções para a compilação de sistemas tolerantes a falhas. Alguns serviços são inerentemente tolerantes a falhas, por exemplo, o Amazon S3, o Amazon DynamoDB, o Amazon SimpleDB, o Amazon SQS, o Amazon Route 53, o Amazon CloudFront, etc. Outros serviços, como o Amazon EC2, o Amazon EBS e o Amazon RDS, fornecem recursos que ajudam a arquitetura altamente disponível e tolerante a falhas. Por exemplo, o Amazon RDS oferece a opção[34] multizonas de disponibilidade que aprimora a disponibilidade do banco de dados ao provisionar e gerenciar automaticamente uma réplica em uma zona de disponibilidade diferente. Para obter mais informações sobre como criar arquiteturas tolerantes a falhas em execução na AWS, leia o whitepaper "Criação de aplicativos tolerantes a falhas na AWS"[35] e os recursos disponíveis no Centro de arquitetura da AWS[36].

Muitos clientes da AWS executam aplicativos de missão crítica na AWS e precisam garantir que sua arquitetura é tolerante a falhas. Como resultado, uma prática importante para todos os sistemas é testar sua capacidade de tolerância a falhas. Enquanto um cenário de teste exercita o sistema (com técnicas semelhantes aos testes de carga), alguns componentes são retirados de propósito para verificar se o sistema é capaz de se recuperar a partir dessa falha simulada. Você pode usar o Console de Gerenciamento da AWS ou a interface de linha de comando para interagir com o ambiente de teste. Por exemplo, você pode encerrar instâncias do Amazon EC2, e, assim, testar se um grupo de Auto Scaling está funcionando conforme o esperado e uma instância de substituição provisionada automaticamente. Você também pode automatizar esse tipo de teste. São melhores práticas usar ferramentas automatizadas que, por exemplo, encerram de forma ocasional e aleatória instâncias do Amazon EC2.

Gerenciamento de recursos

Com a AWS, suas equipes de desenvolvimento e teste podem ter seus próprios recursos dimensionados de acordo com suas necessidades. Provisionar ambientes complexos ou plataformas compostas por várias instâncias pode ser feito facilmente com as pilhas do AWS CloudFormation ou algumas das outras técnicas de automação descritas. Em grandes organizações com várias equipes, é uma boa prática criar uma função interna ou serviço responsável por centralizar e gerenciar recursos de TI em execução na AWS. Essa função normalmente consiste em:

- Promover práticas de desenvolvimento e teste interno descritas aqui
- Desenvolver e manter AMIs de modelo e pilhas do AWS CloudFormation de modelo com as diferentes ferramentas e plataformas usadas na sua organização
- Coletar solicitações de recursos das equipes de projetos e provisionar recursos na AWS de acordo com as políticas da sua organização, incluindo a configuração de rede (por exemplo, Amazon VPC), configurações de segurança (por exemplo, grupos de segurança e credenciais do IAM)
- Monitorar o uso de recursos e cobranças com o Amazon CloudWatch e alocar esses orçamentos para a equipe

Embora você possa usar o Console de Gerenciamento da AWS para realizar as tarefas acima, você pode desenvolver seu próprio portal de gerenciamento e provisionamento interno para uma maior integração com os processos internos. Você pode fazer isso com um dos SDKs da AWS, que permitem acesso programático a recursos em execução na AWS.

[34] Consulte: http://docs.amazonwebservices.com/AmazonRDS/latest/UserGuide/Concepts.DBInstance.html#Concepts.MultiAZ
[35] Disponível em: http://d36cz9buwru1tt.cloudfront.net/AWS_Building_Fault_Tolerant_Applications.pdf
[36] Acesse: http://aws.amazon.com/architecture/

Alocação de custos e várias contas da AWS

Alguns clientes têm considerado útil criar contas específicas para atividades de desenvolvimento e teste. Isso pode ser importante quando o seu ambiente de produção também é executado na AWS e você precisa separar equipes e responsabilidades. Contas separadas são isoladas umas das outras por padrão, para que, por exemplo, os usuários de desenvolvimento e teste não interfiram nos recursos de produção. Para permitir a colaboração, a AWS oferece uma série de recursos que permitem o compartilhamento de recursos entre contas, por exemplo, objetos do Amazon S3, AMIs e snapshots do Amazon EBS.

Para separar e alocar o custo para as várias atividades e fases do ciclo de desenvolvimento e teste, a AWS oferece várias opções. Uma opção é usar contas separadas (por exemplo, para desenvolvimento, teste, preparação e produção), e cada conta terá sua própria fatura. Você também pode consolidar várias contas, por exemplo, para simplificar pagamentos. Outra opção é usar o Relatório de alocação de custos, que permite organizar e controlar seus custos da AWS com a marcação de recurso. No contexto de desenvolvimento e teste, tags podem representar os vários estágios ou equipes do ciclo de desenvolvimento, embora você pode escolher as dimensões que considerar mais úteis.

Conclusão

As práticas de desenvolvimento e teste exigem determinados recursos em certos momentos para o ciclo de desenvolvimento. Em ambientes tradicionais, esses recursos podem simplesmente não estar disponíveis, ou não no prazo necessário. Quando esses recursos estão disponíveis, eles fornecem uma quantidade fixa de capacidade que é insuficiente, especialmente em atividades variáveis, como testes, ou desperdiçada (mas paga), quando os recursos não são usados[37].

A Amazon Web Services oferece uma alternativa econômica para infraestruturas tradicionais de desenvolvimento e teste. Em vez de esperar semanas ou até mesmo meses pelo hardware, você pode provisionar os recursos necessários instantaneamente, expandir o crescimento da carga de trabalho imediatamente e liberar recursos quando não forem mais necessários. Independentemente de os ambientes de desenvolvimento e teste consistirem em algumas ou centenas de instâncias, ou de serem necessários por algumas horas ou 24 horas por dia, 7 dias por semana, você só paga pelo que usar. A AWS é uma plataforma agnóstica de linguagem de programação e sistema operacional, e você pode escolher a plataforma de desenvolvimento ou o modelo de programação usado em sua empresa. Essa flexibilidade permite que você se concentre em seu projeto, não em operar e manter sua infraestrutura.

A AWS também permite possibilidades que eram difíceis de se obter com um hardware tradicional. Você pode automatizar totalmente os recursos na AWS para que os ambientes possam ser provisionados e desativados sem intervenção humana. Você pode iniciar ambientes de desenvolvimento sob demanda; iniciar compilações quando necessário, sem ser limitado pela disponibilidade de recursos; provisionar recursos de teste e orquestrar automaticamente execuções de teste ou campanhas inteiras.

A AWS oferece a você a capacidade de testar e iterar com uma infraestrutura pode ser alterada rapidamente. Suas equipes de projetos têm a liberdade de usar a capacidade de baixo custo para executar qualquer tipo de testes ou testar novas ideias, sem despesas iniciais ou compromissos de longo prazo, tornando a AWS uma plataforma ideal para desenvolvimento e teste.

[37] Consulte: http://aws.amazon.com/economics/

Outras leituras

- Whitepaper: “Checklists operacionais para a AWS” http://media.amazonwebservices.com/AWS_Operational_Checklists.pdf
- Whitepaper: “Como funciona a definição de preços da AWS” http://media.amazonwebservices.com/AWS_Pricing_Overview.pdf
- Centro de arquitetura da AWS – http://aws.amazon.com/architecture
- Whitepapers técnicos da AWS – http://aws.amazon.com/whitepapers/