Melhores práticas para usar o Terraform Provider AWS

AWS Orientação prescritiva

AWS Orientação prescritiva: Melhores práticas para usar o Terraform Provider AWS

Table of Contents

Melhores práticas para usar o Terraform Provider AWS

Michael Begin, DevOps consultor sênior, Amazon Web Services (AWS)

Agosto de 2025 (histórico do documento)

O gerenciamento da infraestrutura como código (IaC) com o Terraform on AWS oferece benefícios importantes, como maior consistência, segurança e agilidade. No entanto, à medida que sua configuração do Terraform cresce em tamanho e complexidade, torna-se fundamental seguir as melhores práticas para evitar armadilhas.

Este guia fornece as melhores práticas recomendadas para usar o Terraform AWS Provider de HashiCorp. Ele orienta você sobre o controle de versão adequado, controles de segurança, back-ends remotos, estrutura de base de código e provedores comunitários para otimizar o Terraform. AWS Cada seção aborda mais detalhes sobre as especificidades da aplicação dessas melhores práticas:

- Segurança
- Backends
- Estrutura e organização da base de código
- AWS Gerenciamento de versões do provedor
- Módulos comunitários

Objetivos

Este guia ajuda você a obter conhecimento operacional sobre o Terraform AWS Provider e aborda as seguintes metas de negócios que você pode alcançar seguindo as melhores práticas de IaC em relação à segurança, confiabilidade, conformidade e produtividade do desenvolvedor.

- Melhore a qualidade e a consistência do código de infraestrutura em todos os projetos do Terraform.
- Acelere a integração do desenvolvedor e a capacidade de contribuir com o código de infraestrutura.
- Aumente a agilidade dos negócios por meio de mudanças mais rápidas na infraestrutura.
- Reduza os erros e o tempo de inatividade relacionados às mudanças na infraestrutura.
- Otimize os custos de infraestrutura seguindo as melhores práticas de IaC.

- Fortaleça sua postura geral de segurança por meio da implementação das melhores práticas.

Público-alvo

O público-alvo deste guia inclui líderes técnicos e gerentes que supervisionam as equipes que usam o Terraform para IaC em. AWS Outros leitores em potencial incluem engenheiros de infraestrutura, DevOps engenheiros, arquitetos de soluções e desenvolvedores que usam ativamente o Terraform para gerenciar a AWS infraestrutura.

Seguir essas melhores práticas economizará tempo e ajudará a desbloquear os benefícios do IaC para essas funções.

Visão geral

Os provedores do Terraform são plug-ins que permitem que o Terraform interaja com diferentes APIs. O Terraform AWS Provider é o plugin oficial para gerenciar AWS infraestrutura como código (IaC) com o Terraform. Ele traduz a sintaxe do Terraform em chamadas de AWS API para criar, ler, atualizar e excluir recursos. AWS

O AWS provedor lida com a autenticação, traduz a sintaxe do Terraform em chamadas de AWS API e fornece recursos em. AWS Você usa um bloco de `provider` código do Terraform para configurar o plug-in do provedor que o Terraform usa para interagir com a AWS API. Você pode configurar vários blocos de AWS provedores para gerenciar recursos em diferentes Contas da AWS regiões.

Aqui está um exemplo de configuração do Terraform que usa vários blocos de AWS provedores com aliases para gerenciar um banco de dados do Amazon Relational Database Service (Amazon RDS) que tem uma réplica em uma região e conta diferentes. Os provedores primários e secundários assumem funções diferentes AWS Identity and Access Management (IAM):

```
# Configure the primary AWS Provider
provider "aws" {
  region = "us-west-1"
  alias  = "primary"
}

# Configure a secondary AWS Provider for the replica Region and account
provider "aws" {
  region      = "us-east-1"
  alias       = "replica"
  assume_role {
    role_arn     = "arn:aws:iam::<replica-account-id>:role/<role-name>"
    session_name = "terraform-session"
  }
}

# Primary Amazon RDS database
resource "aws_db_instance" "primary" {
  provider = aws.primary

  # ... RDS instance configuration
}

# Read replica in a different Region and account
```

```
resource "aws_db_instance" "read_replica" {
  provider = aws.replica

  # ... RDS read replica configuration
  replicate_source_db = aws_db_instance.primary.id
}
```

Neste exemplo:

- O primeiro `provider` bloco configura o AWS provedor principal na `us-west-1` região com o `primary` alias.
- O segundo `provider` bloco configura um AWS provedor secundário na `us-east-1` região com o `replica` alias. Esse provedor é usado para criar uma réplica de leitura do banco de dados principal em uma região e conta diferentes. O `assume_role` bloco é usado para assumir uma função do IAM na conta de réplica. O `role_arn` especifica o Amazon Resource Name (ARN) da função do IAM a ser assumida `session_name` e é um identificador exclusivo para a sessão do Terraform.
- O `aws_db_instance.primary` recurso cria o banco de dados primário do Amazon RDS usando o `primary` provedor na `us-west-1` região.
- O `aws_db_instance.read_replica` recurso cria uma réplica de leitura do banco de dados principal na `us-east-1` região usando o `replica` provedor. O `replicate_source_db` atributo faz referência ao ID do `primary` banco de dados.

Práticas recomendadas de segurança

O gerenciamento adequado da autenticação, dos controles de acesso e da segurança é fundamental para o uso seguro do Terraform AWS Provider. Esta seção descreve as melhores práticas em torno de:

- Funções e permissões do IAM para acesso com privilégios mínimos
- Protegendo credenciais para ajudar a impedir o acesso não autorizado a AWS contas e recursos
- Criptografia de estado remoto para ajudar a proteger dados confidenciais
- Escaneamento de infraestrutura e código-fonte para identificar configurações incorretas
- Controles de acesso para armazenamento remoto de estado
- Aplicação da política do Sentinel para implementar barreiras de governança

Seguir essas melhores práticas ajuda a fortalecer sua postura de segurança ao usar o Terraform para gerenciar AWS a infraestrutura.

Siga o princípio do menor privilégio

O privilégio mínimo é um princípio fundamental de segurança que se refere à concessão somente das permissões mínimas necessárias para que um usuário, processo ou sistema execute as funções pretendidas. É um conceito fundamental em controle de acesso e uma medida preventiva contra acesso não autorizado e possíveis violações de dados.

O princípio do privilégio mínimo é enfatizado várias vezes nesta seção porque está diretamente relacionado à forma como o Terraform autentica e executa ações contra provedores de nuvem, como. AWS

Quando você usa o Terraform para provisionar e gerenciar AWS recursos, ele age em nome de uma entidade (usuário ou função) que exige permissões apropriadas para fazer chamadas de API. Não seguir o menor privilégio abre grandes riscos de segurança:

- Se o Terraform tiver permissões excessivas além do necessário, uma configuração incorreta não intencional poderá fazer alterações ou exclusões indesejadas.
- Concessões de acesso excessivamente permissivas aumentam o escopo do impacto se os arquivos ou credenciais do estado do Terraform forem comprometidos.

- Não seguir o menor privilégio viola as melhores práticas de segurança e os requisitos de conformidade regulatória para conceder o acesso mínimo necessário.

Usar funções do IAM

Use funções do IAM em vez de usuários do IAM sempre que possível para aumentar a segurança com o Terraform AWS Provider. As funções do IAM fornecem credenciais de segurança temporárias que alternam automaticamente, o que elimina a necessidade de gerenciar chaves de acesso de longo prazo. As funções também oferecem controles de acesso precisos por meio de políticas do IAM.

Conceda acesso com privilégios mínimos usando políticas do IAM

Crie cuidadosamente as políticas do IAM para garantir que as funções e os usuários tenham apenas o conjunto mínimo de permissões necessárias para sua carga de trabalho. Comece com uma política vazia e adicione iterativamente serviços e ações permitidos. Para fazer isso:

- Ative o IAM Access Analyzer para avaliar políticas e destacar permissões não utilizadas que podem ser removidas.
- Analise manualmente as políticas para remover quaisquer recursos que não sejam essenciais para a responsabilidade pretendida da função.
- Use variáveis e tags de política do IAM para simplificar o gerenciamento de permissões.

Políticas bem construídas concedem acesso suficiente para cumprir as responsabilidades da carga de trabalho e nada mais. Defina ações no nível da operação e permita chamadas somente para recursos específicos necessários APIs .

Seguir essa prática recomendada reduz o escopo do impacto e segue os princípios fundamentais de segurança de separação de deveres e acesso com privilégios mínimos. Inicie o acesso estrito e aberto gradualmente, conforme necessário, em vez de começar a abrir e tentar restringir o acesso posteriormente.

Assuma funções do IAM para autenticação local

Ao executar o Terraform localmente, evite configurar chaves de acesso estáticas. Em vez disso, use funções do IAM para conceder acesso privilegiado temporariamente sem expor credenciais de longo prazo.

Primeiro, crie uma função do IAM com as permissões mínimas necessárias e adicione uma relação de confiança que permita que a função do IAM seja assumida pela sua conta de usuário ou identidade federada. Isso autoriza o uso temporário da função.

Exemplo de política de relacionamento de confiança:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/terraform-execution"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Em seguida, execute o AWS CLI comando aws sts assume-role para recuperar credenciais de curta duração para a função. Essas credenciais normalmente são válidas por uma hora.

AWS CLI exemplo de comando:

```
aws sts assume-role --role-arn arn:aws:iam::111122223333:role/terraform-execution --
role-session-name terraform-session-example
```

A saída do comando contém uma chave de acesso, uma chave secreta e um token de sessão que você pode usar para se autenticar AWS em:

```
{
    "AssumedRoleUser": {
        "AssumedRoleId": "AROA3XFRBF535PLBIFPI4:terraform-session-example",
        "Arn": "arn:aws:sts::111122223333:assumed-role/terraform-execution/terraform-
session-example"
    },
    "Credentials": {
        "SecretAccessKey": " wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
        "SessionToken": " AQoEXAMPLEH4aoAH0gNCAPyJxz4BlCFFxWNE1OPTgk5TthT
+FvwqnKwRcOIfrRh3c/LTo6UDdyJwOOvEVPvLXCrrrUtdnniCEXAMPLE/
IvU1dYUg2RVAJBanLiHb4IgRmpRV3zrkuWJOgQs8IZZaIv2BXIa2R4OlgkBN9bkUDNCJiBeb/
AXlzBBko7b15fjrBs2+cTQtpZ3CYWFXG8C5zqx37wnOE49mRl/+OtkIKGO7fAE",
```

```
        "Expiration": "2024-03-15T00:05:07Z",
        "AccessKeyId": "ASIAIOSFODNN7EXAMPLE"
    }
}
```

O AWS provedor também pode assumir automaticamente a função.

Exemplo de configuração de provedor para assumir uma função do IAM:

```
provider "aws" {
  assume_role {
    role_arn     = "arn:aws:iam::111122223333:role/terraform-execution"
    session_name = "terraform-session-example"
  }
}
```

Isso concede privilégios elevados estritamente durante a sessão do Terraform. As chaves temporárias não podem ser vazadas porque elas expiram automaticamente após a duração máxima da sessão.

Os principais benefícios dessa prática recomendada incluem segurança aprimorada em comparação com chaves de acesso de longa duração, controles de acesso refinados na função para obter menos privilégios e a capacidade de revogar facilmente o acesso modificando as permissões da função. Ao usar funções do IAM, você também evita ter que armazenar diretamente segredos localmente em scripts ou em disco, o que ajuda a compartilhar a configuração do Terraform com segurança em toda a equipe.

Use funções do IAM para autenticação do Amazon EC2

Ao executar o Terraform a partir de instâncias do Amazon Elastic Compute Cloud (Amazon EC2), evite armazenar credenciais de longo prazo localmente. Em vez disso, use funções e perfis de instância do IAM para conceder permissões de privilégios mínimos automaticamente.

Primeiro, crie uma função do IAM com as permissões mínimas e atribua a função ao perfil da instância. O perfil da instância permite que as instâncias do EC2 herdem as permissões definidas na função. Em seguida, inicie as instâncias especificando esse perfil de instância. A instância será autenticada por meio da função anexada.

Antes de executar qualquer operação do Terraform, verifique se a função está presente nos metadados da instância para confirmar se as credenciais foram herdadas com sucesso.

```
TOKEN=$(curl -s -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600")

curl -H "X-aws-ec2-metadata-token: $TOKEN" -s http://169.254.169.254/latest/meta-data/
iam/security-credentials/
```

Essa abordagem evita a codificação permanente de AWS chaves permanentes em scripts ou na configuração do Terraform dentro da instância. As credenciais temporárias são disponibilizadas para o Terraform de forma transparente por meio da função e do perfil da instância.

Os principais benefícios dessa prática recomendada incluem maior segurança em relação às credenciais de longo prazo, redução da sobrecarga de gerenciamento de credenciais e consistência entre ambientes de desenvolvimento, teste e produção. A autenticação de função do IAM simplifica as execuções do Terraform a partir de instâncias do EC2 e, ao mesmo tempo, impõe o acesso com privilégios mínimos.

Use credenciais dinâmicas para espaços de trabalho do HCP Terraform

O HCP Terraform é um serviço gerenciado fornecido pela HashiCorp que ajuda as equipes a usar o Terraform para provisionar e gerenciar a infraestrutura em vários projetos e ambientes. Ao executar o Terraform no HCP Terraform, use credenciais dinâmicas para simplificar e proteger a autenticação. AWS O Terraform troca automaticamente credenciais temporárias em cada execução sem precisar assumir a função do IAM.

Os benefícios incluem rotação de segredos mais fácil, gerenciamento centralizado de credenciais em todos os espaços de trabalho, permissões com privilégios mínimos e eliminação de chaves codificadas. Confiar em chaves efêmeras com hash aumenta a segurança em comparação com chaves de acesso de longa duração.

Use funções do IAM em AWS CodeBuild

Em AWS CodeBuild, execute suas compilações usando uma função do IAM atribuída ao CodeBuild projeto. Isso permite que cada compilação herde automaticamente as credenciais temporárias da função em vez de usar chaves de longo prazo.

Execute GitHub ações remotamente no HCP Terraform

Configure fluxos de trabalho do GitHub Actions para executar o Terraform remotamente nos espaços de trabalho do HCP Terraform. Confie em credenciais dinâmicas e no bloqueio remoto de estados em vez do gerenciamento de GitHub segredos.

Use GitHub ações com o OIDC e configure a ação Credenciais AWS

Use o padrão OpenID Connect (OIDC) para federar GitHub a identidade do Actions por meio do IAM. Use a ação Configurar AWS Credenciais para trocar o GitHub token por AWS credenciais temporárias sem precisar de chaves de acesso de longo prazo.

Use GitLab com o OIDC e o AWS CLI

Use o padrão OIDC para federar GitLab identidades por meio do IAM para acesso temporário. Ao confiar no OIDC, você evita ter que gerenciar diretamente as chaves de AWS acesso de longo prazo. GitLab As credenciais são trocadas just-in-time, o que melhora a segurança. Os usuários também obtêm menos privilégios de acesso de acordo com as permissões na função do IAM.

Use usuários exclusivos do IAM com ferramentas de automação legadas

Se você tiver ferramentas e scripts de automação que não têm suporte nativo para o uso de funções do IAM, você pode criar usuários individuais do IAM para conceder acesso programático. O princípio do menor privilégio ainda se aplica. Minimize as permissões de políticas e confie em funções separadas para cada pipeline ou script. À medida que você migra para ferramentas ou scripts mais modernos, comece a apoiar funções de forma nativa e faça a transição gradual para elas.

> **Warning**
>
> Os usuários do IAM têm credenciais de longo prazo, o que representa um risco de segurança. Para ajudar a reduzir esse risco, recomendamos que você forneça a esses usuários somente as permissões necessárias para realizar a tarefa e que você os remova quando não forem mais necessários.

Use o plug-in Jenkins AWS Credentials

Use o plug-in AWS Credentials no Jenkins para configurar centralmente e injetar AWS credenciais em compilações dinamicamente. Isso evita a verificação de segredos no controle de origem.

Monitore, valide e otimize continuamente o menor privilégio

Com o tempo, podem ser concedidas permissões adicionais que podem exceder as políticas mínimas exigidas. Analise continuamente o acesso para identificar e remover quaisquer direitos desnecessários.

Monitore continuamente o uso da chave de acesso

Se você não puder evitar o uso de chaves de acesso, use os relatórios de credenciais do IAM para encontrar chaves de acesso não utilizadas com mais de 90 dias e revogue as chaves inativas em contas de usuário e funções de máquina. Alerte os administradores para que confirmem manualmente a remoção das chaves dos funcionários e sistemas ativos.

O monitoramento do uso da chave ajuda a otimizar as permissões porque você pode identificar e remover direitos não utilizados. Quando você segue essa prática recomendada com a rotação da chave de acesso, ela limita a vida útil da credencial e impõe o acesso com privilégios mínimos.

AWS fornece vários serviços e recursos que você pode usar para configurar alertas e notificações para administradores. Aqui estão algumas opções:

- AWS Config: você pode usar AWS Config regras para avaliar as configurações dos seus AWS recursos, incluindo as chaves de acesso do IAM. Você pode criar regras personalizadas para verificar condições específicas, como chaves de acesso não utilizadas com mais de um número específico de dias. Quando uma regra é violada, AWS Config pode iniciar uma avaliação para remediação ou enviar notificações para um tópico do Amazon Simple Notification Service (Amazon SNS).
- AWS Security Hub CSPM: O Security Hub CSPM fornece uma visão abrangente da postura de segurança da sua AWS conta e pode ajudar a detectar e notificar você sobre possíveis problemas de segurança, incluindo chaves de acesso do IAM não utilizadas ou inativas. O Security Hub CSPM pode se integrar à Amazon EventBridge e ao Amazon SNS ou ao Amazon Q Developer em aplicativos de bate-papo para enviar notificações aos administradores.
- AWS Lambda: As funções Lambda podem ser chamadas por vários eventos, incluindo Amazon CloudWatch Events ou AWS Config regras. Você pode escrever funções personalizadas do Lambda para avaliar o uso da chave de acesso do IAM, realizar verificações adicionais e enviar notificações usando serviços como o Amazon SNS ou o Amazon Q Developer em aplicativos de bate-papo.

Valide continuamente as políticas do IAM

Use o IAM Access Analyzer para avaliar as políticas associadas às funções e identificar quaisquer serviços não utilizados ou ações em excesso que foram concedidas. Implemente análises de acesso periódicas para verificar manualmente se as políticas atendem aos requisitos atuais.

Compare a política existente com a política gerada pelo IAM Access Analyzer e remova todas as permissões desnecessárias. Você também deve fornecer relatórios aos usuários e revogar automaticamente as permissões não utilizadas após um período de carência. Isso ajuda a garantir que políticas mínimas permaneçam em vigor.

A revogação proativa e frequente do acesso obsoleto minimiza as credenciais que podem estar em risco durante uma violação. A automação fornece higiene de credenciais e otimização de permissões sustentáveis e de longo prazo. Seguir essa prática recomendada limita o escopo do impacto ao aplicar proativamente o menor privilégio entre AWS identidades e recursos.

Armazenamento remoto seguro de estado

O armazenamento de estado remoto se refere ao armazenamento do arquivo de estado do Terraform remotamente, em vez de localmente, na máquina em que o Terraform está sendo executado. O arquivo de estado é crucial porque acompanha os recursos que são provisionados pelo Terraform e seus metadados.

A falha em proteger o estado remoto pode levar a problemas sérios, como perda de dados de estado, incapacidade de gerenciar a infraestrutura, exclusão inadvertida de recursos e exposição de informações confidenciais que possam estar presentes no arquivo de estado. Por esse motivo, proteger o armazenamento remoto de estado é crucial para o uso do Terraform em nível de produção.

Ative a criptografia e os controles de acesso

Use a criptografia do lado do servidor (SSE) do Amazon Simple Storage Service (Amazon S3) para criptografar o estado remoto em repouso.

Limite o acesso direto aos fluxos de trabalho colaborativos

- Estruture fluxos de trabalho de colaboração no HCP Terraform ou em um CI/CD pipeline dentro do seu repositório Git para limitar o acesso direto ao estado.

- Confie em pull requests, execute aprovações, verificações de políticas e notificações para coordenar as mudanças.

Seguir essas diretrizes ajuda a proteger atributos confidenciais dos recursos e evita conflitos com as alterações dos membros da equipe. A criptografia e as proteções rígidas de acesso ajudam a reduzir a superfície de ataque, e os fluxos de trabalho de colaboração permitem a produtividade.

Use AWS Secrets Manager

Há muitos recursos e fontes de dados no Terraform que armazenam valores secretos em texto simples no arquivo de estado. Evite armazenar segredos no estado - use em vez AWS Secrets Managerdisso.

Em vez de tentar criptografar manualmente valores confidenciais, confie no suporte integrado do Terraform para gerenciamento de estados confidenciais. Ao exportar valores confidenciais para a saída, verifique se os valores estão marcados como confidenciais.

Examine continuamente a infraestrutura e o código-fonte

Examine proativamente a infraestrutura e o código-fonte continuamente em busca de riscos, como credenciais expostas ou configurações incorretas, para fortalecer sua postura de segurança. Resolva as descobertas imediatamente reconfigurando ou corrigindo recursos.

Use AWS serviços para digitalização dinâmica

Use ferramentas AWS nativas, como Amazon Inspector, Amazon Detective e AWS Security Hub CSPMAmazon, GuardDuty para monitorar a infraestrutura provisionada em várias contas e regiões. Agende escaneamentos recorrentes no Security Hub CSPM para rastrear a implantação e o desvio de configuração. Examine instâncias do EC2, funções Lambda, contêineres, buckets S3 e outros recursos.

Realizar análise estática

Incorpore analisadores estáticos, como o Checkov, diretamente nos CI/CD pipelines para escanear o código de configuração do Terraform (HCL) e identificar riscos preventivamente antes da implantação. Isso move as verificações de segurança para um ponto anterior no processo de desenvolvimento (conhecido como deslocamento para a esquerda) e evita a configuração incorreta da infraestrutura.

Garanta uma remediação imediata

Para todas as descobertas do escaneamento, garanta uma correção imediata atualizando a configuração do Terraform, aplicando patches ou reconfigurando os recursos manualmente, conforme apropriado. Reduza os níveis de risco abordando as causas principais.

O uso da varredura de infraestrutura e da varredura de código fornece uma visão em camadas sobre as configurações do Terraform, os recursos provisionados e o código do aplicativo. Isso maximiza a cobertura de risco e conformidade por meio de controles preventivos, de detecção e reativos, ao mesmo tempo em que incorpora a segurança mais cedo ao ciclo de vida de desenvolvimento de software (SDLC).

Imponha verificações de políticas

Use estruturas de código, como as políticas do HashiCorp Sentinel, para fornecer proteções de governança e modelos padronizados para provisionamento de infraestrutura com o Terraform.

As políticas do Sentinel podem definir requisitos ou restrições na configuração do Terraform para se alinharem aos padrões organizacionais e às melhores práticas. Por exemplo, você pode usar as políticas do Sentinel para:

- Exija tags em todos os recursos.
- Restrinja os tipos de instância a uma lista aprovada.
- Aplique variáveis obrigatórias.
- Evite a destruição dos recursos de produção.

A incorporação de verificações de políticas nos ciclos de vida de configuração do Terraform permite a aplicação proativa de padrões e diretrizes de arquitetura. O Sentinel fornece uma lógica política compartilhada que ajuda a acelerar o desenvolvimento e, ao mesmo tempo, evitar práticas não aprovadas.

Práticas recomendadas de back-end

Usar um back-end remoto adequado para armazenar seu arquivo de estado é fundamental para permitir a colaboração, garantir a integridade do arquivo de estado por meio de bloqueio, fornecer backup e recuperação confiáveis, integrar-se aos CI/CD fluxos de trabalho e aproveitar os recursos avançados de segurança, governança e gerenciamento oferecidos por serviços gerenciados, como o HCP Terraform.

O Terraform oferece suporte a vários tipos de back-end, como Kubernetes, HashiCorp Consul e HTTP. No entanto, este guia se concentra no Amazon S3, que é uma solução de back-end ideal para a maioria dos usuários. AWS

Como um serviço de armazenamento de objetos totalmente gerenciado que oferece alta durabilidade e disponibilidade, o Amazon S3 fornece um back-end seguro, escalável e de baixo custo para gerenciar o estado do Terraform on. AWS A presença global e a resiliência do Amazon S3 excedem o que a maioria das equipes pode alcançar com o autogerenciamento do armazenamento estadual. Além disso, a integração nativa com controles de AWS acesso, opções de criptografia, recursos de controle de versão e outros serviços torna o Amazon S3 uma opção conveniente de back-end.

Este guia não fornece orientação de back-end para outras soluções, como Kubernetes ou Consul, porque o público-alvo principal são os clientes. AWS Para equipes que estão totalmente envolvidas Nuvem AWS, o Amazon S3 geralmente é a escolha ideal em relação aos clusters Kubernetes ou Consul. HashiCorp A simplicidade, a resiliência e a forte AWS integração do armazenamento de estado do Amazon S3 fornecem uma base ideal para a maioria dos usuários que AWS seguem as melhores práticas. As equipes podem aproveitar a durabilidade, as proteções de backup e a disponibilidade dos AWS serviços para manter o estado remoto do Terraform altamente resiliente.

Seguir as recomendações de back-end nesta seção resultará em bases de código mais colaborativas do Terraform, limitando o impacto de erros ou modificações não autorizadas. Ao implementar um back-end remoto bem arquitetado, as equipes podem otimizar os fluxos de trabalho do Terraform.

Melhores práticas:

- Use o Amazon S3 para armazenamento remoto
- Facilite a colaboração em equipe
- Separe os back-ends para cada ambiente
- Monitore ativamente a atividade do estado remoto

Use o Amazon S3 para armazenamento remoto

Armazenar o estado do Terraform remotamente no Amazon S3 e implementar o bloqueio de estado e a verificação de consistência usando o Amazon DynamoDB oferecem grandes benefícios em relação ao armazenamento local de arquivos. O estado remoto permite colaboração em equipe, rastreamento de alterações, proteções de backup e bloqueio remoto para maior segurança.

Usar o Amazon S3 com a classe de armazenamento S3 Standard (padrão) em vez de armazenamento local efêmero ou soluções autogerenciadas fornece 99,999999999% de durabilidade e proteções de disponibilidade de 99,99% para evitar perda acidental de dados estaduais. AWS serviços gerenciados, como o Amazon S3 e o DynamoDB, fornecem acordos de nível de serviço (SLAs) que excedem o que a maioria das organizações pode alcançar ao autogerenciar o armazenamento. Confie nessas proteções para manter os back-ends remotos acessíveis.

Ativar bloqueio remoto de estado

O bloqueio de estado restringe o acesso para evitar operações de gravação simultâneas e reduz os erros de modificações simultâneas feitas por vários usuários. O Terraform oferece suporte a dois mecanismos de bloqueio para back-ends do Amazon S3:

- Bloqueio de estado nativo do Amazon S3 (recomendado): disponível desde o Terraform 1.10.0; usa os recursos de bloqueio nativos do Amazon S3
- Bloqueio de estado do DynamoDB (obsoleto): abordagem antiga que será removida nas futuras versões do Terraform

```
terraform {
  backend "s3" {
    bucket       = "myorg-terraform-states"
    key          = "myapp/production/tfstate"
    region       = "us-east-1"
    use_lockfile = true
  }
}
```

Para compatibilidade com versões anteriores durante a migração, você pode configurar o bloqueio do Amazon S3 e do DynamoDB simultaneamente. No entanto, recomendamos que você migre

para o bloqueio nativo do Amazon S3 porque o bloqueio baseado no DynamoDB está se tornando obsoleto.

Ative o controle de versão e os backups automáticos

Para proteção adicional, habilite o controle de versão e os backups automáticos usando os back-ends do Amazon AWS Backup S3. O controle de versão preserva todas as versões anteriores do estado sempre que alterações são feitas. Também permite restaurar instantâneos anteriores do estado de funcionamento, se necessário, para reverter alterações indesejadas ou se recuperar de acidentes.

Restaure versões anteriores, se necessário

Os buckets de estado versionados do Amazon S3 facilitam a reversão de alterações restaurando um snapshot anterior de estado válido. Isso ajuda a proteger contra alterações acidentais e fornece recursos adicionais de backup.

Use o HCP Terraform

O HCP Terraform fornece uma alternativa de back-end totalmente gerenciada para configurar seu próprio armazenamento de estado. O HCP Terraform gerencia automaticamente o armazenamento seguro de estado e criptografia enquanto desbloqueia recursos adicionais.

Quando você usa o HCP Terraform, o estado é armazenado remotamente por padrão, o que permite o compartilhamento e o bloqueio de estados em toda a sua organização. Controles detalhados de políticas ajudam você a restringir o acesso e as mudanças de estado.

Recursos adicionais incluem integrações de controle de versão, proteções de políticas, automação de fluxo de trabalho, gerenciamento de variáveis e integrações de login único com SAML. Você também pode usar a política do Sentinel como código para implementar controles de governança.

Embora o HCP Terraform exija o uso de uma plataforma de software como serviço (SaaS), para muitas equipes, os benefícios de segurança, controles de acesso, verificações automatizadas de políticas e recursos de colaboração o tornam a escolha ideal em relação ao armazenamento de estado autogerenciado com o Amazon S3 ou o DynamoDB.

A fácil integração com serviços como GitHub e GitLab com configurações menores também atrai usuários que adotam totalmente as ferramentas de nuvem e SaaS para melhorar os fluxos de trabalho da equipe.

Facilite a colaboração em equipe

Use back-ends remotos para compartilhar dados de estado com todos os membros da sua equipe do Terraform. Isso facilita a colaboração porque dá a toda a equipe visibilidade das mudanças na infraestrutura. Protocolos de back-end compartilhados combinados com a transparência do histórico de estados simplificam o gerenciamento interno de mudanças. Todas as mudanças na infraestrutura passam pelo pipeline estabelecido, o que aumenta a agilidade dos negócios em toda a empresa.

Melhore a responsabilidade usando AWS CloudTrail

AWS CloudTrail Integre-se ao bucket do Amazon S3 para capturar chamadas de API feitas ao bucket estadual. Filtre CloudTrail eventos para rastrear `PutObject` `DeleteObject,` e outras chamadas relevantes.

CloudTrail os registros mostram a AWS identidade do principal que fez cada chamada de API para mudança de estado. A identidade do usuário pode ser associada a uma conta de máquina ou a membros da equipe que interagem com o armazenamento de back-end.

Combine CloudTrail os registros com o controle de versão estadual do Amazon S3 para vincular as alterações de infraestrutura ao principal que as aplicou. Ao analisar várias revisões, você pode atribuir qualquer atualização à conta da máquina ou ao membro responsável da equipe.

Se ocorrer uma alteração não intencional ou disruptiva, o controle de versão de estado fornece recursos de reversão. CloudTrail rastreia a alteração até o usuário para que você possa discutir melhorias preventivas.

Também recomendamos que você aplique as permissões do IAM para limitar o acesso ao bucket estadual. No geral, o controle de versão e o CloudTrail monitoramento do S3 oferecem suporte à auditoria em todas as mudanças na infraestrutura. As equipes obtêm melhores recursos de responsabilidade, transparência e auditoria no histórico estadual do Terraform.

Separe os back-ends para cada ambiente

Use back-ends Terraform distintos para cada ambiente de aplicativo. Backends separados isolam o estado entre desenvolvimento, teste e produção.

Reduza o escopo do impacto

O estado de isolamento ajuda a garantir que as mudanças em ambientes inferiores não afetem a infraestrutura de produção. Acidentes ou experimentos em ambientes de desenvolvimento e teste têm impacto limitado.

Restringir acesso à produção

Bloqueie as permissões do back-end do estado de produção para acesso somente leitura para a maioria dos usuários. Limite quem pode modificar a infraestrutura de produção para as funções de CI/CD tubulação e vidro quebrado.

Simplifique os controles de acesso

O gerenciamento de permissões no nível do back-end simplifica o controle de acesso entre ambientes. Usar buckets S3 distintos para cada aplicativo e ambiente significa que amplas permissões de leitura ou gravação podem ser concedidas em buckets de back-end inteiros.

Evite espaços de trabalho compartilhados

Embora você possa usar os espaços de trabalho do Terraform para separar estados entre ambientes, back-ends distintos fornecem um isolamento mais forte. Se você tiver espaços de trabalho compartilhados, os acidentes ainda podem afetar vários ambientes.

Manter os back-ends do ambiente totalmente isolados minimiza o impacto de qualquer falha ou violação. Backends separados também alinham os controles de acesso ao nível de sensibilidade do ambiente. Por exemplo, você pode fornecer proteção de gravação para o ambiente de produção e acesso mais amplo para ambientes de desenvolvimento e teste.

Monitore ativamente a atividade do estado remoto

O monitoramento contínuo da atividade do estado remoto é fundamental para detectar possíveis problemas precocemente. Procure desbloqueios, alterações ou tentativas de acesso anômalos.

Receba alertas sobre desbloqueios suspeitos

A maioria das mudanças de estado deve ocorrer por meio de CI/CD pipelines. Gere alertas se os desbloqueios de estado ocorrerem diretamente nas estações de trabalho do desenvolvedor, o que pode sinalizar alterações não autorizadas ou não testadas.

Monitore as tentativas de acesso

Falhas de autenticação em compartimentos de estado podem indicar atividade de reconhecimento. Observe se várias contas estão tentando acessar o estado ou se aparecem endereços IP incomuns, o que indica credenciais comprometidas.

Melhores práticas para organização e estrutura de base de código

A estrutura e a organização adequadas da base de código são essenciais à medida que o uso do Terraform cresce em grandes equipes e empresas. Uma base de código bem arquitetada permite a colaboração em grande escala e, ao mesmo tempo, aprimora a capacidade de manutenção.

Esta seção fornece recomendações sobre modularidade, convenções de nomenclatura, documentação e padrões de codificação do Terraform que oferecem suporte à qualidade e consistência.

A orientação inclui dividir a configuração em módulos reutilizáveis por ambiente e componentes, estabelecer convenções de nomenclatura usando prefixos e sufixos, documentar módulos e explicar claramente as entradas e saídas e aplicar regras de formatação consistentes usando verificações de estilo automatizadas.

As melhores práticas adicionais abrangem a organização lógica de módulos e recursos em uma hierarquia estruturada, a catalogação de módulos públicos e privados na documentação e a abstração de detalhes de implementação desnecessários em módulos para simplificar o uso.

Ao implementar diretrizes de estrutura de base de código sobre modularidade, documentação, padrões e organização lógica, você pode oferecer suporte à ampla colaboração entre equipes e, ao mesmo tempo, manter o Terraform sustentável à medida que o uso se espalha pela organização. Ao aplicar convenções e padrões, você pode evitar a complexidade de uma base de código fragmentada.

Práticas recomendadas:

- Implemente uma estrutura de repositório padrão
- Estrutura para modularidade
- Siga as convenções de nomenclatura
- Use recursos de anexos
- Use tags padrão
- Atenda aos requisitos de registro do Terraform
- Use fontes de módulos recomendadas
- Siga os padrões de codificação

Implemente uma estrutura de repositório padrão

Recomendamos que você implemente o layout de repositório a seguir. A padronização dessas práticas de consistência em todos os módulos melhora a capacidade de descoberta, a transparência, a organização e a confiabilidade, ao mesmo tempo que permite a reutilização em muitas configurações do Terraform.

- Módulo ou diretório raiz: esse deve ser o ponto de entrada principal para os módulos raiz e reutilizáveis do Terraform e espera-se que seja exclusivo. Se você tiver uma arquitetura mais complexa, poderá usar módulos aninhados para criar abstrações leves. Isso ajuda você a descrever a infraestrutura em termos de sua arquitetura, em vez de diretamente, em termos de objetos físicos.
- README: O módulo raiz e todos os módulos aninhados devem ter arquivos README. Esse arquivo deve ser nomeado`README.md`. Ele deve conter uma descrição do módulo e para que ele deve ser usado. Se você quiser incluir um exemplo de uso desse módulo com outros recursos, coloque-o em um `examples` diretório. Considere incluir um diagrama que descreva os recursos de infraestrutura que o módulo pode criar e seus relacionamentos. Use terraform-docs para gerar automaticamente entradas ou saídas do módulo.
- main.tf: Este é o ponto de entrada principal. Para um módulo simples, todos os recursos podem ser criados nesse arquivo. Para um módulo complexo, a criação de recursos pode estar espalhada por vários arquivos, mas todas as chamadas de módulo aninhadas devem estar no `main.tf` arquivo.
- variables.tf e outputs.tf: esses arquivos contêm as declarações de variáveis e saídas. Todas as variáveis e resultados devem ter descrições de uma ou duas frases que expliquem seu propósito. Essas descrições são usadas para documentação. Para obter mais informações, consulte a HashiCorp documentação para configuração de variáveis e configuração de saída.
  - Todas as variáveis devem ter um tipo definido.
  - A declaração da variável também pode incluir um argumento padrão. Se a declaração incluir um argumento padrão, a variável será considerada opcional e o valor padrão será usado se você não definir um valor ao chamar o módulo ou executar o Terraform. O argumento padrão requer um valor literal e não pode referenciar outros objetos na configuração. Para tornar uma variável obrigatória, omita um padrão na declaração da variável e considere se a configuração `nullable = false` faz sentido.
  - Para variáveis que têm valores independentes do ambiente (como`disk_size`), forneça valores padrão.

  - Para variáveis que tenham valores específicos do ambiente (como`project_id`), não forneça valores padrão. Nesse caso, o módulo de chamada deve fornecer valores significativos.
  - Use padrões vazios para variáveis como cadeias de caracteres ou listas vazias somente quando deixar a variável vazia for uma preferência válida que o subjacente APIs não rejeita.
  - Seja criterioso no uso de variáveis. Parametrize valores somente se eles precisarem variar para cada instância ou ambiente. Ao decidir se deseja expor uma variável, certifique-se de ter um caso de uso concreto para alterar essa variável. Se houver apenas uma pequena chance de que uma variável seja necessária, não a exponha.
    - Adicionar uma variável com um valor padrão é compatível com versões anteriores.
    - A remoção de uma variável é incompatível com versões anteriores.
    - Nos casos em que um literal é reutilizado em vários lugares, você deve usar um valor local sem expô-lo como uma variável.
  - Não passe saídas diretamente por meio de variáveis de entrada, pois isso impede que elas sejam adicionadas adequadamente ao gráfico de dependências. Para garantir que dependências implícitas sejam criadas, certifique-se de que as saídas façam referência aos atributos dos recursos. Em vez de referenciar diretamente uma variável de entrada para uma instância, passe o atributo.
- locals.tf: esse arquivo contém valores locais que atribuem um nome a uma expressão, portanto, um nome pode ser usado várias vezes em um módulo em vez de repetir a expressão. Os valores locais são como as variáveis locais temporárias de uma função. As expressões em valores locais não se limitam a constantes literais; elas também podem fazer referência a outros valores no módulo, incluindo variáveis, atributos de recursos ou outros valores locais, para combiná-los.
- providers.tf: Esse arquivo contém o bloco terraform e os blocos provider. `provider`os blocos devem ser declarados somente nos módulos raiz pelos consumidores dos módulos.

  Se você estiver usando o HCP Terraform, adicione também um bloco de nuvem vazio. O `cloud` bloco deve ser configurado inteiramente por meio de variáveis de ambiente e credenciais de variáveis de ambiente como parte de um CI/CD pipeline.
- versions.tf: Esse arquivo contém o bloco required_providers. Todos os módulos do Terraform devem declarar quais provedores são necessários para que o Terraform possa instalar e usar esses provedores.
- data.tf: para uma configuração simples, coloque as fontes de dados ao lado dos recursos que fazem referência a elas. Por exemplo, se você estiver buscando uma imagem para ser usada na execução de uma instância, coloque-a ao lado da instância em vez de coletar recursos de dados

em seu próprio arquivo. Se o número de fontes de dados ficar muito grande, considere movê-las para um `data.tf` arquivo dedicado.

- arquivos.tfvars: para módulos raiz, você pode fornecer variáveis não confidenciais usando um arquivo. `.tfvars` Para consistência, nomeie os arquivos variáveis`terraform.tfvars`. Coloque valores comuns na raiz do repositório e valores específicos do ambiente na pasta. `envs/`
- Módulos aninhados: os módulos aninhados devem existir no `modules/` subdiretório. Qualquer módulo aninhado que tenha um `README.md` é considerado utilizável por um usuário externo. Se a `README.md` não existir, o módulo será considerado somente para uso interno. Módulos aninhados devem ser usados para dividir o comportamento complexo em vários módulos pequenos que os usuários podem escolher cuidadosamente.

  Se o módulo raiz incluir chamadas para módulos aninhados, essas chamadas devem usar caminhos relativos, por exemplo, `./modules/sample-module` para que o Terraform os considere parte do mesmo repositório ou pacote, em vez de baixá-los novamente separadamente.

  Se um repositório ou pacote contiver vários módulos aninhados, o ideal é que eles possam ser compostos pelo chamador, em vez de chamarem diretamente uns aos outros e criarem uma árvore de módulos profundamente aninhada.
- Exemplos: exemplos de uso de um módulo reutilizável devem existir no `examples/` subdiretório na raiz do repositório. Para cada exemplo, você pode adicionar um README para explicar o objetivo e o uso do exemplo. Exemplos de submódulos também devem ser colocados no `examples/` diretório raiz.

  Como os exemplos geralmente são copiados em outros repositórios para personalização, os blocos de módulos devem ter sua origem definida para o endereço que um chamador externo usaria, não para um caminho relativo.
- Arquivos com nome de serviço: os usuários geralmente desejam separar os recursos do Terraform por serviço em vários arquivos. Essa prática deve ser desencorajada tanto quanto possível, e os recursos devem ser definidos em `main.tf` seu lugar. No entanto, se uma coleção de recursos (por exemplo, funções e políticas do IAM) exceder 150 linhas, é razoável dividi-la em seus próprios arquivos, como`iam.tf`. Caso contrário, todo o código do recurso deverá ser definido no`main.tf`.
- Scripts personalizados: use scripts somente quando necessário. O Terraform não contabiliza nem gerencia o estado dos recursos criados por meio de scripts. Use scripts personalizados somente quando os recursos do Terraform não forem compatíveis com o comportamento desejado. Coloque scripts personalizados chamados pelo Terraform em um `scripts/` diretório.

- Scripts auxiliares: organize scripts auxiliares que não são chamados pelo Terraform em um diretório. `helpers/` Documente scripts auxiliares no `README.md` arquivo com explicações e exemplos de invocações. Se os scripts auxiliares aceitarem argumentos, forneça verificação e `--help` saída de argumentos.
- Arquivos estáticos: arquivos estáticos aos quais o Terraform faz referência, mas não executa (como scripts de inicialização carregados em instâncias do EC2) devem ser organizados em um `files/` diretório. Coloque documentos longos em arquivos externos, separados do HCL. Referencie-os com a função file ().
- Modelos: Para arquivos que a função templatefile do Terraform lê, use a extensão de arquivo. `.tftpl` Os modelos devem ser colocados em um `templates/` diretório.

Estrutura do módulo raiz

O Terraform sempre é executado no contexto de um único módulo raiz. Uma configuração completa do Terraform consiste em um módulo raiz e na árvore de módulos secundários (que inclui os módulos que são chamados pelo módulo raiz, quaisquer módulos chamados por esses módulos e assim por diante).

Exemplo básico do layout do módulo raiz do Terraform:

```
.
### data.tf
### envs
#   ### dev
#   #   ### terraform.tfvars
#   ### prod
#   #   ### terraform.tfvars
#   ### test
#       ### terraform.tfvars
### locals.tf
### main.tf
### outputs.tf
### providers.tf
### README.md
### terraform.tfvars
### variables.tf
### versions.tf
```

Estrutura de módulo reutilizável

Os módulos reutilizáveis seguem os mesmos conceitos dos módulos raiz. Para definir um módulo, crie um novo diretório para ele e coloque os `.tf` arquivos nele, assim como você definiria um módulo raiz. O Terraform pode carregar módulos a partir de caminhos relativos locais ou de repositórios remotos. Se você espera que um módulo seja reutilizado por várias configurações, coloque-o em seu próprio repositório de controle de versão. É importante manter a árvore de módulos relativamente plana para facilitar a reutilização dos módulos em diferentes combinações.

Exemplo básico de layout de módulo reutilizável do Terraform:

```
.
### data.tf
### examples
#   ### multi-az-new-vpc
#   #   ### data.tf
#   #   ### locals.tf
#   #   ### main.tf
#   #   ### outputs.tf
#   #   ### providers.tf
#   #   ### README.md
#   #   ### terraform.tfvars
#   #   ### variables.tf
#   #   ### versions.tf
#   #   ### vpc.tf
#   ### single-az-existing-vpc
#   #   ### data.tf
#   #   ### locals.tf
#   #   ### main.tf
#   #   ### outputs.tf
#   #   ### providers.tf
#   #   ### README.md
#   #   ### terraform.tfvars
#   #   ### variables.tf
#   #   ### versions.tf
### iam.tf
### locals.tf
### main.tf
### outputs.tf
### README.md
### variables.tf
### versions.tf
```

Estrutura para modularidade

Em princípio, você pode combinar quaisquer recursos e outras construções em um módulo, mas o uso excessivo de módulos aninhados e reutilizáveis pode tornar sua configuração geral do Terraform mais difícil de entender e manter, portanto, use esses módulos com moderação.

Quando fizer sentido, divida sua configuração em módulos reutilizáveis que elevam o nível de abstração descrevendo um novo conceito em sua arquitetura que é construído a partir de tipos de recursos.

Ao modularizar sua infraestrutura em definições reutilizáveis, busque conjuntos lógicos de recursos em vez de componentes individuais ou coleções excessivamente complexas.

Não envolva recursos únicos

Você não deve criar módulos que envolvam outros tipos de recursos únicos. Se você tiver problemas para encontrar um nome para seu módulo que seja diferente do nome do tipo de recurso principal dentro dele, seu módulo provavelmente não está criando uma nova abstração – está adicionando complexidade desnecessária. Em vez disso, use o tipo de recurso diretamente no módulo de chamada.

Encapsular relacionamentos lógicos

Agrupe conjuntos de recursos relacionados, como bases de rede, camadas de dados, controles de segurança e aplicativos. Um módulo reutilizável deve encapsular partes da infraestrutura que funcionem juntas para habilitar um recurso.

Mantenha a herança estável

Ao agrupar módulos em subdiretórios, evite aprofundar mais de um ou dois níveis. Estruturas de herança profundamente aninhadas complicam as configurações e a solução de problemas. Os módulos devem se basear em outros módulos, não construir túneis por meio deles.

Ao focar os módulos em agrupamentos lógicos de recursos que representam padrões de arquitetura, as equipes podem configurar rapidamente bases de infraestrutura confiáveis. Equilibre a abstração sem excesso de engenharia ou simplificação.

Recursos de referência em resultados

Para cada recurso definido em um módulo reutilizável, inclua pelo menos uma saída que faça referência ao recurso. Variáveis e saídas permitem inferir dependências entre módulos e recursos. Sem nenhuma saída, os usuários não podem solicitar adequadamente seu módulo em relação às configurações do Terraform.

Módulos bem estruturados que fornecem consistência de ambiente, agrupamentos orientados por propósitos e referências de recursos exportados permitem a colaboração em grande escala com o Terraform em toda a organização. As equipes podem montar a infraestrutura a partir de blocos de construção reutilizáveis.

Não configure provedores

Embora os módulos compartilhados herdem os provedores dos módulos de chamada, os módulos não devem definir as configurações do provedor sozinhos. Evite especificar blocos de configuração do provedor nos módulos. Essa configuração só deve ser declarada uma vez globalmente.

Declare os fornecedores necessários

Embora as configurações do provedor sejam compartilhadas entre os módulos, os módulos compartilhados também devem declarar seus próprios requisitos de provedor. Essa prática permite que o Terraform garanta que haja uma única versão do provedor compatível com todos os módulos na configuração e especifique o endereço de origem que serve como identificador global (independente do módulo) para o provedor. No entanto, os requisitos específicos do provedor do módulo não especificam nenhuma das configurações que determinam quais endpoints remotos o provedor acessará, como um. Região da AWS

Ao declarar os requisitos de versão e evitar a configuração codificada do provedor, os módulos oferecem portabilidade e reutilização em todas as configurações do Terraform usando provedores compartilhados.

Para módulos compartilhados, defina as versões mínimas necessárias do provedor em um bloco required_providers. `versions.tf`

Para declarar que um módulo requer uma versão específica do AWS provedor, use um `required_providers` bloco dentro de um `terraform` bloco:

```
terraform {
```

```
  required_version = ">= 1.0.0"

  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = ">= 4.0.0"
    }
  }
}
```

Se um módulo compartilhado suportar somente uma versão específica do AWS provedor, use o operador de restrição pessimista (~>), que permite que somente o componente da versão mais à direita seja incrementado:

```
terraform {
  required_version = ">= 1.0.0"

  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.0"
    }
  }
}
```

Neste exemplo, `~> 4.0` permite a instalação de `4.57.1` e`4.67.0`, mas não`5.0.0`. Para obter mais informações, consulte Sintaxe de restrição de versão na HashiCorp documentação.

Siga as convenções de nomenclatura

Nomes claros e descritivos simplificam sua compreensão das relações entre os recursos no módulo e a finalidade dos valores de configuração. A consistência com as diretrizes de estilo melhora a legibilidade tanto para os usuários quanto para os mantenedores do módulo.

Siga as diretrizes para nomeação de recursos

- Use snake_case (onde os termos em minúsculas são separados por sublinhados) para que todos os nomes de recursos correspondam aos padrões de estilo do Terraform. Essa prática garante a consistência com a convenção de nomenclatura para tipos de recursos, tipos de fonte de dados e outros valores predefinidos. Essa convenção não se aplica aos argumentos de nomes.

- Para simplificar as referências a um recurso que é o único desse tipo (por exemplo, um único balanceador de carga para um módulo inteiro), nomeie o recurso `main` ou `this` para maior clareza.
- Use nomes significativos que descrevam a finalidade e o contexto do recurso e que ajudem a diferenciar recursos semelhantes (por exemplo, para o banco de dados principal e `primary` `read_replica` para uma réplica de leitura do banco de dados).
- Use nomes singulares, não plurais.
- Não repita o tipo de recurso no nome do recurso.

Siga as diretrizes para nomenclatura de variáveis

- Adicione unidades aos nomes das entradas, variáveis locais e saídas que representem valores numéricos, como tamanho do disco ou tamanho da RAM (por exemplo, `ram_size_gb` para o tamanho da RAM em gigabytes). Essa prática deixa clara a unidade de entrada esperada para os mantenedores da configuração.
- Use unidades binárias, como MiB e GiB, para tamanhos de armazenamento, e unidades decimais, como MB ou GB, para outras métricas.
- Dê nomes positivos às variáveis booleanas, como`enable_external_access`.

Use recursos de anexos

Alguns recursos têm pseudo-recursos incorporados como atributos. Sempre que possível, você deve evitar usar esses atributos de recursos incorporados e, em vez disso, usar o recurso exclusivo para anexar esse pseudo-recurso. Esses relacionamentos de recursos podem causar cause-and-effect problemas exclusivos para cada recurso.

Usando um atributo incorporado (evite esse padrão):

```
resource "aws_security_group" "allow_tls" {
  ...
  ingress {
    description      = "TLS from VPC"
    from_port        = 443
    to_port          = 443
    protocol         = "tcp"
    cidr_blocks      = [aws_vpc.main.cidr_block]
```

```
    ipv6_cidr_blocks = [aws_vpc.main.ipv6_cidr_block]
  }

  egress {
    from_port        = 0
    to_port          = 0
    protocol         = "-1"
    cidr_blocks      = ["0.0.0.0/0"]
    ipv6_cidr_blocks = ["::/0"]
  }
}
```

Usando recursos de anexo (preferencial):

```
resource "aws_security_group" "allow_tls" {
  ...
}

resource "aws_security_group_rule" "example" {
  type              = "ingress"
  description      = "TLS from VPC"
  from_port        = 443
  to_port          = 443
  protocol         = "tcp"
  cidr_blocks      = [aws_vpc.main.cidr_block]
  ipv6_cidr_blocks = [aws_vpc.main.ipv6_cidr_block]
  security_group_id = aws_security_group.allow_tls.id
}
```

Use tags padrão

Atribua tags a todos os recursos que podem aceitar tags. O Terraform AWS Provider tem uma fonte de dados aws_default_tags que você deve usar dentro do módulo raiz.

Considere adicionar as tags necessárias a todos os recursos criados por um módulo do Terraform. Aqui está uma lista de possíveis tags para anexar:

- Nome: Nome do recurso legível por humanos
- AppId: o ID do aplicativo que usa o recurso
- AppRole: A função técnica do recurso; por exemplo, “servidor web” ou “banco de dados”

- AppPurpose: a finalidade comercial do recurso; por exemplo, “interface de usuário de front-end” ou “processador de pagamento”
- Ambiente: o ambiente de software, como desenvolvimento, teste ou produção
- Projeto: Os projetos que usam o recurso
- CostCenter: Quem cobrar pelo uso de recursos

Atenda aos requisitos de registro do Terraform

Um repositório de módulos deve atender a todos os requisitos a seguir para que possa ser publicado em um registro do Terraform.

Você deve sempre seguir esses requisitos, mesmo que não esteja planejando publicar o módulo em um registro a curto prazo. Ao fazer isso, você pode publicar o módulo em um registro posteriormente sem precisar alterar a configuração e a estrutura do repositório.

- Nome do repositório: Para um repositório de módulos, use o nome de três partes`terraform-aws-<NAME>`, que `<NAME>` reflete o tipo de infraestrutura que o módulo gerencia. O `<NAME>` segmento pode conter hífens adicionais (por exemplo,`terraform-aws-iam-terraform-roles`).
- Estrutura padrão do módulo: O módulo deve seguir a estrutura padrão do repositório. Isso permite que o registro inspecione seu módulo e gere documentação, acompanhe o uso de recursos e muito mais.
  - Depois de criar o repositório Git, copie os arquivos do módulo para a raiz do repositório. Recomendamos que você coloque cada módulo destinado a ser reutilizável na raiz de seu próprio repositório, mas você também pode referenciar módulos de subdiretórios.
  - Se você estiver usando o HCP Terraform, publique os módulos que devem ser compartilhados no registro da sua organização. O registro gerencia os downloads e controla o acesso com os tokens da API HCP Terraform, para que os consumidores não precisem acessar o repositório de origem do módulo, mesmo quando executam o Terraform na linha de comando.
- Localização e permissões: o repositório deve estar em um dos seus provedores de sistema de controle de versão (VCS) configurado, e a conta de usuário do HCP Terraform VCS deve ter acesso de administrador ao repositório. O registro precisa de acesso de administrador para criar os webhooks para importar novas versões do módulo.
- tags x.y.z para lançamentos: pelo menos uma tag de lançamento deve estar presente para que você publique um módulo. O registro usa tags de lançamento para identificar as versões do

módulo. Os nomes das tags de lançamento devem usar controle de versão semântico, que você pode opcionalmente prefixar com um v (por exemplo, e). `v1.1.0` `1.1.0` O registro ignora as tags que não se parecem com números de versão. Para obter mais informações sobre a publicação de módulos, consulte a documentação do Terraform.

Para obter mais informações, consulte Preparando um repositório de módulos na documentação do Terraform.

Use fontes de módulos recomendadas

O Terraform usa o `source` argumento em um bloco de módulo para encontrar e baixar o código-fonte de um módulo filho.

Recomendamos que você use caminhos locais para módulos estreitamente relacionados que tenham o objetivo principal de eliminar elementos de código repetidos e usar um registro de módulo nativo do Terraform ou um provedor de VCS para módulos que devem ser compartilhados por várias configurações.

Os exemplos a seguir ilustram os tipos de fonte mais comuns e recomendados para compartilhar módulos. Os módulos de registro oferecem suporte ao controle de versão. Você deve sempre fornecer uma versão específica, conforme mostrado nos exemplos a seguir.

Registro

Registro do Terraform:

```
module "lambda" {
  source = "github.com/terraform-aws-modules/terraform-aws-lambda.git?
ref=e78cdf1f82944897ca6e30d6489f43cf24539374" #--> v4.18.0

  ...

}
```

Ao fixar hashes de confirmação, você pode evitar o desvio de registros públicos que são vulneráveis a ataques na cadeia de suprimentos.

Terraform do HCP:

```
module "eks_karpenter" {
```

```
  source = "app.terraform.io/my-org/eks/aws"
  version = "1.1.0"

  ...

  enable_karpenter = true
}
```

Terraform Enterprise:

```
module "eks_karpenter" {
  source = "terraform.mydomain.com/my-org/eks/aws"
  version = "1.1.0"

  ...

  enable_karpenter = true
}
```

Provedores de VCS

Os provedores de VCS apoiam o `ref` argumento de selecionar uma revisão específica, conforme mostrado nos exemplos a seguir.

GitHub (HTTPS):

```
module "eks_karpenter" {
  source = "github.com/my-org/terraform-aws-eks.git?ref=v1.1.0"

  ...

  enable_karpenter = true
}
```

Repositório Git genérico (HTTPS):

```
module "eks_karpenter" {
  source = "git::https://example.com/terraform-aws-eks.git?ref=v1.1.0"

  ...

  enable_karpenter = true
```

```
}
```

Repositório Git genérico (SSH):

> **Warning**
>
> Você precisa configurar as credenciais para acessar repositórios privados.

```
module "eks_karpenter" {
  source = "git::ssh://username@example.com/terraform-aws-eks.git?ref=v1.1.0"

  ...

  enable_karpenter = true
}
```

Siga os padrões de codificação

Aplique regras e estilos consistentes de formatação do Terraform em todos os arquivos de configuração. Aplique padrões usando verificações de estilo automatizadas em CI/CD pipelines. Quando você incorpora as melhores práticas de codificação nos fluxos de trabalho da equipe, as configurações permanecem legíveis, sustentáveis e colaborativas à medida que o uso se espalha amplamente pela organização.

Siga as diretrizes de estilo

- Formate todos os arquivos (`.tf`arquivos) do Terraform com o comando terraform fmt para corresponder HashiCorp aos padrões de estilo.
- Use o comando terraform validate para verificar a sintaxe e a estrutura da sua configuração.
- Analise estaticamente a qualidade do código usando TFLint. Esse linter verifica as melhores práticas do Terraform, além da formatação, e falha nas compilações quando encontra erros.

Configurar ganchos de pré-confirmação

Configure ganchos de pré-confirmação do lado do cliente que executam`terraform fmt`, `tflint`e `checkov`, e outras digitalizações de código e verificações de estilo antes de permitir

confirmações. Essa prática ajuda você a validar a conformidade com os padrões mais cedo nos fluxos de trabalho do desenvolvedor.

Use estruturas de pré-confirmação, como pré-confirmação, para adicionar linting, formatação e digitalização de código do Terraform como ganchos em sua máquina local. Os ganchos são executados em cada confirmação do Git e falham na confirmação se as verificações não forem aprovadas.

Mover as verificações de estilo e qualidade para ganchos locais de pré-confirmação fornece feedback rápido aos desenvolvedores antes que as mudanças sejam introduzidas. Os padrões se tornam parte do fluxo de trabalho de codificação.

Práticas recomendadas para gerenciamento AWS de versões do Provider

O gerenciamento cuidadoso das versões do AWS Provider e dos módulos Terraform associados é fundamental para a estabilidade. Esta seção descreve as melhores práticas em relação às restrições de versão e atualizações.

Melhores práticas:

- Adicionar verificações automáticas de versão
- Monitore novos lançamentos
- Contribua com fornecedores

Adicionar verificações automáticas de versão

Adicione verificações de versão para provedores do Terraform em seus pipelines de CI/CD para validar a fixação da versão e falhe nas compilações se a versão for indefinida.

- Adicione verificações de TFlint em pipelines de CI/CD para verificar as versões do provedor que não tenham restrições fixas de versão principal/secundária definidas. Use o plug-in de conjunto de regras TFlint para o Terraform AWS Provider, que fornece regras para detectar possíveis erros e verifica as melhores práticas sobre recursos. AWS
- Falha em execuções de CI que detectam versões não fixadas do provedor para evitar que atualizações implícitas cheguem à produção.

Monitore novos lançamentos

- Monitore as notas de lançamento e os feeds do changelog do provedor. Receba notificações sobre novos lançamentos principais/secundários.
- Avalie as notas de versão em busca de possíveis alterações significativas e avalie seu impacto em sua infraestrutura existente.
- Atualize primeiro as versões secundárias em ambientes que não sejam de produção para validá-las antes de atualizar o ambiente de produção.

Ao automatizar as verificações de versão em pipelines e monitorar novas versões, você pode detectar atualizações não suportadas com antecedência e dar tempo às suas equipes para avaliar o impacto de novas versões principais/secundárias antes de atualizar os ambientes de produção.

Contribua com fornecedores

Contribua ativamente com o HashiCorp AWS Provedor relatando defeitos ou solicitando recursos em GitHub problemas:

- Abra problemas bem documentados no repositório do AWS Provider para detalhar quaisquer bugs encontrados ou funcionalidades que estejam faltando. Forneça etapas reproduzíveis.
- Solicite e vote em aprimoramentos para expandir os recursos do AWS provedor para gerenciar novos serviços.
- Consulte pull requests emitidos ao contribuir com propostas de correções para defeitos ou aprimoramentos do fornecedor. Link para problemas relacionados.
- Siga as diretrizes de contribuição no repositório para convenções de codificação, padrões de teste e documentação.

Ao retribuir aos provedores que você usa, você pode fornecer informações diretas em seu roteiro e ajudar a melhorar sua qualidade e seus recursos para todos os usuários.

Melhores práticas para módulos comunitários

Usar módulos de forma eficaz é fundamental para gerenciar configurações complexas do Terraform e promover a reutilização. Esta seção fornece as melhores práticas sobre módulos, dependências, fontes, abstração e contribuições da comunidade.

Melhores práticas:

- Descubra os módulos da comunidade
- Entenda as dependências
- Use fontes confiáveis
- Contribua com os módulos da comunidade

Descubra os módulos da comunidade

Pesquise no Registro do Terraform e em outras fontes os AWS módulos existentes que possam resolver seu caso de uso antes de criar um novo módulo. GitHub Procure opções populares que tenham atualizações recentes e sejam mantidas ativamente.

Use variáveis para personalização

Ao usar módulos comunitários, passe as entradas por meio de variáveis em vez de bifurcar ou modificar diretamente o código-fonte. Substitua os padrões quando necessário, em vez de alterar os componentes internos do módulo.

A bifurcação deve se limitar a contribuir com correções ou recursos para o módulo original para beneficiar a comunidade em geral.

Entenda as dependências

Antes de usar o módulo, revise o código-fonte e a documentação para identificar dependências:

- Provedores obrigatórios: observe as versões do AWS Kubernetes ou de outros provedores que o módulo exige.
- Módulos aninhados: verifique se há outros módulos usados internamente que introduzem dependências em cascata.

- Fontes de dados externas: observe as APIs, plug-ins personalizados ou dependências de infraestrutura nas quais o módulo depende.

Ao mapear a árvore completa de dependências diretas e indiretas, você pode evitar surpresas ao usar o módulo.

Use fontes confiáveis

O fornecimento de módulos do Terraform de editores não verificados ou desconhecidos apresenta um risco significativo. Use módulos somente de fontes confiáveis.

- Dê preferência aos módulos certificados do Terraform Registry que são publicados por criadores verificados, como AWS ou HashiCorp parceiros.
- Para módulos personalizados, revise o histórico do editor, os níveis de suporte e a reputação de uso, mesmo que o módulo seja da sua própria organização.

Ao não permitir módulos de fontes desconhecidas ou não verificadas, você pode reduzir o risco de injetar vulnerabilidades ou problemas de manutenção em seu código.

Assinar notificações do

Inscreva-se para receber notificações de lançamentos de novos módulos de editores confiáveis:

- Observe os repositórios do GitHub módulo para receber alertas sobre novas versões do módulo.
- Monitore os blogs e registros de alterações dos editores em busca de atualizações.
- Receba notificações proativas para novas versões de fontes verificadas e altamente avaliadas, em vez de obter atualizações implicitamente.

O consumo de módulos somente de fontes confiáveis e o monitoramento de alterações fornecem estabilidade e segurança. Módulos aprovados aumentam a produtividade e minimizam o risco da cadeia de suprimentos.

Contribua com os módulos da comunidade

Envie correções e aprimoramentos para os módulos da comunidade hospedados em: GitHub

- Abra solicitações pull em módulos para resolver defeitos ou limitações que você encontra em seu uso.
- Solicite que novas configurações de melhores práticas sejam adicionadas aos módulos OSS existentes criando problemas.

Contribuir com os módulos da comunidade aprimora os padrões codificados e reutilizáveis para todos os profissionais do Terraform.

Perguntas frequentes

P: Por que focar no AWS provedor?

R. O AWS Provider é um dos provedores mais amplamente usados e complexos para provisionamento de infraestrutura no Terraform. Seguir essas melhores práticas ajuda os usuários a otimizar o uso do provedor para o AWS ambiente.

P: Sou novato no Terraform. Posso usar este guia?

R. O guia é para pessoas que são novas no Terraform, bem como para profissionais mais avançados que desejam aprimorar suas habilidades. As práticas melhoram os fluxos de trabalho dos usuários em qualquer estágio do aprendizado.

P: Quais são algumas das principais práticas recomendadas abordadas?

R. As principais práticas recomendadas incluem o uso de funções do IAM em vez de chaves de acesso, fixação de versões, incorporação de testes automatizados, bloqueio remoto de estados, rotação de credenciais, contribuição de volta aos provedores e organização lógica de bases de código.

P: Onde posso aprender mais sobre o Terraform?

R. A seção Recursos inclui links para a documentação oficial do HashiCorp Terraform e fóruns da comunidade. Use os links para saber mais sobre os fluxos de trabalho avançados do Terraform.

Próximas etapas

Aqui estão algumas das próximas etapas possíveis depois de ler este guia:

- Se você já tiver uma base de código do Terraform, revise sua configuração e identifique as áreas que poderiam ser aprimoradas com base nas recomendações fornecidas neste guia. Por exemplo, analise as melhores práticas para implementar back-ends remotos, separar o código em módulos, usar a fixação de versões e assim por diante, e valide-as em sua configuração.
- Se você não tiver uma base de código existente do Terraform, use essas melhores práticas ao estruturar sua nova configuração. Siga os conselhos sobre gerenciamento de estado, autenticação, estrutura de código e assim por diante desde o início.
- Tente usar alguns dos módulos da HashiCorp comunidade mencionados neste guia para ver se eles simplificam seus padrões de arquitetura. Os módulos permitem níveis mais altos de abstração, para que você não precise reescrever recursos comuns.
- Habilite linting, verificações de segurança, verificações de políticas e ferramentas de teste automatizadas para reforçar algumas das melhores práticas em relação à segurança, conformidade e qualidade do código. Ferramentas como TFlint, tfsec e Checkov podem ajudar.
- Analise a documentação mais recente do AWS Provider para ver se há novos recursos ou funcionalidades que possam ajudar a otimizar o uso do Terraform. Mantenha-se atualizado sobre as novas versões do AWS Provedor.
- Para obter orientação adicional, consulte a documentação, o guia de melhores práticas e o guia de estilo do Terraform no HashiCorp site.

Recursos

Referências

Os links a seguir fornecem material de leitura adicional para o Terraform AWS Provider e usar o Terraform for IaC on. AWS

- Terraform AWS Provider (HashiCorp documentação)
- Módulos Terraform para AWS serviços (Terraform Registry)
- The AWS and HashiCorp Partnership (postagem HashiCorp no blog)
- Credenciais dinâmicas com o AWS provedor (documentação do HCP Terraform)
- Bloqueio de estado do DynamoDB (documentação do Terraform)
- Aplique a política com o Sentinel (documentação do Terraform)

Ferramentas

As ferramentas a seguir ajudam a melhorar a qualidade do código e a automação das configurações do Terraform no AWS, conforme recomendado neste guia de melhores práticas.

Qualidade do código:

- Checkov: Escaneia o código do Terraform para identificar configurações incorretas antes da implantação.
- TFlint: identifica possíveis erros, sintaxe obsoleta e declarações não utilizadas. Esse linter também pode aplicar as AWS melhores práticas e convenções de nomenclatura.
- terraform-docs: gera documentação dos módulos do Terraform em vários formatos de saída.

Ferramentas de automação:

- HCP Terraform: ajuda as equipes a criar versões, colaborar e criar fluxos de trabalho do Terraform com verificações de políticas e portas de aprovação.
- Atlantis: uma ferramenta de automação de pull request do Terraform de código aberto para validar alterações no código.

- CDK para Terraform: uma estrutura que permite usar linguagens familiares, como TypeScript Python, Java, C# e Go, em vez da Linguagem de HashiCorp Configuração (HCL) para definir, provisionar e testar sua infraestrutura do Terraform como código.

Histórico do documento

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

Alteração	Descrição	Data
Atualização de bloqueio remoto de estado	A seção de melhores práticas de back-end foi atualizada para refletir o bloqueio de estado nativo do Amazon S3, que agora é a abordagem recomendada.	26 de agosto de 2025
Publicação inicial	—	28 de maio de 2024

AWS Glossário de orientação prescritiva

A seguir estão os termos comumente usados em estratégias, guias e padrões fornecidos pela Orientação AWS Prescritiva. Para sugerir entradas, use o link Fornecer feedback no final do glossário.

Números

7 Rs

Sete estratégias comuns de migração para mover aplicações para a nuvem. Essas estratégias baseiam-se nos 5 Rs identificados pela Gartner em 2011 e consistem em:

- Refatorar/rearquitetar: mova uma aplicação e modifique sua arquitetura aproveitando ao máximo os recursos nativos de nuvem para melhorar a agilidade, a performance e a escalabilidade. Isso normalmente envolve a portabilidade do sistema operacional e do banco de dados. Exemplo: migrar seu banco de dados Oracle on-premises para o Amazon Aurora Edição Compatível com PostgreSQL.
- Redefinir a plataforma (mover e redefinir [mover e redefinir (lift-and-reshape]): mova uma aplicação para a nuvem e introduza algum nível de otimização a fim de aproveitar os recursos da nuvem. Exemplo: migrar seu banco de dados Oracle on-premises para o Amazon Relational Database Service (Amazon RDS) para Oracle na Nuvem AWS.
- Recomprar (drop and shop): mude para um produto diferente, normalmente migrando de uma licença tradicional para um modelo SaaS. Exemplo: migrar seu sistema de gerenciamento de relacionamento com o cliente (CRM) para o Salesforce.com.
- Redefinir a hospedagem (mover sem alterações [lift-and-shift])mover uma aplicação para a nuvem sem fazer nenhuma alteração a fim de aproveitar os recursos da nuvem. Exemplo: migrar seu banco de dados Oracle on-premises para o Oracle em uma instância do EC2 na Nuvem AWS.
- Realocar (mover o hipervisor sem alterações [hypervisor-level lift-and-shift]): mover a infraestrutura para a nuvem sem comprar novo hardware, reescrever aplicações ou modificar suas operações existentes. Você migra servidores de uma plataforma on-premises para um serviço de nuvem para a mesma plataforma. Exemplo: Migrar um Microsoft Hyper-V aplicativo para o. AWS
- Reter (revisitar): mantenha as aplicações em seu ambiente de origem. Isso pode incluir aplicações que exigem grande refatoração, e você deseja adiar esse trabalho para um

momento posterior, e aplicações antigas que você deseja manter porque não há justificativa comercial para migrá-las.

- Retirar: desative ou remova aplicações que não são mais necessárias em seu ambiente de origem.

A

ABAC

Consulte controle de acesso baseado em atributo.

serviços abstraídos

Veja serviços gerenciados.

ACID

Veja atomicidade, consistência, isolamento, durabilidade.

migração ativa-ativa

Um método de migração de banco de dados no qual os bancos de dados de origem e de destino são mantidos em sincronia (por meio de uma ferramenta de replicação bidirecional ou operações de gravação dupla), e ambos os bancos de dados lidam com transações de aplicações conectadas durante a migração. Esse método oferece suporte à migração em lotes pequenos e controlados, em vez de exigir uma substituição única. É mais flexível, mas exige mais trabalho do que a migração ativa-passiva.

migração ativa-passiva

Um método de migração de banco de dados em que os bancos de dados de origem e de destino são mantidos em sincronia, mas somente o banco de dados de origem manipula as transações das aplicações conectadas, enquanto os dados são replicados no banco de dados de destino. O banco de dados de destino não aceita nenhuma transação durante a migração.

AGGREGATE FUNCTION

Uma função SQL que opera em um grupo de linhas e calcula um único valor de retorno para o grupo. Exemplos de funções agregadas incluem `SUM` e `MAX`.

AI

Veja inteligência artificial.

AIOps

Veja operações de inteligência artificial.

anonimização

O processo de excluir permanentemente informações pessoais em um conjunto de dados. A anonimização pode ajudar a proteger a privacidade pessoal. Dados anônimos não são mais considerados dados pessoais.

antipadrões

Uma solução frequentemente usada para um problema recorrente em que a solução é contraproducente, ineficaz ou menos eficaz do que uma alternativa.

controle de aplicações

Uma abordagem de segurança que permite o uso somente de aplicações aprovadas para ajudar a proteger um sistema contra malware.

portfólio de aplicações

Uma coleção de informações detalhadas sobre cada aplicação usada por uma organização, incluindo o custo para criar e manter a aplicação e seu valor comercial. Essas informações são fundamentais para o processo de descoberta e análise de portfólio e ajudam a identificar e priorizar as aplicações a serem migradas, modernizadas e otimizadas.

inteligência artificial (IA)

O campo da ciência da computação que se dedica ao uso de tecnologias de computação para desempenhar funções cognitivas normalmente associadas aos humanos, como aprender, resolver problemas e reconhecer padrões. Para obter mais informações, consulte O que é inteligência artificial?

operações de inteligência artificial (AIOps)

O processo de usar técnicas de machine learning para resolver problemas operacionais, reduzir incidentes operacionais e intervenção humana e aumentar a qualidade do serviço. Para obter mais informações sobre como AIOps é usado na estratégia de AWS migração, consulte o guia de integração de operações.

criptografia assimétrica

Um algoritmo de criptografia que usa um par de chaves, uma chave pública para criptografia e uma chave privada para descriptografia. É possível compartilhar a chave pública porque ela não é usada na descriptografia, mas o acesso à chave privada deve ser altamente restrito.

atomicidade, consistência, isolamento, durabilidade (ACID)

Um conjunto de propriedades de software que garantem a validade dos dados e a confiabilidade operacional de um banco de dados, mesmo no caso de erros, falhas de energia ou outros problemas.

controle de acesso por atributo (ABAC)

A prática de criar permissões minuciosas com base nos atributos do usuário, como departamento, cargo e nome da equipe. Para obter mais informações, consulte ABAC AWS na documentação AWS Identity and Access Management (IAM).

fonte de dados autorizada

Um local onde você armazena a versão principal dos dados, que é considerada a fonte de informações mais confiável. Você pode copiar dados da fonte de dados autorizada para outros locais com o objetivo de processar ou modificar os dados, como anonimizá-los, redigi-los ou pseudonimizá-los.

Zona de disponibilidade

Um local distinto dentro de um Região da AWS que está isolado de falhas em outras zonas de disponibilidade e fornece conectividade de rede barata e de baixa latência a outras zonas de disponibilidade na mesma região.

AWS Estrutura de adoção da nuvem (AWS CAF)

Uma estrutura de diretrizes e melhores práticas AWS para ajudar as organizações a desenvolver um plano eficiente e eficaz para migrar com sucesso para a nuvem. AWS O CAF organiza a orientação em seis áreas de foco chamadas perspectivas: negócios, pessoas, governança, plataforma, segurança e operações. As perspectivas de negócios, pessoas e governança têm como foco habilidades e processos de negócios; as perspectivas de plataforma, segurança e operações concentram-se em habilidades e processos técnicos. Por exemplo, a perspectiva das pessoas tem como alvo as partes interessadas que lidam com recursos humanos (RH), funções de pessoal e gerenciamento de pessoal. Nessa perspectiva, o AWS CAF fornece orientação para desenvolvimento, treinamento e comunicação de pessoas para ajudar a preparar a organização para a adoção bem-sucedida da nuvem. Para obter mais informações, consulte o site da AWS CAF e o whitepaper da AWS CAF.

AWS Estrutura de qualificação da carga de trabalho (AWS WQF)

Uma ferramenta que avalia as cargas de trabalho de migração do banco de dados, recomenda estratégias de migração e fornece estimativas de trabalho. AWS O WQF está incluído com AWS

Schema Conversion Tool ()AWS SCT. Ela analisa esquemas de banco de dados e objetos de código, código de aplicações, dependências e características de performance, além de fornecer relatórios de avaliação.

B

bot malicioso

Um bot destinado a causar disrupção ou danos a indivíduos ou organizações.

BCP

Veja planejamento de continuidade de negócios

gráfico de comportamento

Uma visualização unificada e interativa do comportamento e das interações de recursos ao longo do tempo. É possível usar um gráfico de comportamento com o Amazon Detective para examinar tentativas de login malsucedidas, chamadas de API suspeitas e ações similares. Para obter mais informações, consulte Dados em um gráfico de comportamento na documentação do Detective.

sistema big-endian

Um sistema que armazena o byte mais significativo antes. Veja também endianness.

classificação binária

Um processo que prevê um resultado binário (uma de duas classes possíveis). Por exemplo, seu modelo de ML pode precisar prever problemas como “Este e-mail é ou não é spam?” ou “Este produto é um livro ou um carro?”

filtro de bloom

Uma estrutura de dados probabilística e eficiente em termos de memória que é usada para testar se um elemento é membro de um conjunto.

blue/green deployment (implantação azul/verde)

Uma estratégia de implantação em que você cria dois ambientes separados, mas idênticos. Você executa a versão atual da aplicação em um ambiente (azul) e a nova versão da aplicação no outro ambiente (verde). Essa estratégia ajuda você a reverter rapidamente com o mínimo de impacto.

bot

Uma aplicação de software que executa tarefas automatizadas na internet e simula a atividade ou interação humana. Alguns bots são úteis ou benéficos, como crawlers da web que indexam informações na internet. Outros bots, conhecidos como bots maliciosos, têm como objetivo causar disrupção ou danos a indivíduos ou organizações.

botnet

Redes de bots infectadas por malware e sob o controle de uma única parte, conhecidas como bot herder ou operador de bots. Os botnets são o mecanismo mais conhecido para escalar bots e seu impacto.

ramo

Uma área contida de um repositório de código. A primeira ramificação criada em um repositório é a ramificação principal. Você pode criar uma nova ramificação a partir de uma ramificação existente e, em seguida, desenvolver recursos ou corrigir bugs na nova ramificação. Uma ramificação que você cria para gerar um recurso é comumente chamada de ramificação de recurso. Quando o recurso estiver pronto para lançamento, você mesclará a ramificação do recurso de volta com a ramificação principal. Para obter mais informações, consulte Sobre filiais (GitHub documentação).

Acesso de emergência

Em circunstâncias excepcionais e por meio de um processo aprovado, um meio rápido para um usuário obter acesso a um Conta da AWS que ele normalmente não tem permissão para acessar. Para obter mais informações, consulte o indicador Implement break-glass procedures nas orientações do AWS Well-Architected.

estratégia brownfield

A infraestrutura existente em seu ambiente. Ao adotar uma estratégia brownfield para uma arquitetura de sistema, você desenvolve a arquitetura de acordo com as restrições dos sistemas e da infraestrutura atuais. Se estiver expandindo a infraestrutura existente, poderá combinar as estratégias brownfield e greenfield.

cache do buffer

A área da memória em que os dados acessados com mais frequência são armazenados.

capacidade de negócios

O que uma empresa faz para gerar valor (por exemplo, vendas, atendimento ao cliente ou marketing). As arquiteturas de microsserviços e as decisões de desenvolvimento podem

ser orientadas por recursos de negócios. Para obter mais informações, consulte a seção Organizados de acordo com as capacidades de negócios do whitepaper Executar microsserviços conteinerizados na AWS.

planejamento de continuidade de negócios (BCP)

Um plano que aborda o impacto potencial de um evento disruptivo, como uma migração em grande escala, nas operações e permite que uma empresa retome as operações rapidamente.

C

CAF

Veja AWS Cloud Adoption Framework.

implantação canário

O lançamento lento e incremental de uma versão para usuários finais. Quando estiver confiante, você implanta a nova versão e substitui a versão atual por completo.

CCoE

Veja Centro de Excelência da Nuvem.

CDC

Veja captura de dados de alteração.

captura de dados de alterações (CDC)

O processo de rastrear alterações em uma fonte de dados, como uma tabela de banco de dados, e registrar metadados sobre a alteração. É possível usar o CDC para várias finalidades, como auditar ou replicar alterações em um sistema de destino para manter a sincronização.

engenharia do caos

Introduzir intencionalmente falhas ou eventos disruptivos para testar a resiliência de um sistema. Você pode usar AWS Fault Injection Service (AWS FIS) para realizar experimentos que estressam suas AWS cargas de trabalho e avaliar sua resposta.

CI/CD

Veja integração e entrega contínuas.

classificação

Um processo de categorização que ajuda a gerar previsões. Os modelos de ML para problemas de classificação predizem um valor discreto. Os valores discretos são sempre diferentes uns dos outros. Por exemplo, um modelo pode precisar avaliar se há ou não um carro em uma imagem.

criptografia no lado do cliente

Criptografia de dados localmente, antes que o alvo os AWS service (Serviço da AWS) receba.

Centro de excelência em nuvem (CCoE)

Uma equipe multidisciplinar que impulsiona os esforços de adoção da nuvem em toda a organização, incluindo o desenvolvimento de práticas recomendas de nuvem, a mobilização de recursos, o estabelecimento de cronogramas de migração e a liderança da organização em transformações em grande escala. Para obter mais informações, consulte as publicações CCo E no blog de estratégia Nuvem AWS corporativa.

computação em nuvem

A tecnologia de nuvem normalmente usada para armazenamento de dados remoto e gerenciamento de dispositivos de IoT. A computação em nuvem é normalmente conectada à tecnologia de computação de borda.

modelo operacional em nuvem

Em uma organização de TI, o modelo operacional usado para criar, amadurecer e otimizar um ou mais ambientes de nuvem. Para obter mais informações, consulte Criar seu modelo operacional de nuvem.

estágios de adoção da nuvem

As quatro fases pelas quais as organizações normalmente passam ao migrar para a Nuvem AWS:

- Projeto: executar alguns projetos relacionados à nuvem para fins de prova de conceito e aprendizado
- Fundação — Fazer investimentos fundamentais para escalar sua adoção da nuvem (por exemplo, criar uma landing zone, definir um CCo E, estabelecer um modelo de operações)
- Migração: migrar aplicações individuais
- Reinvenção: otimizar produtos e serviços e inovar na nuvem

Esses estágios foram definidos por Stephen Orban na postagem do blog The Journey Toward Cloud-First & the Stages of Adoption no blog de estratégia Nuvem AWS empresarial. Para obter

informações sobre como eles se relacionam com a estratégia de AWS migração, consulte o guia de preparação para migração.

CMDB

Veja banco de dados de gerenciamento de configuração.

repositório de código

Um local onde o código-fonte e outros ativos, como documentação, amostras e scripts, são armazenados e atualizados por meio de processos de controle de versão. Os repositórios de nuvem comuns incluem o GitHub ou o Bitbucket Cloud. Cada versão do código é chamada de ramificação. Em uma estrutura de microsserviços, cada repositório é dedicado a uma única peça de funcionalidade. Um único pipeline de CI/CD pode usar vários repositórios.

cache frio

Um cache de buffer que está vazio, não está bem preenchido ou contém dados obsoletos ou irrelevantes. Isso afeta a performance porque a instância do banco de dados deve ler da memória principal ou do disco, um processo que é mais lento do que a leitura do cache do buffer.

dados frios

Dados que raramente são acessados e geralmente são históricos. Ao consultar esse tipo de dados, consultas lentas geralmente são aceitáveis. Mover esses dados para níveis ou classes de armazenamento de baixo desempenho e menos caros pode reduzir os custos.

visão computacional (CV)

Um campo de IA que usa machine learning para analisar e extrair informações de formatos visuais, como vídeos e imagens digitais. Por exemplo, a Amazon SageMaker AI fornece algoritmos de processamento de imagem para CV.

desvio de configuração

Em uma workload, uma alteração de configuração em relação ao estado esperado. Isso pode fazer com que a workload se torne incompatível e, normalmente, é gradual e não intencional.

banco de dados de gerenciamento de configuração (CMDB)

Um repositório que armazena e gerencia informações sobre um banco de dados e seu ambiente de TI, incluindo componentes de hardware e software e suas configurações. Normalmente, os dados de um CMDB são usados no estágio de descoberta e análise do portfólio da migração.

pacote de conformidade

Um conjunto de AWS Config regras e ações de remediação que você pode montar para personalizar suas verificações de conformidade e segurança. Você pode implantar um pacote de conformidade como uma entidade única em uma Conta da AWS região ou em uma organização usando um modelo YAML. Para obter mais informações, consulte Pacotes de conformidade na documentação. AWS Config

integração contínua e entrega contínua (CI/CD)

O processo de automatizar os estágios de origem, criação, teste, preparação e produção do processo de lançamento do software. CI/CD é comumente descrito como um pipeline. CI/CD pode ajudá-lo a automatizar processos, melhorar a produtividade, melhorar a qualidade do código e entregar com mais rapidez. Para obter mais informações, consulte Benefícios da entrega contínua. CD também pode significar implantação contínua. Para obter mais informações, consulte Entrega contínua versus implantação contínua.

CV

Veja visão computacional.

D

dados em repouso

Dados estacionários em sua rede, por exemplo, dados que estão em um armazenamento.

classificação de dados

Um processo para identificar e categorizar os dados em sua rede com base em criticalidade e confidencialidade. É um componente crítico de qualquer estratégia de gerenciamento de riscos de segurança cibernética, pois ajuda a determinar os controles adequados de proteção e retenção para os dados. A classificação de dados é um componente do pilar de segurança no AWS Well-Architected Framework. Para obter mais informações, consulte Classificação de dados.

desvio de dados

Uma variação significativa entre os dados de produção e os dados usados para treinar um modelo de ML ou uma alteração significativa nos dados de entrada ao longo do tempo. O desvio de dados pode reduzir a qualidade geral, a precisão e a imparcialidade das previsões do modelo de ML.

dados em trânsito

Dados que estão se movendo ativamente pela sua rede, como entre os recursos da rede.

data mesh

Um framework de arquitetura que fornece propriedade de dados distribuída e descentralizada com gerenciamento e governança centralizados.

minimização de dados

O princípio de coletar e processar apenas os dados estritamente necessários. Praticar a minimização de dados no Nuvem AWS pode reduzir os riscos de privacidade, os custos e a pegada de carbono de sua análise.

perímetro de dados

Um conjunto de proteções preventivas em seu AWS ambiente que ajudam a garantir que somente identidades confiáveis acessem recursos confiáveis das redes esperadas. Para obter mais informações, consulte Construindo um perímetro de dados em. AWS

pré-processamento de dados

A transformação de dados brutos em um formato que seja facilmente analisado por seu modelo de ML. O pré-processamento de dados pode significar a remoção de determinadas colunas ou linhas e o tratamento de valores ausentes, inconsistentes ou duplicados.

proveniência dos dados

O processo de rastrear a origem e o histórico dos dados ao longo de seu ciclo de vida, por exemplo, como os dados foram gerados, transmitidos e armazenados.

titular dos dados

Um indivíduo cujos dados estão sendo coletados e processados.

data warehouse

Um sistema de gerenciamento de dados compatível com business intelligence, como analytics. Os data warehouses geralmente contêm grandes quantidades de dados históricos e geralmente são usados para consultas e análises.

linguagem de definição de dados (DDL)

Instruções ou comandos para criar ou modificar a estrutura de tabelas e objetos em um banco de dados.

linguagem de manipulação de dados (DML)

Instruções ou comandos para modificar (inserir, atualizar e excluir) informações em um banco de dados.

DDL

Veja linguagem de definição de banco de dados.

deep ensemble

A combinação de vários modelos de aprendizado profundo para gerar previsões. Os deep ensembles podem ser usados para produzir uma previsão mais precisa ou para estimar a incerteza nas previsões.

Aprendizado profundo

Um subcampo do ML que usa várias camadas de redes neurais artificiais para identificar o mapeamento entre os dados de entrada e as variáveis-alvo de interesse.

defense-in-depth

Uma abordagem de segurança da informação na qual uma série de mecanismos e controles de segurança são cuidadosamente distribuídos por toda a rede de computadores para proteger a confidencialidade, a integridade e a disponibilidade da rede e dos dados nela contidos. Ao adotar essa estratégia AWS, você adiciona vários controles em diferentes camadas da AWS Organizations estrutura para ajudar a proteger os recursos. Por exemplo, uma defense-in-depth abordagem pode combinar autenticação multifatorial, segmentação de rede e criptografia.

administrador delegado

Em AWS Organizations, um serviço compatível pode registrar uma conta de AWS membro para administrar as contas da organização e gerenciar as permissões desse serviço. Essa conta é chamada de administrador delegado para esse serviço Para obter mais informações e uma lista de serviços compatíveis, consulte Serviços que funcionam com o AWS Organizations na documentação do AWS Organizations .

implantação

O processo de criar uma aplicação, novos recursos ou correções de código disponíveis no ambiente de destino. A implantação envolve a implementação de mudanças em uma base de código e, em seguida, a criação e execução dessa base de código nos ambientes da aplicação

ambiente de desenvolvimento

Veja ambiente.

controle detectivo

Um controle de segurança projetado para detectar, registrar e alertar após a ocorrência de um evento. Esses controles são uma segunda linha de defesa, alertando você sobre eventos de segurança que contornaram os controles preventivos em vigor. Para obter mais informações, consulte Controles detectivos em Como implementar controles de segurança na AWS.

mapeamento do fluxo de valor de desenvolvimento (DVSM)

Um processo usado para identificar e priorizar restrições que afetam negativamente a velocidade e a qualidade em um ciclo de vida de desenvolvimento de software. O DVSM estende o processo de mapeamento do fluxo de valor originalmente projetado para práticas de manufatura enxuta. Ele se concentra nas etapas e equipes necessárias para criar e movimentar valor por meio do processo de desenvolvimento de software.

gêmeo digital

Uma representação virtual de um sistema real, como um prédio, fábrica, equipamento industrial ou linha de produção. Os gêmeos digitais oferecem suporte à manutenção preditiva, ao monitoramento remoto e à otimização da produção.

tabela de dimensões

Em um esquema em estrela, uma tabela menor que contém atributos de dados sobre dados quantitativos em uma tabela de fatos. Os atributos da tabela de dimensões geralmente são campos de texto ou números discretos que se comportam como texto. Esses atributos normalmente são usados para restringir consultas, filtrar e rotular conjuntos de resultados.

desastre

Um evento que impede que uma workload ou sistema cumpra seus objetivos de negócios em seu local principal de implantação. Esses eventos podem ser desastres naturais, falhas técnicas ou o resultado de ações humanas, como configuração incorreta não intencional ou ataque de malware.

Recuperação de desastres (RD)

A estratégia e o processo que você usa para minimizar o tempo de inatividade e a perda de dados causados por um disastre. Para obter mais informações, consulte Recuperação de desastres de cargas de trabalho em AWS: Recuperação na nuvem no AWS Well-Architected Framework.

DML

Veja linguagem de manipulação de banco de dados.

design orientado por domínio

Uma abordagem ao desenvolvimento de um sistema de software complexo conectando seus componentes aos domínios em evolução, ou principais metas de negócios, atendidos por cada componente. Esse conceito foi introduzido por Eric Evans em seu livro, Design orientado por domínio: lidando com a complexidade no coração do software (Boston: Addison-Wesley Professional, 2003). Para obter informações sobre como usar o design orientado por domínio com o padrão strangler fig, consulte Modernizar incrementalmente os serviços web herdados do Microsoft ASP.NET (ASMX) usando contêineres e o Amazon API Gateway.

DR

Veja recuperação de desastres.

Detecção da oscilação

Rastreamento de desvios de uma configuração de linha de base. Por exemplo, você pode usar AWS CloudFormation para detectar desvios nos recursos do sistema ou AWS Control Tower para detectar mudanças em seu landing zone que possam afetar a conformidade com os requisitos de governança.

DVSM

Veja mapeamento do fluxo de valor de desenvolvimento.

E

EDA

Veja análise exploratória de dados.

EDI

Veja intercâmbio eletrônico de dados.

computação de borda

A tecnologia que aumenta o poder computacional de dispositivos inteligentes nas bordas de uma rede de IoT. Quando comparada com a computação em nuvem, a computação de borda pode reduzir a latência da comunicação e melhorar o tempo de resposta.

intercâmbio eletrônico de dados (EDI)

A troca automatizada de documentos comerciais entre organizações. Para obter mais informações, consulte O que é EDI (Intercâmbio eletrônico de dados)?.

criptografia

Um processo de computação que transforma dados de texto simples, legíveis por humanos, em texto cifrado.

chave de criptografia

Uma sequência criptográfica de bits aleatórios que é gerada por um algoritmo de criptografia. As chaves podem variar em tamanho, e cada chave foi projetada para ser imprevisível e exclusiva.

endianismo

A ordem na qual os bytes são armazenados na memória do computador. Os sistemas big-endian armazenam o byte mais significativo antes. Os sistemas little-endian armazenam o byte menos significativo antes.

endpoint

Veja endpoint de serviço.

serviço de endpoint

Um serviço que pode ser hospedado em uma nuvem privada virtual (VPC) para ser compartilhado com outros usuários. Você pode criar um serviço de endpoint com AWS PrivateLink e conceder permissões a outros diretores Contas da AWS ou a AWS Identity and Access Management (IAM). Essas contas ou entidades principais podem se conectar ao serviço de endpoint de maneira privada criando endpoints da VPC de interface. Para obter mais informações, consulte Criar um serviço de endpoint na documentação do Amazon Virtual Private Cloud (Amazon VPC).

planejamento de recursos empresariais (ERP)

Um sistema que automatiza e gerencia os principais processos de negócios (como contabilidade, MES e gerenciamento de projetos) para uma empresa.

criptografia envelopada

O processo de criptografar uma chave de criptografia com outra chave de criptografia. Para obter mais informações, consulte Criptografia de envelope na documentação AWS Key Management Service (AWS KMS).

ambiente

Uma instância de uma aplicação em execução. Estes são tipos comuns de ambientes na computação em nuvem:

- ambiente de desenvolvimento: uma instância de uma aplicação em execução que está disponível somente para a equipe principal responsável pela manutenção da aplicação. Ambientes de desenvolvimento são usados para testar mudanças antes de promovê-las para ambientes superiores. Esse tipo de ambiente às vezes é chamado de ambiente de teste.
- ambientes inferiores: todos os ambientes de desenvolvimento para uma aplicação, como aqueles usados para compilações e testes iniciais.
- ambiente de produção: uma instância de uma aplicação em execução que os usuários finais podem acessar. Em um CI/CD pipeline, o ambiente de produção é o último ambiente de implantação.
- ambientes superiores: todos os ambientes que podem ser acessados por usuários que não sejam a equipe principal de desenvolvimento. Isso pode incluir um ambiente de produção, ambientes de pré-produção e ambientes para testes de aceitação do usuário.

epic

Em metodologias ágeis, categorias funcionais que ajudam a organizar e priorizar seu trabalho. Os epics fornecem uma descrição de alto nível dos requisitos e das tarefas de implementação. Por exemplo, os épicos de segurança AWS da CAF incluem gerenciamento de identidade e acesso, controles de detetive, segurança de infraestrutura, proteção de dados e resposta a incidentes. Para obter mais informações sobre epics na estratégia de migração da AWS , consulte o guia de implementação do programa.

ERP

Veja planejamento de recursos empresariais.

análise exploratória de dados (EDA)

O processo de analisar um conjunto de dados para entender suas principais características. Você coleta ou agrega dados e, em seguida, realiza investigações iniciais para encontrar padrões, detectar anomalias e verificar suposições. O EDA é realizado por meio do cálculo de estatísticas resumidas e da criação de visualizações de dados.

F

tabela de fatos

A tabela central em um esquema em estrela. Ela armazena dados quantitativos sobre as operações comerciais. Normalmente, uma tabela de fatos contém dois tipos de colunas: as que contêm medidas e as que contêm uma chave externa para uma tabela de dimensões.

Antecipar-se à falha

Uma filosofia que usa testes frequentes e incrementais para reduzir o ciclo de vida do desenvolvimento. É uma parte essencial de uma abordagem ágil.

delimitação de isolamento contra falhas

No Nuvem AWS, um limite, como uma zona de disponibilidade, Região da AWS um plano de controle ou um plano de dados, que limita o efeito de uma falha e ajuda a melhorar a resiliência das cargas de trabalho. Para obter mais informações, consulte AWS Fault Isolation Boundaries.

ramificação de recursos

Veja ramificação.

recursos

Os dados de entrada usados para fazer uma previsão. Por exemplo, em um contexto de manufatura, os recursos podem ser imagens capturadas periodicamente na linha de fabricação.

importância do recurso

O quanto um recurso é importante para as previsões de um modelo. Isso geralmente é expresso como uma pontuação numérica que pode ser calculada por meio de várias técnicas, como Shapley Additive Explanations (SHAP) e gradientes integrados. Para obter mais informações, consulte Interpretabilidade do modelo de aprendizado de máquina com AWS.

transformação de recursos

O processo de otimizar dados para o processo de ML, incluindo enriquecer dados com fontes adicionais, escalar valores ou extrair vários conjuntos de informações de um único campo de dados. Isso permite que o modelo de ML se beneficie dos dados. Por exemplo, se a data “2021-05-27 00:15:37” for dividida em “2021”, “maio”, “quinta” e “15”, isso poderá ajudar o algoritmo de aprendizado a aprender padrões diferenciados associados a diferentes componentes de dados.

prompt few shot

Fornecer a um LLM um pequeno número de exemplos que demonstram a tarefa e o resultado desejado antes de solicitar que ele execute uma tarefa semelhante. Essa técnica é uma aplicação do aprendizado em contexto, em que os modelos aprendem com exemplos (shots) incorporados aos prompts. Prompts few-shot podem ser eficazes para tarefas que exigem formatação, raciocínio ou conhecimento de domínio específicos. Veja também prompts zero-shot.

FGAC

Veja controle de acesso refinado.

Controle de acesso refinado (FGAC)

O uso de várias condições para permitir ou negar uma solicitação de acesso.

migração flash-cut

Um método de migração de banco de dados que usa replicação contínua de dados via captura de dados de alteração para migrar os dados no menor tempo possível, em vez de usar uma abordagem em fases. O objetivo é reduzir ao mínimo o tempo de inatividade.

FM

Veja modelo de base.

modelo de base (FM)

Uma grande rede neural de aprendizado profundo que vem treinando em grandes conjuntos de dados generalizados e não rotulados. FMs são capazes de realizar uma ampla variedade de tarefas gerais, como entender a linguagem, gerar texto e imagens e conversar em linguagem natural. Para obter mais informações, consulte O que são modelos de base?.

G

IA generativa

Um subconjunto de modelos de IA que foram treinados em grandes quantidades de dados e que podem usar um simples prompt de texto para criar novos artefatos e conteúdo, como imagens, vídeos, texto e áudio. Para obter mais informações, consulte O que é IA generativa?.

bloqueio geográfico

Veja restrições geográficas.

restrições geográficas (bloqueio geográfico)

Na Amazon CloudFront, uma opção para impedir que usuários em países específicos acessem distribuições de conteúdo. É possível usar uma lista de permissões ou uma lista de bloqueios para especificar países aprovados e banidos. Para obter mais informações, consulte Restringir a distribuição geográfica do seu conteúdo na CloudFront documentação.

Fluxo de trabalho do GitFlow

Uma abordagem na qual ambientes inferiores e superiores usam ramificações diferentes em um repositório de código-fonte. O fluxo de trabalho do Gitflow é considerado legado, e o fluxo de trabalho trunk-based é a abordagem moderna e preferencial.

golden image

Um snapshot de um sistema ou software usado como modelo para implantar novas instâncias desse sistema ou software. Por exemplo, na manufatura, uma golden image pode ser usada para provisionar software em vários dispositivos e ajudar a melhorar a velocidade, a escalabilidade e a produtividade nas operações de fabricação de dispositivos.

estratégia greenfield

A ausência de infraestrutura existente em um novo ambiente. Ao adotar uma estratégia greenfield para uma arquitetura de sistema, é possível selecionar todas as novas tecnologias sem a restrição da compatibilidade com a infraestrutura existente, também conhecida como brownfield. Se estiver expandindo a infraestrutura existente, poderá combinar as estratégias brownfield e greenfield.

barreira de proteção

Uma regra de alto nível que ajuda a governar recursos, políticas e conformidade em todas as unidades organizacionais ()OUs. Barreiras de proteção preventivas impõem políticas para garantir o alinhamento a padrões de conformidade. Elas são implementadas usando políticas de controle de serviço e limites de permissões do IAM. Barreiras de proteção detectivas detectam violações de políticas e problemas de conformidade e geram alertas para remediação. Eles são implementados usando AWS Config, AWS Security Hub CSPM, Amazon GuardDuty AWS Trusted Advisor, Amazon Inspector e verificações personalizadas AWS Lambda .

H

HA

Veja alta disponibilidade.

migração heterogênea de bancos de dados

Migrar seu banco de dados de origem para um banco de dados de destino que usa um mecanismo de banco de dados diferente (por exemplo, Oracle para Amazon Aurora). A migração heterogênea geralmente faz parte de um esforço de redefinição da arquitetura, e converter

o esquema pode ser uma tarefa complexa. O AWS fornece o AWS SCT para ajudar nas conversões de esquemas.

alta disponibilidade (HA)

A capacidade de uma workload operar continuamente, sem intervenção, em caso de desafios ou desastres. Os sistemas AH são projetados para realizar o failover automático, oferecer consistentemente desempenho de alta qualidade e lidar com diferentes cargas e falhas com impacto mínimo no desempenho.

modernização de historiador

Uma abordagem usada para modernizar e atualizar os sistemas de tecnologia operacional (OT) para melhor atender às necessidades do setor de manufatura. Um historiador é um tipo de banco de dados usado para coletar e armazenar dados de várias fontes em uma fábrica.

dados de hold-out

Uma parte dos dados históricos rotulados que são retidos de um conjunto de dados usado para treinar um modelo de machine learning. Você pode usar dados de hold-out para avaliar a performance do modelo comparando as predições do modelo com os dados de retenção.

migração homogênea de bancos de dados

Migrar seu banco de dados de origem para um banco de dados de destino que compartilha o mesmo mecanismo de banco de dados (por exemplo, Microsoft SQL Server para Amazon RDS para SQL Server). A migração homogênea geralmente faz parte de um esforço de redefinição da hospedagem ou da plataforma. É possível usar utilitários de banco de dados nativos para migrar o esquema.

dados quentes

Dados acessados com frequência, como dados em tempo real ou dados translacionais recentes. Esses dados normalmente exigem uma camada ou classe de armazenamento de alto desempenho para fornecer respostas rápidas às consultas.

hotfix

Uma correção urgente para um problema crítico em um ambiente de produção. Devido à sua urgência, um hotfix geralmente é feito fora do fluxo de trabalho normal de DevOps lançamento.

período de hipercuidados

Imediatamente após a substituição, o período em que uma equipe de migração gerencia e monitora as aplicações migradas na nuvem para resolver quaisquer problemas. Normalmente,

a duração desse período é de 1 a 4 dias. No final do período de hipercuidados, a equipe de migração normalmente transfere a responsabilidade pelas aplicações para a equipe de operações de nuvem.

eu

IaC

Veja infraestrutura como código.

Política baseada em identidade

Uma política anexada a um ou mais diretores do IAM que define suas permissões no Nuvem AWS ambiente.

aplicação ociosa

Uma aplicação que tem um uso médio de CPU e memória entre 5 e 20% em um período de 90 dias. Em um projeto de migração, é comum retirar essas aplicações ou retê-las on-premises.

IIoT

Veja Internet das Coisas Industrial.

infraestrutura imutável

Um modelo que implanta uma nova infraestrutura para workloads de produção em vez de atualizar, aplicar patches ou modificar a infraestrutura existente. Infraestruturas imutáveis são inerentemente mais consistentes, confiáveis e preditivas do que infraestruturas mutáveis. Para obter mais informações, consulte a prática recomendada Implantar usando infraestrutura imutável no AWS Well-Architected Framework.

VPC de entrada (admissão)

Em uma arquitetura de AWS várias contas, uma VPC que aceita, inspeciona e roteia conexões de rede de fora de um aplicativo. A Arquitetura de Referência de AWS Segurança recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

migração incremental

Uma estratégia de substituição na qual você migra a aplicação em pequenas partes, em vez de realizar uma única substituição completa. Por exemplo, é possível mover inicialmente

apenas alguns microsserviços ou usuários para o novo sistema. Depois de verificar se tudo está funcionando corretamente, mova os microsserviços ou usuários adicionais de forma incremental até poder descomissionar seu sistema herdado. Essa estratégia reduz os riscos associados a migrações de grande porte.

Indústria 4.0

Um termo que foi introduzido por Klaus Schwab em 2016 para se referir à modernização dos processos de manufatura por meio de avanços em conectividade, dados em tempo real, automação, analytics e IA/ML.

infraestrutura

Todos os recursos e ativos contidos no ambiente de uma aplicação.

Infraestrutura como código (IaC)

O processo de provisionamento e gerenciamento da infraestrutura de uma aplicação por meio de um conjunto de arquivos de configuração. A IaC foi projetada para ajudar você a centralizar o gerenciamento da infraestrutura, padronizar recursos e escalar rapidamente para que novos ambientes sejam reproduzíveis, confiáveis e consistentes.

Internet industrial das coisas (IIoT)

O uso de sensores e dispositivos conectados à Internet nos setores industriais, como manufatura, energia, automotivo, saúde, ciências biológicas e agricultura. Para obter mais informações, consulte Criando uma estratégia de transformação digital industrial da Internet das Coisas (IIoT).

VPC de inspeção

Em uma arquitetura de AWS várias contas, uma VPC centralizada que gerencia as inspeções do tráfego de rede entre VPCs (na mesma ou em diferentes Regiões da AWS) a Internet e as redes locais. A Arquitetura de Referência de AWS Segurança recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

Internet das coisas (IoT)

A rede de objetos físicos conectados com sensores ou processadores incorporados que se comunicam com outros dispositivos e sistemas pela Internet ou por uma rede de comunicação local. Para obter mais informações, consulte O que é IoT?

interpretabilidade

Uma característica de um modelo de machine learning que descreve o grau em que um ser humano pode entender como as previsões do modelo dependem de suas entradas. Para obter mais informações, consulte Interpretabilidade do modelo de aprendizado de máquina com AWS.

IoT

Veja Internet das Coisas.

Biblioteca de informações de TI (ITIL)

Um conjunto de práticas recomendadas para fornecer serviços de TI e alinhar esses serviços a requisitos de negócios. A ITIL fornece a base para o ITSM.

Gerenciamento de serviços de TI (ITSM)

Atividades associadas a design, implementação, gerenciamento e suporte de serviços de TI para uma organização. Para obter informações sobre a integração de operações em nuvem com ferramentas de ITSM, consulte o guia de integração de operações.

ITIL

Veja biblioteca de informações de TI.

ITSM

Veja gerenciamento de serviços de TI.

L

controle de acesso baseado em etiqueta (LBAC)

Uma implementação do controle de acesso obrigatório (MAC) em que os usuários e os dados em si recebem explicitamente um valor de etiqueta de segurança. A interseção entre a etiqueta de segurança do usuário e a etiqueta de segurança dos dados determina quais linhas e colunas podem ser vistas pelo usuário.

zona de pouso

Uma landing zone é um AWS ambiente bem arquitetado, com várias contas, escalável e seguro. Um ponto a partir do qual suas organizações podem iniciar e implantar rapidamente workloads e aplicações com confiança em seu ambiente de segurança e infraestrutura. Para obter mais

informações sobre zonas de pouso, consulte Configurar um ambiente da AWS com várias contas seguro e escalável.

grande modelo de linguagem (LLM)

Um modelo de IA de aprendizado profundo pré-treinado em uma grande quantidade de dados. Um LLM pode realizar várias tarefas, como responder a perguntas, resumir documentos, traduzir texto para outros idiomas e completar frases. Para obter mais informações, consulte O que são LLMs.

migração de grande porte

Uma migração de 300 servidores ou mais.

LBAC

Veja controle de acesso baseado em rótulo.

privilégio mínimo

A prática recomendada de segurança de conceder as permissões mínimas necessárias para executar uma tarefa. Para obter mais informações, consulte Aplicar permissões de privilégios mínimos na documentação do IAM.

mover sem alterações (lift-and-shift)

Veja 7 Rs.

sistema little-endian

Um sistema que armazena o byte menos significativo antes. Veja também endianness.

LLM

Veja grande modelo de linguagem.

ambientes inferiores

Veja ambiente.

M

machine learning (ML)

Um tipo de inteligência artificial que usa algoritmos e técnicas para reconhecimento e aprendizado de padrões. O ML analisa e aprende com dados gravados, por exemplo, dados da

Internet das Coisas (IoT), para gerar um modelo estatístico baseado em padrões. Para obter mais informações, consulte Machine learning.

ramificação principal

Veja ramificação.

Malware

Software projetado para comprometer a segurança ou a privacidade do computador. O malware pode interromper os sistemas do computador, vazar informações sensíveis ou obter acesso não autorizado. Exemplos de malware incluem vírus, worms, ransomware, cavalos de Troia, spyware e keyloggers.

Serviços gerenciados

Serviços da AWS para o qual AWS opera a camada de infraestrutura, o sistema operacional e as plataformas, e você acessa os endpoints para armazenar e recuperar dados. O Amazon Simple Storage Service (Amazon S3) e o Amazon DynamoDB são exemplos de serviços gerenciados. Eles também são conhecidos como serviços abstraídos.

sistema de execução de manufatura (MES)

Um sistema de software para rastrear, monitorar, documentar e controlar processos de produção que convertem matérias-primas em produtos acabados no chão de fábrica.

MAP

Veja Programa de Aceleração da Migração.

mecanismo

Um processo completo em que você cria uma ferramenta, impulsiona a adoção da ferramenta e, em seguida, inspeciona os resultados para fazer ajustes. Um mecanismo é um ciclo que se reforça e se aprimora à medida que opera. Para obter mais informações, consulte Construindo mecanismos no AWS Well-Architected Framework.

conta de membro

Todos, Contas da AWS exceto a conta de gerenciamento, que fazem parte de uma organização em AWS Organizations. Uma conta só pode ser membro de uma organização de cada vez.

MES

Veja sistema de execução de manufatura.

Transporte de Telemetria de Enfileiramento de Mensagens (MQTT)

Um protocolo de comunicação leve machine-to-machine (M2M), baseado no padrão de publicação/assinatura, para dispositivos de IoT com recursos limitados.

microsserviço

Um serviço pequeno e independente que se comunica de forma bem definida APIs e normalmente é de propriedade de equipes pequenas e independentes. Por exemplo, um sistema de seguradora pode incluir microsserviços que mapeiam as capacidades comerciais, como vendas ou marketing, ou subdomínios, como compras, reclamações ou análises. Os benefícios dos microsserviços incluem agilidade, escalabilidade flexível, fácil implantação, código reutilizável e resiliência. Para obter mais informações, consulte Integração de microsserviços usando serviços sem AWS servidor.

arquitetura de microsserviços

Uma abordagem à criação de aplicações com componentes independentes que executam cada processo de aplicação como um microsserviço. Esses microsserviços se comunicam por meio de uma interface bem definida usando leveza. APIs Cada microsserviço nessa arquitetura pode ser atualizado, implantado e escalado para atender à demanda por funções específicas de uma aplicação. Para obter mais informações, consulte Implementação de microsserviços em. AWS

Programa de Aceleração da Migração (MAP)

Um AWS programa que fornece suporte de consultoria, treinamento e serviços para ajudar as organizações a criar uma base operacional sólida para migrar para a nuvem e ajudar a compensar o custo inicial das migrações. O MAP inclui uma metodologia de migração para executar migrações legadas de forma metódica e um conjunto de ferramentas para automatizar e acelerar cenários comuns de migração.

migração em escala

O processo de mover a maior parte do portfólio de aplicações para a nuvem em ondas, com mais aplicações sendo movidas em um ritmo mais rápido a cada onda. Essa fase usa as práticas recomendadas e lições aprendidas nas fases anteriores para implementar uma fábrica de migração de equipes, ferramentas e processos para agilizar a migração de workloads por meio de automação e entrega ágeis. Esta é a terceira fase da estratégia de migração para a AWS.

fábrica de migração

Equipes multifuncionais que simplificam a migração de workloads por meio de abordagens automatizadas e ágeis. As equipes da fábrica de migração geralmente incluem operações,

analistas e proprietários de negócios, engenheiros de migração, desenvolvedores e DevOps profissionais que trabalham em sprints. Entre 20 e 50% de um portfólio de aplicações corporativas consiste em padrões repetidos que podem ser otimizados por meio de uma abordagem de fábrica. Para obter mais informações, consulte discussão sobre fábricas de migração e o guia do Cloud Migration Factory neste conjunto de conteúdo.

metadados de migração

As informações sobre a aplicação e o servidor necessárias para concluir a migração. Cada padrão de migração exige um conjunto de metadados de migração diferente. Exemplos de metadados de migração incluem a sub-rede, o grupo de segurança e AWS a conta de destino.

padrão de migração

Uma tarefa de migração repetível que detalha a estratégia de migração, o destino da migração e a aplicação ou o serviço de migração usado. Exemplo: rehospede a migração para o Amazon EC2 AWS com o Application Migration Service.

Avaliação de Portfólio para Migração (MPA)

Uma ferramenta on-line que fornece informações para validar o caso de negócios para migrar para a Nuvem AWS. O MPA fornece avaliação detalhada do portfólio (dimensionamento correto do servidor, preços, comparações de TCO, análise de custos de migração), bem como planejamento de migração (análise e coleta de dados de aplicações, agrupamento de aplicações, priorização de migração e planejamento de ondas). A ferramenta MPA (requer login) está disponível gratuitamente para todos os AWS consultores e consultores parceiros da APN.

Avaliação de Preparação para Migração (MRA)

O processo de obter insights sobre o status de prontidão de uma organização para a nuvem, identificar pontos fortes e fracos e criar um plano de ação para fechar as lacunas identificadas, usando o CAF. AWS Para mais informações, consulte o guia de preparação para migração. A MRA é a primeira fase da estratégia de migração para a AWS.

estratégia de migração

A abordagem usada para migrar uma workload para a Nuvem AWS. Para obter mais informações, veja a entrada 7 Rs neste glossário e consulte Mobilize sua organização para acelerar migrações em grande escala.

ML

Veja machine learning.

modernização

Transformar uma aplicação desatualizada (herdada ou monolítica) e sua infraestrutura em um sistema ágil, elástico e altamente disponível na nuvem para reduzir custos, ganhar eficiência e aproveitar as inovações. Para obter mais informações, consulte Strategy for modernizing applications in the Nuvem AWS.

avaliação de preparação para modernização

Uma avaliação que ajuda a determinar a preparação para modernização das aplicações de uma organização. Ela identifica benefícios, riscos e dependências e determina o quão bem a organização pode acomodar o estado futuro dessas aplicações. O resultado da avaliação é um esquema da arquitetura de destino, um roteiro que detalha as fases de desenvolvimento e os marcos do processo de modernização e um plano de ação para abordar as lacunas identificadas. Para obter mais informações, consulte Evaluating modernization readiness for applications in the Nuvem AWS.

aplicações monolíticas (monólitos)

Aplicações que são executadas como um único serviço com processos fortemente acoplados. As aplicações monolíticas apresentam várias desvantagens. Se um recurso da aplicação apresentar um aumento na demanda, toda a arquitetura deverá ser escalada. Adicionar ou melhorar os recursos de uma aplicação monolítica também se torna mais complexo quando a base de código cresce. Para resolver esses problemas, é possível criar uma arquitetura de microsserviços. Para obter mais informações, consulte Decompor monólitos em microsserviços.

MPA

Veja Avaliação do Portfólio para Migração.

MQTT

Veja Transporte de Telemetria de Enfileiramento de Mensagens.

classificação multiclasse

Um processo que ajuda a gerar previsões para várias classes (prevendo um ou mais de dois resultados). Por exemplo, um modelo de ML pode perguntar “Este produto é um livro, um carro ou um telefone?” ou “Qual categoria de produtos é mais interessante para este cliente?”

infraestrutura mutável

Um modelo que atualiza e modifica a infraestrutura existente para workloads de produção. Para melhorar a consistência, confiabilidade e previsibilidade, o AWS Well-Architected Framework recomenda o uso de infraestrutura imutável como uma prática recomendada.

O

OAC

Veja controle de acesso de origem.

OAI

Veja identidade de acesso de origem.

OCM

Veja gerenciamento de alterações organizacionais.

migração offline

Um método de migração no qual a workload de origem é desativada durante o processo de migração. Esse método envolve tempo de inatividade prolongado e geralmente é usado para workloads pequenas e não críticas.

OI

Veja integração de operações.

Ola

Veja acordo de nível operacional.

migração online

Um método de migração no qual a workload de origem é copiada para o sistema de destino sem ser colocada offline. As aplicações conectadas à workload podem continuar funcionando durante a migração. Esse método envolve um tempo de inatividade nulo ou mínimo e normalmente é usado para workloads essenciais para a produção.

OPC-UA

Veja Open Process Communications - Unified Architecture.

Open Process Communications - Unified Architecture (OPC-UA)

Um protocolo de comunicação machine-to-machine (M2M) para automação industrial. O OPC-UA fornece um padrão de interoperabilidade com esquemas de criptografia, autenticação e autorização de dados.

acordo de nível operacional (OLA)

Um acordo que esclarece o que os grupos funcionais de TI prometem oferecer uns aos outros para apoiar um acordo de serviço (SLA).

análise de prontidão operacional (ORR)

Uma lista de verificação de perguntas e práticas recomendadas associadas que ajudam você a entender, avaliar, prevenir ou reduzir o escopo de incidentes e possíveis falhas. Para obter mais informações, consulte Operational Readiness Reviews (ORR) no AWS Well-Architected Framework.

tecnologia operacional (TO)

Sistemas de hardware e software que trabalham com o ambiente físico para controlar operações, equipamentos e infraestrutura industriais. Na manufatura, a integração dos sistemas de tecnologia da informação (TI) e tecnologia operacional (TO) é o foco principal das transformações da Indústria 4.0.

integração de operações (OI)

O processo de modernização das operações na nuvem, que envolve planejamento de preparação, automação e integração. Para obter mais informações, consulte o guia de integração de operações.

trilha organizacional

Uma trilha criada por ela AWS CloudTrail registra todos os eventos de todos Contas da AWS em uma organização em AWS Organizations. Essa trilha é criada em cada Conta da AWS que faz parte da organização e monitora a atividade em cada conta. Para obter mais informações, consulte Criação de uma trilha para uma organização na CloudTrail documentação.

gerenciamento de alterações organizacionais (OCM)

Uma estrutura para gerenciar grandes transformações de negócios disruptivas de uma perspectiva de pessoas, cultura e liderança. O OCM ajuda as organizações a se prepararem e fazerem a transição para novos sistemas e estratégias, acelerando a adoção de alterações, abordando questões de transição e promovendo mudanças culturais e organizacionais. Na estratégia de AWS migração, essa estrutura é chamada de aceleração de pessoas, devido à velocidade de mudança exigida nos projetos de adoção da nuvem. Para obter mais informações, consulte o guia do OCM.

controle de acesso de origem (OAC)

Em CloudFront, uma opção aprimorada para restringir o acesso para proteger seu conteúdo do Amazon Simple Storage Service (Amazon S3). O OAC oferece suporte a todos os buckets S3 Regiões da AWS, criptografia do lado do servidor com AWS KMS (SSE-KMS) e solicitações dinâmicas ao bucket S3. `PUT` `DELETE`

Identidade do acesso de origem (OAI)

Em CloudFront, uma opção para restringir o acesso para proteger seu conteúdo do Amazon S3. Quando você usa o OAI, CloudFront cria um principal com o qual o Amazon S3 pode se autenticar. Os diretores autenticados podem acessar o conteúdo em um bucket do S3 somente por meio de uma distribuição específica. CloudFront Veja também OAC, que fornece um controle de acesso mais granular e aprimorado.

ORR

Veja análise de prontidão operacional.

OT

Veja tecnologia operacional.

VPC de saída (egresso)

Em uma arquitetura de AWS várias contas, uma VPC que gerencia conexões de rede que são iniciadas de dentro de um aplicativo. A Arquitetura de Referência de AWS Segurança recomenda configurar sua conta de rede com entrada, saída e inspeção VPCs para proteger a interface bidirecional entre seu aplicativo e a Internet em geral.

P

limite de permissões

Uma política de gerenciamento do IAM anexada a entidades principais do IAM para definir as permissões máximas que o usuário ou perfil podem ter. Para obter mais informações, consulte Limites de permissões na documentação do IAM.

Informações de identificação pessoal (PII)

Informações que, quando visualizadas diretamente ou combinadas com outros dados relacionados, podem ser usadas para inferir razoavelmente a identidade de um indivíduo. Exemplos de PII incluem nomes, endereços e informações de contato.

PII

Veja informações de identificação pessoal.

manual

Um conjunto de etapas predefinidas que capturam o trabalho associado às migrações, como a entrega das principais funções operacionais na nuvem. Um manual pode assumir a forma de scripts, runbooks automatizados ou um resumo dos processos ou etapas necessários para operar seu ambiente modernizado.

PLC

Veja controlador lógico programável.

PLM

Veja gerenciamento do ciclo de vida do produto.

política

Um objeto que pode definir permissões (veja política baseada em identidade), especificar condições de acesso (veja políta baseada em recurso) ou definir as permissões máximas para todas as contas em uma organização no AWS Organizations (veja política de controle de serviços).

persistência poliglota

Escolher de forma independente a tecnologia de armazenamento de dados de um microsserviço com base em padrões de acesso a dados e outros requisitos. Se seus microsserviços tiverem a mesma tecnologia de armazenamento de dados, eles poderão enfrentar desafios de implementação ou apresentar baixa performance. Os microsserviços serão implementados com mais facilidade e alcançarão performance e escalabilidade melhores se usarem o armazenamento de dados mais bem adaptado às suas necessidades.

avaliação do portfólio

Um processo de descobrir, analisar e priorizar o portfólio de aplicações para planejar a migração. Para obter mais informações, consulte Avaliar a preparação para a migração.

predicado

Uma condição de consulta que retorna `true` ou `false`, normalmente localizada em uma cláusula `WHERE`.

pushdown de predicados

Uma técnica de otimização de consultas de banco de dados que filtra os dados na consulta antes da transferência. Isso reduz a quantidade de dados que devem ser recuperados e processados do banco de dados relacional e melhora a performance das consultas.

controle preventivo

Um controle de segurança projetado para evitar que um evento ocorra. Esses controles são a primeira linha de defesa para ajudar a evitar acesso não autorizado ou alterações indesejadas em sua rede. Para obter mais informações, consulte Controles preventivos em Como implementar controles de segurança na AWS.

principal (entidade principal)

Uma entidade AWS que pode realizar ações e acessar recursos. Essa entidade geralmente é um usuário raiz para um Conta da AWS, uma função do IAM ou um usuário. Para obter mais informações, consulte Entidade principal em Termos e conceitos de perfis na documentação do IAM.

Privacidade por design

Uma abordagem em engenharia de sistemas que leva em consideração a privacidade em todo o processo de desenvolvimento.

zonas hospedadas privadas

Um contêiner que contém informações sobre como você deseja que o Amazon Route 53 responda às consultas de DNS para um domínio e seus subdomínios em um ou mais. VPCs Para obter mais informações, consulte Como trabalhar com zonas hospedadas privadas na documentação do Route 53.

controle proativo

Um controle de segurança desenvolvido para evitar a implantação de recursos não conformes. Esses controles verificam os recursos antes de serem provisionados. Se o recurso não estiver em conformidade com o controle, ele não será provisionado. Para obter mais informações, consulte o guia de referência de controles na AWS Control Tower documentação e consulte Controles proativos em Implementação de controles de segurança em AWS.

gerenciamento do ciclo de vida do produto (PLM)

O gerenciamento de dados e processos de um produto em todo o seu ciclo de vida, desde a concepção, o desenvolvimento e o lançamento, passando pelo crescimento e maturidade, até o declínio e a remoção.

ambiente de produção

Veja ambiente.

controlador lógico programável (PLC)

Na manufatura, um computador altamente confiável e adaptável que monitora as máquinas e automatiza os processos de fabricação.

encadeamento de prompts

Uso da saída de um prompt do LLM como entrada para o próximo prompt para gerar respostas melhores. Essa técnica é usada para dividir uma tarefa complexa em subtarefas, ou para refinar ou expandir iterativamente uma resposta preliminar. Isso ajuda a melhorar a precisão e a relevância das respostas de um modelo e permite resultados mais granulares e personalizados.

pseudonimização

O processo de substituir identificadores pessoais em um conjunto de dados por valores de espaço reservado. A pseudonimização pode ajudar a proteger a privacidade pessoal. Os dados pseudonimizados ainda são considerados dados pessoais.

publish/subscribe (pub/sub)

Um padrão que permite comunicações assíncronas entre microsserviços para melhorar a escalabilidade e a capacidade de resposta. Por exemplo, em um MES baseado em microsserviços, um microsserviço pode publicar mensagens de eventos em um canal em que outros microsserviços possam assinar. O sistema pode adicionar novos microsserviços sem alterar o serviço de publicação.

Q

plano de consulta

Uma série de etapas, como instruções, usadas para acessar os dados em um sistema de banco de dados relacional SQL.

regressão de planos de consultas

Quando um otimizador de serviço de banco de dados escolhe um plano menos adequado do que escolhia antes de uma determinada alteração no ambiente de banco de dados ocorrer. Isso pode ser causado por alterações em estatísticas, restrições, configurações do ambiente, associações de parâmetros de consulta e atualizações do mecanismo de banco de dados.

R

Matriz RACI

Veja responsável, aprovador, consultado, informado (RACI).

RAG

Veja geração aumentada via recuperação.

ransomware

Um software mal-intencionado desenvolvido para bloquear o acesso a um sistema ou dados de computador até que um pagamento seja feito.

Matriz RASCI

Veja responsável, aprovador, consultado, informado (RACI).

RCAC

Veja controle de acesso por linha e coluna.

réplica de leitura

Uma cópia de um banco de dados usada somente para leitura. É possível encaminhar consultas para a réplica de leitura e reduzir a carga no banco de dados principal.

Redefinir arquitetura

Veja 7 Rs.

objetivo de ponto de recuperação (RPO).

O máximo período de tempo aceitável desde o último ponto de recuperação de dados. Isso determina o que é considerado uma perda aceitável de dados entre o último ponto de recuperação e a interrupção do serviço.

objetivo de tempo de recuperação (RTO)

O máximo atraso aceitável entre a interrupção e a restauração do serviço.

refatorar

Veja 7 Rs.

Região

Uma coleção de AWS recursos em uma área geográfica. Cada um Região da AWS é isolado e independente dos outros para fornecer tolerância a falhas, estabilidade e resiliência. Para obter informações, consulte Specify which Regiões da AWS your account can use.

regressão

Uma técnica de ML que prevê um valor numérico. Por exemplo, para resolver o problema de "Por qual preço esta casa será vendida?" um modelo de ML pode usar um modelo de regressão linear para prever o preço de venda de uma casa com base em fatos conhecidos sobre a casa (por exemplo, a metragem quadrada).

redefinir a hospedagem

Veja 7 Rs.

versão

Em um processo de implantação, o ato de promover mudanças em um ambiente de produção.

realocar

Veja 7 Rs.

redefinir a plataforma

Veja 7 Rs.

recomprar

Veja 7 Rs.

resiliência

A capacidade de uma aplicação de resistir ou se recuperar de interrupções. Alta disponibilidade e recuperação de desastres são considerações comuns ao planejar a resiliência na Nuvem AWS. Para obter mais informações, consulte Nuvem AWS Resilience.

política baseada em recurso

Uma política associada a um recurso, como um bucket do Amazon S3, um endpoint ou uma chave de criptografia. Esse tipo de política especifica quais entidades principais têm acesso permitido, ações válidas e quaisquer outras condições que devem ser atendidas.

matriz responsável, accountable, consultada, informada (RACI)

Uma matriz que define as funções e responsabilidades de todas as partes envolvidas nas atividades de migração e nas operações de nuvem. O nome da matriz é derivado dos tipos de responsabilidade definidos na matriz: responsável (R), responsabilizável (A), consultado (C) e informado (I). O tipo de suporte (S) é opcional. Se você incluir suporte, a matriz será chamada de matriz RASCI e, se excluir, será chamada de matriz RACI.

controle responsivo

Um controle de segurança desenvolvido para conduzir a remediação de eventos adversos ou desvios em relação à linha de base de segurança. Para obter mais informações, consulte Controles responsivos em Como implementar controles de segurança na AWS.

reter

Veja 7 Rs.

Retirada

Veja 7 Rs.

Geração Aumentada de Recuperação (RAG)

Uma tecnologia de IA generativa em que um LLM faz referência a uma fonte de dados autorizada que está fora de suas fontes de dados de treinamento antes de gerar uma resposta. Por exemplo, um modelo RAG pode realizar uma pesquisa semântica na base de conhecimento ou nos dados personalizados de uma organização. Para obter mais informações, consulte O que é RAG (geração aumentada via recuperação)?.

alternância

O processo de atualizar periodicamente um segredo para dificultar o acesso de um invasor às credenciais.

controle de acesso por linha e coluna (RCAC)

O uso de expressões SQL básicas e flexíveis que tenham regras de acesso definidas. O RCAC consiste em permissões de linha e máscaras de coluna.

RPO

Veja objetivo de ponto de recuperação.

RTO

Veja objetivo de tempo de recuperação.

runbook

Um conjunto de procedimentos manuais ou automatizados necessários para realizar uma tarefa específica. Eles são normalmente criados para agilizar operações ou procedimentos repetitivos com altas taxas de erro.

S

SAML 2.0

Um padrão aberto que muitos provedores de identidade (IdPs) usam. Esse recurso permite o login único federado (SSO), para que os usuários possam fazer login Console de gerenciamento da AWS ou chamar as operações da AWS API sem que você precise criar um usuário no IAM para todos em sua organização. Para obter mais informações sobre a federação baseada em SAML 2.0, consulte Sobre a federação baseada em SAML 2.0 na documentação do IAM.

SCADA

Veja controle de supervisão e aquisição de dados.

SCP

Veja política de controle de serviço.

secret

Em AWS Secrets Manager, informações confidenciais ou restritas, como uma senha ou credenciais de usuário, que você armazena de forma criptografada. Consiste no valor secreto e em seus metadados. O valor secreto pode ser binário, uma única string ou várias strings. Para obter mais informações, consulte What's in a Secrets Manager secret? na documentação do Secrets Manager.

segurança desde a concepção

Uma abordagem em engenharia de sistemas que leva em consideração a segurança em todo o processo de desenvolvimento.

controle de segurança

Uma barreira de proteção técnica ou administrativa que impede, detecta ou reduz a capacidade de uma ameaça explorar uma vulnerabilidade de segurança. Existem quatro tipos primários de controles de segurança: preventivos, detectivos, responsivos e proativos.

hardening da segurança

O processo de reduzir a superfície de ataque para torná-la mais resistente a ataques. Isso pode incluir ações como remover recursos que não são mais necessários, implementar a prática recomendada de segurança de conceder privilégios mínimos ou desativar recursos desnecessários em arquivos de configuração.

sistema de gerenciamento de eventos e informações de segurança (SIEM)

Ferramentas e serviços que combinam sistemas de gerenciamento de informações de segurança (SIM) e gerenciamento de eventos de segurança (SEM). Um sistema SIEM coleta, monitora e analisa dados de servidores, redes, dispositivos e outras fontes para detectar ameaças e violações de segurança e gerar alertas.

automação de resposta de segurança

Uma ação predefinida e programada projetada para responder ou remediar automaticamente um evento de segurança. Essas automações servem como controles de segurança responsivos ou detectivos que ajudam você a implementar as melhores práticas AWS de segurança. Exemplos de ações de resposta automatizada incluem a modificação de um grupo de segurança da VPC, a aplicação de patches em uma instância do Amazon EC2 ou a alternância de credenciais.

Criptografia do lado do servidor

Criptografia dos dados em seu destino, por AWS service (Serviço da AWS) quem os recebe.

política de controle de serviços (SCP)

Uma política que fornece controle centralizado sobre as permissões de todas as contas em uma organização em AWS Organizations. SCPs defina barreiras ou estabeleça limites nas ações que um administrador pode delegar a usuários ou funções. Você pode usar SCPs como listas de permissão ou listas de negação para especificar quais serviços ou ações são permitidos ou proibidos. Para obter mais informações, consulte Políticas de controle de serviço na AWS Organizations documentação.

service endpoint (endpoint de serviço)

O URL do ponto de entrada para um AWS service (Serviço da AWS). Você pode usar o endpoint para se conectar programaticamente ao serviço de destino. Para obter mais informações, consulte Endpoints do AWS service (Serviço da AWS) na Referência geral da AWS.

acordo de serviço (SLA)

Um acordo que esclarece o que uma equipe de TI promete fornecer aos clientes, como tempo de atividade e performance do serviço.

indicador de nível de serviço (SLI)

Uma avaliação de um aspecto de performance de um serviço, como taxa de erro, disponibilidade ou throughput.

objetivo de nível de serviço (SLO)

Uma métrica alvo que representa a integridade de um serviço, conforme avaliado por um indicador de nível de serviço.

modelo de responsabilidade compartilhada

Um modelo que descreve a responsabilidade com a qual você compartilha AWS pela segurança e conformidade na nuvem. AWS é responsável pela segurança da nuvem, enquanto você é responsável pela segurança na nuvem. Para obter mais informações, consulte o Modelo de responsabilidade compartilhada.

SIEM

Veja sistema de gerenciamento de eventos e informações de segurança.

ponto único de falha (SPOF)

Uma falha em um único componente crítico de uma aplicação que pode interromper o sistema.

SLA

Veja cacordo de serviço.

SLI

Veja indicador de nível de serviço.

SLO

Veja objetivo de nível de serviço.

split-and-seed modelo

Um padrão para escalar e acelerar projetos de modernização. À medida que novos recursos e lançamentos de produtos são definidos, a equipe principal se divide para criar novas equipes de produtos. Isso ajuda a escalar os recursos e os serviços da sua organização, melhora a produtividade do desenvolvedor e possibilita inovações rápidas. Para obter mais informações, consulte Phased approach to modernizing applications in the Nuvem AWS.

SPOF

Veja ponto único de falha.

esquema em estrela

Uma estrutura organizacional de banco de dados que usa uma grande tabela de fatos para armazenar dados transacionais ou medidos e usa uma ou mais tabelas dimensionais menores para armazenar atributos de dados. Essa estrutura foi projetada para ser usada em um data warehouse ou para fins de inteligência comercial.

padrão strangler fig

Uma abordagem à modernização de sistemas monolíticos que consiste em reescrever e substituir incrementalmente a funcionalidade do sistema até que o sistema herdado possa ser desativado. Esse padrão usa a analogia de uma videira que cresce e se torna uma árvore estabelecida e, eventualmente, supera e substitui sua hospedeira. O padrão foi apresentado por Martin Fowler como forma de gerenciar riscos ao reescrever sistemas monolíticos. Para ver um exemplo de como aplicar esse padrão, consulte Modernizar incrementalmente os serviços Web herdados do Microsoft ASP.NET (ASMX) usando contêineres e o Amazon API Gateway.

sub-rede

Um intervalo de endereços IP na VPC. Cada sub-rede fica alocada em uma única zona de disponibilidade.

controle supervisório e aquisição de dados (SCADA)

Na manufatura, um sistema que usa hardware e software para monitorar ativos físicos e operações de produção.

symmetric encryption (criptografia simétrica)

Um algoritmo de criptografia que usa a mesma chave para criptografar e descriptografar dados.

testes sintéticos

Testar um sistema de forma que simule as interações do usuário para detectar possíveis problemas ou monitorar a performance. Você pode usar o Amazon CloudWatch Synthetics para criar esses testes.

prompt do sistema

Uma técnica para fornecer contexto, instruções ou orientações a um LLM a fim de direcionar seu comportamento. Os prompts do sistema ajudam a definir o contexto e a estabelecer regras para interações com os usuários.

T

tags

Pares de valores-chave que atuam como metadados para organizar seus recursos. AWS As tags podem ajudar você a gerenciar, identificar, organizar, pesquisar e filtrar recursos da . Para obter mais informações, consulte Marcar seus recursos do AWS.

variável-alvo

O valor que você está tentando prever no ML supervisionado. Ela também é conhecida como variável de resultado. Por exemplo, em uma configuração de fabricação, a variável-alvo pode ser um defeito do produto.

lista de tarefas

Uma ferramenta usada para monitorar o progresso por meio de um runbook. Uma lista de tarefas contém uma visão geral do runbook e uma lista de tarefas gerais a serem concluídas. Para cada tarefa geral, ela inclui o tempo estimado necessário, o proprietário e o progresso.

ambiente de teste

Veja ambiente.

treinamento

O processo de fornecer dados para que seu modelo de ML aprenda. Os dados de treinamento devem conter a resposta correta. O algoritmo de aprendizado descobre padrões nos dados de treinamento que mapeiam os atributos dos dados de entrada no destino (a resposta que você deseja prever). Ele gera um modelo de ML que captura esses padrões. Você pode usar o modelo de ML para obter previsões de novos dados cujo destino você não conhece.

gateway de trânsito

Um hub de trânsito de rede que você pode usar para interconectar sua rede com VPCs a rede local. Para obter mais informações, consulte O que é um gateway de trânsito na AWS Transit Gateway documentação.

fluxo de trabalho baseado em troncos

Uma abordagem na qual os desenvolvedores criam e testam recursos localmente em uma ramificação de recursos e, em seguida, mesclam essas alterações na ramificação principal. A ramificação principal é então criada para os ambientes de desenvolvimento, pré-produção e produção, sequencialmente.

Acesso confiável

Conceder permissões a um serviço que você especifica para realizar tarefas em sua organização AWS Organizations e em suas contas em seu nome. O serviço confiável cria um perfil vinculado ao serviço em cada conta, quando esse perfil é necessário, para realizar tarefas de gerenciamento para você. Para obter mais informações, consulte Usando AWS Organizations com outros AWS serviços na AWS Organizations documentação.

tuning (ajustar)

Alterar aspectos do processo de treinamento para melhorar a precisão do modelo de ML. Por exemplo, você pode treinar o modelo de ML gerando um conjunto de rótulos, adicionando rótulos e repetindo essas etapas várias vezes em configurações diferentes para otimizar o modelo.

equipe de duas pizzas

Uma pequena DevOps equipe que você pode alimentar com duas pizzas. Uma equipe de duas pizzas garante a melhor oportunidade possível de colaboração no desenvolvimento de software.

U

incerteza

Um conceito que se refere a informações imprecisas, incompletas ou desconhecidas que podem minar a confiabilidade dos modelos preditivos de ML. Há dois tipos de incertezas: a incerteza epistêmica é causada por dados limitados e incompletos, enquanto a incerteza aleatória é causada pelo ruído e pela aleatoriedade inerentes aos dados.

tarefas indiferenciadas

Também conhecido como trabalho pesado, trabalho necessário para criar e operar um aplicativo, mas que não fornece valor direto ao usuário final nem oferece vantagem competitiva. Exemplos de tarefas indiferenciadas incluem aquisição, manutenção e planejamento de capacidade.

ambientes superiores

Veja ambiente.

V

aspiração

Uma operação de manutenção de banco de dados que envolve limpeza após atualizações incrementais para recuperar armazenamento e melhorar a performance.

controle de versões

Processos e ferramentas que rastreiam mudanças, como alterações no código-fonte em um repositório.

emparelhamento da VPC

Uma conexão entre duas VPCs que permite rotear o tráfego usando endereços IP privados. Para ter mais informações, consulte O que é emparelhamento de VPC? na documentação da Amazon VPC.

Vulnerabilidade

Uma falha de software ou hardware que compromete a segurança do sistema.

W

cache quente

Um cache de buffer que contém dados atuais e relevantes que são acessados com frequência. A instância do banco de dados pode ler do cache do buffer, o que é mais rápido do que ler da memória principal ou do disco.

dados mornos

Dados acessados raramente. Ao consultar esse tipo de dados, consultas moderadamente lentas geralmente são aceitáveis.

função de janela

Uma função SQL que executa um cálculo em um grupo de linhas que se relacionam de alguma forma com o registro atual. As funções de janela são úteis para processar tarefas, como calcular uma média móvel ou acessar o valor das linhas com base na posição relativa da linha atual.

workload

Uma coleção de códigos e recursos que geram valor empresarial, como uma aplicação voltada para o cliente ou um processo de backend.

workstreams

Grupos funcionais em um projeto de migração que são responsáveis por um conjunto específico de tarefas. Cada workstream é independente, mas oferece suporte aos outros workstreams do projeto. Por exemplo, o workstream de portfólio é responsável por priorizar aplicações, planejar ondas e coletar metadados de migração. O workstream de portfólio entrega esses ativos ao workstream de migração, que então migra os servidores e as aplicações.

WORM

Veja gravação única e várias leituras.

WQF

Veja AWS Workload Qualification Framework.

gravação única e várias leituras (WORM)

Um modelo de armazenamento que grava dados uma única vez e evita que os dados sejam excluídos ou modificados. Os usuários autorizados podem ler os dados quantas vezes forem necessárias, mas não podem alterá-los. Essa infraestrutura de armazenamento de dados é considerada imutável.

Z

exploração de dia zero

Um ataque, normalmente malware, que tira proveito de uma vulnerabilidade zero-day.

vulnerabilidade de dia zero

Uma falha ou vulnerabilidade não mitigada em um sistema de produção. Os agentes de ameaças podem usar esse tipo de vulnerabilidade para atacar o sistema. Os desenvolvedores frequentemente ficam cientes da vulnerabilidade como resultado do ataque.

prompt zero shot

Fornecer a um LLM instruções para realizar uma tarefa, mas sem exemplos (shots) que possam ajudar a orientá-lo. O LLM deve usar seu conhecimento pré-treinado para lidar com a tarefa. A

eficácia dos prompts zero-shot depende da complexidade da tarefa e da qualidade do prompt. Veja também prompts few-shot.

aplicação zumbi

Uma aplicação que tem um uso médio de CPU e memória inferior a 5%. Em um projeto de migração, é comum retirar essas aplicações.

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