Projetando para a nuvem: práticas recomendadas

Janeiro de 2010

Última atualização - Janeiro de 2011

Jinesh Varia

jvaria@amazon.com

Introdução

Os arquitetos de software estão há vários anos descobrindo e implementando vários conceitos e práticas recomenda das para criar aplicativos altamente escaláveis. Atualmente na "era do tera", esses conceitos são ainda mais aplicáveis por causa dos conjuntos de dados em contínuo crescimento, padrões de tráfego imprevisíveis e a procura por tempos de resposta mais rápidos. Este whitepaper vai reforçar e reiterar alguns destes conceitos tradicionais e discutir como pode evoluir no contexto da computação em nuvem. Também serão discutidos alguns conceitos sem precedentes, como a elasticidade que surgiu devido à natureza dinâmica da nuvem.

Este artigo é direcionado para os arquitetos de nuvem, que estão se preparando para mover um aplicativo empresarial de um ambiente físico fixo para um ambiente virtualizado em nuvem. O foco deste whitepaper é destacar conceitos, princípios e práticas recomendadas na criação de novos aplicativos em nuvem ou na migração de aplicativos existente para a nuvem.

Histórico

Como um arquiteto de nuvem, é importante compreender os benefícios de computação em nuvem. Nesta seção, você vai aprender alguns dos benefícios técnicos e comerciais da computação em nuvem e dos diferentes serviços atualmente disponíveis na AWS.

Benefícios comerciais da computação em nuvem

Existem alguns benefícios comerciais claros para a criação de aplicativos na nuvem. Alguns destes são listados aqui:

Quase zero de investimento em infraestrutura adiantada: se você precisa construir um sistema em grande escala isso pode custar uma fortuna em investimento em imóveis, segurança física, hardware (racks, servidores, routers, fontes de alimentação de backup), gerenciamento de hardware (gerenciamento de energia, resfriamento) e a equipe de operações. Devido aos elevados custos iniciais, o projeto normalmente exige várias etapas de aprovações de gerenciamento antes que possa ser iniciado. Agora, com o utilitário de estilo de computação em nuvem, não há custo fixo ou custo inicial.

Infraestrutura just-in-time: no passado, se seu aplicativo tornava-se popular e seus sistemas ou sua infraestrutura não podiam ser dimensionados, você tornava-se uma vítima do seu próprio sucesso. Por outro lado, se você investiu pesado e não alcançou a popularidade, você tornou-se uma vítima de seu fracasso. Com a implantação de aplicativos na nuvem com autoprovisionamento just-in-time, você não precisa se preocupar com a capacidade de pré-aquisição para sistemas de grande escala. Isso aumenta a agilidade, reduz o risco e os custos operacionais porque você dimensiona apenas de acordo com o seu *crescimento* e paga somente pelo que você utiliza.

Utilização de recursos mais eficientes: os administradores de sistema normalmente se preocupam com a aquisição de hardware (ao serem executados fora da capacidade) e com uma maior utilização de infraestrutura (quando têm capacidade ociosa e excessiva). Com a nuvem, eles podem gerenciar recursos de forma mais eficaz e eficiente por terem os aplicativos para solicitar e renunciar a recursos on demand.

Avaliação de custo baseado no uso: com o utilitário de estilo de preços, você será cobrado apenas pela infraestrutura que tiver sido utilizada. Você não está pagando pela infraestrutura alocada mas não utilizada. Isso acrescenta uma nova dimensão à redução de custos. Você pode ver uma economia imediata (por vezes, logo na fatura do próximo mês) quando você implanta um patch de otimização para atualizar seu aplicativo de nuvem. Por exemplo, se uma camada de cache pode reduzir as suas solicitações de dados em 70%, a economia começa a ser gerada imediatamente e você percebe a recompensa na próxima fatura. Além disso, se você estiver criando plataformas no topo da nuvem, você pode passar sobre a mesma estrutura de custos baseada no uso flexível, variável para seus próprios clientes.

Redução do tempo de entrega: a paralelização é uma das grandes maneiras de acelerar o processamento. Se um trabalho com computação ou dados de uso intensivo que podem ser executados em paralelo podem levar 500 horas para processar em uma máquina, com as arquiteturas de nuvem [6], seria possível gerar e iniciar 500 instâncias e processar o mesmo trabalho em 1 hora. Ter uma infraestrutura elástica disponível fornece o aplicativo com a capacidade de explorar a paralelização de forma econômica, reduzindo o tempo de mercado.

Benefícios técnicos do computação em nuvem

Alguns dos benefícios técnicos do computação em nuvem inclui:

Automação – "Infraestrutura programável": você pode criar sistemas de compilação e implementação reproduzíveis, aproveitando a infraestrutura programável (API-driven).

Auto-scaling: você pode dimensionar os seus aplicativos para diminuir ou expandir para atender a sua demanda inesperada sem qualquer intervenção humana. O dimensionamento automático incentiva a automação e as unidades mais eficientes.

Dimensionamento proativo: dimensione o seu aplicativo para diminuir e expandir para atender a sua demanda antecipada com a adequada compreensão do planejamento de seus padrões de tráfego para que você mantenha seus custos baixos enquanto em está dimensionando.

Ciclo de vida de desenvolvimento mais eficiente: os sistemas de produção podem ser facilmente clonados para uso como ambientes de teste e desenvolvimento. Ambientes de teste podem ser facilmente promovidos para ambientes de produção.

Melhorar a possibilidade de teste: nunca fique sem hardware para testes. Inserir e automatizar testes em todas as etapas do processo de desenvolvimento. Você pode gerar um "laboratório de teste instantâneo" com ambientes pré-configurados somente para a duração da fase de testes.

Recuperação de desastres e continuidade de negócios: a nuvem fornece uma opção de menor custo para a manutenção de uma frota de servidores DR e armazenamento de dados. Com a nuvem, você pode aproveitar a geo-distribuição e replicar o ambiente em outro local em poucos minutos.

"Excesso" de tráfego para a nuvem: com alguns cliques e técnicas de balanceamento de carga efetiva, você pode criar um aplicativo completo à prova de excesso ao direcionar o excesso de tráfego para a nuvem.

Noções básicas sobre o Amazon Web Services Cloud

A nuvem da Amazon Web Services (AWS) fornece uma infraestrutura altamente confiável e escalável para a implantação de soluções de escala da web, com o mínimo de custos com suporte e administração e mais flexibilidade do que se espera de sua própria infraestrutura, tanto no local como em uma instalação de datacenter.

Hoje a AWS oferece variedade de serviços de infraestrutura. O diagrama abaixo irá apresentar-lhe a terminologia da AWS para ajudá-lo a entender como o seu aplicativo pode interagir com diferentes serviços da Amazon Web Services e como eles interagem uns com os outros.

[1]O Amazon Elastic Compute Cloud (Amazon EC2) é um serviço da Web que fornece uma capacidade de computação redimensionável em nuvem. Você pode agrupar o sistema operacional, o software de aplicativo e as definições de configuração associadas em uma *Amazon Machine Image* (AMI). Você pode utilizar essas AMIs para configurar várias *instâncias* virtualizadas, bem como encerrá-las utilizando as chamadas simples de serviço web para dimensionar a capacidade de reduzir ou expandir rapidamente, alterando o seu requisito de capacidade. Você pode comprar as

[1] Mais informações sobre o Amazon EC2 estão disponíveis em http://aws.amazon.com/ec2

Instâncias On Demand cujo pagamento é calculado por hora ou as *Instâncias Reservadas* cujo pagamento é efetuado uma única vez e de um pequeno valor e receber uma taxa mais baixa de utilização para executar a instância do que para uma instância On-Demand ou também pode comprar as *Instâncias Spot* para as quais é possível solicitar redução de valores pela capacidade não utilizada e reduzir ainda mais o seu custo. As instâncias podem ser iniciadas em uma ou mais *regiões* geográficas. Cada região tem várias *Zonas de disponibilidade*. As Zonas de disponibilidade são as posições distintas que são projetadas para serem isoladas das falhas em outras Zonas da disponibilidade e fornecem rede de conectividade acessível e de baixa latência para outras Zonas de disponibilidade da mesma região.

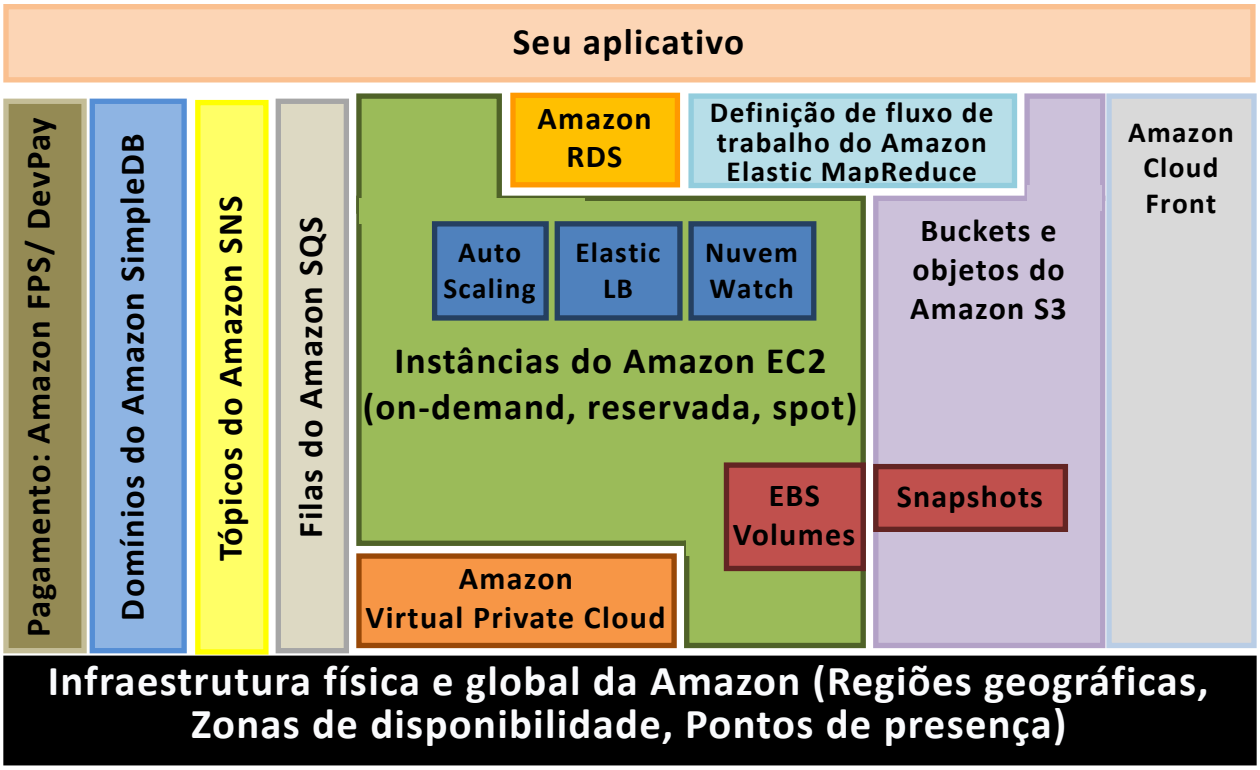

Figura 1: Amazon Web Services

Os endereços do *Elastic IP* permitem que você atribua um endereço IP estático de forma programática a uma instância. Você pode habilitar o monitoramento em uma instância do Amazon EC2 utilizando o Amazon CloudWatch[2] a fim de ganhar visibilidade na utilização de recursos, desempenho operacional e padrões de demanda global (incluindo métricas como a utilização do CPU, leituras e gravações de discos e tráfego de rede). Você pode criar um *grupo Auto Scaling* utilizando o recurso Auto-scaling[3] para dimensionar automaticamente a sua capacidade em determinadas condições com base em métricas que o Amazon CloudWatch coleta. Você também pode distribuir o tráfego de entrada, criando um *Elastic Load Balancer* utilizando o serviço Elastic Load Balancing[4]. Os volumes do Amazon Elastic Block Storage (EBS)[5] fornecem armazenamento persistente anexado à rede para instâncias do Amazon EC2. *Snapshots* consistentes nos volumes EBS atuais podem ser criadas e armazenadas no Amazon Simple Storage Service (Amazon S3)[6].

[2] Mais informações sobre o Amazon CloudWatch estão disponíveis em http://aws.amazon.com/cloudwatch/
[3] Mais informações sobre o recurso Auto-scaling estão disponíveis em http://aws.amazon.com/auto-scaling
[4] Mais informações sobre o recurso Elastic Load Balancing estão disponíveis em http://aws.amazon.com/elasticloadbalancing
[5] Mais informações sobre o Elastic Block Store estão disponíveis em http://aws.amazon.com/ebs
[6] Mais informações sobre o Amazon S3 estão disponíveis em http://aws.amazon.com/s3

O Amazon S3 é um armazenamento de dados distribuído e altamente durável. Com uma interface simples de serviços da web, você pode armazenar e recuperar grandes quantidades de dados como *objetos* em *buckets* (containers) a qualquer momento, de qualquer lugar na web utilizando verbos HTTP padrão. As cópias de objetos podem ser distribuídas e armazenadas em cache em 14 *pontos de presença* ao redor do mundo, criando uma *distribuição* utilizando o serviço Amazon CloudFront[7] – um serviço da web para entrega de conteúdo (estático ou de streaming). O Amazon SimpleDB[8] é um serviço da web que oferece uma funcionalidade principal de pesquisa de banco de dados em tempo real e de consulta simples de dados estruturados, sem a complexidade operacional. Você organiza seus dados estruturados em*domínios* e pode executar consultas em todos os dados armazenados em um domínio específico. Os domínios são conjuntos de*itens* que são descritos por *pares de valores de atributos*. O serviço de banco de dados relacional[9] (Amazon RDS) facilita a configuração, a operação e o dimensionamento de seu banco de dados relacional em nuvem. Você pode iniciar uma *Instância de banco de dados* e obter acesso a um banco de dados MySQL completo sem se preocupar com tarefas comuns de administração de banco de dados como backups, gerenciamento de patch, etc.

O Amazon Simple Queue Service (Amazon SQS)[10] oferece uma fila hospedada altamente escalável e confiável para o armazenamento de *mensagens* à medida em que elas transitam entre computadores.

O Amazon Simple Notifications Service (Amazon SNS)[11] fornece uma maneira simples para notificar aplicativos ou pessoas a partir da nuvem criando *tópicos* e utilizando um protocolo de publicação de assinatura.

O Amazon Elastic MapReduce[12] fornece uma estrutura Hadoop hospedada sendo executada na infraestrutura de escala da web do Amazon Elastic Compute Cloud (Amazon EC2) e Amazon Simple Storage Service (Amazon S3) e permite que você crie *JobFlows* personalizados. JobFlow é uma sequência de *etapas* do MapReduce.

O Amazon Virtual Private Cloud (Amazon VPC)[13] permite que você amplie a sua rede corporativa em uma nuvem privada contida na AWS. O Amazon VPC utiliza o modo Tunnel IPSec que permite que você crie uma conexão segura entre um gateway no seu datacenter e um gateway na AWS.

O Amazon Route53 é um serviço DNS altamente escalável que permite que você gerencie os seus registros DNS, criando um *HostedZone* para cada domínio que você gostaria de gerenciar.

O AWS Identity and Access Management (IAM)[14] permite que você crie múltiplos usuários e gerencie permissões para cada um desses usuários a partir de sua conta da AWS. O IAM é integrado nativamente nos serviços AWS. Nenhuma das APIs de serviço foi alterada para suportar a IAM, aplicativos de saída e as ferramentas projetadas a partir de APIs de serviço da AWS continuam a funcionar enquanto a IAM estiver sendo usada.

A AWS também oferece diversos serviços de pagamento e faturamento[15] que aproveitam a infraestrutura de pagamento da Amazon.

---

[7] Mais informações sobre o Amazon CloudFront estão disponíveis em http://aws.amazon.com/cloudfront
[8] Mais informações sobre o Amazon SimpleDB estão disponíveis em http://aws.amazon.com/simpledb
[9] Mais informações sobre o Amazon RDS estão disponíveis em http://aws.amazon.com/rds
[10] Mais informações sobre o Amazon SQS estão disponíveis em http://aws.amazon.com/sqs e
[11] Mais informações sobre o Amazon SNS estão disponíveis em http://aws.amazon.com/sns
[12] Mais informações sobre o Amazon ElasticMapReduce estão disponíveis em http://aws.amazon.com/sns
[13] Mais informações sobre o Amazon Virtual Private Cloud estão disponíveis em http://aws.amazon.com/vpc
[14] Mais informações sobre o Amazon Identity and Access Management estão disponíveis em http://aws.amazon.com/iam
[15] Mais informações sobre o Amazon Flexible Payments Service estão disponíveis em http://aws.amazon.com/fps e sobre Amazon DevPay estão disponíveis em http://aws.amazon.com/devpay

Todos os serviços de infraestrutura da AWS oferecem um utilitário de estilo de preços sem exigir contratos ou compromissos a longo prazo. Por exemplo, você paga por hora pelo uso da instância do Amazon EC2 e paga pelo gigabyte para armazenamento e transferência de dados no caso do Amazon S3. Mais informações sobre cada um desses serviços e seus definições de preço pague somente pelo o que usar estão disponíveis no site da AWS.

Observe que o uso da nuvem da AWS não requer sacrificar a flexibilidade e o controle a que você já está acostumado:

- Você é livre para utilizar o modelo de programação, linguagem ou sistema operacional (Windows, OpenSolaris ou qualquer tipo de Linux) de sua escolha.
- Você é livre para escolher os produtos da AWS que melhor satisfazem as suas necessidades — você pode utilizar qualquer um dos serviços individualmente ou em qualquer combinação.
- A AWS fornece recursos redimensionáveis (armazenamento, largura de banda e computação), e por isso você está livre para utilizar muito ou pouco e só paga o que você de fato utilizar.
- Você é livre para utilizar as ferramentas de gerenciamento de sistema utilizadas no passado e ampliar o seu datacenter para dentro da nuvem.

Conceitos de nuvem

A nuvem reforça alguns conceitos antigos de criação de arquiteturas [13] da Internet altamente escaláveis e introduz alguns novos conceitos que mudam completamente o modo pelo qual os aplicativos são criados e implantados. Assim, quando você passar do conceito para a implementação, você pode obter a sensação de que "Tudo mudou, mas nada é diferente". A nuvem altera vários processos, padrões, práticas, filosofias e reforça alguns princípios tradicionais de arquitetura orientados ao serviço que você aprendeu pois eles são ainda mais importantes do que antes. Nesta seção, você verá alguns desses novos conceitos de nuvem e conceitos SOA renovados.

Os aplicativos tradicionais foram projetados com base em alguns paradigmas pré-concebidos que tinham relevância econômica e arquitetônica no momento em que foram desenvolvidos. A nuvem traz algumas filosofias novas que você precisa entender e que são argumentadas abaixo:

Construção de arquiteturas escaláveis

É fundamental construir uma arquitetura escalável, a fim de tirar proveito de uma infraestrutura escalável.

A nuvem é projetada para fornecer escalabilidade infinita de forma conceitual. No entanto, você não pode aproveitar todas as escalabilidades em infraestrutura se a sua arquitetura não é escalável. Ambos devem trabalhar juntos. Você precisará identificar os componentes monolíticos e os afunilamentos em sua arquitetura, identificar as áreas onde você não pode aproveitar os recursos de provisionamento On-Demand e trabalhar para *refatorar* o seu aplicativo para aproveitar a infraestrutura escalável e tirar proveito da nuvem.

Características de um aplicativo verdadeiramente escalável:

- O aumento nos recursos gera um aumento proporcial no desempenho
- Um serviço escalável é capaz de lidar com heterogeneidade
- Um serviço escalável é operacionalmente eficiente
- Um serviço escalável é flexível
- Um serviço escalável deve tornar-se mais rentável quando cresce (custo por unidade diminui com o aumento do número de unidades)

Estas são as informações que devem se transformar em uma parte inerente do seu aplicativo e se você projetar a sua arquitetura com as características acima em mente, logo em seguida a sua arquitetura e infraestrutura trabalharão juntas para disponibilizar a você a escalabilidade que está procurando.

Noções básicas sobre a elasticidade

O gráfico abaixo ilustra as diferentes abordagens que um arquiteto de nuvem pode realizar para dimensionar os seus aplicativos a fim de atender a demanda.

Abordagem de expansão: sem se preocupar com a arquitetura de aplicativo escalável e investindo fortemente em computadores maiores e mais poderosos (dimensionamento vertical) para atender a demanda. Essa abordagem normalmente funciona até certo ponto, mas poderia custar uma fortuna (ver "Despesas enormes de capital" no diagrama) ou a demanda poderia aumentar a capacidade antes que seja implantado um novo "supercomputador" (ver "Você acaba de perder seus clientes" no diagrama).

Abordagem de ampliação tradicional: criando uma arquitetura que se expande horizontalmente e investindo na infraestrutura em pequenas partes. A maioria das empresas e aplicativos da web em grande escala segue esse padrão para distribuir seus componentes de aplicativos, particionando seus conjuntos de dados e utilizando um projeto orientado a serviços. Esta abordagem é muitas vezes mais eficaz do que uma abordagem de expansão. No entanto, sso requer ainda prever a demanda em intervalos regulares e, em seguida, a implantar um infraestrutura em partes para atender à demanda. Isso muitas vezes leva ao excesso de capacidade ("redução de dinheiro") e um monitoramento manual constante ("redução de ritmo de trabalho"). Além disso, ele normalmente não funciona se o aplicativo usado excessivamente (muitas vezes referido como o efeito Slashdot[16]).

Observação: ambas as abordagens têm custos iniciais e ambas são reativas na natureza.

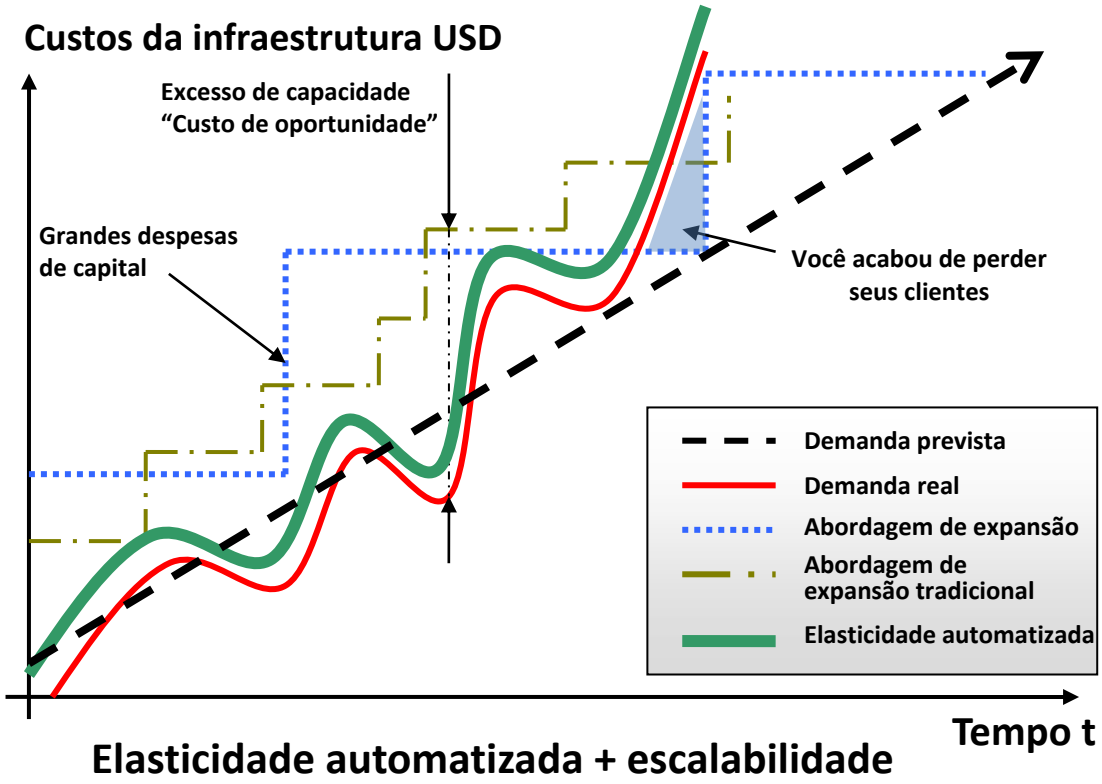

Figura 2: elasticidade automatizada

[16] http://en.wikipedia.org/wiki/Slashdot_effect

A infraestrutura tradicional geralmente exige uma previsão da quantidade de recursos computacionais que o seu aplicativo utilizará ao longo de um período de vários anos. Se você subestimar, os seus aplicativos não terão a potência suficiente para lidar com o tráfego inesperado, resultando potencialmente na insatisfação do cliente. Se você superestimar, perderá dinheiro com recursos supérfluos.

No entanto, a natureza elástica e on demand da *abordagem de nuvem* (elasticidade automatizada), permite que a infraestrutura possa ser alinhada (como ela se expande e reduz) com a demanda real, aumentando assim a utilização global e reduzindo os custos.

A elasticidade é uma das propriedades fundamentais da nuvem. A elasticidade é o poder para dimensionar recursos computacionais diminuindo ou expandindo facilmente e com o mínimo de atrito. É importante compreender que a elasticidade acabará por conduzir a maioria dos benefícios da nuvem. Como um arquiteto de nuvem, você precisará internalizar este conceito e trabalhar em sua arquitetura de aplicativo a fim de aproveitar a nuvem ao máximo.

Tradicionalmente, os aplicativos foram projetados para a infraestrutura fixa, rígida e pré-provisionada. As empresas nunca tiveram a necessidade de provisionar e instalar servidores em base diária. Como resultado, a maioria das arquiteturas de software não atendem a implementação rápida ou a redução de hardware. O tempo de provisionamento e investimento inicial até a aquisição de novos recursos era demasiadamente elevado, os arquitetos de software nunca investiram tempo e recursos na otimização para a utilização do hardware. Era aceitável se o hardware no qual o aplicativo estava sendo executado fosse subutilizado. A noção de “elasticidade” dentro de uma arquitetura foi esquecida porque a ideia de ter novos recursos em minutos não foi possível.

Com a nuvem, essa mentalidade precisa mudar. A computação em nuvem simplifica o processo de aquisição dos recursos necessários; não há nenhuma necessidade de fazer pedidos antes do tempo e de manter o hardware não utilizado em cativeiro. Em vez disso, os arquitetos de nuvem podem solicitar o que precisam poucos minutos antes de precisarem ou automatizar o processo de aquisição, aproveitando a vasta escala e o tempo de resposta rápida da nuvem. O mesmo se aplica para liberar os recursos desnecessários ou subutilizados quando você não precisa deles.

Se você não pode abraçar a mudança e implementar a elasticidade na sua arquitetura de aplicativo, você será incapaz de aproveitar a nuvem ao máximo. Assim como um arquiteto de nuvem, você deve pensar de maneira criativa pensando de que maneiras você pode implementar a elasticidade no seu aplicativo. Por exemplo, a infraestrutura que costumava executar compilações diárias todas as noites e realizar testes de regressão e unidade todas as noites às 2h durante duas horas (muitas vezes denominadas como “Caixa de controle de qualidade da compilação”) ficava como ociosa pelo resto do dia. Agora, com a infraestrutura elástica, pode-se executar compilações noturnas em caixas que estão “ativas” e está sendo pago apenas por 2 horas por noite. Da mesma forma, um problema interno no aplicativo da web de emissão de tíquetes que sempre é utilizado para executar em capacidade máxima (5 servidores 24 x 7 x 365) para atender a demanda durante o dia agora pode ser configurado também para ser executado sob demanda (5 servidores das 9h as 17h e 2 servidores das 17h as 9h) com base no padrão de tráfego.

Projetar arquiteturas inteligentes de nuvens elásticas, para que a infraestrutura seja executada somente quando você precisa dela, é uma arte em si. A elasticidade deve ser um dos requisitos do projeto arquitetônico ou uma propriedade do sistema. Perguntas que você precisa fazer: quais os componentes ou as camadas em minha arquitetura de aplicativo podem se tornar elásticas? O que será necessário para tornar esse componente *elástico*? Qual será o impacto da implementação da elasticidade à minha arquitetura geral do sistema?

Na próxima seção, você verá técnicas específicas para implementar a elasticidade em seus aplicativos. Para aproveitar efetivamente os benefícios da nuvem, é importante um arquiteto com essa mentalidade.

Não temendo restrições

Quando você decidir mover os seus aplicativos para a nuvem e tentar mapear as suas especificações de sistema para aquelas disponíveis na nuvem, você vai notar que a nuvem pode não ter a especificação exata do recurso que você tem no local. Por exemplo, “a nuvem não fornece um X de memória RAM em um servidor” ou “Meu banco de dados precisa ter mais IOPS do que posso obter em uma única instância”.

Você deve compreender que a nuvem fornece *recursos abstratos* e eles se tornam poderosos quando você os combina com o modelo de provisionamento on demand. Você não deve ficar com receio e nem constrangido, quando estiver utilizando recursos de nuvem porque é importante compreender que, mesmo que você não tenha uma réplica exata do seu hardware no ambiente da nuvem, você poderá obter mais desses recursos em nuvem para compensar essa necessidade.

Por exemplo, se a nuvem não lhe fornecer RAM com exatidão ou maior quantidade em um servidor, tente usar um cache distribuído do tipo *memcached*[17] ou particionar seus dados em vários servidores. Se seus bancos de dados precisam de mais E/S por segundo e não mapearem diretamente a da nuvem, há várias recomendações que você pode escolher de acordo com seu tipo de dados e utilização. Se for um aplicativo de leitura frequente, você pode distribuir a carga de leitura através de uma frota de escravos sincronizados. Como alternativa, você pode usar um algoritmo de *fragmentação* [10] que direciona os dados aonde eles precisam estar ou você pode usar várias soluções de clustering de bancos de dados.

Recapitulando, quando você combina os recursos de provisionamento sob demanda com a flexibilidade, você vai notar que aparentes restrições na verdade poderão ser entendidas como formas que realmente irão melhorar a escalabilidade e o desempenho geral do sistema.

Administração virtual

O advento da nuvem mudou o papel do administrador do sistema para um "administrador de sistema virtual". Isso simplesmente significa que tarefas rotineiras realizadas por esses administradores agora tornaram-se ainda mais interessantes à medida que eles aprendem mais os sobre aplicativos e decidem o que é melhor para a empresa como um todo. O administrador do sistema já não tem a necessidade de provisionar servidores, instalar software e conectar dispositivos de rede já que todo esse trabalho maçante é substituído por alguns cliques e linhas de comando. A nuvem incentiva a automação porque a infraestrutura é programável. Os administradores de sistema precisam mover a roda da tecnologia e aprender como gerenciar recursos abstratos de nuvem abstrata utilizando scripts.

Da mesma forma, a função do administrador do banco de dados se altera para administrador de banco de dados virtual na qual ele/ela gerencia recursos através de um console baseado na web, executa scripts que adicionam novas capacidades por meio de programação no caso dos recursos de hardware de banco de dados se esgotarem e automatiza os processos de rotina. O DBA virtual deve agora aprender novos métodos de implantação (imagens de máquina virtual), adotar novos modelos (paralelização de consulta, geo-redundância e replicação assíncrona [11]), repensar a arquitetura de abordagem para dados (fragmentação [9], particionamento horizontal [13], federação [14]) e aproveitamento das diferentes opções de armazenamento disponíveis na nuvem para os diferentes tipos de conjuntos de dados.

Na empresa tradicional, os desenvolvedores de aplicativos podem não trabalhar em estreita colaboração com os administradores de rede e administradores de rede podem não ter a menor ideia sobre o aplicativo. Como resultado, possíveis otimizações na camada de rede e na camada de arquitetura do aplicativo poderão negligenciadas. Com a nuvem, as duas funções se fundiram a uma em razoável extensão. Projetando a arquitetura de futuras aplicações, as empresas devem encorajar mais polinização cruzada de conhecimento entre as duas funções e entender que elas estão em processo de fusão.

[17] http://www.danga.com/memcached/

Práticas recomendadas em nuvem

Nesta seção, você conhecerá as práticas recomendadas que lhe ajudarão a construir um aplicativo na nuvem.

Se prepare para falhas e nada falhará

Regra de ouro: seja um pessimista ao projetar arquiteturas na nuvem; presuma que tudo irá falhar. Em outras palavras, sempre projete, implemente e implante para recuperação automatizada de falha.

Principalmente, presuma que seu hardware *falhará*. Presuma que Pressuponha que algum desastre *irá* atingir seu aplicativo. Pressuponha que você *será* solicitado com mais do que o número esperado de pedidos por segundo algum dia. Pressuponha que com o tempo o seu software aplicativo falhará também. Sendo um pessimista, você acaba refletindo sobre estratégias de recuperação ao longo do tempo de design, o que ajuda na criação de um sistema melhor como um todo.

Se você perceber que as coisas falham ao longo do tempo e incorporar esse pensamento na sua arquitetura, crie mecanismos para lidar com essa falha antes que o desastre chegue para lidar com uma infraestrutura escalável, você vai acabar criando uma arquitetura tolerante a falhas que é otimizada para a nuvem.

Questões que você precisa levantar: o que acontecerá se um nó em seu sistema falhar? Como você reconhece esta falha? Como eu faço para substituir o nó? Para quais tipos de cenários que eu tenho que me planejar? Quais são meus pontos únicos de falha? Se um balanceador de carga está disposto na frente de uma matriz de servidores de aplicativos, o que acontecerá se o balanceador de carga falhar? Se não houver mestre e escravos em sua arquitetura, o que acontecerá se o nó mestre falhar? Como é que ocorre o failover e como é que um escravo novo é instanciado e colocado em sincronia com o mestre?

Da mesma forma que projetar para falhas de hardware, você também deve projetar para falhas de software. Questões que você precisa levantar: O que acontecerá com o meu aplicativo se os seus serviços dependentes mudarem suas interfaces? E se o serviço de downstream expirar ou retornar uma exceção? E se as chaves de cache crescerem além do limite de memória de uma instância?

Crie mecanismos para lidar com essa falha. Por exemplo, as seguintes estratégias podem ajudar em caso de falha:

1. Ter um backup coerente e estratégia de restauração para os seus dados e automatizá-la
2. Construir linhas de processo que se retomam na reinicialização
3. Permitir que o estado do sistema se ressincronize através de recargas de mensagens de filas
4. Mantenha imagens virtuais pré-configuradas e pré-otimizadas para suporte (2) e (3) no lançamento/inicialização
5. Evite sessões in memory ou contexto de monitoração de estado de usuário, mova tudo isso para armazenamentos de dados.

Boas arquiteturas de nuvem devem ser a prova de reboots e reinicalizações. Em GrepTheWeb (discutido no artigo Cloud Architectures [6]), usando uma combinação do Amazon SQS e do Amazon SimpleDB, a arquitetura geral do controlador é muito resistente aos tipos de falhas listados nesta seção. Por exemplo, se a instância na qual o thread controlador estava sendo executado for desativada, ela pode ser reativada e retomar o seu estado anterior, como se nada tivesse acontecido. Isso foi alcançado com a criação de uma Amazon Machine Image (AMI) pré-configurada, que, quando iniciada, retira da fila todas as mensagens do Amazon SQS e lê seus estados a partir de um domínio em reboot do Amazon SimpleDB.

Projetar com uma suposição de que o hardware subjacente irá falhar, pode prepará-lo para o futuro quando isso realmente ocorrer.

Este princípio de design o ajudará a projetar aplicativos de fácil operação, assim como foi destacado no artigo de Hamilton [11]. Se você puder estender este princípio para medir e equilibrar cargas dinamicamente e de forma proativa, pode ser que você consiga gerenciar a variação no desempenho de disco e de rede que existe devido à natureza multilocatária da nuvem.

Táticas específicas da AWS para implementar essa prática recomendada:

1. Enfrente um failover com tranquilidade usando Elastic IPs: Elastic IP é um endereço IP estático que é dinamicamente remapeável. Você pode rapidamente remapear e fazer um failover para outro conjunto de servidores para que o seu tráfego seja direcionado para os novos servidores. Isso funciona muito bem quando você deseja fazer atualizações para versões mais novas ou em caso de falhas de hardware
2. Utilize várias Zonas de disponibilidade: as Zonas de disponibilidade são conceitualmente semelhantes a datacenters lógicos. Ao implantar sua arquitetura em várias Zonas de disponibilidade, você pode garantir alta disponibilidade. Utilize a funcionalidade de implantação do Amazon RDS Multi-AZ [21] para replicar automaticamente atualizações de banco de dados em várias Zonas de disponibilidade.
3. Mantenha uma Amazon Machine Image para que você possa facilmente restaurar e clonar ambientes em uma Zona de disponibilidade diferente; Mantenha vários bancos de dados secundários nas Zonas de disponibilidade e configure uma replicação dinâmica.
4. Utilize o Amazon CloudWatch (ou várias ferramentas de monitoramento em tempo real de código aberto) para obter mais visibilidade e tomar as medidas apropriadas em caso de degradação de desempenho ou falha de hardware. Configure um grupo de Auto scaling para manter um tamanho fixo de frota para que este substitua as instâncias com problemas do Amazon EC2 por novas.
5. Utilize o Amazon EBS e configure trabalhos cron para que as snapshots incrementais sejam automaticamente enviadas para o Amazon S3 e os dados sejam mantidos independentes de suas instâncias.
6. Utilize o Amazon RDS e defina o período de retenção para os backups, para que ele possa realizar backups automatizados.

Separe seus componentes

A nuvem reforça o princípio de design SOA de que *quanto mais separados estiverem os componentes do sistema, muito mais e de melhor maneira eles se dimensionarão*.

A chave é construir componentes que não sejam tão dependentes uns dos outros, para que se um componente for desativado (falhar), estiver suspenso (não responder) ou permanecer ocupado (lento para responder) por algum motivo, os outros componentes do sistema são construídos, a fim de continuar a trabalhar como se nenhuma falha estivesse acontecendo. Basicamente, o acoplamento mais solto isola a várias camadas e componentes do seu aplicativo para que cada componente interaja de forma assíncrona com os outros e os trate como uma “caixa preta”. Por exemplo, no caso da arquitetura do aplicativo da web, você pode isolar o servidor de aplicativo do servidor web e de banco de dados. O servidor de aplicativo não sabe sobre seu servidor web e vice-versa, isto oferece uma dissociação entre essas camadas e não há nenhuma dependência em nível de código ou de perspectivas funcionais. No caso da arquitetura de processamento em lote, você pode criar componentes *assíncronos* que são independentes uns dos outros.

Perguntas que você precisa fazer: qual componente de negócios ou recurso poderia ser isolado do aplicativo atual monolítico e pode ser executado de forma autônoma separadamente? E, em seguida, como posso adicionar mais instâncias desse componente sem quebrar meu sistema atual e ao mesmo tempo atender a mais usuários? Quanto esforço será necessário para encapsular o componente para que ele possa interagir com outros componentes de forma assíncrona?

A dissociação entre seus componentes, a construção de sistemas *assíncronos* e o dimensionamento horizontal tornam-se muito importantes no contexto da nuvem. Isso não somente permitirá que você amplie seu sistema adicionando mais instâncias do mesmo componente, mas também permitirá que você crie modelos híbridos inovadores nos quais alguns componentes continuam a ser executados no local, enquanto outros componentes podem tirar proveito do dimensionamento em nuvem e usar a nuvem para potência de computação adicional e largura de banda. Dessa forma, com o mínimo esforço, você pode direcionar "excesso" de tráfego para a nuvem através da implementação de táticas de balanceamento de carga inteligente.

É possível criar um sistema de baixo acoplamento usando *filas de mensagens*. Se uma fila/buffer é usada para conectar quaisquer dois componentes, será possível oferecer suporte a simultaneidade, a alta disponibilidade e a picos de carga. Como resultado, o sistema como um todo continua a ser executado mesmo se partes de componentes estão temporariamente indisponíveis. Se um componente é desativado ou fica temporariamente indisponível, o sistema armazenará as mensagens em buffer e as processará quando o componente for reativado.

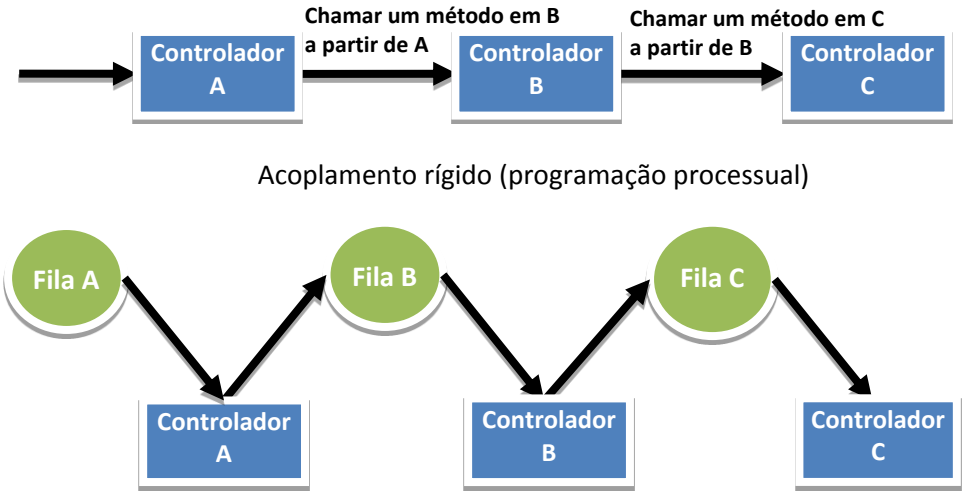

Figura 3: dissociação entre componentes usando filas

Você observará um grande uso de filas na arquitetura GrepTheWeb descritas em detalhes no artigo Cloud Architectures [6]. Em GrepTheWeb, se muitas solicitações de repente chegam ao servidor (uma situação de sobrecarga induzida pela Internet) ou o processamento de expressões regulares leva mais tempo do que a média (lenta taxa de resposta de um componente), as filas do Amazon SQS armazenam as solicitações em buffer de forma durável para que esses atrasos não afetem outros componentes.

Táticas específicas da AWS para implementar essa prática recomendada:

1. Use o Amazon SQS para isolar componentes [18]
2. Use o Amazon SQS como armazenamento em buffers entre componentes [18]
3. Projete cada componente a fim de que exponha uma interface de serviço e seja responsável por sua própria capacidade de expansão em todas as dimensões adequadas e para que interaja com outros componentes de forma assíncrona
4. Inclua a construção lógica de um componente em uma Amazon Machine Image para que ele possa ser implantado com mais frequência
5. Deixe seus aplicativos o mais sem monitoração de estado possível. Armazene o estado de sessão fora do componente (no Amazon SimpleDB, se apropriado)

Implemente elasticidade

A nuvem traz um novo conceito de elasticidade para seus aplicativos. A elasticidade pode ser implementada de três maneiras:

1. Dimensionamento proativo cíclico: dimensionamento periódico que ocorre em intervalo fixo (diária, semanal, mensal ou trimestralmente)
2. Dimensionamento proativo baseado em evento: executa o dimensionamento apenas quando você está esperando uma grande onda de solicitações de tráfego devido a um evento de negócios agendado (novo lançamento de produto, campanhas de marketing)
3. Auto-scaling baseado em demanda. Usando um serviço de monitoramento, seu sistema pode enviar disparadores para tomar as medidas apropriadas para que ele possa se expandir ou se reduzir com base em métricas (utilização dos servidores ou rede e/s, por exemplo)

Para implementar a "Elasticidade", é preciso primeiro automatizar o processo de implantação e simplificar a configuração e o processo de compilação. Isso assegurará que o sistema pode ser expandido sem qualquer intervenção humana.

Isso resultará em benefícios de custo imediato, à medida que a utilização geral é aumentada ao garantir que seus recursos estejam alinhados com a demanda, em vez de sendo executados potencialmente em servidores que são subutilizados.

Automatize sua infraestrutura

Um dos benefícios mais importantes do uso de um ambiente de nuvem é a capacidade de usar APIs da nuvem para automatizar o processo de implantação. É recomendável que você disponha de tempo para criar um processo automatizado de implantação no início durante o processo de migração e não espere até o final. Criar um processo de implantação automatizada e repetível ajudará a reduzir erros e facilitará um processo de atualização eficiente e escalável.

Para automatizar o processo de implantação:

- Criar uma biblioteca de "receitas" – pequenos scripts utilizados frequentemente (para instalação e configuração)
- Gerencie a configuração e o processo de implantação usando agentes incluídos em uma AMI
- Inicialização de suas instâncias

Inicialização de suas instâncias

Faça com que suas instâncias lhe façam uma pergunta ao inicializar "quem sou eu e qual é a minha função"? Cada instância deveria ter uma função ("servidor de banco de dados", "servidor de aplicativos", "servidor secundário" no caso de um aplicativo da Web) para desempenhar no ambiente. Esta função pode ser passada como um argumento durante a inicialização, que instrui a AMI sobre quando instanciar e quais os passos a tomar depois de ser iniciada. Na inicialização, as instâncias devem pegar os recursos necessários (código, scripts, configuração) com base na função e "anexá-los" para um cluster para servir a sua função. Benefícios da inicialização de suas instâncias:

1. Recrie o ambiente (Dev, preparo, produção) em alguns cliques e com um esforço mínimo
2. Mais controle sobre seus recursos abstratos baseados em nuvem
3. Reduzir erros de implantação induzidos pelo homem
4. Crie um ambiente de auto correção e de auto descobrimento que é mais resistente a falhas de hardware

Táticas específicas da AWS para automatizar sua infraestrutura

1. Defina grupos de Auto-scalling para diferentes clusters usando o recurso de Auto-scaling da Amazon no Amazon EC2.
2. Monitore suas métricas de sistema (CPU, memória, E/S de disco, E/S de rede) usando o Amazon CloudWatch e tome as medidas apropriadas (iniciar novas AMIs dinamicamente usando o serviço de Auto-scaling) ou envie notificações.
3. Armazene e recupere informações de configuração de máquina dinamicamente: utilize o Amazon SimpleDB para buscar dados de configuração durante o tempo de inicialização de uma instância (eg. sequências de conexão de banco de dados). O SimpleDB também pode ser usado para armazenar informações sobre uma instância, como seu endereço IP, o nome da máquina e a função.
4. Crie um processo de compilação que insira as últimas versões em um bucket no Amazon S3. Faça o download da versão mais recente de um aplicativo que ocorreu durante a inicialização do sistema.
5. Invista na construção de ferramentas de gerenciamento de recursos (Scripts automatizados, imagens pré-configuradas) ou use ferramentas de gerenciamento de configuração inteligente de código aberto como Chef [18], Puppet[19], CFEngine[20] ou Genome[21].
6. Inclua sistema operacional suficiente (JeOS [22]) e suas dependências de software em uma Amazon Machine Image para que seja mais fácil gerenciar e manter. Passe parâmetros ou arquivos de configuração no momento da inicialização e recupere dados do usuário[23] e metadados de instância após a inicialização.
7. Reduza a agregação e o tempo de inicialização ao iniciar a partir de volumes do Amazon EBS[24] e anexar vários volumes do Amazon EBS a uma instância. Crie snapshots de volumes comuns e compartilhe snapshots [25] entre contas, sempre que possível.
8. Componentes do aplicativo não devem presumir integridade ou localização do hardware no qual estão sendo executados. Por exemplo, anexe dinamicamente o endereço IP de um novo nó ao cluster. Faça um failover automaticamente e inicie um novo clone em caso de falha.

Pense paralelo

A nuvem faz a paralelização sem esforço. Se ela está solicitando dados de nuvem, armazenando dados para a nuvem, processando dados (ou executando trabalhos) na nuvem, como um arquiteto de nuvem, você precisará internalizar o conceito de paralelização ao projetar arquiteturas na nuvem. É aconselhável não apenas implementar a paralelização sempre que possível, mas também automatizá-la pois a nuvem permite que você crie um processo repetitivo muito facilmente.

[18] Mais informações sobre o Chef podem ser encontradas em http://wiki.opscode.com/display/chef/Home
[19] Mais informações sobre o Puppet podem ser encontradas em http://reductivelabs.com/trac/puppet/
[20] Mais informações sobre o CFEngine podem ser encontradas em http://www.cfengine.org/
[21] Mais informações sobre o Genome podem ser encontradas em http://genome.et.redhat.com/
[22] http://en.wikipedia.org/wiki/Just_enough_operating_system
[23] Metadados de instância e dados de usuários podem ser encontrados em http://docs.amazonwebservices.com/AWSEC2/latest/DeveloperGuide/index.html?AESDG-chapter-instancedata.html
[24] Mais informações sobre o recurso de Iniciar a partir do Amazon EBS podem ser encontradas em http://developer.amazonwebservices.com/connect/entry.jspa?externalID=3121
[25] Veja como compartilhar um Snapshot em http://aws.amazon.com/ebs/

Quando se trata de acesso a dados (recuperando e armazenando), a nuvem é projetada para manipular operações massivamente paralelas. Para atingir o máximo desempenho e taxa de transferência, você deve aproveitar o*pedido de paralelização*. Ao realizar o multi-threading de suas solicitações usando threads simultâneos armazenará ou buscará os dados mais rapidamente do que solicita-los em sequência. Assim, sempre que possível, os processos de um aplicativo em nuvem devem ser feitos de maneira segura para o thread através de uma filosofia de não compartilhamento e de aproveitamento de multi-threading.

Quando se trata de processamento ou de solicitações em execução na nuvem, torna-se ainda mais importante aproveitar a paralelização. Uma prática recomendada geral, no caso de um aplicativo da web, é distribuir as solicitações de entrada em vários servidores web assíncronos usando o balanceador de carga. Em caso de aplicativo de processamento em lote, você pode dominar nós que podem gerar vários nós de trabalho secundário nessa tarefa de processos em paralelo (como em estruturas de processamento distribuído como o Hadoop [26])

A beleza da nuvem aparece quando você combina elasticidade e paralelização. Seu aplicativo de nuvem pode trazer um cluster de instâncias de computação que é provisionado em minutos com apenas algumas chamadas de API, pode realizar um trabalho, executando tarefas em paralelo, pode armazenar os resultados e pode encerrar todas as instâncias. O aplicativo GrepTheWeb discutido em [6] é um exemplo disso.

Táticas específicas da AWS para paralelização:

1. Realize o lançamento de vários threads de seu Amazon S3 conforme detalhamento no artigo sobre Práticas recomendadas [2]
2. Realize o lançamento de vários threads de suas solicitações GET e BATCHPUT do Amazon SimpleDB [3] [4] [5]
3. Crie um JobFlow usando o serviço Amazon Elastic MapReduce para cada um dos seus processos diários de lote (indexação, análise de logs, etc.) que calcularão o trabalho em paralelo e economizarão tempo
4. Use o serviço do Elastic Load Balancing e distribua sua carga entre vários servidores de aplicativos web *dinamicamente*

Mantenha os dados dinâmicos mais próximo da computação e os dados estáticos mais próximos do usuário final

Em geral é uma boa prática manter seus dados o mais próximo possível para seus elementos de computação ou processamento a fim de reduzir a latência. Na nuvem, essa prática recomendada é ainda mais relevante e importante porque muitas vezes você tem de lidar com latências de Internet. Além disso, na nuvem, você está pagando pela largura de banda dentro e fora da nuvem por gigabyte de transferência de dados e o custo pode aumentar muito rapidamente.

Se uma grande quantidade de dados que precisa ser processada reside fora da nuvem, pode ser mais barato e mais rápido "enviar" e transferir os dados para a nuvem primeiro e, em seguida, fazer a computação. Por exemplo, no caso de um aplicativo de armazenamento de dados, é aconselhável mover o conjunto de dados para a nuvem e, em seguida, executar consultas paralelas com relação ao conjunto de dados. No caso de aplicativos web que armazenam e recuperam dados de bancos de dados relacionais, é aconselhável mover o banco de dados, bem como o servidor de aplicativo para a nuvem todos ao mesmo tempo.

[26] http://hadoop.apache.org/

Se os dados são gerados em nuvem, em seguida, os aplicativos que consomem os dados também devem ser implantados em nuvem para que eles possam aproveitar a transferência de dados em nuvem e baixa latência. Por exemplo, no caso de um aplicativo web de e-commerce que gera logs e dados de sequência de cliques, é aconselhável executar o analisador de log e relatório de motores em nuvem.

Por outro lado, se os dados são estáticos e não mudarão com frequência (por exemplo, imagens, vídeo, áudio, PDFs, JS, arquivos CSS), é aconselhável tirar proveito de um serviço de entrega de conteúdo para que os dados estáticos sejam armazenadas em cache em um local periférico mais próximo do usuário final (solicitante) diminuindo assim a latência de acesso. Devido ao cache, um serviço de entrega de conteúdo fornece acesso mais rápido aos objetos populares.

Táticas específicas da AWS para implementar essa prática recomendada:

1. Insira seus dados de discos rígidos no Amazon usando o serviço de Import/Export[27]. Pode ser mais barato e mais rápido para mover grandes quantidades de dados usando o sneakernet[28] do que para carregar usando a Internet
2. Utilize a mesma Zona de disponibilidade para lançar um cluster de máquinas
3. Crie uma distribuição de seu bucket do Amazon S3 e deixe o conteúdo dos caches do Amazon CloudFront desse bucket em todos os 14 pontos de presença ao redor do mundo

Práticas recomendadas de segurança

Em um ambiente de multi-locatário, arquitetos de nuvem frequentemente demonstram preocupações com a segurança. *A segurança deve ser implementada em cada nível da sua arquitetura de aplicativo em nuvem.* A segurança física normalmente é abordada por seu provedor de serviços (Whitepaper de segurança [7]), que é um benefício adicional do uso da nuvem. Segurança de rede e de aplicativo é de sua responsabilidade e você deve implementar as práticas recomendadas aplicáveis ao seu negócio. Neste whitepaper, você aprenderá sobre algumas ferramentas específicas, recursos e orientações sobre como proteger o seu aplicativo em nuvem no ambiente da AWS. É recomendável aproveitar essas ferramentas e recursos mencionados para implementar a segurança básica e então implementar as práticas recomendadas de segurança adicional usando os métodos padrão conforme apropriado ou como achar melhor.

Proteja seus dados em trânsito

Se você precisar trocar informações confidenciais entre um navegador e um servidor web, configure o SSL em sua instância de servidor. Você precisará de um certificado de uma autoridade de certificação externa como o VeriSign[29] ou o Entrust[30]. A chave pública incluída no certificado autentica o seu servidor para o browser e serve como base para criar a chave de sessão compartilhada usada para criptografar os dados em ambas as direções.

Crie uma Virtual Private Cloud fazendo algumas chamadas de linha de comando (usando Amazon VPC). Isso permitirá que você use os seus próprios recursos isolados logicamente dentro da nuvem da AWS e, em seguida, conecte esses recursos diretamente ao seu próprio datacenter usando as conexões criptografadas IPSec VPN padrão do setor.

Você também pode configurar [15] um servidor OpenVPN em uma instância de Amazon EC2 e instalar o cliente OpenVPN em todos usuário dos PCs.

[27] Mais informações sobre o Amazon Import Export Services podem ser encontradas em http://aws.amazon.com/importexport
[28] http://en.wikipedia.org/wiki/Sneakernet
[29] http://www.verisign.com/ssl/
[30] http://www.entrust.net/ssl-products.htm

Proteja seus dados em repouso

Se você estiver preocupado sobre como armazenar dados confidenciais em nuvem, você deve criptografar os dados (arquivos individuais) antes de enviá-los para a nuvem. Por exemplo, criptografar os dados usando qualquer ferramenta de código aberto[31] ou comercial[32] baseada em PGP antes de armazená-los como objetos do Amazon S3 e descriptografá-lo após o download. Isso geralmente é uma boa prática ao criar aplicativos compatíveis com HIPPA [8] que precisam armazenar Informações protegidas de saúde (PHI).

No Amazon EC2, a criptografia de arquivo depende do sistema operacional. As instâncias do Amazon EC2 que executam o Windows podem usar o recurso interno do Encrypting File System (EFS) [16]. Este recurso abordará a criptografia e a descriptografia de arquivos e pastas automaticamente e tornará o processo transparente para os usuários [19].
No entanto, apesar do nome, o EFS não criptografa o sistema de arquivos inteiro; em vez disso, ele criptografa arquivos individuais. Se você precisa de um volume completamente criptografado, considere o uso do produto de código aberto TrueCrypt[33]; ele se integrará muito bem com volumes do EBS formatados pelo NTFS. As instâncias do Amazon EC2 executando Linux podem montar volumes do EBS usando sistemas de arquivo criptografado usando uma variedade de abordagens (EncFS[34], Loop-AES[35], dm-crypt[36], TrueCrypt[37]). Da mesma forma, as instâncias do Amazon EC2 executando o OpenSolaris podem aproveitar o ZFS[38] Encryption Support [20]. Independentemente de qual abordagem você escolha, a criptografia de arquivos e volumes no Amazon EC2 ajuda a proteger os arquivos e os dados de log para que somente os usuários e processos no servidor possam ver os dados em texto descriptografado, mas nada nem ninguém fora do servidor ver apenas os dados criptografados.

Não importa qual sistema operacional ou tecnologia você escolha, a criptografia de dados em repouso apresenta um desafio: gerenciar as chaves usadas para criptografar os dados. Se você perder as chaves, perderá seus dados para sempre e se as suas chaves forem comprometidas, os dados podem estar em risco. Portanto, certifique-se de estudar os recursos de gerenciamento de chaves de qualquer produto que você escolha e estabeleça um procedimento que minimize o risco de perda de chaves.

Além de proteger os seus dados contra espionagem, também considere como protegê-los contra desastres. Tire snapshots periódicos dos volumes do Amazon EBS para assegurar que é altamente durável e disponível. Os snapshots são incrementados naturalmente e armazenados no Amazon S3 (separar por geo-localização) e podem ser restaurados com alguns cliques ou chamadas de linha de comando.

Proteger suas credenciais da AWS

A AWS fornece dois tipos de credenciais de segurança: acessar chaves da AWS e certificados x.509. Sua chave de acesso da AWS tem duas partes: sua *ID de chave de acesso* e sua *chave de acesso secreta*. Ao usar o REST ou a API de consulta, você tem que usar sua chave de acesso secreta para calcular uma assinatura para incluir no seu pedido de autenticação. Para evitar a violação em curso, todas as solicitações devem ser enviadas por HTTPS.

[31] http://www.gnupg.org
[32] http://www.pgp.com/
[33] http://www.truecrypt.org/
[34] http://www.arg0.net/encfs
[35] http://loop-aes.sourceforge.net/loop-AES.README
[36] http://www.saout.de/misc/dm-crypt/
[37] http://www.truecrypt.org/
[38] http://www.opensolaris.org/os/community/zfs/

Se sua Amazon Machine Image (AMI) está executando processos que precisam se comunicar com outros serviços web da AWS (sondagem a fila do Amazon SQS ou para ler objetos do Amazon S3, por exemplo), um erro de concepção comum é incorporar as credenciais AWS na AMI. Em vez de incorporadas as credenciais devem ser passadas como argumentos durante o lançamento e a criptografia antes de serem enviados pela conexão [17].

Se sua chave de acesso secreta for comprometida, você deve obter uma nova girando para uma nova[39] ID de chave de acesso secreta. Como uma boa prática, é recomendável que você incorpore um mecanismo de rotação de chaves em sua arquitetura de aplicativo para que você possa usá-lo em uma base regular ou ocasionalmente (quando o funcionário insatisfeito deixa a empresa) para garantir que as chaves comprometidas não possam durar por muito tempo.

Como alternativa, você pode usar os certificados x.509 para autenticação para determinados serviços da AWS. O arquivo de certificado contém sua chave pública em um corpo de certificado DER codificado em base64. Um arquivo separado contém a chave privada correspondente codificado em base64 PKCS#8.

A AWS oferece suporte a autenticação multi-factor[40] como um protetor adicional para trabalhar com as informações de sua conta em aws.amazon.com e AWS Management Console[41].

Gerencie vários usuários e suas permissões com a IAM

O AWS Identity and Access Management (IAM)[42] permite que você crie vários usuários e gerencie permissões para cada um desses usuários a partir de sua conta da AWS. Um usuário é uma identidade (dentro de sua conta da AWS) com credenciais de segurança exclusivas que podem ser usadas para acessar os serviços da AWS. O IAM elimina a necessidade de compartilhar senhas ou chaves de acesso e facilita a ativação e a desativação de acesso de um Usuário, conforme apropriado.

O IAM permite que você implemente melhores práticas de segurança, tais como menor privilégio, conceder credenciais únicas para cada usuário dentro de sua conta AWS e só conceder permissão para acessar os serviços e recursos da AWS necessários para os usuários realizarem seu trabalho Como padrão, o IAM é seguro; os novos usuários não devem acessar os recursos do AWS até que permissões sejam explicitamente concedidas.

A IAM é integrada nativamente nos serviços da AWS. Nenhuma API de serviço foi alterada para oferecer suporte ao IAM e aplicações e ferramentas baseadas em APIs de serviço da AWS que continuarão a funcionar quando o IAM estiver sendo usado. Os aplicativos só precisam começar a usar as chaves de acesso geradas para um novo usuário.

Você deve minimizar o uso de suas credenciais de conta AWS, tanto quanto possíveis quando estiver interagindo com seus serviços AWS e aproveitar as credenciais de usuário IAM para acessar recursos e serviços da AWS.

Proteja seu aplicativo

Cada instância do Amazon EC2 está protegida por um ou mais *Grupos de segurança*[43], chamado conjuntos de regras que especificam em qual ingresso (ou seja, entrada) o tráfego de rede deve ser entregue à sua instância. Você pode especificar portas TCP e UDP, tipos ICMP, códigos e endereços de origem. Os grupos de segurança oferecem proteção básica com um firewall para instâncias em execução. Por exemplo, as instâncias que pertencem a um aplicativo web podem ter as seguintes configurações de grupo de segurança:

---

[39] http://aws.amazon.com/about-aws/whats-new/2009/08/31/seamlessly-rotate-your-access-credentials/
[40] Mais informações sobre o Multi-factor Authentication estão disponíveis em http://aws.amazon.com/pt/mfa/
[41] AWS Management Console http://aws.amazon.com/console/
[42] Mais informações em http://aws.amazon.com.com/iam
[43] Mais informações sobre o grupo de segurança estão disponíveis em http://docs.amazonwebservices.com/AWSEC2/2009-07-15/UserGuide/index.html?using-network-security.html

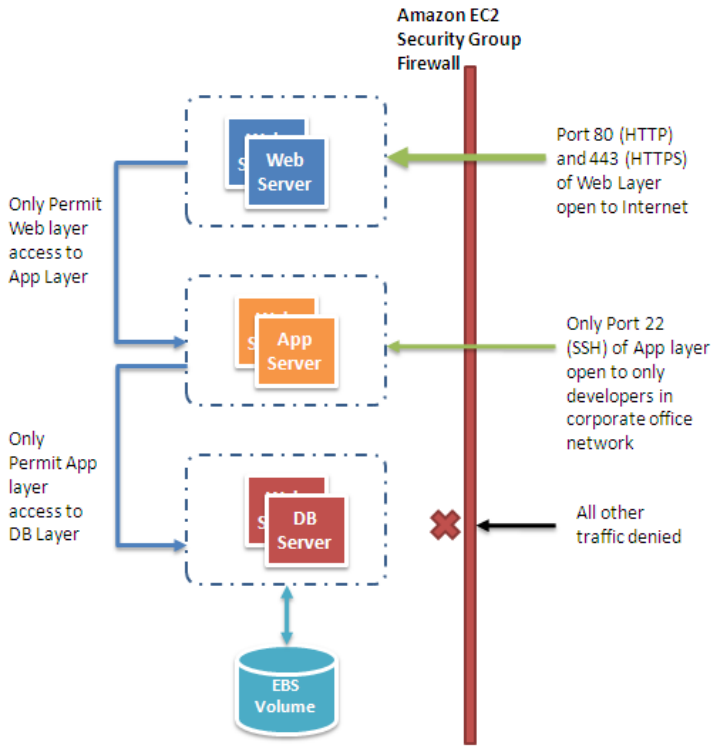

Figura4: Protegendo seu aplicativo web usando os grupos de segurança do Amazon EC2

Outra forma de restringir o tráfego de entrada é configurar firewalls baseados em software em suas instâncias. Instâncias do Windows podem usar o firewall interno[44]. As instâncias do Linux podem usar o *netfilter*[45] e o *iptables*.

Ao longo do tempo, erros no software são descobertos e exigem patches de correção. Certifique-se de seguir as seguintes orientação básicas para maximizar a segurança do seu aplicativo:

- Fazer downloads regularmente dos patches do site do fornecedor e atualizar suas AMIs
- Reimplantar instâncias de AMIs novas e testar os aplicativos para garantir que os patches não corrompem nada. Certifique-se de que a AMI mais recente está implantada em *todas* as instâncias
- Investa em scripts de teste para que você possa executar verificações de segurança periodicamente e automatizar o processo
- Verifique se o software de terceiros está definido com as configurações mais seguras
- Nunca execute seus processos logado como *raiz* ou *Administrador* a menos que seja absolutamente necessário

Toda as práticas segurança padrão da era pré-nuvem como a adoção de boas práticas de codificação, isolamento os dados confidenciais ainda são aplicáveis e devem ser implementadas.

Em contrapartida, a nuvem retira a complexidade da segurança física e lhe dá o controle através de ferramentas e recursos para que você possa proteger o seu aplicativo.

---

[44] http://technet.microsoft.com/en-us/library/cc779199(WS.10).aspx, March 2003
[45] http://www.netfilter.org/

Direções para pesquisas futuras

O dia não é longe demais quando os aplicativos não estão mais ligados ao hardware físico. Da mesma forma que para ligar um microondas não é necessário qualquer conhecimento de electricidade, qualquer um deve ser capaz de *conectar* um aplicativo em nuvem para receber a energia que ele precisa para executar, como um utilitário. Como arquiteto, você irá gerir os recursos abstratos de computação, de armazenamento e de rede em vez de servidores físicos. Os aplicativos continuarão a funcionar mesmo se o hardware físico subjacente falhar, for removido ou substituído. Os aplicativos se adaptarão aos padrões de demanda flutuante por implantação de recursos *instantaneamente* e automaticamente, alcançando assim níveis mais altos de utilização em todos os momentos. Escalabilidade, segurança, alta disponibilidade, tolerância a falhas, capacidade de teste e elasticidade serão propriedades configuráveis da arquitetura do aplicativo e serão uma parte intrínseca e automatizada da plataforma na qual são criadas.

No entanto, nós não estamos lá ainda. Hoje, você pode criar aplicativos em nuvem com algumas dessas qualidades implementando as práticas recomendadas destacadas no artigo. As práticas recomendadas em arquiteturas de computação em nuvem continuarão a evoluir e como pesquisadores, não devemos apenas nos concentrar no aprimoramento da nuvem, mas também na construção de ferramentas, tecnologias e processos que facilitarão para desenvolvedores e arquitetos a conexão de aplicativos em nuvem.

Conclusão

Este artigo forneceu orientações prescritivas para arquitetos de nuvem para criar aplicativos de nuvem eficientes.

Concentrando-se em conceitos e práticas recomendadas, como projetar para falha, dissociação entre os componentes do aplicativo, compreensão e implementação de elasticidade, combinando com a paralelização e a integração de segurança em todos os aspectos da arquitetura do aplicativo, os arquitetos de nuvem podem compreender as considerações de design necessárias para criar aplicativos de nuvem altamente escaláveis.

A nuvem da AWS oferece serviços de infraestrutura altamente confiáveis e você só paga pelo o que usar. As táticas específicas da AWS destacadas no artigo ajudarão no desenvolvimento de aplicativos em nuvem usando esses serviços. Como pesquisador, é aconselhável que você utilize estes serviços comerciais, aprenda com o trabalho dos outros, amplie, melhore e invente computação em nuvem.

Agradecimentos

O autor é profundamente grato a Jeff Barr, Steve Riley, Paul Horvath, Prashant Sridharan e Marvin Scot por fornecerem comentários sobre os primeiros rascunhos do presente artigo. Agradecimento especial a Matt Tavis por fornecer informações valiosas. Sem as suas contribuições, o artigo não teria sido possível.

Parte do conteúdo deste whitepaper é trecho de um capítulo escrito pelo mesmo autor que aparece no livro 'Cloud Computing: Paradigms and Patterns', Copyright © 2010 John Wiley & Sons, Inc.

Referências e leitura complementar

1. **Amazon S3 Team, Práticas recomendadas para uso do Amazon S3,** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1904, 2008-11-26
2. **Amazon S3 Team, Amazon S3 Error Best Practices,** http://docs.amazonwebservices.com/AmazonS3/latest/index.html?ErrorBestPractices.html, 2006-03-01
3. **Amazon SimpleDB Team, Query 201: Tips and Tricks for Amazon SimpleDB Query,** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1232&categoryID=176, 2008-02-07
4. **Amazon SimpleDB Team, Building for Performance and Reliability with Amazon SimpleDB,** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1394&categoryID=176, 2008-04-11
5. **Amazon SimpleDB Team, Query 101: Building Amazon SimpleDB Queries,** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=1231&categoryID=176, 2008-02-07
6. **J. Varia, Cloud Architectures,** http://jineshvaria.s3.amazonaws.com/public/cloudarchitectures-varia.pdf, 2007-07-01
7. **Amazon Security Team, Overview of Security Processes,** http://awsmedia.s3.amazonaws.com/pdf/AWS_Security_Whitepaper.pdf, 2009-06-01
8. **Amazon Web Services Team, Creating HIPPA-Compliant Medical Data Applications With AWS,** http://awsmedia.s3.amazonaws.com/AWS_HIPAA_Whitepaper_Final.pdf, 2009-04-01
9. D. Obasanjo, Building Scalable Databases: Pros and Cons of Various Database Sharding Schemes, http://www.25hoursaday.com/weblog/2009/01/16/BuildingScalableDatabasesProsAndConsOfVariousDatabaseShardingSchemes.aspx, 2009-01-16
10. D. Pritchett, Shard Lessons, http://www.addsimplicity.com/adding_simplicity_an_engi/2008/08/shard-lessons.html, 2008-08-24
11. J. Hamilton, On Designing and Deploying Internet-Scale Services, 2007, *21st Large Installation System Administration conference (LISA 'O7)*, http://mvdirona.com/jrh/talksAndPapers/JamesRH_Lisa.pdf
12. J. Dean and S. Ghemawat, MapReduce: Simplified data processing on large clusters2004-12-01, In Proc. of the 6th OSDI, http://labs.google.com/papers/mapreduce-osdi04.pdf
13. T. Schlossnagle, *Scalable Internet Architectures*, Sams Publishing, 2006-07-31,
14. M. Lurie, The Federation: Database Interoperability, http://www.ibm.com/developerworks/data/library/techarticle/0304lurie/0304lurie.html, 2003-04-23
15. E. Hammond, Escaping Restrictive/Untrusted Networks with OpenVPN on EC2, http://alestic.com/2009/05/openvpn-ec2, 2009-05-02
16. R. Bragg, The Encrypting File System, http://technet.microsoft.com/en-us/library/cc700811.aspx, 2009
17. S. Swidler, How to keep your AWS credentials on an EC2 instance securely, http://clouddevelopertips.blogspot.com/2009/08/how-to-keep-your-aws-credentials-on-ec2.html, 2009-08-31
18. **Amazon SQS Team, Building Scalable, Reliable Amazon EC2 Applications with Amazon SQS,** http://sqs-public-images.s3.amazonaws.com/Building_Scalabale_EC2_applications_with_SQS2.pdf, 2008
19. Microsoft Support Team, Best Practices For Encrypting File System (Windows), http://support.microsoft.com/kb/223316, 2009
20. Solaris Security Team, ZFS Encryption Project (OpenSolaris), http://www.opensolaris.org/os/project/zfs-crypto/, 2009-05-01

21. **Amazon RDS Team, Amazon RDS Multi-AZ Deployments,** http://docs.amazonwebservices.com/AmazonRDS/latest/DeveloperGuide/Concepts.DBInstance.html#Concepts.MultiAZ. 2010-05-15

22. **Amazon SimpleDB Team, Amazon SimpleDB Consistency Enhancements** http://developer.amazonwebservices.com/connect/entry.jspa?externalID=3572 2010-02-24