Alta Disponibilidade: Demandas de Disponibilidade AWS

Resumo

Referência bibliográfica

- AWS. Amazon Web Services. **Computação em nuvem com a AWS**. 2021. Disponível em: <https://aws.amazon.com/pt/what-is-aws/> Acesso em: 07/05/2022.
- AWS. **Pilar Confiabilidade**: AWS Well-Architected Framework. 2020. Disponível em: <https://docs.aws.amazon.com/pt_br/wellarchitected/latest/reliability-pillar/wellarchitected-reliability-pillar.pdf#plan-for-disaster-recovery-dr> Acesso em: 07/05/2022.

Compreensão da Necessidade de Disponibilidade

Para que a alta disponibilidade seja corretamente implementada deve ser aprofundada, ou seja, não será eficiente que seja tratada em um contexto geral, devendo ser abordada de forma profunda avaliando os recursos de cada sistema. Assim como usuários e aplicações apresentam diferentes requisitos que impactam na alta disponibilidade, dentro de cada aplicação, seus diversos recursos e funções vão variar em sua demanda por recursos,

> Por exemplo, alguns sistemas podem priorizar a capacidade de receber e armazenar novos dados antes de recuperar dados existentes. Outros sistemas priorizam operações em tempo real a operações que mudam a configuração ou o ambiente de um sistema. Os serviços podem ter requisitos de disponibilidade muito altos durante determinados horários do dia, mas podem tolerar períodos muito mais longos de interrupção fora desses horários. (AWS 2020, p .07).

Desta forma, aplicativos e sistemas devem ser desconstruídos para que cada uma de suas funções e recursos sejam avaliados em termos de suas necessidades de largura de banda, memória, armazenamento, tempo de resposta, ping e outros parâmetros de rede. Desta forma os esforços e o custo da implementação da alta disponibilidade são focados, mais assertivos e podem ser concluídos mais rapidamente.

Podemos refletir melhor nos benefícios desta aplicação mais detalhada em realidades como a de provedores de recursos em nuvem como a própria AWS, que apresenta mais de 200 serviços diferentes, milhões de clientes e datacenters (zonas de disponibilidade) espalhados pelo mundo. Neste contexto AWS a implementação feita para a Alta Disponibilidade demandou este detalhamento, como forma de otimizar o investimento em sua infraestrutura. Podemos compreender a estruturação da AWS em suas ações como nos planos de controle e plano de dados:

> Na AWS, normalmente dividimos os serviços em “plano de dados” e “plano de controle”. O plano de dados é responsável por prestar serviço em tempo real, enquanto os planos de controle são usados para configurar o ambiente. Por exemplo, instâncias do Amazon EC2, bancos de dados do Amazon RDS e operações de leitura/gravação de tabelas do Amazon DynamoDB são operações de plano de dados. (AWS 2020, p. 07).

Podemos perceber que a abordagem da análise minuciosa da alta disponibilidade de cada componente de um sistema é feita também pelos clientes dos serviços em nuvem, pois eles também buscam a alta disponibilidade, mas cuidam para que sua implementação tenha o menor custo, avaliando criticamente o uso de cada recurso como os oferecidos pela AWS. É uma relação de custo e benefício, pois demanda recursos ao mesmo tempo em que precisa controlar o consumo de tais recursos, conforme defende AWS (2020, p. 08)

> Muitos dos clientes da AWS adotam uma abordagem similar para avaliar de modo crítico os aplicativos e identificar subcomponentes com diferentes necessidades de disponibilidade. As metas de design de disponibilidade então são elaboradas para os diferentes aspectos, e os esforços de trabalho adequados são executados para projetar o sistema. A AWS tem experiência significativa no desenvolvimento de aplicativos com uma variedade de metas de design de disponibilidade, incluindo serviços com 99,999% de disponibilidade ou mais. Os arquitetos de soluções (SAs) da AWS podem ajudá-lo a projetar adequadamente conforme suas metas de disponibilidade.

Portanto a visão que a empresa consumidora de recursos em nuvem deve ter deste detalhamento de alta disponibilidade tem relação com a criticidade dos recursos que demanda, ou

seja, alocar mais onde é critico para que seu desempenho seja otimizado, e manter em níveis modestos os demais recursos como forma de controlar gastos.

Planejamento da Topologia de Rede Para a Alta Disponibilidade

Por topologia podemos compreender que define a posição, a ordem com que os dispositivos da rede são conectados, e isso indica a existência de diversas estratégias que podem ser elencadas para que este processo também facilite na implementação da alta disponibilidade. A distribuição física dos componentes indica onde estarão alguns recursos, o que promove a existência de volumes de recursos em proporção diferente em alguns locais, o que impacta na carga de trabalho,

> Muitas vezes, as cargas de trabalho estão presentes em vários ambientes. Dentre eles estão vários ambientes de nuvem (acessíveis publicamente e privados) e possivelmente sua infraestrutura de datacenter existente. Os planos devem incluir considerações de rede, como conectividade dentro dos sistemas e entre eles, gerenciamento de endereços IP públicos e privados e resolução de nomes de domínio. (AWS 2020, p.11).

A topologia faz com que o processo de alocação dos endereços IP dos componentes e computadores tenha algumas especificidades, o que impacta no seu planejamento e também na forma com que serão posteriormente analisadas as falhas da rede. Um recurso que ajuda nesta questão é a aplicação das redes privadas na nuvem, um dos recursos oferecidos pela AWS, o AWS VPC

> A Amazon Virtual Private Cloud (Amazon VPC) permite provisionar uma seção isolada e privada da nuvem AWS, onde é possível executar recursos da AWS em uma rede virtual. Use conectividade de rede altamente disponível em seus endpoints públicos de carga de trabalho: esses endpoints e o roteamento para eles devem ser altamente disponíveis. Para que isso seja possível, use DNS altamente disponível, Redes de entrega de conteúdo (CDNs), API Gateway, balanceamento de carga ou proxies reversos. (AWS 2020, p.11).

A oferta dos *endpoints* públicos pode ser feita em diversos recursos da AWS, como “O Amazon Route 53, o AWS Global Accelerator, o Amazon CloudFront, o Amazon API Gateway e o Elastic Load Balancing (ELB). (AWS 2020, p.11). Portanto, com sistemas como estes e a topologia avaliada, fica simplificado o processo de balanceamento de carga, outro importante recurso da alta disponibilidade.

Para a AWS é importante a oferta de *endpoints* de alta disponibilidade onde são elencados recursos como o *Elastic Load Balancing* que faz o balanceamento de carga pulverizando recursos nas zonas de disponibilidade de uma região. Este recurso usa o roteamento TCP na camada 4 ou simplesmente na camada 7 de apresentação com o protocolo HTTP e integra o AWS Auto Scaling. Com este último ele passa a oferecer a criação de uma infraestrutura que se autorrepara e é capaz de lidar eficientemente com o aumento de tráfego e liberação de recursos conforme determina o conceito de elasticidade. E como parte da topologia e configurações de endereçamento da rede a AWS oferece o Route 53,

> [...] serviço de Domain Name System (DNS) escalável e altamente disponível que conecta as solicitações de usuários à infraestrutura em execução na AWS, como instâncias do Amazon EC2, load balancers do Elastic Load Balancing ou buckets do Amazon S3. Além disso, também pode ser usado para direcionar os usuários para a infraestrutura fora da AWS. (AWS 2020, p.11).

Embora seja diferente da topologia em redes locais, a topologia da nuvem apresenta nos recursos do provedor a possibilidade de ser altamente disponível, algo impensado em redes que dependem de sua conexão física para que isso seja garantido. Cabeamento é um processo que apresenta certa simplicidade o que induz a certo abandono em projetos de redes o que causa interrupções nos serviços, mas que pode ser contornado quando em uma estrutura de rede.

Arquitetura Orientada a serviço para a Carga de Trabalho

A alta disponibilidade pode ser ampliada com o apoio de conceitos como a SOA (Service-Oriented Architecture - Arquitetura orientada por serviços) ou a arquitetura de microsserviços, pois ambas permitem oferecer cargas de trabalho de elevada confiabilidade e escalabilidade. Um dos fatores que mais contribuem para estes dois atributos é a capacidade destes serviços de serem altamente reutilizáveis através de interfaces como as API´s e, no caso dos microsserviços, existe o benefício de que são criados para serem simples e leves, algo visto em prática na infraestrutura da AWS:

> As interfaces SOA usam padrões de comunicação comuns para que possam ser rapidamente incorporadas a novas cargas de trabalho. A SOA substituiu a prática de construção de arquiteturas monolíticas, que consistiam em unidades interdependentes e indivisíveis. Na AWS, sempre

> usamos a SOA, mas agora adotamos a criação de nossos sistemas usando microsserviços. Embora os microsserviços tenham várias qualidades interessantes, o principal benefício para disponibilidade é que eles são menores e mais simples. Eles permitem diferenciar a disponibilidade exigida de diferentes serviços e, portanto, concentrar os investimentos mais especificamente nos microsserviços que têm as maiores necessidades de disponibilidade.

A própria infraestrutura AWS usa este conceito de microsserviços, como quando exibe ao usuário os detalhes de um produto, algo feito com centenas destes serviços. Este detalhe permite que os principais microsserviços, como os que exibem a precificação do produto, sejam altamente disponíveis, enquanto outros menos essenciais possam ser excluídos. Como falar de microsserviços é falar em não aplicar a arquitetura monolítica, podemos deixar aqui a sugestão de evitar esta segunda em favorecimento da primeira, conforme defende AWS (2020, p. 14)

> Escolha como segmentar a carga de trabalho: a arquitetura monolítica deve ser evitada. Em vez dela, escolha entre SOA e microsserviços. Ao fazer cada escolha, analise os benefícios em relação às complexidades. O que é ideal para um novo produto a caminho do seu primeiro lançamento não se aplica a uma carga de trabalho que foi criada para escalabilidade a partir das necessidades iniciais. Os benefícios de usar segmentos menores incluem maior agilidade, flexibilidade organizacional e escalabilidade. As complexidades incluem possível maior latência, depuração mais complexa e maior carga operacional.

Nos casos onde é inevitável produzir algo monolítico, tal sistema deve ser construído com a capacidade de ser modular e assim poder ser futuramente convertido para SOA ou microsserviço, o que deve ser uma necessidade assim que o volume de requisições começar a exigir que sejam provisionados novos recursos ao sistema monolítico, algo custoso e complexo, pois,

> A SOA e os microsserviços oferecem respectivamente segmentação menor, que é preferida por ser uma arquitetura moderna escalável e confiável. Porém, existem compensações a serem consideradas especialmente ao implantar uma arquitetura de microsserviços. Uma delas é que você agora tem uma arquitetura de computação distribuída que pode dificultar o cumprimento de requisitos de latência do usuário final, e existe uma complexidade adicional na depuração e no rastreamento de interações. (AWS 2020, p.14).

Arquiteturas como SOA e microsserviços facilitam na criação de sistemas distribuídos, algo que até pode ser realizado no monolítico, mas de implementação inviável em projetos de custo reduzido ou com equipes pequenas.

Os Sistema Distribuídos e as formas de se evitar, mitigar e resistir a falhas

Construir um sistema distribuído coloca o projeto na dependência das redes de comunicação de forma mais intensa que outras opções de implementação, mas oferecem a vantagem de operar de forma eficiente na distribuição da carga de trabalho e na forma com que lidam com perdas de pacotes ou variações de latência. Quando sistemas distribuídos são utilizados existe ganho na alta disponibilidade pela melhora de indicadores como o MTBF (*Mean Time Between Failures* - Tempo médio entre falhas). Desta forma, AWS complementa afirmando que,

> [...] sistemas distribuídos em tempo real rígidos exigem respostas síncronas e rápidas, enquanto os sistemas em tempo real flexíveis têm uma janela de tempo para resposta maior, de minutos ou mais. Os sistemas off-line gerenciam as respostas por meio do processamento em lote ou assíncrono. Os sistemas distribuídos em tempo real rígidos têm os requisitos de confiabilidade mais rigorosos. (AWS 2020, p. 15).

Mas sistemas distribuídos apresentam alguma complexidade, principalmente se tal distribuição for feita em tempo real, nos denominados serviços de solicitação/resposta, pois constantemente ocorrem solicitações inesperadas o que força respostas rápidas como as que ocorrem na infraestrutura de cobrança de cartões de crédito, por exemplo, e neste sentido, AWS (2020, p. 15), afirma que,

> Implementar dependências com acoplamento fraco: dependências, como sistemas de enfileiramento, sistemas de streaming, fluxos de trabalho e load balancers, têm acoplamento fraco. O baixo acoplamento ajuda a isolar o comportamento de um componente de outros componentes que dependem dele, aumentando a resiliência e a agilidade. (AWS 2020, p. 15).

O foco aqui é evitar o acoplamento dos componentes, deixando o acoplamento apenas para a interface que chama estes serviços. De acordo com AWS (2020, p. 15), esta implementação com baixo acoplamento entre dependências permite "isolar uma falha em uma dependência para não afetar a outra. O baixo acoplamento permite adicionar mais código ou recursos a um componente enquanto minimiza o risco para componentes que dependem dele." Este recurso melhora a escalabilidade geral do sistema.

O baixo acoplamento é um importante atributo das aplicações distribuídas e o faz justamente por prevenir que serviços dependam uns dos outros, o que permite uma escalabilidade ampliada e individualizada às necessidades de cada sistema. Mas ações devem ser tomadas nos momentos onde os microsserviços precisam se comunicar, de forma a não prejudicar a resiliência, outro componente da alta disponibilidade.

> Para melhorar ainda mais a resiliência por meio do baixo acoplamento, torne as interações de componentes assíncronas sempre que possível. Esse modelo é adequado para qualquer interação que não precise de uma resposta imediata e em que uma confirmação de que uma solicitação foi registrada será suficiente. Envolve um componente que gera eventos e outro que os consome. Os dois componentes não se integram por meio de interação direta ponto a ponto, mas geralmente por meio de uma camada de armazenamento durável intermediária, como uma fila do SQS ou uma plataforma de dados de streaming, como o Amazon Kinesis ou o AWS Step Functions. (AWS 2020, p. 15).

Aqui apresentamos mais um recurso capaz de aprimorar aplicações distribuídas e seu impacto na alta disponibilidade: a criação de interações assíncronas, para que exista um mediador capaz de eficientemente conectar os recursos que não são conectados diretamente.

Exercícios

1. A disponibilidade de um aplicativo, normalmente é considerada como um objetivo único, ou seja, considera o aplicativo como um todo. Entretanto, neste tipo de análise não são consideradas os diferentes aspectos deste aplicativo, e que podem apresentar requisitos de disponibilidade diferentes. Dessa forma, é necessário que este aplicativo seja dividido em partes, de modo a analisar quais são os requisitos de disponibilidade de cada uma delas.
   Neste contexto, qual os benefícios desta divisão de aplicativos em partes para a promoção da alta disponibilidade?

   a) A possibilidade de reconstrução deste aplicativo de um modo mais redundante.
   b) A concentração de esforços e custos de disponibilidade de acordo com as necessidades específicas.
   c) A possibilidade de análise de quais são os momentos de indisponibilidade de cada parte do aplicativo.
   d) A possibilidade de priorizar todo o sistema de acordo com requisitos mais rígidos.
   e) Concentrar esforços e despesas na análise do aplicativo de forma global.

2. A AWS (Amazon Web Services) é um serviço de computação em nuvem desenvolvido que foi desenvolvido pela empresa Amazon para oferecer serviços computacionais sob demanda com custos otimizados e alta disponibilidade. Os inúmeros serviços disponibilizados pela AWS proporcionam para as organizações, a possibilidade de usufruir de recursos computacionais de alta tecnologia, dentre eles, o Amazon Virtual Cloud (Amazom VPC). Qual a funcionalidade do Amazon Virtual Cloud (Amazon VPC) para o planejamento de uma rede de alta disponibilidade?

   a) Oferece balanceamento de carga entre as zonas de disponibilidade, executando o roteamento da camada 4 (TCP).
   b) Conecta as solicitações dos usuários a infraestrutura que está em execução na AWS.
   c) Provisiona uma seção isolada e privada da nuvem AWS para a execução de recursos AWS em uma rede virtual.
   d) Utilizado para realizar o direcionamento de tráfego para endpoints ideias pela rede global AWS.
   e) Fornece proteção automática contra ataques de negação sem custos adicionais para endpoints de serviços AWS.

3. As arquiteturas orientadas a serviços (SOA) e a arquitetura de microsserviços são utilizadas para criar cargas de trabalho altamente escaláveis e confiáveis. A plataforma AWS (Amazon Web Services), se utiliza de microsserviços na criação de seus sistemas, por oferecerem importantes benefícios para disponibilidade. Assinale a alternativa que apresente quais são os benefícios para a disponibilidade na utilização de microsserviços na criação de sistemas computacionais:

    a) Permite diferenciar a disponibilidade que é exigida por diferentes serviços.
    b) Permite a concentração de investimentos nos microsserviços de menor necessidade de disponibilidade.
    c) Os microsserviços tornam os componentes maiores e com estrutura e funcionalidade mais complexa.
    d) Por tornarem componentes de softwares reutilizáveis por meio de interfaces de serviços.
    e) Por usar padrões de comunicação comuns de forma a serem incorporados de forma fácil nas cargas de trabalho.

4. A caracterização de cargas de trabalho, dentro de um sistema computacional, tem como principal objetivo a melhoria de desempenho dos seus sistemas. Conceituada como a capacidade dos sistemas computacionais de manipular e realizar o processamento do trabalho, que são enviadas para os determinados componentes como servidores, que recebem esta carga esperada em sua concepção. A segmentação da carga de trabalho, representa a distribuição das cargas determinadas para os dispositivos devidos. Na escolha de como segmentar a carga de trabalho, qual arquitetura deve ser evitada:

    a) Arquitetura Monolítica.
    b) Arquitetura de microsserviços.
    c) Arquitetura Orientada a serviço.
    d) Arquitetura de carga.
    e) Arquitetura em camadas.

5. Os sistemas distribuídos representam sistemas que possuem componentes que se localizam em dispositivos, como computadores, que estão conectados em rede que estabelecem conexão por meio de troca de mensagens entre seus componentes, tendo a internet como principal representante desse tipo de sistema. Os sistemas distribuídos, apresentam as dependências, como sistemas de enfileiramento, sistemas de streaming, fluxos de trabalho e load balancers, e que encontram grandes

benefícios com a implementação de baixos acoplamentos. Qual benefício para os sistemas distribuídos, da implementação de um baixo acoplamento entre dependências?

a) O baixo acoplamento promove a dependência de componentes.
b) O isolamento de uma falha em uma dependência de modo que a outra não seja afetada.
c) Auxilia na difusão de uma falha em diversas dependências, promovendo seu controle.
d) Permite remover códigos ou recursos de um componente
e) Potencializa os riscos para componentes que dependem de componentes que tiveram códigos adicionados.

6. Em um sistema computacional, a resiliência representa a capacidade destes sistemas manter sus funcionalidade durante eventos extremos, podendo ainda se recuperar, voltando a operar de forma normal após este evento. Mesmo em sistema resilientes, ainda é possível promover melhorias através do baixo acoplamento de componentes, auxiliando assim na construção de um sistema ainda mais eficiente. Como é possível promover melhorias na resiliência através do baixo acoplamento?

a) Removendo recursos a um componente, minimizando os riscos do componente dependente.
b) Diminuindo a implementação adjacente da dependência entre componentes.
c) Sempre que possível é necessário tornar a interação dos componentes assíncronas.
d) É necessário que obrigatoriamente a interação dos componentes seja simultânea.
e) É necessário que os componentes interajam indiretamente.

Gabarito

1. **Letra B.**

   A alternativa está correta, pois dividir aplicativos em partes de forma que estas partes sejam analisadas suas necessidades de disponibilidade separadamente, permitem que esforços e custos de disponibilidade (que não são baixos) sejam concentrados de acordo com necessidades específicas, sendo melhor aproveitados.

2. **Letra C.**

   Alternativa está correta, pois o Amazon Virtual Cloud (Amazon VPC) permite a inicialização de recursos AWS em uma rede virtual provisionada pela própria AWS.

3. **Letra A.**

   Alternativa está correta, pois os microsserviços possibilitam a diferenciação da disponibilidade que é exigida por cada serviço, permitindo a concentração de investimentos nos microsserviços que possuem uma maior necessidade de disponibilidade.

4. **Letra A.**

   Alternativa está correta, pois a arquitetura monolítica, diferentemente das arquiteturas SOA e de microsserviços não oferecem segmentação menor, pois é único e não pode ser dividido, entretanto é até possível iniciar com uma arquitetura monolítica, desde que ela seja modular e tenha a capacidade de evoluir para SOA ou microsserviços.

5. **Letra B.**

   A alternativa está correta, pois a implementação de um baixo acoplamento entre dependências, como sistemas de enfileiramento, sistemas de streaming, fluxos de trabalho entre outros, pode isolar possíveis falhas em uma determinada dependência, para que outra não seja afetada.

6. **Letra C.**

   A Alternativa está correta, pois como forma de melhorar ainda mais a resiliência através do baixo acoplamento, é necessário tornar as interações dos componentes assíncronas, ou seja, não simultâneas, sendo o modelo adequado para interações que não necessitem de uma resposta imediata.