WordPress: melhores práticas na AWS

Arquitetura de referência para websites escaláveis com WordPress

Fevereiro de 2018

Avisos

Este documento é fornecido apenas para fins informativos. Ele relaciona as atuais ofertas de produtos e práticas da AWS na data de emissão deste documento, que estão sujeitas a alterações sem aviso prévio. Os clientes são responsáveis por fazer sua própria avaliação independente das informações neste documento e de qualquer uso dos produtos ou serviços da AWS, cada um dos quais é fornecido “como está”, sem garantia de qualquer tipo, expressa ou implícita. Este documento não cria quaisquer garantias, representações, compromissos contratuais, condições ou promessas da AWS, suas afiliadas, fornecedores ou licenciadores. As responsabilidades e obrigações da AWS para com seus clientes são controladas por contratos da AWS, e este documento não modifica nem faz parte de qualquer contrato entre a AWS e seus clientes.

Sumário

Resumo

Este whitepaper fornece aos administradores de sistema orientações específicas sobre como começar a usar o WordPress na AWS e como melhorar tanto a eficiência de custos da implantação quanto a experiência do usuário final. Ele também descreve uma arquitetura de referência que aborda os requisitos comuns de escalabilidade e alta disponibilidade.

Introdução

O WordPress é uma ferramenta de código aberto e sistema de gerenciamento de conteúdo (CMS) baseado em PHP e MySQL que é usado para alimentar desde blogs pessoais até sites de alto tráfego.

A primeira versão do WordPress foi lançada em 2003, e não foi construída com modernas infraestruturas baseadas em nuvem elásticas e escaláveis. Através do trabalho da comunidade WordPress e do lançamento de vários módulos do WordPress, os recursos desta solução CMS estão em constante expansão. Hoje é possível criar uma arquitetura do WordPress que aproveita muitos dos benefícios da Nuvem AWS.

Implantação simples

Para blogs ou sites com pouco tráfego, sem requisitos rigorosos de alta disponibilidade, uma implantação simples de um único servidor pode ser adequada. Esta implantação não é a arquitetura mais resiliente ou escalonável, mas é a maneira mais rápida e econômica de colocar seu site em funcionamento.

Considerações

Vamos começar com a implantação de um único servidor web. Pode haver ocasiões em que você a supere, por exemplo:

- A máquina virtual em que seu site WordPress é implantado será um ponto único de falha. Um problema com essa instância causará uma perda de serviço para o seu site.
- O dimensionamento de recursos para melhorar o desempenho só pode ser alcançado com o “dimensionamento vertical”, ou seja, aumentando o tamanho da máquina virtual que executa o site do WordPress.

Abordagens disponíveis

A AWS tem um número de diferentes opções para o provisionamento de máquinas virtuais. Existem três maneiras principais para hospedar seu próprio site do WordPress na AWS:

- Amazon Lightsail
- Amazon Elastic Compute Cloud (Amazon EC2)
- AWS Marketplace

O Amazon Lightsail é um serviço que permite que você inicie rapidamente um servidor virtual privado (uma instância do Lightsail) para hospedar um site do WordPress.[1] Lightsail é a maneira mais fácil de começar, se você não precisar de tipos de instâncias altamente configuráveis ou acesso a recursos avançados de rede.

O Amazon EC2 é um serviço da Web que fornece uma capacidade de computação redimensionável para que você possa iniciar um servidor virtual em minutos.[2] O Amazon EC2 fornece mais opções de configuração e gerenciamento do que o Lightsail, que é desejável em arquiteturas mais avançados. Você tem acesso administrativo para suas instâncias EC2 e pode instalar todos os pacotes de software que escolher, incluindo o WordPress.

O AWS Marketplace é uma loja online onde você pode encontrar, comprar e implementar rapidamente softwares que são executados na AWS. [3] Você pode usar a implantação do 1-Click para lançar imagens pré-configuradas do WordPress diretamente no Amazon EC2 em sua própria conta da AWS em apenas alguns minutos. Há vários fornecedores do Marketplace oferecendo instâncias do WordPress prontas para execução.

Abordaremos a opção Lightsail como a implementação recomendada para um site WordPress de servidor único.

Amazon Lightsail

O Lightsail é a maneira mais fácil de começar na AWS, para desenvolvedores, pequenas empresas, alunos e outros usuários que precisam de uma solução simples de servidor virtual privado (VPS).

O serviço abstrai muitos dos elementos mais complexos do gerenciamento de infraestrutura do usuário. É, portanto, um ponto de partida ideal se você tiver menos experiência em infraestrutura ou quando precisar se concentrar em executar seu website e um produto simplificado for suficiente para suas necessidades.

Com o Amazon Lightsail, você pode escolher sistemas operacionais Windows ou Linux/Unix e aplicativos populares da Web, incluindo o WordPress, e implantá-los com um único clique a partir de modelos pré-configurados.

À medida que suas necessidades crescem, você terá a capacidade de sair com tranquilidade dos limites iniciais e se conectar a serviços adicionais de banco de dados, armazenamento de objetos, armazenamento em cache e distribuição de conteúdo da AWS.

Selecionando um plano de preços do Amazon Lightsail

Um plano do Lightsail define o custo mensal dos recursos do Lightsail que você usará para hospedar seu site WordPress. Há vários planos disponíveis para cobrir uma variedade de casos de uso, com vários níveis de recursos de CPU, memória, armazenamento em unidade de estado sólido (SSD). e transferência de dados.[4] Se o seu site é complexo pode ser necessário uma instância maior com mais recursos. Você pode fazer isso, migrando seu servidor para um plano maior usando o console da web[5] ou conforme descrito na documentação da Amazon Lightsail CLI.[6]

Instalação do WordPress

O Lightsail fornece modelos para aplicativos comumente usados, como o WordPress. Este modelo é um excelente ponto de partida para executar o seu próprio site WordPress, já que vem pré-instalado com a maioria dos softwares que você precisa. Você pode instalar software adicional ou personalizar a configuração do software usando o terminal do navegador ou o seu próprio cliente SSH, ou por meio da interface da Web de administração do WordPress. Para obter mais informações sobre como gerenciar o WordPress no Lightsail, consulte o Guia da iniciação do WordPress na documentação da sua instância do Amazon Lightsail.[7] Quando terminar de personalizar seu site WordPress, recomendamos que você tire um snapshot da sua instância.

Um snapshot é uma maneira de criar uma imagem de backup da instância da Lightsail.[8] Ele é uma cópia do disco do sistema e também armazena a configuração original da máquina (memória, CPU, tamanho do disco e taxa de transferência de dados). Os snapshots podem ser usados para reverter para uma configuração válida após uma implantação ou atualização incorreta.

Esse snapshot permitirá que você recupere seu servidor, se necessário, mas também para iniciar novas instâncias com as mesmas personalizações.

Recuperação de falha

Um único servidor da Web é um ponto único de falha; portanto, você deve garantir o backup dos dados do seu site. O mecanismo de snapshot descrito anteriormente também pode ser usado para essa finalidade. Para fazer a recuperação de falha, você pode restaurar uma nova instância a partir do snapshot mais recente. Para reduzir a quantidade de dados que podem ser perdidos durante uma restauração, seus snapshots devem ser o mais recente possível.

Para minimizar o potencial de perda de dados, certifique-se de que os snapshots estão sendo tirados regularmente. Isso pode ser feito ao automatizar snapshots usando a solução do Lightsail Auto Snapshots.[9]

Recomendamos que você use um endereço de IP estático - um endereço de IP público fixo que seja dedicado à sua conta do Lightsail. Se você precisar substituir a instância por outra, poderá reatribuir o IP estático para a nova instância. Dessa forma, você não precisa reconfigurar todos os sistemas externos (como registros DNS) para apontar para um novo endereço IP, toda vez que desejar substituir sua instância.

Como melhorar o desempenho e a eficiência de custos

Você poderá, por fim, superar sua implantação de servidor único. Você precisará considerar opções para melhorar o desempenho do seu site. Antes de migrar para uma implantação escalonável e com vários servidores - conforme discutiremos mais adiante neste documento -, há vários desempenhos e eficiências de custo

que você poderá aplicar. Estas são boas práticas que você deve seguir de qualquer maneira, mesmo se mudar para uma arquitetura multisservidor.

As seções a seguir apresentam várias opções que podem melhorar aspectos do desempenho e escalabilidade do website do WordPress. Alguns podem ser aplicados a uma implantação de servidor único, enquanto muitos aproveitam a escalabilidade de servidores múltiplos. Muitas dessas modificações exigem o uso de um ou mais plugins do WordPress. Embora várias opções estejam disponíveis, W3 Total Cache é uma escolha popular que combina muitas dessas modificações em um único plugin.[10]

Acelerar a entrega de conteúdo

Qualquer site WordPress precisa fornecer uma mistura de conteúdo estático e dinâmico. O conteúdo estático inclui imagens, arquivos JavaScript ou folhas de estilo. O conteúdo dinâmico inclui qualquer coisa gerada no lado do servidor usando o código PHP do WordPress, por exemplo, elementos de seu site que são gerados do banco de dados ou personalizados para cada visualizador. Um aspecto importante da experiência do usuário final é a latência de rede envolvida ao entregar conteúdo anterior para os usuários em todo o mundo. Acelerar a entrega de conteúdo anterior melhora a experiência do usuário final, especialmente os usuários distribuídos geograficamente por todo o mundo. Isso pode ser obtido com uma rede de entrega de conteúdo (CDN), como o Amazon CloudFront.

O Amazon CloudFront é um serviço da Web que fornece uma maneira fácil e econômica para distribuir conteúdo com baixa latência e alta velocidade de transferência de dados por meio de vários pontos de presença em todo o mundo.[11] As solicitações de visualizadores são encaminhadas automaticamente para um ponto adequado do CloudFront para reduzir a latência de localização de borda.[12] Se o conteúdo puder ser armazenado em cache (por alguns segundos, minutos ou até mesmo dias) e já estiver armazenado em uma determinada localização de borda, o CloudFront o entrega imediatamente. Se o conteúdo não deve ser armazenado em cache, expirou ou não está nesse local de borda, o CloudFront recupera o conteúdo de uma ou mais fontes de verdade, referido como a origem (neste caso, a instância do Lightsail) na configuração do CloudFront. Essa recuperação ocorre em conexões de rede otimizadas, que funcionem para acelerar a entrega de conteúdo em seu site. Além de melhorar a experiência do usuário final, o modelo que discutimos também reduz a carga nos servidores de origem e tem o potencial de gerar economias significativas de custos.

Descarregamento de conteúdo estático

Isso inclui CSS, JavaScript e arquivos de imagem - aqueles que fazem parte dos seus temas do WordPress ou os arquivos de mídia enviados pelos administradores de conteúdo. Todos esses arquivos podem ser armazenados no Amazon Simple Storage Service (Amazon S3) usando um plug-in como o W3 Total Cache e servidos aos usuários de maneira escalável e altamente disponível. O Amazon S3 oferece uma infraestrutura de armazenamento de dados altamente escalável, confiável e de baixa latência a um custo muito baixo, que é acessível por meio de APIs REST.[13] Da Amazon S3 armazena seus objetos de forma redundante, não apenas em vários dispositivos, mas também em várias instalações em uma região do Amazon S3, proporcionando assim níveis excepcionalmente altos de durabilidade.

Isso tem o efeito colateral positivo de descarregar essa carga de trabalho de sua instância do Lightsail e permitir que ela se concentre na geração de conteúdo dinâmico. Isso reduz a carga no servidor. Mais adiante neste documento, também veremos que esse é um passo importante para criar uma arquitetura stateless (e por que isso é um pré-requisito para que possamos implementar o Auto Scaling).

Você pode, em seguida, configurar o Amazon S3 como uma origem para o CloudFront para melhorar a entrega desses ativos estáticos a usuários em todo o mundo. Embora o WordPress não esteja integrado ao Amazon S3 e ao CloudFront, vários plug-ins adicionam suporte a esses serviços (por exemplo, W3 Total Cache).

Conteúdo dinâmico

O conteúdo dinâmico inclui a saída de scripts PHP do WordPress do lado do servidor. Também pode ser servido via CloudFront, configurando o site WordPress como uma origem. Como isso incluirá conteúdo personalizado, você precisará configurar o CloudFront para encaminhar certos cookies HTTP e cabeçalhos HTTP como parte de uma solicitação para seu servidor de origem personalizado. O CloudFront usa os valores de cookie encaminhados como parte da chave que identifica um objeto exclusivo em seu cache. Para garantir que você maximize a eficiência de cache, você deve configurar CloudFront apenas para encaminhar os cookies http e cabeçalhos HTTP que realmente variam o conteúdo (não os cookies que são usados apenas no lado do cliente ou por aplicativos de terceiros, por exemplo, para Web Analytics).

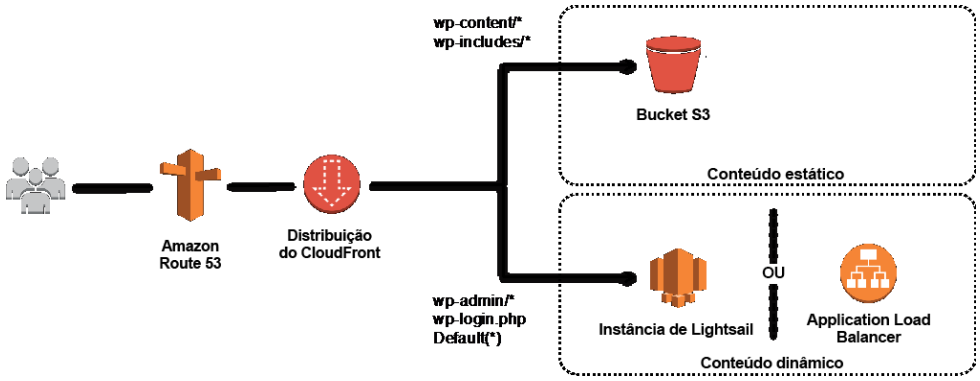

Figura 1: Entrega inteira do Web site através de CloudFront

Na Figura 1 você pode ver que agora temos duas origens: uma para conteúdo estático e outra para conteúdo dinâmico. Para obter detalhes da implementação, consulte o Apêndice A.

O CloudFront usa cabeçalhos de controle de cache padrão para identificar se e por quanto tempo ele deve armazenar em cache respostas HTTP específicas. Os mesmos cabeçalhos de controle de cache também são usados por navegadores da Web para decidir quando e por quanto tempo armazenar o conteúdo localmente para uma experiência de usuário final ótima (por exemplo, um arquivo `.css` que já está baixado não será rebaixado sempre que um visitante voltar a visualizar uma página). Você pode configurar cabeçalhos de controle de cache no nível do servidor Web (por exemplo, através de arquivos `.htaccess` ou modificações do arquivo `httpd.conf`) ou instalar um plugin WordPress (por exemplo, W3 total cache) para ditar como esses cabeçalhos são definidos para conteúdo estático e dinâmico.

Cache de banco de dados

O cache do banco de dados pode reduzir significativamente a latência e aumentar a taxa de transferência para cargas de trabalho de aplicativos com muita leitura, como o WordPress. O desempenho do aplicativo é aprimorado armazenando dados acessados com frequência na memória para acesso de baixa latência (por exemplo, os resultados de consultas de banco de dados com utilização intensiva de E/S). Quando uma grande porcentagem das consultas é fornecida a partir do cache, o número de consultas que precisam atingir o banco de dados é reduzido, resultando em um custo menor associado à execução do banco de dados.

Embora o WordPress tenha recursos de armazenamento em cache limitados para fora da caixa, uma variedade de plugins oferece suporte à integração com o Memcached, um sistema de memória em cache amplamente adotado. O plugin W3 Total Cache é um bom exemplo.[14]

Nos cenários mais simples, você pode instalar o Memcached do servidor web e capturar o resultado como um novo snapshot. Nesse caso, você é responsável por executar as tarefas administrativas associadas a um cache.

Outra opção é aproveitar um serviço gerenciado, como o Amazon ElastiCache[15] e evitar que a carga operacional sobrecarregue. O ElastiCache facilita a implantação, a operação e o dimensionamento de um cache de memória distribuído na nuvem. Você pode encontrar informações sobre como se conectar ao nós do seu cluster do ElastiCache na documentação do Amazon ElastiCache.[16]

Se você estiver usando o Lightsail e deseja acessar um cluster do ElastiCache em sua conta da AWS de forma privada, você pode o fazer usando o emparelhamento de VPCs. Você pode encontrar instruções para ativar o emparelhamento de VPC na documentação do Lightsail.[17]

Cache de bytecode

Cada vez que um script PHP é executada, ele é analisado e compilado. Ao usar um cache de bytes do PHP, a saída da compilação PHP é armazenado na RAM, para que o mesmo script não precise ser compilado de novo e de novo. Isso reduz a sobrecarga relacionados à execução de scripts PHP, o que resulta em melhor desempenho e reduz os requisitos de CPU.

Um cache de bytecode pode ser instalado em qualquer instância do Lightsail que hospede o WordPress e pode reduzir significativamente a carga. Para PHP 5.5 e posterior, recomendamos o uso de OPcache, uma extensão integrada com essa versão do PHP.[18]

Observe que o OPcache está habilitado por padrão no modelo Bitnami WordPress Lightsail, portanto, nenhuma outra ação é necessária.

Implantação elástica

Há muitos cenários em que a implantação de um único servidor pode não ser suficiente para o seu site. Nessas situações, você precisará de uma arquitetura escalonável de vários servidores.

Arquitetura de referência

Há uma arquitetura de referência disponível no GitHub que descreve as melhores práticas para a implantação do WordPress na AWS e inclui um conjunto de modelos do CloudFormation para que você possa começar a trabalhar rapidamente.[19] A arquitetura a seguir é baseada nessa arquitetura de referência. O restante desta seção analisará os motivos por trás das escolhas de arquitetura.

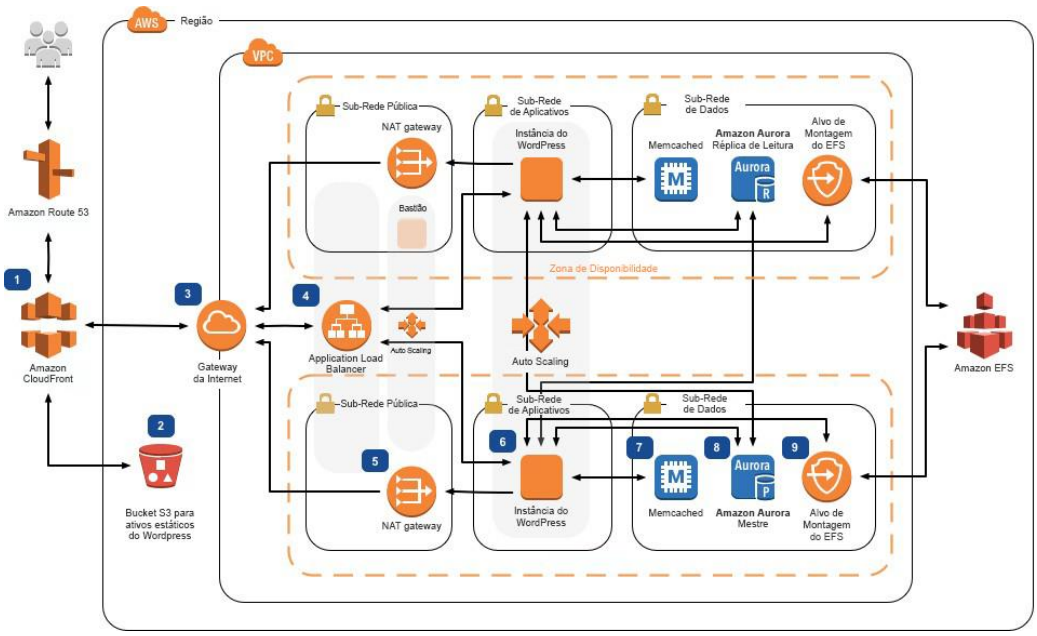

Figura 2: Arquitetura de referência para hospedagem do WordPress na AWS

Componentes da arquitetura

A arquitetura de referência na Figura 2 ilustra uma prática recomendada de implantação completa para um site do WordPress na AWS. Ela começa com cache de borda no **Amazon CloudFront** (1) para armazenar conteúdo em cache próximo aos usuários finais para uma entrega mais rápida. CloudFront puxa o conteúdo estático de um **S3 Bucket** (2) e conteúdo dinâmico de um **load balancer** de aplicativo (4) na frente das instâncias da Web. As instâncias

da Web são executadas em um **grupo de Auto Scaling** de instâncias do **Amazon EC2** (6). Um cluster do **ElastiCache** (7) armazena dados consultados frequentemente para acelerar as respostas. Uma instância MySQL do **Amazon Aurora** (8) hospeda o banco de dados do WordPress. As instâncias do WordPress EC2 acessam dados WordPress compartilhados em um sistema de arquivos da **Amazon EFS** através de um **destino de montagem do EFS** (9) em cada zona de disponibilidade. Um **gateway de internet** (3) permite a comunicação entre recursos em sua VPC e a Internet. **Gateways NAT** (5) em cada zona de disponibilidade ativam instâncias EC2 em sub-redes privadas (App e Data) para acesso à internet.

No **Amazon VPC**, existem dois tipos de sub-redes: públicas (**Public Subnet**) e privadas (**App Subnet** e **Data Subnet**). Os recursos implantados nas sub-redes públicas receberão um endereço IP público e serão publicamente visíveis na internet. O **load balancer do aplicativo** (4) e um host de bastião para administração são implantados aqui. Os recursos implantados nas sub-redes privadas recebem apenas um endereço IP privado e, portanto, não são visíveis publicamente na Internet, melhorando a segurança desses recursos. As **instâncias de servidor web do WordPress** (6), **instâncias de cluster do ElastiCache** (7), **instâncias de banco de dados MySQL do Aurora** (8) e **destinos de montagem do EFS** (9) são implantados em sub-redes privadas.

O restante desta seção aborda cada uma dessas considerações em mais detalhes.

Escalando a camada da web

Para evoluir sua arquitetura de servidor único para um multisservidor, escalável, existem cinco componentes principais que você precisará usar: instâncias do EC2, imagens de máquina da Amazon (AMIs), balanceadores de carga, Auto Scaling e verificações de integridade.

A AWS fornece uma grande variedade de tipos de instância do EC2 para que você possa escolher a melhor configuração do servidor em relação ao desempenho e ao custo. De um modo geral, o tipo de instância otimizado para computação (por exemplo, C4) pode ser uma boa opção para um servidor Web WordPress. Você pode implantar suas instâncias em várias zonas de disponibilidade dentro de uma região para aumentar a confiabilidade da arquitetura geral.

Como você tem o controle completo de sua instância do EC2, você pode fazer logon com o acesso root para instalar e configurar todos os componentes de software necessários para executar um site do WordPress. Depois de concluir essas etapas, você pode salvar essa configuração como uma AMI, que você pode usar para iniciar novas instâncias com todas as personalizações feitas.

Para distribuir solicitações de usuário final para vários nós de servidor Web, você precisa de uma solução de balanceamento de carga. A AWS fornece este recursos através de Elastic Load Balancing (ELB), um serviço altamente disponível que distribui o tráfego para vários EC2[20]. Como seu site estará servindo conteúdo para seus usuários via HTTP ou HTTPS, recomendamos que você faça uso do load balancer de aplicativo, um load balancer de camada de aplicativo com roteamento de conteúdo e a capacidade de executar vários sites WordPress em diferentes domínios, se necessário.

ELB oferece suporte à distribuição de solicitações em várias zonas de disponibilidade dentro de uma região da AWS. Você também pode configurar uma verificação de integridade para que o load balancer do aplicativo pare automaticamente o envio de tráfego para instâncias individuais que falharam (por exemplo, devido a um problema de hardware ou falha de software). Recomendamos o uso da página de login de administração do WordPress (`/wp-login.php`) para a verificação de integridade, pois essa página confirmará tanto que o servidor web está em execução e que o servidor da Web está configurado para servir arquivos PHP corretamente. Você pode optar por criar uma página de verificação de integridade personalizado que verifica outros recursos dependentes, como banco de dados e recursos de cache. Para obter mais informações, consulte Verificações de integridade para seus grupos de destino no *Application Load Balancer Guide*.[21]

Elasticidade é uma característica essencial da Nuvem AWS. Você pode lançar mais capacidade de computação (por exemplo, servidores Web) quando você precisar dele e executar menos quando você não precisar. O Auto Scaling é um serviço da AWS que ajuda você a automatizar o provisionamento para escalar a capacidade do Amazon EC2 para cima ou para baixo de acordo com as condições que você definir, sem necessidade de intervenção manual.[22] Você pode configurar o Auto Scaling para que o número de instâncias do EC2 que você está usando aumente facilmente durante picos de demanda para manter o desempenho e diminua automaticamente quando o tráfego diminui, a fim de minimizar custos. O ELB dinâmico também oferece suporte à adição e a

remoção de hosts do Amazon EC2 a partir da rotação de balanceamento de carga. O ELB em si também irá crescer dinamicamente e reduzir a capacidade de balanceamento de carga para ajustar as demandas de tráfego sem intervenção manual.

Camada de web stateless

Para tirar proveito de vários servidores Web em uma configuração de Auto Scaling, sua camada da Web deve ser apátrida. Um aplicativo stateless é aquele que não precisa de nenhum conhecimento de interações anteriores e não armazena nenhuma informação de sessão. No caso do WordPress, isso significa que todos os usuários finais recebem a mesma resposta, independentemente do servidor web que processou a sua solicitação. Um aplicativo stateless pode dimensionar horizontalmente, uma vez que qualquer solicitação pode ser atendidas por qualquer um dos recursos de computação disponíveis (ou seja, instâncias de servidor web). Quando essa capacidade não é mais necessária, qualquer recurso individual pode ser encerrado com segurança (após ase tarefas em execução serem encerradas). Esses recursos não precisam estar cientes da presença de seus pares- tudo o que é necessário é uma maneira de distribuir a carga de trabalho para eles.

Quando se trata de armazenamento de dados de sessão do usuário, o núcleo do WordPress é completamente stateless, porque se baseia em cookies que são armazenados no navegador da web do cliente. O armazenamento de sessão não é uma preocupação, a menos que você não tenha instalado nenhum código personalizado (por exemplo, um plug-in do WordPress) que, em vez disso, dependa de sessões nativas do PHP.

No entanto, o WordPress foi projetado originalmente para ser executado em um único servidor. Como resultado, ele armazena alguns dados no sistema de arquivos local do servidor. Ao executar o WordPress em uma configuração de vários servidores, isso cria um problema porque há inconsistência entre servidores web. Por exemplo, se um usuário faz upload de uma nova imagem, ela só é armazenado em um dos servidores.

Isso demonstra por que precisamos, para melhorar a configuração de execução padrão do WordPress, mover dados importantes para armazenamento compartilhado. A arquitetura de melhores práticas terá, portanto, um banco de dados como uma camada separada fora do servidor da Web e utilizará

armazenamento compartilhado para armazenar uploads de usuários, temas e plug-ins.

Armazenamento compartilhado (Amazon S3 e Amazon EFS)

Por padrão, o WordPress armazena uploads de usuário no sistema de arquivos local e, portanto, não é stateless. Portanto, é necessário mover a instalação do WordPress e todas as personalizações do usuário (como configuração, plugins, temas e uploads gerados pelo usuário) para uma plataforma de dados compartilhados para ajudar a reduzir a carga nos servidores da Web e para tornar a camada da web stateless.

O Amazon Elastic File System (Amazon EFS) fornece sistemas de arquivos de rede escalonáveis para uso com instâncias EC2.[23] Os sistemas de arquivos EFS são distribuídos por um número não restrito de servidores de armazenamento, permitindo que os sistemas de arquivos cresçam com elasticidade e permitindo o acesso massivo de paralelo a partir de instâncias EC2. O design distribuído do Amazon EFS evita gargalos e restrições inerentes a servidores de arquivos tradicionais.

Ao mover todo o diretório de instalação do WordPress para um sistema de arquivos EFS e montá-lo em cada uma de suas instâncias EC2 quando inicializar, seu site WordPress e todos os seus dados serão automaticamente armazenados em um sistema de arquivos distribuídos que não é dependente de qualquer nenhuma instância EC2 , tornando sua camada da Web completamente stateless. O benefício dessa arquitetura é que você não precisa instalar plugins e temas em cada nova execução da instância, e você pode significativamente acelerar a instalação e a recuperação de instâncias do WordPress. Também é mais fácil implementar alterações plugins e temas no WordPress, conforme descrito na seção Considerações de Implantação deste documento.

Para garantir o melhor desempenho do seu site ao executar a partir de um sistema de arquivos do EFS, verifique as definições de configuração recomendada para o Amazon EFS e OPcache na Arquitetura de Referência da AWS para WordPress.[24]

Você também tem a opção de descarregar todos os ativos estáticos, como imagens, CSS e arquivos JavaScript, para um bucket do S3 com o armazenamento em cache do CloudFront na frente. O mecanismo para fazer isso em uma arquitetura do servidor é exatamente o mesmo que para uma

arquitetura de servidor único, como discutido na seção deste whitepaper Conteúdo Estático. Os benefícios são os mesmos que na arquitetura de um único servidor — você pode descarregar o trabalho associado ao serviço de ativos estáticos para Amazon S3 e CloudFront, permitindo assim que seus servidores Web se foquem na geração de conteúdo dinâmico e atendam mais solicitações de usuário por servidor Web.

Nível de dados (Amazon Aurora e Amazon ElastiCache)

Com a instalação do WordPress armazenado em um sistema de arquivos distribuído, escalável, compartilhado de rede, e ativos estáticos sendo servidos a partir de Amazon S3, podemos agora concentrar a nossa atenção sobre o componente monitoração restante: o banco de dados. Como com a camada de armazenamento, o banco de dados não deve ser dependente de qualquer servidor único, portanto, não podemos hospedá-lo em um dos servidores Web. Em vez disso, vamos hospedar o banco de dados do WordPress no Amazon Aurora.

O Amazon Aurora é um banco de dados relacional compatível com MySQL e PostgreSQL, construído para a nuvem que combina o desempenho e a disponibilidade de bancos de dados comerciais high-end com a simplicidade e a rentabilidade dos bancos de dados Open Source. A Aurora MySQL aumenta o desempenho e a disponibilidade do MySQL, integrando firmemente o mecanismo de banco de dados com um sistema de armazenamento distribuído feito por propósito, apoiado pelo SSD. É tolerante a falhas e autocura, replica seis cópias de seus dados em três zonas de disponibilidade, é projetado para disponibilidade maior que 99,99% e faz backup contínuo de seus dados no Amazon S3. Amazon Aurora é projetado para detectar automaticamente falhas de banco de dados e reiniciar sem a necessidade de recuperação de falhas ou para reconstruir o cache de banco de dados.

O Amazon Aurora fornece uma série de tipos de instâncias para atender a diferentes perfis de aplicativos, incluindo instâncias otimizadas para memória e rajadas.[25] Para melhorar o desempenho de seu banco de dados, você pode selecionar um tipo de instância grande para fornecer mais recursos de CPU e memória.

O Amazon Aurora gerencia automaticamente o failover entre a instância principal e as réplicas do Aurora para que seus aplicativos possam retomar operações de banco de dados o mais rápido possível sem intervenção administrativa manual. O Failover normalmente leva menos de 30 segundos. Depois de criar pelo menos uma réplica do Aurora, conecte-se à instância

principal usando o endpoint de cluster para permitir que o failover automático do aplicativo no caso da instância principal falhar. Você pode criar até 15 réplicas de leitura de baixa latência em três zonas de disponibilidade.

À medida que seu banco de dados aumenta, seu cache de banco de dados também precisará ser dimensionado. Como discutido anteriormente na seção Cache de Banco de Dados, o ElastiCache tem recursos para dimensionar o cache em vários nós em um cluster do ElastiCache, e em várias zonas de disponibilidade em uma região para melhorar a disponibilidade. À medida que você escala seu cluster ElastiCache, deve garantir que configure seu plugin de cache para se conectar usando o ponto de extremidade de configuração para que o WordPress possa usar novos nós de cluster à medida que s são adicionados e parar de usar nós de cluster antigos à medida que são removidos. Você também precisará configurar os servidores da Web para usar o ElastiCache Cluster Client para PHP e atualizar sua AMI para armazenar essa alteração.[26]

Conclusão

A AWS apresenta muitas opções de arquitetura para a execução do WordPress. A opção mais simples é uma instalação de servidor único para sites de baixo tráfego. Para sites mais avançadas, os administradores do site podem adicionar várias outras opções, cada uma representando um aumento incremental em termos de disponibilidade e escalabilidade. Os administradores podem selecionar os recursos mais adequados para suas necessidades e seu orçamento.

Contribuidores

As pessoas e organizações a seguir contribuíram com este documento:

- Paul Lewis, Solutions Architect, Amazon Web Services
- Ronan Guilfoyle, Solutions Architect, Amazon Web Services
- Andreas Chatzakis, Solutions Architect Manager, Amazon Web Services

Revisões do documento

Data	Descrição
Fevereiro de 2018	Atualizado para esclarecer mensagens de produto do Amazon Aurora.
Dezembro de 2017	Atualizado para incluir serviços da AWS executadas desde a primeira publicação.
Dezembro de 2014	Primeira publicação.

Apêndice A: configuração do CloudFront

Para obter o melhor desempenho e eficiência ao usar o Amazon CloudFront com seu site WordPress, é importante configurar o site corretamente para os diferentes tipos de conteúdo que estão sendo servidos.

Origens e Comportamentos

Uma origem é um local em que o CloudFront envia solicitações de conteúdo que distribui através localizações de borda.[27] Você pode apontar o CloudFront para o local no qual você está armazenando o conteúdo estático (na arquitetura de referência acima este é o Amazon S3) usando uma origem do Amazon S3. Você pode apontar CloudFront para o seu conteúdo dinâmico (na implantação de um único servidor acima esta é uma instância Lightsail, ou na arquitetura de referência acima este é o load balancer do aplicativo) usando uma origem personalizada. Quando você usa o Amazon S3 como origem para sua distribuição, você precisa usar uma política de bucket para tornar o conteúdo publicamente acessível.[28]

Comportamentos permitem que você defina regras que controlam como o CloudFront armazena o conteúdo e, por sua vez, determinam quão eficaz o cache será.[29] Os comportamentos Comportamentos permitem que você controle o protocolo e os métodos HTTP pelos quais seu site é acessível. Eles também permitem que você controle se deve passar cabeçalhos HTTP, cookies ou cadeias de consulta para o seu back-end (e, em caso afirmativo, quais). Os comportamentos podem se aplicar a padrões de caminho de URL específicos.

Criação de Usuário do IAM

Você precisará criar um usuário do AWS Identity and Access Management (IAM) para o plugin do WordPress armazenar os ativos estáticos no Amazon S3. Você pode seguir este guia para Criação de um usuário do IAM na sua conta da AWS.[30]

Observação: As funções do IAM fornecem uma melhor maneira de gerenciar o acesso aos recursos da AWS, mas no momento o plugin W3 Total Cache não é compatível com as funções do IAM.[31]

Anote as credenciais de segurança do usuário e armazene-as de maneira segura - você precisará delas mais tarde.

Criação de Bucket do S3

Você também precisará criar um bucket do Amazon S3 na região de sua escolha. Você pode seguir este guia para criar um bucket do Amazon S3.[32] Habilitar hospedagem do site estático para o bucket, seguindo o guia para configurar um bucket para hospedagem de site.[33]

Crie uma política do IAM para fornecer o usuário do IAM criado anteriormente para acessar o bucket do S3 especificado e anexar a política ao usuário do IAM. Você pode seguir este guia para Gerenciar políticas do IAM para criar a política a seguir:[34]

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "Stmt1389783689000",
   "Effect": "Allow",
   "Principal": "*",
   "Action": [
    "s3:DeleteObject",
    "s3:GetObject",
    "s3:GetObjectAcl",
    "s3:ListBucket",
    "s3:PutObject",
    "s3:PutObjectAcl"
   ],
   "Resource": [
    "arn:aws:s3:::wp-demo",
    "arn:aws:s3:::wp-demo/*"
   ]
  }
 ]
}
```

Criação de distribuição do CloudFront

Em seguida, você precisará criar uma distribuição na web do CloudFront seguindo o guia Lista de tarefas para criar uma distribuição na web.[35] Quando você cria uma nova distribuição do CloudFront, você cria automaticamente uma origem e um comportamento padrão, que você usará para conteúdo dinâmico. Também criaremos quatro comportamentos adicionais para personalizar ainda mais a maneira como as duas solicitações estáticas e dinâmicas são tratadas. A tabela a seguir resume as propriedades de configuração para os cinco comportamentos.

	Static	Dynamic (admin)	Dynamic (front end)
Paths	wp-content/* wp-includes/*	wp-admin/* wp-login.php	default (*)
Protocols	HTTP and HTTPS	Redirect to HTTPS	HTTP and HTTPS
HTTP methods	GET, HEAD	ALL	ALL
HTTP headers	NONE	ALL	Host CloudFront-Forwarded-Proto CloudFront-Is-Desktop-Viewer CloudFront-Is-Mobile-Viewer CloudFront-Is-Tablet-Viewer
Cookies	NONE	ALL	comment_* wordpress_* wp-settings-*
Query strings	YES (invalidation)	YES	YES

Tabela 1: Resumo dos comportamentos de propriedade de configuração para o CloudFront

Para o comportamento padrão, recomendamos a seguinte configuração:

- Permitir que a política de protocolo de origem **corresponda ao visualizador**, de modo que se os telespectadores se conectarem a CloudFront usando HTTPS, o CloudFront se conectará à sua origem usando HTTPS também, obtendo criptografia de ponta a ponta. Observe que isso exige que você instale um certificado SSL confiável no load balancer conforme explicado no Guia do Desenvolvedor do Amazon CloudFront.[36]
- Permitir todos os métodos HTTP, pois as partes dinâmicas do site exigem solicitações GET e POST (por exemplo, para dar suporte ao POST para os formulários de envio de comentários).

- Encaminhe apenas os cookies que variam a saída do WordPress, por exemplo, `wordpress_*`, `wp-settings-*` e `comment_*`. Você precisará estender essa lista se tiver instalado quaisquer plug-ins que dependam de outros cookies que não estejam na lista.
- Encaminhará somente os cabeçalhos HTTP que afetam a saída do WordPress, por exemplo, **Host**, **CloudFront-Forwarded-Proto**, é - **CloudFront-Desktop-Viewer**, **CloudFront-for-Mobile-Viewer** e **CloudFront-for-Tablet-Viewer**. O cabeçalho de **host** permite que vários sites do WordPress sejam hospedados na mesma origem; o cabeçalho **CloudFront-Forwarded-Proto** permite que diferentes versões de páginas sejam armazenadas em cache, dependendo se são acessados via HTTP ou HTTPS; e os cabeçalhos do **CloudFront-for-Desktop-Viewer**, do **CloudFront-for-Mobile-Viewer** e do **CloudFront-for-Tablet-Viewer** permitem que você personalize a saída dos temas com base no tipo de dispositivo do usuário final.
- Encaminhe e armazene em cache com base em todas as strings de consulta, pois o WordPress depende delas, e elas podem ser usadas para invalidar objetos em cache.

Se você deseja servir seu site com um nome de domínio personalizado (ou seja, * .cloudfront.net), insira os URIs apropriados em **nomes de domínio alternativos** em configurações de distribuição. Nesse caso, você também precisará de um certificado SSL para seu nome de domínio personalizado. O certificados SSL pode ser solicitado gratuitamente por meio do AWS Certificate Manager e configurados em um CloudFront distribution.[37]

Agora você precisa criar mais dois comportamentos de cache para conteúdo dinâmico: uma para a página de login (padrão de caminho: `wp-login.php`) e uma para o painel admin (padrão de caminho: `wp-admin /*`). Esses dois comportamentos têm configurações idênticas, como se segue:

- Aplique uma política de protocolo de visualização apenas de HTTPS.
- Permita todos os métodos HTTP.
- Cache baseado em todos os cabeçalhos HTTP.
- Encaminhe todos os cookies.
- Encaminhe e armazene em cache com base em todas as strings de consulta.

A razão por trás dessa configuração é que essa seção do website é altamente personalizada e, normalmente, tem apenas alguns usuários, portanto, a eficiência do armazenamento em cache não é a principal preocupação aqui. O objetivo é manter a configuração simples para garantir a máxima compatibilidade com quaisquer plugins instalados, transmitindo todos os cookies e cabeçalhos para a origem.

Neste ponto, a configuração do CloudFront para conteúdo dinâmico está concluído. No entanto, antes de adicionar a configuração para conteúdo estático, o site do WordPress deve ser configurado adequadamente, o que é abordado a seguir.

Instalação e configuração do plugin do WordPress

Neste exemplo, o plugin W3 Total Cache (W3TC) é usado para armazenar os ativos estáticos no Amazon S3. No entanto, há outros plugins disponíveis com recursos semelhantes. Se você deseja usar uma alternativa você pode adaptar as etapas abaixo de acordo com a necessidade. As etapas a seguir apenas se referem a recursos ou configurações relevantes para este exemplo. Uma descrição detalhada de todas as configurações está além do escopo deste documento. Consulte a página plugin W3 Total Cache em wordpress.org para obter mais informações.

Instale e ative o plugin W3TC do painel de administração do WordPress. Navegue até a seção Configurações gerais da configuração do plugin, e certifique-se de que o **Cache do Navegador** e **CDN** esteja ativado. Na lista suspensa da configuração CDN, selecione **Origin Push: Amazon CloudFront** (este terá o Amazon S3 como sua origem).

Navegue até a seção cache do navegador da configuração do plugin e habilite os **cabeçalhos de expiração**, **controle de cache** e **tag de entidade** (ETag). Ative também a opção **impedir o armazenamento em cache de objetos após a alteração de configurações**, para que uma nova sequência de consulta seja gerada e anexada aos objetos sempre que as configurações forem alteradas.

Navegue até a seção CDN da configuração do plugin e insira as credenciais de segurança do usuário do IAM que você criou anteriormente, bem como o

nome do bucket do S3. Se você estiver servindo seu site por meio do URL do CloudFront, insira o nome do domínio de distribuição na caixa relevante. Caso contrário, digite um ou mais CNAMEs para seu nome de domínio personalizado (s).

Por fim, você deve exportar a biblioteca de mídia e carregar os arquivos wp-includes, de tema e personalizados no Amazon S3 usando o plug-in W3TC. Essas funções de upload estão disponíveis na seção **geral** da página de configuração **CDN**.

Criação de origem estática

Agora que os arquivos estáticos estão armazenados no Amazon S3, volte para a configuração do CloudFront no console do CloudFront e configure o Amazon S3 como a origem de conteúdo estático. Para fazer isso, adicione uma segunda origem apontando para o bucket do S3 que você criou para essa finalidade. Em seguida, crie mais dois comportamentos de cache, um para cada uma das duas pastas (`wp-content` e `wp-inclui`) que devem usar a origem do S3 em vez de origem padrão para conteúdo dinâmico. Configure os dois da mesma maneira:

- Servir apenas solicitações HTTP GET.
- O Amazon S3 não varia sua saída com base em cookies ou cabeçalhos HTTP, portanto, você pode melhorar o armazenamento a eficiência ao não encaminha-los para a origem através do CloudFront.
- Apesar do fato de que esses comportamentos servem apenas para conteúdo estático (que não aceita parâmetros), você encaminhará as strings de consulta para a origem. Isso é assim para que você possa usar strings de consulta como identificadores de versão para invalidar instantaneamente, por exemplo, arquivos CSS mais antigos ao implantar novas versões. Para obter mais informações, consulte o Guia do desenvolvedor do Amazon CloudFront.[38]

Observação: depois de adicionar os comportamentos de origem estática à sua distribuição CloudFront, verifique a ordem para garantir os comportamentos `wp-login.php` e `wp-admin/*` têm precedência maior do que os comportamentos de conteúdo estático. Caso contrário, você poderá observar comportamento estranho ao acessar o painel de administração.

Apêndice B: backup e recuperação

Recuperar-se de falhas no AWS é mais rápido e mais fácil de fazer em comparação com ambientes de hospedagem tradicional. Por exemplo, você pode executar uma instância de substituição em minutos em resposta a uma falha de hardware, ou você pode fazer uso de failover automatizado em muitos de nossos serviços gerenciados para negar o impacto de uma reinicialização devido a manutenção de rotina.

No entanto, você ainda precisa garantir que está fazendo backup de dados corretos para recuperá-lo com êxito. A fim de restabelecer a disponibilidade de um site WordPress, você deve ser capaz de recuperar os seguintes componentes:

- O sistema operacional (SO) e os serviços de instalação e configuração (Apache, MySQL, etc.)
- Código de aplicativo e configuração do WordPress
- Temas e plugins do WordPress
- Uploads (por exemplo, arquivos de mídia para postagens)
- Conteúdos de database (postagens, comentários, etc.)

Como catalogados no whitepaper Abordagens de backup e recuperação usando o Amazon Web Services, a AWS fornece uma variedade de métodos para fazer backup e restaurar seus dados e bens por aplicativo web.[39]

Já discutimos a utilização de snapshots Lightsail para proteger todos os dados armazenados no armazenamento local da instância. Se o seu site do WordPress for executado apenas na instância do Lightsail, os instantâneos comuns do Lightsail deverão ser suficientes para que você recupere o site do WordPress em sua totalidade. No entanto, você ainda vai perder todas as

alterações aplicadas ao seu site desde que o último snapshot foi tirado, se você restaurar a partir de um snapshot.

Em uma implantação de vários servidores, será necessário fazer backup de cada um dos componentes discutidos anteriormente usando mecanismos diferentes. Cada componente pode ter um requisito diferente para a frequência de backup, por exemplo, a instalação e configuração do sistema operacional e WordPress vai mudar muito menos frequentemente do que o conteúdo gerado pelo usuário e, portanto, pode ser apoiado com menos frequência, sem perder dados em caso de recuperação.

Para fazer backup do sistema operacional e instalação e configuração dos serviços, e o código e configuração do aplicativo WordPress, você pode criar uma AMI de uma instância do EC2 corretamente configurada. AMIs pode servir dois propósitos: atuar como um backup do estado de instância e atuar como um modelo ao lançar novas instâncias.

Para fazer backup do código e da configuração do aplicativo WordPress, você precisará fazer uso de backups de AMIs e Aurora (mais a seguir).

Para fazer o backup dos temas e plugins do WordPress instalados em seu site, você precisa fazer o backup do bucket do Amazon S3 ou do sistema de arquivos do Amazon EFS em que estão armazenados.

- Para temas e plugins armazenados em um bucket do S3, você pode habilitar a replicação através de regiões para que todos os objetos carregados para o bucket primário sejam replicados automaticamente para o bucket de backup em outra AWS Region.[40] A Recuperação através de Regiões requer que o versionamento esteja habilitado em ambos os buckets de origem e de destino, o que fornece uma camada adicional de proteção e permite que você reverta para uma versão anterior de um determinado objeto na sua bucket.[41]
- Para temas e plug-ins armazenados em um sistema de arquivos do EFS, você pode criar um AWS Data Pipeline para copiar dados do seu sistema de arquivos do EFS de produção para outro sistema de arquivos do EFS, conforme descrito na página de documentação Backup de um sistema de arquivos do EFS.[42] Você também pode fazer backup de um sistema de arquivos do EFS usando qualquer aplicativo de backup com o qual já esteja familiarizado.

- Para fazer backup de uploads de usuário, você deve seguir as etapas descritas anteriormente para o backup dos temas e plugins do WordPress.
- Para fazer backup de conteúdo do banco de dados você precisa usar o Aurora backup.[43] O Aurora faz o backup do volume de cluster automaticamente e retém dados de restauração pelo tempo de duração do *período de retenção de backup*. Os backups do Aurora são contínuos e incrementais para que você possa rapidamente restaurar para qualquer ponto dentro do período de retenção de backup. Não ocorre nenhum impacto no desempenho nem interrupção no serviço do banco de dados, pois os dados de backup estão sendo gravados. Você pode especificar um período de retenção de backup, de 1 a 35 dias. Você também pode criar snapshots manuais de database, que são mantidos até que vocês os exclua. Manual de snapshots de banco de dados são úteis para backups e arquivamentos de longo prazos.[44]

Apêndice C: implantação de novos plugins e temas

Poucos sites permanecem estáticos. Na maioria dos casos, você irá adicionar periodicamente temas publicamente disponíveis e os plug-ins do WordPress ou atualizar para uma versão mais recente do WordPress. Em outros casos, você irá desenvolver seus próprios temas personalizados e plugins do zero.

Sempre que você faz uma mudança estrutural da sua instalação do WordPress existe um certo risco de se introduzirem problemas inesperados. No mínimo, você deve ter um backup do seu código de aplicativo, configuração e banco de dados antes de aplicar qualquer alteração significativa (como instalar um novo plugin). Para sites de negócios ou outros valores, você deve testar essas alterações primeiro em um ambiente de preparação separado. Com a AWS, é muito fácil replicar a configuração de seu ambiente de produção e executar todo o processo de implantação de maneira segura. Depois de concluir seus testes, você pode simplesmente desmontar seu ambiente de teste e parar de pagar por esses recursos. Posteriormente, discutimos algumas considerações específicas do WordPress. Para obter mais informações sobre as melhores práticas de desenvolvimento e teste na AWS, consulte o whitepaper Desenvolvimento e Teste na Amazon Web Services.[45]

Alguns plugins gravam informações de configuração na tabela de banco de dados `wp_options` (ou introduzem alterações no esquema do banco de dados), enquanto outros criam arquivos de configuração no diretório de instalação do WordPress. Como movemos o banco de dados e o armazenamento para plataformas compartilhadas, essas alterações ficarão imediatamente disponíveis para todas as instâncias em execução sem qualquer esforço adicional de sua parte.

Ao implantar novos temas no WordPress, um pouco mais de esforço pode ser necessário. Se você estiver apenas usando o Amazon EFS para armazenar todos os arquivos de instalação do WordPress, os novos temas estarão imediatamente disponíveis para todas as instâncias em execução. No entanto, se você estiver descarregando conteúdo estático para o Amazon S3, precisará processar uma cópia deles no local correto do intervalo. Plugins como W3 Total Cache oferecem uma forma de iniciar manualmente essa tarefa. Como alternativa, você pode automatizar essa etapa como parte de um processo de compilação.

Como os recursos de tema podem ser armazenados em cache no CloudFront e no navegador, você precisa de uma maneira de invalidar versões mais antigas ao implantar as alterações. A melhor maneira de fazer isso é incluindo algum tipo de identificador de versão no objeto. Esse identificador pode ser uma string de consulta com a marca de data e hora ou uma string aleatória. Se você usar o plugin W3 Total Cache, você pode atualizar uma string de consulta de mídia que é anexado aos URLs de arquivos de mídia.

Observações

[1] https://amazonlightsail.com/

[2] https://aws.amazon.com/ec2/

[3] https://aws.amazon.com/marketplace/

[4] https://amazonlightsail.com/pricing/

[5] https://lightsail.aws.amazon.com/ls/docs/how-to/article/lightsail-how-to-create-instance-from-snapshot

[6] https://lightsail.aws.amazon.com/ls/docs/how-to/article/lightsail-how-to-create-larger-instance-from-snapshot-using-aws-cli

[7] https://lightsail.aws.amazon.com/ls/docs/getting-started/article/getting-started-with-wordpress-and-lightsail

[8] https://lightsail.aws.amazon.com/ls/docs/overview/article/understanding-instance-snapshots-in-amazon-lightsail

[9] https://github.com/awslabs/lightsail-auto-snapshots

[10] https://wordpress.org/plugins/w3-total-cache/

[11] https://aws.amazon.com/cloudfront/

[12] https://aws.amazon.com/cloudfront/details/#edge-locations

[13] https://aws.amazon.com/s3/

[14] https://memcached.org/

[15] https://aws.amazon.com/elasticache/

[16] https://docs.aws.amazon.com/AmazonElastiCache/latest/UserGuide/GettingStarted.ConnectToCacheNode.html

[17] https://lightsail.aws.amazon.com/ls/docs/how-to/article/lightsail-how-to-set-up-vpc-peering-with-aws-resources

[18] http://php.net/manual/en/book.opcache.php

[19] https://github.com/awslabs/aws-refarch-wordpress

[20] https://aws.amazon.com/elasticloadbalancing/

[21] https://docs.aws.amazon.com/elasticloadbalancing/latest/application/target-group-health-checks.html

[22] https://aws.amazon.com/autoscaling/

[23] https://aws.amazon.com/efs/details/

[24] https://github.com/awslabs/aws-refarch-wordpress#opcache

[25] https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.DBInstanceClass.html

[26] https://docs.aws.amazon.com/AmazonElastiCache/latest/UserGuide/Appendix.PHPAutoDiscoverySetup.html

[27] https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/DownloadDistS3AndCustomOrigins.html

[28] https://docs.aws.amazon.com/AmazonS3/latest/dev/WebsiteAccessPermissionsReqd.html

[29] https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/RequestAndResponseBehavior.html

[30] https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users_create.html

[31] http://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2.html

[32] https://docs.aws.amazon.com/AmazonS3/latest/user-guide/create-bucket.html

[33] https://docs.aws.amazon.com/AmazonS3/latest/dev/HowDoIWebsiteConfiguration.html

[34] https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_manage.html

[35] https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/distribution-web-creating.html

[36] https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/using-https-cloudfront-to-custom-origin.html

[37] https://docs.aws.amazon.com/acm/latest/userguide/gs-acm-request.html

[38] https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/ReplacingObjects.html

[39] https://do.awsstatic.com/whitepapers/Storage/Backup_and_Recovery_Approaches_Using_AWS.pdf

[40] https://docs.aws.amazon.com/AmazonS3/latest/dev/crr.html

[41] https://docs.aws.amazon.com/AmazonS3/latest/dev/Versioning.html

[42] https://docs.aws.amazon.com/efs/latest/ug/efs-backup.html

[43] http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Aurora.Managing.html#Aurora.Managing.Backups

[44] https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_CreateSnapshot.html

[45] https://do.awsstatic.com/whitepapers/aws-development-test-environments.pdf