Bewährte Methoden für die Verwendung des Terraform Providers AWS

# AWS Präskriptive Leitlinien

# AWS Präskriptive Leitlinien: Bewährte Methoden für die Verwendung des Terraform Providers AWS

# Table of Contents

# Bewährte Methoden für die Verwendung des Terraform AWS-Anbieters

Michael Begin, leitender DevOps Berater, Amazon Web Services (AWS)

Mai 2024 (Verlauf der Dokumente)

Die Verwaltung von Infrastructure as Code (IaC) mit aktiviertem Terraform AWS bietet wichtige Vorteile wie verbesserte Konsistenz, Sicherheit und Agilität. Da Ihre Terraform-Konfiguration jedoch an Größe und Komplexität zunimmt, wird es wichtig, bewährte Methoden zu befolgen, um Fallstricke zu vermeiden.

Dieses Handbuch enthält empfohlene Best Practices für die Verwendung des AWS Terraform Providers von. HashiCorp Es führt Sie durch die richtige Versionierung, Sicherheitskontrollen, Remote-Backends, Codebasisstruktur und Community-Anbieter, auf denen Terraform optimiert werden kann. AWS Jeder Abschnitt befasst sich eingehender mit den Einzelheiten der Anwendung dieser bewährten Methoden:

- Sicherheit
- Backends
- Struktur und Organisation der Codebasis
- AWS Versionsverwaltung für Anbieter
- Community-Module

## Ziele

Dieser Leitfaden hilft Ihnen dabei, betriebliches Wissen über den Terraform AWS Provider zu erwerben, und befasst sich mit den folgenden Geschäftszielen, die Sie erreichen können, indem Sie die bewährten IaC-Methoden in Bezug auf Sicherheit, Zuverlässigkeit, Compliance und Entwicklerproduktivität befolgen.

- Verbessern Sie die Qualität und Konsistenz des Infrastrukturcodes in allen Terraform-Projekten.
- Beschleunigen Sie das Onboarding von Entwicklern und die Fähigkeit, zum Infrastrukturcode beizutragen.
- Erhöhen Sie die geschäftliche Flexibilität durch schnellere Infrastrukturänderungen.

- Reduzieren Sie Fehler und Ausfallzeiten im Zusammenhang mit Infrastrukturänderungen.
- Optimieren Sie die Infrastrukturkosten, indem Sie die Best Practices von IaC befolgen.
- Stärken Sie Ihre allgemeine Sicherheitslage durch die Implementierung von Best Practices.

# Zielpublikum

Zur Zielgruppe dieses Leitfadens gehören technische Leiter und Manager, die Teams beaufsichtigen, auf denen Terraform for IaC verwendet wird. AWS Zu den weiteren potenziellen Lesern gehören Infrastrukturingenieure, DevOps Ingenieure, Lösungsarchitekten und Entwickler, die Terraform aktiv zur Verwaltung der Infrastruktur verwenden. AWS

Die Einhaltung dieser bewährten Methoden spart Zeit und trägt dazu bei, die Vorteile von IaC für diese Rollen zu nutzen.

# Übersicht

Terraform-Anbieter sind Plugins, die es Terraform ermöglichen, mit verschiedenen APIs zu interagieren. Der Terraform AWS Provider ist das offizielle Plugin für die Verwaltung von AWS Infrastructure as Code (IaC) mit Terraform. Es übersetzt die Terraform-Syntax in AWS API-Aufrufe zum Erstellen, Lesen, Aktualisieren und Löschen von Ressourcen. AWS

Der AWS Anbieter kümmert sich um die Authentifizierung, übersetzt die Terraform-Syntax in AWS API-Aufrufe und stellt Ressourcen bereit. AWS Sie verwenden einen `provider` Terraform-Codeblock, um das Anbieter-Plugin zu konfigurieren, das Terraform für die Interaktion mit der API verwendet. AWS Sie können mehrere AWS Provider-Blöcke konfigurieren, um Ressourcen in verschiedenen Regionen zu verwalten. AWS-Konten

Hier ist ein Beispiel für eine Terraform-Konfiguration, die mehrere AWS Provider-Blöcke mit Aliassen verwendet, um eine Amazon Relational Database Service (Amazon RDS) -Datenbank zu verwalten, die über ein Replikat in einer anderen Region und einem anderen Konto verfügt. Der primäre und der sekundäre Anbieter nehmen unterschiedliche AWS Identity and Access Management Rollen (IAM) ein:

```
# 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
}
```

In diesem Beispiel:

- Der erste `provider` Block konfiguriert den primären AWS Anbieter in der `us-west-1` Region mit dem Alias. `primary`
- Der zweite `provider` Block konfiguriert einen sekundären AWS Anbieter in der `us-east-1` Region mit dem Alias. `replica` Dieser Anbieter wird verwendet, um eine Lesereplik der Primärdatenbank in einer anderen Region und einem anderen Konto zu erstellen. Der `assume_role` Block wird verwendet, um eine IAM-Rolle im Replikatkonto anzunehmen. Der `role_arn` gibt den Amazon-Ressourcennamen (ARN) der zu übernehmenden IAM-Rolle an und `session_name` ist ein eindeutiger Bezeichner für die Terraform-Sitzung.
- Die `aws_db_instance.primary` Ressource erstellt die primäre Amazon RDS-Datenbank mithilfe des `primary` Anbieters in der `us-west-1` Region.
- Die `aws_db_instance.read_replica` Ressource erstellt mithilfe des `replica` Anbieters eine Lesereplik der Primärdatenbank in der `us-east-1` Region. Das `replicate_source_db` Attribut verweist auf die ID der `primary` Datenbank.

# Bewährte Methoden für die Gewährleistung der Sicherheit

Die ordnungsgemäße Verwaltung von Authentifizierung, Zugriffskontrollen und Sicherheit ist für die sichere Nutzung des Terraform Providers von entscheidender Bedeutung. AWS In diesem Abschnitt werden bewährte Methoden in folgenden Bereichen beschrieben:

- IAM-Rollen und -Berechtigungen für den Zugriff mit den geringsten Rechten
- Sicherung von Anmeldeinformationen, um unbefugten Zugriff auf Konten und Ressourcen zu verhindern AWS
- Verschlüsselung per Fernzugriff zum Schutz sensibler Daten
- Scannen von Infrastruktur und Quellcode zur Identifizierung von Fehlkonfigurationen
- Zugriffskontrollen für den Remote-State-Speicher
- Durchsetzung der Richtlinien von Sentinel zur Implementierung von Leitplanken

Die Einhaltung dieser bewährten Methoden trägt dazu bei, Ihre Sicherheitslage zu stärken, wenn Sie Terraform zur Verwaltung der Infrastruktur verwenden. AWS

## Folgen Sie dem Prinzip der geringsten Rechte

Die geringsten Rechte sind ein grundlegendes Sicherheitsprinzip, das sich darauf bezieht, dass einem Benutzer, Prozess oder System nur die Mindestberechtigungen gewährt werden, die erforderlich sind, damit ein Benutzer, Prozess oder System die vorgesehenen Funktionen ausführen kann. Es ist ein Kernkonzept der Zugriffskontrolle und eine vorbeugende Maßnahme gegen unbefugten Zugriff und potenzielle Datenschutzverletzungen.

Das Prinzip der geringsten Rechte wird in diesem Abschnitt mehrfach hervorgehoben, da es sich direkt darauf bezieht, wie Terraform sich authentifiziert und Aktionen gegen Cloud-Anbieter wie ausführt. AWS

Wenn Sie Terraform zur Bereitstellung und Verwaltung von AWS Ressourcen verwenden, handelt es im Namen einer Entität (Benutzer oder Rolle), die entsprechende Berechtigungen für API-Aufrufe benötigt. Die Nichteinhaltung der geringsten Rechte birgt große Sicherheitsrisiken:

- Wenn Terraform über zu viele Berechtigungen verfügt, die über die erforderlichen Berechtigungen hinausgehen, kann eine unbeabsichtigte Fehlkonfiguration zu unerwünschten Änderungen oder Löschungen führen.

- Zu freizügige Zugriffsberechtigungen erhöhen den Wirkungsbereich, wenn Terraform-Statusdateien oder Anmeldeinformationen kompromittiert werden.
- Die Nichteinhaltung der geringsten Rechte verstößt gegen bewährte Sicherheitsverfahren und die Einhaltung gesetzlicher Vorschriften zur Gewährung eines nur minimal erforderlichen Zugriffs.

# Verwenden von IAM-Rollen

Verwenden Sie nach Möglichkeit IAM-Rollen anstelle von IAM-Benutzern, um die Sicherheit mit dem AWS Terraform Provider zu erhöhen. IAM-Rollen stellen temporäre Sicherheitsanmeldedaten bereit, die automatisch rotieren, sodass die Verwaltung langfristiger Zugriffsschlüssel entfällt. Rollen bieten außerdem präzise Zugriffskontrollen durch IAM-Richtlinien.

## Gewähren Sie mithilfe von IAM-Richtlinien Zugriff mit geringsten Rechten

Erstellen Sie sorgfältig IAM-Richtlinien, um sicherzustellen, dass Rollen und Benutzer nur über die Mindestberechtigungen verfügen, die für ihre Arbeitslast erforderlich sind. Beginnen Sie mit einer leeren Richtlinie und fügen Sie schrittweise zulässige Dienste und Aktionen hinzu. Um dies zu erreichen:

- Aktivieren Sie IAM Access Analyzer, um Richtlinien auszuwerten und ungenutzte Berechtigungen hervorzuheben, die entfernt werden können.
- Überprüfen Sie die Richtlinien manuell, um alle Funktionen zu entfernen, die für die vorgesehene Verantwortung der Rolle nicht unbedingt erforderlich sind.
- Verwenden Sie IAM-Richtlinienvariablen und -Tags, um die Rechteverwaltung zu vereinfachen.

Gut ausgearbeitete Richtlinien gewähren gerade genug Zugriff, um die Aufgaben des Workloads zu erfüllen, mehr nicht. Definieren Sie Aktionen auf Vorgangsebene und lassen Sie Aufrufe nur für bestimmte Ressourcen zu, die benötigt werden APIs .

Die Befolgung dieser bewährten Methode reduziert das Ausmaß der Auswirkungen und entspricht den grundlegenden Sicherheitsprinzipien der Aufgabentrennung und des geringsten Zugriffs. Fangen Sie bei Bedarf schrittweise mit striktem und offenem Zugang an, anstatt zunächst offen zu beginnen und zu versuchen, den Zugriff später einzuschränken.

## Nehmen Sie IAM-Rollen für die lokale Authentifizierung an

Wenn Sie Terraform lokal ausführen, vermeiden Sie die Konfiguration statischer Zugriffsschlüssel. Verwenden Sie stattdessen IAM-Rollen, um vorübergehend privilegierten Zugriff zu gewähren, ohne langfristige Anmeldeinformationen preiszugeben.

Erstellen Sie zunächst eine IAM-Rolle mit den erforderlichen Mindestberechtigungen und fügen Sie eine Vertrauensbeziehung hinzu, sodass die IAM-Rolle von Ihrem Benutzerkonto oder Ihrer föderierten Identität übernommen werden kann. Dadurch wird die temporäre Nutzung der Rolle autorisiert.

Beispiel für eine Vertrauensbeziehungsrichtlinie:

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

Führen Sie dann den AWS CLI Befehl aws sts assume-role aus, um kurzlebige Anmeldeinformationen für die Rolle abzurufen. Diese Anmeldeinformationen sind in der Regel eine Stunde lang gültig.

AWS CLI Beispiel für einen Befehl:

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

Die Ausgabe des Befehls enthält einen Zugriffsschlüssel, einen geheimen Schlüssel und ein Sitzungstoken, mit denen Sie sich authentifizieren können für AWS:

```
{
    "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"
    }
}
```

Der AWS Anbieter kann die Übernahme der Rolle auch automatisch übernehmen.

Beispiel für die Anbieterkonfiguration für die Übernahme einer IAM-Rolle:

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

Dadurch werden erweiterte Rechte ausschließlich für die Dauer der Terraform-Sitzung gewährt. Die temporären Schlüssel können nicht durchgesickert werden, da sie nach der maximalen Sitzungsdauer automatisch ablaufen.

Zu den wichtigsten Vorteilen dieser bewährten Methode gehören eine verbesserte Sicherheit im Vergleich zu langlebigen Zugriffsschlüsseln, detaillierte Zugriffskontrollen für die Rolle mit den geringsten Rechten und die Möglichkeit, den Zugriff einfach zu widerrufen, indem die Berechtigungen der Rolle geändert werden. Durch die Verwendung von IAM-Rollen müssen Sie außerdem Geheimnisse nicht direkt lokal in Skripten oder auf der Festplatte speichern, sodass Sie die Terraform-Konfiguration sicher im gesamten Team gemeinsam nutzen können.

## Verwenden Sie IAM-Rollen für die Amazon-Authentifizierung EC2

Wenn Sie Terraform von Amazon Elastic Compute Cloud (Amazon EC2) -Instances aus ausführen, vermeiden Sie es, langfristige Anmeldeinformationen lokal zu speichern. Verwenden Sie stattdessen

IAM-Rollen und Instanzprofile, um automatisch Berechtigungen mit den geringsten Rechten zu gewähren.

Erstellen Sie zunächst eine IAM-Rolle mit den Mindestberechtigungen und weisen Sie die Rolle dem Instanzprofil zu. Das Instanzprofil ermöglicht es EC2 Instanzen, die in der Rolle definierten Berechtigungen zu erben. Starten Sie dann Instances, indem Sie dieses Instanzprofil angeben. Die Instanz authentifiziert sich über die zugeordnete Rolle.

Bevor Sie Terraform-Operationen ausführen, stellen Sie sicher, dass die Rolle in den Instanz-Metadaten vorhanden ist, um sicherzustellen, dass die Anmeldeinformationen erfolgreich vererbt wurden.

```
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/
```

Bei diesem Ansatz wird vermieden, dass permanente AWS Schlüssel in Skripten oder in der Terraform-Konfiguration innerhalb der Instanz fest codiert werden. Die temporären Anmeldeinformationen werden Terraform transparent über die Instanzrolle und das Profil zur Verfügung gestellt.

Zu den wichtigsten Vorteilen dieser bewährten Methode gehören eine verbesserte Sicherheit gegenüber langfristigen Anmeldeinformationen, ein geringerer Aufwand für die Verwaltung von Anmeldeinformationen und die Konsistenz zwischen Entwicklungs-, Test- und Produktionsumgebungen. Die IAM-Rollenauthentifizierung vereinfacht die Ausführung von EC2 Terraform-Instanzen aus und erzwingt gleichzeitig den Zugriff mit den geringsten Rechten.

## Verwenden Sie dynamische Anmeldeinformationen für HCP Terraform-Arbeitsbereiche

HCP Terraform ist ein verwalteter Service von, der Teams dabei unterstützt HashiCorp , Terraform zur Bereitstellung und Verwaltung der Infrastruktur für mehrere Projekte und Umgebungen zu verwenden. Wenn Sie Terraform in HCP Terraform ausführen, verwenden Sie dynamische Anmeldeinformationen, um die Authentifizierung zu vereinfachen und zu sichern. AWS Terraform tauscht bei jedem Lauf automatisch temporäre Anmeldeinformationen aus, ohne dass eine IAM-Rollenübernahme erforderlich ist.

Zu den Vorteilen gehören eine einfachere geheime Rotation, eine zentralisierte Verwaltung von Anmeldeinformationen in allen Arbeitsbereichen, Berechtigungen mit den geringsten Rechten und der Wegfall fest codierter Schlüssel. Die Verwendung von kurzlebigen Hash-Schlüsseln erhöht die Sicherheit im Vergleich zu langlebigen Zugriffsschlüsseln.

### Verwenden Sie IAM-Rollen in AWS CodeBuild

Führen Sie Ihre Builds unter Verwendung einer IAM-Rolle aus, die dem CodeBuild Projekt zugewiesen ist. AWS CodeBuild Dadurch kann jeder Build automatisch temporäre Anmeldeinformationen von der Rolle erben, anstatt langfristige Schlüssel zu verwenden.

### Führen Sie GitHub Aktionen remote auf HCP Terraform aus

Konfigurieren Sie GitHub Aktions-Workflows, um Terraform remote auf HCP Terraform-Workspaces auszuführen. Verlassen Sie sich auf dynamische Anmeldeinformationen und Fernsperren von Zuständen statt auf die Verwaltung von Geheimnissen. GitHub

### Verwenden Sie GitHub Aktionen mit OIDC und konfigurieren Sie die AWS Aktion „Anmeldeinformationen“

Verwenden Sie den OpenID Connect (OIDC) -Standard, um die Identität von GitHub Actions über IAM zu bündeln. Verwenden Sie die Aktion AWS Anmeldeinformationen konfigurieren, um das GitHub Token gegen temporäre AWS Anmeldeinformationen auszutauschen, ohne langfristige Zugriffsschlüssel zu benötigen.

### Verwenden Sie es GitLab mit OIDC und AWS CLI

Verwenden Sie den OIDC-Standard, um GitLab Identitäten über IAM für den temporären Zugriff zu föderieren. Wenn Sie sich auf OIDC verlassen, müssen Sie langfristige Zugriffsschlüssel nicht direkt innerhalb von OIDC verwalten. AWS GitLab Anmeldeinformationen werden ausgetauscht just-in-time, was die Sicherheit verbessert. Je nach den Berechtigungen in der IAM-Rolle erhalten Benutzer außerdem Zugriff mit den geringsten Rechten.

## Verwenden Sie einzigartige IAM-Benutzer mit älteren Automatisierungstools

Wenn Sie über Automatisierungstools und Skripts verfügen, die keine native Unterstützung für die Verwendung von IAM-Rollen bieten, können Sie individuelle IAM-Benutzer erstellen, um

programmatischen Zugriff zu gewähren. Das Prinzip der geringsten Rechte gilt weiterhin. Reduzieren Sie die Anzahl der Richtlinienberechtigungen und verlassen Sie sich auf separate Rollen für jede Pipeline oder jedes Skript. Beginnen Sie bei der Migration zu moderneren Tools oder Skripten damit, Rollen nativ zu unterstützen, und wechseln Sie schrittweise zu diesen Rollen.

> ⚠ Warning
>
> IAM-Benutzer verfügen über langfristige Anmeldeinformationen, die ein Sicherheitsrisiko darstellen. Um dieses Risiko zu minimieren, empfehlen wir, diesen Benutzern nur die Berechtigungen zu gewähren, die sie für die Ausführung der Aufgabe benötigen, und diese Benutzer zu entfernen, wenn sie nicht mehr benötigt werden.

### Verwenden Sie das Plugin Jenkins AWS Credentials

Verwenden Sie das AWS Credentials-Plugin in Jenkins, um Anmeldeinformationen zentral zu konfigurieren und dynamisch in Builds AWS einzufügen. Dadurch wird vermieden, dass Geheimnisse in die Quellcodeverwaltung eingecheckt werden.

## Kontinuierliche Überwachung, Validierung und Optimierung der geringsten Rechte

Im Laufe der Zeit können zusätzliche Berechtigungen erteilt werden, die die erforderlichen Mindestrichtlinien überschreiten können. Analysieren Sie kontinuierlich den Zugriff, um unnötige Berechtigungen zu identifizieren und zu entfernen.

### Überwachen Sie kontinuierlich die Verwendung von Zugriffsschlüsseln

Wenn Sie die Verwendung von Zugriffsschlüsseln nicht vermeiden können, verwenden Sie IAM-Anmeldedatenberichte, um nach ungenutzten Zugriffsschlüsseln zu suchen, die älter als 90 Tage sind, und inaktive Schlüssel sowohl für Benutzerkonten als auch für Computerrollen zu widerrufen. Weisen Sie die Administratoren darauf hin, das Entfernen von Schlüsseln für aktive Mitarbeiter und Systeme manuell zu bestätigen.

Durch die Überwachung der Schlüsselnutzung können Sie Ihre Berechtigungen optimieren, da Sie ungenutzte Berechtigungen identifizieren und entfernen können. Wenn Sie diese bewährte Methode bei der Rotation von Zugriffsschlüsseln befolgen, wird die Lebensdauer der Anmeldeinformationen begrenzt und der Zugriff mit den geringsten Rechten durchgesetzt.

AWS stellt mehrere Dienste und Funktionen bereit, mit denen Sie Warnmeldungen und Benachrichtigungen für Administratoren einrichten können. Hier sind einige Optionen:

- AWS Config: Sie können AWS Config Regeln verwenden, um die Konfigurationseinstellungen Ihrer AWS Ressourcen, einschließlich der IAM-Zugriffsschlüssel, auszuwerten. Sie können benutzerdefinierte Regeln erstellen, um nach bestimmten Bedingungen zu suchen, z. B. nach ungenutzten Zugriffsschlüsseln, die älter als eine bestimmte Anzahl von Tagen sind. Wenn eine Regel verletzt wird, AWS Config kann eine Bewertung zur Behebung gestartet oder Benachrichtigungen an ein Amazon Simple Notification Service (Amazon SNS) -Thema gesendet werden.
- AWS Security Hub: Security Hub bietet einen umfassenden Überblick über den Sicherheitsstatus Ihres AWS Kontos und kann Ihnen helfen, potenzielle Sicherheitsprobleme zu erkennen und Sie darüber zu informieren, einschließlich ungenutzter oder inaktiver IAM-Zugangsschlüssel. Security Hub kann mit Amazon EventBridge und Amazon SNS oder Amazon Q Developer in Chat-Anwendungen integriert werden, um Benachrichtigungen an Administratoren zu senden.
- AWS Lambda: Lambda-Funktionen können durch verschiedene Ereignisse aufgerufen werden, einschließlich Amazon CloudWatch Events oder AWS Config Regeln. Sie können benutzerdefinierte Lambda-Funktionen schreiben, um die Verwendung von IAM-Zugriffsschlüsseln zu bewerten, zusätzliche Prüfungen durchzuführen und Benachrichtigungen zu senden, indem Sie Dienste wie Amazon SNS oder Amazon Q Developer in Chat-Anwendungen verwenden.

## Überprüfen Sie kontinuierlich die IAM-Richtlinien

Verwenden Sie IAM Access Analyzer, um Richtlinien zu bewerten, die Rollen zugeordnet sind, und um ungenutzte Dienste oder überflüssige Aktionen zu identifizieren, denen gewährt wurde. Führen Sie regelmäßige Zugriffsprüfungen durch, um manuell zu überprüfen, ob die Richtlinien den aktuellen Anforderungen entsprechen.

Vergleichen Sie die bestehende Richtlinie mit der von IAM Access Analyzer generierten Richtlinie und entfernen Sie alle nicht benötigten Berechtigungen. Sie sollten den Benutzern auch Berichte zur Verfügung stellen und ungenutzte Berechtigungen nach Ablauf einer Übergangsfrist automatisch widerrufen. Auf diese Weise wird sichergestellt, dass nur minimale Richtlinien in Kraft bleiben.

Durch den proaktiven und häufigen Widerruf veralteter Zugriffe werden die Anmeldedaten minimiert, die bei einer Sicherheitsverletzung gefährdet sein könnten. Die Automatisierung sorgt für eine nachhaltige, langfristige Hygiene von Anmeldedaten und eine Optimierung der

Zugriffsberechtigungen. Die Einhaltung dieser bewährten Methode begrenzt den Wirkungsbereich, indem proaktiv die geringsten Rechte für AWS Identitäten und Ressourcen durchgesetzt werden.

# Sicherer Remote-State-Speicher

Remote-State-Speicher bezieht sich auf das Remote-Speichern der Terraform-Statusdatei statt lokal auf dem Computer, auf dem Terraform ausgeführt wird. Die Statusdatei ist von entscheidender Bedeutung, da sie die von Terraform bereitgestellten Ressourcen und deren Metadaten verfolgt.

Wenn der Remotestatus nicht gesichert wird, kann dies zu schwerwiegenden Problemen wie dem Verlust von Statusdaten, der Unfähigkeit, die Infrastruktur zu verwalten, dem versehentlichen Löschen von Ressourcen und der Offenlegung vertraulicher Informationen führen, die möglicherweise in der Statusdatei enthalten sind. Aus diesem Grund ist die Sicherung des Remote-State-Speichers für die Nutzung von Terraform in Produktionsqualität von entscheidender Bedeutung.

## Aktivieren Sie Verschlüsselung und Zugriffskontrollen

Verwenden Sie die serverseitige Verschlüsselung (SSE) von Amazon Simple Storage Service (Amazon S3), um den Remote-Status im Ruhezustand zu verschlüsseln.

## Beschränken Sie den direkten Zugriff auf kollaborative Workflows

- Strukturieren Sie Workflows für die Zusammenarbeit in HCP Terraform oder in einer CI/CD-Pipeline innerhalb Ihres Git-Repositorys, um den direkten Statuszugriff einzuschränken.
- Verlassen Sie sich auf Pull-Requests, führen Sie Genehmigungen, Richtlinienprüfungen und Benachrichtigungen durch, um Änderungen zu koordinieren.

Die Einhaltung dieser Richtlinien trägt dazu bei, vertrauliche Ressourcenattribute zu schützen und Konflikte mit Änderungen der Teammitglieder zu vermeiden. Verschlüsselung und strenger Zugriffsschutz tragen dazu bei, die Angriffsfläche zu reduzieren, und Workflows für die Zusammenarbeit fördern die Produktivität.

# Verwenden AWS Secrets Manager

Es gibt viele Ressourcen und Datenquellen in Terraform, die geheime Werte im Klartext in der Statusdatei speichern. Vermeiden Sie es, Geheimnisse im Status zu speichern — verwenden Sie stattdessen. AWS Secrets Manager

Anstatt zu versuchen, sensible Werte manuell zu verschlüsseln, sollten Sie sich auf die integrierte Unterstützung von Terraform für die Verwaltung sensibler Zustände verlassen. Stellen Sie beim Exportieren sensibler Werte in die Ausgabe sicher, dass die Werte als sensibel gekennzeichnet sind.

# Scannen Sie kontinuierlich Infrastruktur und Quellcode

Scannen Sie sowohl die Infrastruktur als auch den Quellcode proaktiv und kontinuierlich auf Risiken wie offengelegte Anmeldeinformationen oder Fehlkonfigurationen, um Ihre Sicherheitslage zu verbessern. Korrigieren Sie die Ergebnisse umgehend, indem Sie Ressourcen neu konfigurieren oder patchen.

## Verwenden Sie AWS Dienste für dynamisches Scannen

Verwenden Sie AWS native Tools wie Amazon Inspector AWS Security Hub, Amazon Detective und Amazon, GuardDuty um die bereitgestellte Infrastruktur konten- und regionsübergreifend zu überwachen. Planen Sie wiederkehrende Scans in Security Hub, um Bereitstellungs- und Konfigurationsabweichungen zu verfolgen. Scannen Sie EC2 Instanzen, Lambda-Funktionen, Container, S3-Buckets und andere Ressourcen.

## Führen Sie eine statische Analyse durch

Betten Sie statische Analysatoren wie Checkov direkt in CI/CD-Pipelines ein, um den Terraform-Konfigurationscode (HCL) zu scannen und Risiken vor der Bereitstellung präventiv zu identifizieren. Dadurch werden Sicherheitsüberprüfungen an einen früheren Punkt im Entwicklungsprozess verlagert (als Verlagerung nach links bezeichnet) und eine falsch konfigurierte Infrastruktur verhindert.

## Sorgen Sie für eine schnelle Problembehebung

Stellen Sie bei allen Scanergebnissen eine schnelle Behebung sicher, indem Sie entweder die Terraform-Konfiguration aktualisieren, Patches anwenden oder Ressourcen je nach Bedarf manuell neu konfigurieren. Senken Sie das Risiko, indem Sie die Grundursachen angehen.

Sowohl das Scannen von Infrastrukturen als auch das Scannen von Code bieten umfassende Einblicke in die Terraform-Konfigurationen, die bereitgestellten Ressourcen und den Anwendungscode. Durch präventive, detektive und reaktive Kontrollen werden Risiken und die Einhaltung gesetzlicher Vorschriften maximiert und gleichzeitig die Sicherheit bereits zu einem früheren Zeitpunkt in den Softwareentwicklungszyklus (SDLC) integriert.

# Setzen Sie Richtlinienprüfungen durch

Verwenden Sie Code-Frameworks wie HashiCorp Sentinel-Richtlinien, um Governance-Richtlinien und standardisierte Vorlagen für die Infrastrukturbereitstellung mit Terraform bereitzustellen.

Sentinel-Richtlinien können Anforderungen oder Einschränkungen für die Terraform-Konfiguration definieren, um sie an die Unternehmensstandards und Best Practices anzupassen. Sie können Sentinel-Richtlinien beispielsweise verwenden, um:

- Erfordert Tags für alle Ressourcen.
- Beschränken Sie die Instanztypen auf eine Liste mit genehmigten Instanzen.
- Erzwingen Sie obligatorische Variablen.
- Verhindern Sie die Zerstörung von Produktionsressourcen.

Die Einbettung von Richtlinienprüfungen in die Lebenszyklen der Terraform-Konfiguration ermöglicht die proaktive Durchsetzung von Standards und Architekturrichtlinien. Sentinel bietet eine gemeinsame Richtlinienlogik, die dazu beiträgt, die Entwicklung zu beschleunigen und gleichzeitig nicht genehmigte Praktiken zu verhindern.

# Bewährte Methoden im Backend

Die Verwendung eines geeigneten Remote-Backends zum Speichern Ihrer Statusdatei ist entscheidend für die Zusammenarbeit, die Sicherstellung der Integrität von Statusdateien durch Sperren, die Bereitstellung zuverlässiger Backups und Wiederherstellungen, die Integration in CI/CD-Workflows und die Nutzung der erweiterten Sicherheits-, Governance- und Verwaltungsfunktionen, die von verwalteten Diensten wie HCP Terraform angeboten werden.

Terraform unterstützt verschiedene Backend-Typen wie Kubernetes, Consul und HTTP. HashiCorp Dieser Leitfaden konzentriert sich jedoch auf Amazon S3, eine optimale Backend-Lösung für die meisten AWS Benutzer.

Als vollständig verwalteter Objektspeicherservice, der eine hohe Beständigkeit und Verfügbarkeit bietet, bietet Amazon S3 ein sicheres, skalierbares und kostengünstiges Backend für die Verwaltung von Terraform State On. AWS Die globale Präsenz und Widerstandsfähigkeit von Amazon S3 übertreffen das, was die meisten Teams durch die Selbstverwaltung von State Storage erreichen können. Darüber hinaus ist Amazon S3 aufgrund der nativen Integration mit AWS Zugriffskontrollen, Verschlüsselungsoptionen, Versionierungsfunktionen und anderen Diensten eine praktische Backend-Wahl.

Dieser Leitfaden bietet keine Backend-Anleitungen für andere Lösungen wie Kubernetes oder Consul, da die Hauptzielgruppe Kunden sind. AWS Für Teams, die voll im Einsatz sind AWS Cloud, ist Amazon S3 in der Regel die ideale Wahl gegenüber Kubernetes- oder HashiCorp Consul-Clustern. Die Einfachheit, Stabilität und enge AWS Integration von Amazon S3 State Storage bieten eine optimale Grundlage für die meisten Benutzer, die AWS Best Practices befolgen. Teams können die Robustheit, den Backup-Schutz und die Verfügbarkeit von AWS Diensten nutzen, um den Terraform-Fernstatus äußerst widerstandsfähig zu halten.

Die Befolgung der Backend-Empfehlungen in diesem Abschnitt führt zu einer besseren Zusammenarbeit der Terraform-Codebasen und begrenzt gleichzeitig die Auswirkungen von Fehlern oder unbefugten Änderungen. Durch die Implementierung eines gut konzipierten Remote-Backends können Teams die Terraform-Workflows optimieren.

Bewährte Verfahren:

- Verwenden Sie Amazon S3 für den Remotespeicher
- Erleichtern Sie die Zusammenarbeit im Team
- Trennen Sie die Backends für jede Umgebung

- Überwachen Sie aktiv die Aktivitäten im Fernstatus

# Verwenden Sie Amazon S3 für den Remotespeicher

Das Remote-Speichern des Terraform-Status in Amazon S3 und die Implementierung von Zustandssperren und Konsistenzprüfungen mithilfe von Amazon DynamoDB bieten große Vorteile gegenüber der lokalen Dateispeicherung. Remote State ermöglicht die Zusammenarbeit im Team, die Nachverfolgung von Änderungen, den Schutz von Backups und das Sperren per Fernzugriff für mehr Sicherheit.

Die Verwendung von Amazon S3 mit der Speicherklasse S3 Standard (Standard) anstelle von kurzlebigem lokalem Speicher oder selbstverwalteten Lösungen bietet eine Beständigkeit von 99,999999999% und einen Verfügbarkeitsschutz von 99,99%, um versehentlichen Datenverlust zu verhindern. AWS Managed Services wie Amazon S3 und DynamoDB bieten Service Level Agreements (SLAs), die weit über das hinausgehen, was die meisten Unternehmen erreichen können, wenn sie ihren Speicher selbst verwalten. Verlassen Sie sich auf diese Schutzmaßnahmen, um den Zugriff auf Remote-Backends zu gewährleisten.

## Aktivieren Sie die Fernzustandssperrung

DynamoDB-Sperren schränken den Zustandszugriff ein, um gleichzeitige Schreibvorgänge zu verhindern. Dadurch werden gleichzeitige Änderungen durch mehrere Benutzer verhindert und Fehler reduziert.

Beispiel für eine Backend-Konfiguration mit State-Locking:

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

## Aktivieren Sie Versionierung und automatische Backups

Für zusätzlichen Schutz aktivieren Sie die automatische Versionierung und Backups mithilfe von AWS Backup Amazon S3 S3-Backends. Bei der Versionierung werden alle vorherigen Versionen

des Status beibehalten, wenn Änderungen vorgenommen werden. Außerdem können Sie bei Bedarf frühere Snapshots des Arbeitsstatus wiederherstellen, um unerwünschte Änderungen rückgängig zu machen oder die Daten nach Unfällen wiederherzustellen.

### Stellen Sie bei Bedarf frühere Versionen wieder her

Versionierte Amazon S3 S3-State-Buckets machen es einfach, Änderungen rückgängig zu machen, indem ein früherer zweifelsfrei funktionierender Status-Snapshot wiederhergestellt wird. Dies schützt vor versehentlichen Änderungen und bietet zusätzliche Backup-Funktionen.

### Verwenden Sie HCP Terraform

HCP Terraform bietet eine vollständig verwaltete Backend-Alternative zur Konfiguration Ihres eigenen Statusspeichers. HCP Terraform kümmert sich automatisch um die sichere Speicherung von Status und Verschlüsselung und schaltet gleichzeitig zusätzliche Funktionen frei.

Wenn Sie HCP Terraform verwenden, wird der Status standardmäßig remote gespeichert, was die gemeinsame Nutzung und Sperrung von Status in Ihrer gesamten Organisation ermöglicht. Detaillierte Richtlinienkontrollen helfen Ihnen dabei, den Zugriff auf Bundesstaaten und deren Änderungen einzuschränken.

Zu den weiteren Funktionen gehören Integrationen zur Versionskontrolle, Richtlinien, Workflow-Automatisierung, Variablenverwaltung und Single-Sign-On-Integrationen mit SAML. Sie können die Sentinel-Richtlinie auch als Code verwenden, um Governance-Kontrollen zu implementieren.

HCP Terraform erfordert zwar die Verwendung einer Software-as-a-Service (SaaS) -Plattform, aber für viele Teams ist es aufgrund der Vorteile in Bezug auf Sicherheit, Zugriffskontrollen, automatisierte Richtlinienprüfungen und Funktionen für die Zusammenarbeit eine optimale Wahl gegenüber dem selbstverwaltenden Statusspeicher mit Amazon S3 oder DynamoDB.

Die einfache Integration mit Diensten wie GitHub und GitLab mit geringer Konfiguration spricht auch Benutzer an, die Cloud- und SaaS-Tools für bessere Team-Workflows voll nutzen.

## Erleichtern Sie die Zusammenarbeit im Team

Verwenden Sie Remote-Backends, um Statusdaten mit allen Mitgliedern Ihres Terraform-Teams auszutauschen. Dies erleichtert die Zusammenarbeit, da das gesamte Team so Einblick in Infrastrukturänderungen hat. Gemeinsam genutzte Backend-Protokolle in Kombination mit

der Transparenz des Statusverlaufs vereinfachen das interne Änderungsmanagement. Alle Infrastrukturänderungen durchlaufen die etablierte Pipeline, wodurch die geschäftliche Flexibilität im gesamten Unternehmen erhöht wird.

### Verbessern Sie die Rechenschaftspflicht, indem Sie AWS CloudTrail

AWS CloudTrail Integrieren Sie es in den Amazon S3 S3-Bucket, um API-Aufrufe an den State-Bucket zu erfassen. Filtern Sie CloudTrail Ereignisse, die Sie verfolgen `DeleteObject`, möchten`PutObject`, und andere relevante Aufrufe.

CloudTrail In den Protokollen wird die AWS Identität des Prinzipals angezeigt, der jeden API-Aufruf zur Statusänderung ausgeführt hat. Die Identität des Benutzers kann einem Computerkonto oder Teammitgliedern zugeordnet werden, die mit dem Backend-Speicher interagieren.

Kombinieren Sie CloudTrail Protokolle mit der Amazon S3 S3-Statusversionierung, um Infrastrukturänderungen dem Prinzipal zuzuordnen, der sie angewendet hat. Durch die Analyse mehrerer Revisionen können Sie alle Aktualisierungen dem Computerkonto oder dem verantwortlichen Teammitglied zuordnen.

Wenn eine unbeabsichtigte oder störende Änderung eintritt, bietet die State-Versionierung Rollback-Funktionen. CloudTrail verfolgt die Änderung bis zum Benutzer, sodass Sie vorbeugende Verbesserungen besprechen können.

Wir empfehlen außerdem, IAM-Berechtigungen durchzusetzen, um den Zugriff auf State-Buckets einzuschränken. Insgesamt unterstützt die Versionierung und CloudTrail Überwachung von S3 die Prüfung aller Infrastrukturänderungen. Teams erhalten verbesserte Rechenschaftspflicht, Transparenz und Auditfunktionen im Hinblick auf den Terraform-Statusverlauf.

## Trennen Sie die Backends für jede Umgebung

Verwenden Sie für jede Anwendungsumgebung unterschiedliche Terraform-Backends. Separate Backends isolieren den Status zwischen Entwicklung, Test und Produktion.

### Reduzieren Sie den Umfang der Auswirkungen

Durch die Isolierung des Zustands wird sichergestellt, dass sich Änderungen in niedrigeren Umgebungen nicht auf die Produktionsinfrastruktur auswirken. Unfälle oder Experimente in Entwicklungs- und Testumgebungen haben nur begrenzte Auswirkungen.

## Beschränken Sie den Produktionszugriff

Sperren Sie die Berechtigungen für das Backend im Produktionsstatus für die meisten Benutzer auf schreibgeschützten Zugriff. Beschränken Sie, wer die Produktionsinfrastruktur auf die CI/CD-Pipeline und die Rollen von Break Glass ändern kann.

## Vereinfachen Sie die Zutrittskontrollen

Die Verwaltung von Berechtigungen auf Backend-Ebene vereinfacht die Zugriffskontrolle zwischen Umgebungen. Die Verwendung unterschiedlicher S3-Buckets für jede Anwendung und Umgebung bedeutet, dass umfassende Lese- oder Schreibberechtigungen für ganze Backend-Buckets gewährt werden können.

## Vermeiden Sie gemeinsam genutzte Arbeitsbereiche

Sie können zwar Terraform-Arbeitsbereiche verwenden, um den Status zwischen Umgebungen zu trennen, aber unterschiedliche Backends sorgen für eine stärkere Isolierung. Wenn Sie gemeinsam genutzte Arbeitsbereiche haben, können sich Unfälle immer noch auf mehrere Umgebungen auswirken.

Durch die vollständige Isolierung der Umgebungs-Backends werden die Auswirkungen einzelner Ausfälle oder Sicherheitsverletzungen minimiert. Separate Backends passen außerdem die Zugriffskontrollen an die Sensibilitätsstufe der Umgebung an. Sie können beispielsweise Schreibschutz für die Produktionsumgebung und einen breiteren Zugriff für Entwicklungs- und Testumgebungen bereitstellen.

# Überwachen Sie aktiv die Aktivitäten im Fernstatus

Die kontinuierliche Überwachung der Aktivitäten im Fernzugriff ist entscheidend für die frühzeitige Erkennung potenzieller Probleme. Achten Sie auf ungewöhnliche Entsperrungen, Änderungen oder Zugriffsversuche.

## Lassen Sie sich bei verdächtigen Entsperrungen benachrichtigen

Die meisten Statusänderungen sollten über CI/CD-Pipelines erfolgen. Generieren Sie Warnmeldungen, wenn Statusentsperrungen direkt über Entwickler-Workstations erfolgen, was auf nicht autorisierte oder ungetestete Änderungen hinweisen könnte.

# Überwachen Sie Zugriffsversuche

Authentifizierungsfehler bei Status-Buckets können auf Aufklärungsaktivitäten hinweisen. Beachten Sie, ob mehrere Konten versuchen, auf den Status zuzugreifen, oder wenn ungewöhnliche IP-Adressen angezeigt werden, was auf kompromittierte Anmeldeinformationen hinweist.

# Bewährte Methoden für die Struktur und Organisation der Codebasis

Die richtige Struktur und Organisation der Codebasis ist entscheidend, da die Nutzung von Terraform in großen Teams und Unternehmen zunimmt. Eine gut strukturierte Codebasis ermöglicht eine Zusammenarbeit in großem Maßstab und verbessert gleichzeitig die Wartbarkeit.

Dieser Abschnitt enthält Empfehlungen zur Modularität von Terraform, zu Namenskonventionen, zur Dokumentation und zu Codierungsstandards, die Qualität und Konsistenz unterstützen.

Zu den Anleitungen gehören die Aufteilung der Konfiguration in wiederverwendbare Module nach Umgebung und Komponenten, die Festlegung von Namenskonventionen mithilfe von Präfixen und Suffixen, die Dokumentation von Modulen und die klare Erläuterung von Eingaben und Ausgaben sowie die Anwendung konsistenter Formatierungsregeln mithilfe automatisierter Stilprüfungen.

Weitere bewährte Methoden umfassen die logische Organisation von Modulen und Ressourcen in einer strukturierten Hierarchie, die Katalogisierung öffentlicher und privater Module in der Dokumentation und die Zusammenfassung unnötiger Implementierungsdetails in Modulen, um die Verwendung zu vereinfachen.

Durch die Implementierung von Richtlinien für die Codebasisstruktur in Bezug auf Modularität, Dokumentation, Standards und logische Organisation können Sie eine breite Zusammenarbeit zwischen Teams unterstützen und gleichzeitig dafür sorgen, dass Terraform auch dann gewartet werden kann, wenn sich die Nutzung innerhalb eines Unternehmens verbreitet. Durch die Durchsetzung von Konventionen und Standards können Sie die Komplexität einer fragmentierten Codebasis vermeiden.

Bewährte Verfahren:

- Implementieren Sie eine Standard-Repository-Struktur
- Struktur für Modularität
- Beachten Sie die Namenskonventionen
- Verwenden Sie Ressourcen für Dateianhänge
- Verwenden Sie Standard-Tags
- Erfüllen Sie die Terraform-Registrierungsanforderungen
- Verwenden Sie die empfohlenen Modulquellen
- Befolgen Sie die Codierungsstandards

# Implementieren Sie eine Standard-Repository-Struktur

Wir empfehlen Ihnen, das folgende Repository-Layout zu implementieren. Die modulübergreifende Standardisierung dieser Konsistenzpraktiken verbessert die Auffindbarkeit, Transparenz, Organisation und Zuverlässigkeit und ermöglicht gleichzeitig die Wiederverwendung in vielen Terraform-Konfigurationen.

- Stammmodul oder Verzeichnis: Dies sollte der primäre Einstiegspunkt sowohl für Terraform-Root - als auch für wiederverwendbare Terraform-Module sein und es wird erwartet, dass es einzigartig ist. Wenn Sie eine komplexere Architektur haben, können Sie verschachtelte Module verwenden, um einfache Abstraktionen zu erstellen. Auf diese Weise können Sie die Infrastruktur anhand ihrer Architektur beschreiben, anstatt sie direkt anhand physischer Objekte zu beschreiben.
- README: Das Root-Modul und alle verschachtelten Module sollten README-Dateien enthalten. Diese Datei muss benannt werden. `README.md` Sie sollte eine Beschreibung des Moduls und dessen Verwendungszweck enthalten. Wenn Sie ein Beispiel für die Verwendung dieses Moduls mit anderen Ressourcen hinzufügen möchten, legen Sie es in ein `examples` Verzeichnis. Erwägen Sie, ein Diagramm beizufügen, das die Infrastrukturressourcen, die das Modul möglicherweise erstellt, und deren Beziehungen darstellt. Verwenden Sie Terraform-Docs, um automatisch Eingaben oder Ausgaben des Moduls zu generieren.
- main.tf: Dies ist der primäre Einstiegspunkt. Für ein einfaches Modul könnten alle Ressourcen in dieser Datei erstellt werden. Bei einem komplexen Modul kann die Ressourcenerstellung auf mehrere Dateien verteilt sein, aber alle verschachtelten Modulaufrufe sollten sich in der `main.tf` Datei befinden.
- variables.tf und outputs.tf: Diese Dateien enthalten die Deklarationen für Variablen und Ausgaben. Alle Variablen und Ausgaben sollten Beschreibungen mit einem oder zwei Sätzen enthalten, die ihren Zweck erläutern. Diese Beschreibungen werden zur Dokumentation verwendet. Weitere Informationen finden Sie in der HashiCorp Dokumentation zur Variablenkonfiguration und zur Ausgabekonfiguration.
  - Alle Variablen müssen einen definierten Typ haben.
  - Die Variablendeklaration kann auch ein Standardargument enthalten. Wenn die Deklaration ein Standardargument enthält, wird die Variable als optional betrachtet, und der Standardwert wird verwendet, wenn Sie beim Aufrufen des Moduls oder beim Ausführen von Terraform keinen Wert festlegen. Das Standardargument erfordert einen Literalwert und kann nicht auf andere Objekte in der Konfiguration verweisen. Um eine Variable erforderlich zu machen, lassen Sie einen Standard in der Variablendeklaration weg und überlegen Sie, ob die Einstellung `nullable = false` sinnvoll ist.

  - Geben Sie für Variablen mit umgebungsunabhängigen Werten (z. B.`disk_size`) Standardwerte an.
  - Geben Sie für Variablen mit umgebungsspezifischen Werten (z. B.`project_id`) keine Standardwerte an. In diesem Fall muss das aufrufende Modul aussagekräftige Werte bereitstellen.
  - Verwenden Sie leere Standardwerte für Variablen wie leere Zeichenketten oder Listen nur, wenn das Leerlassen der Variablen eine gültige Einstellung ist, die von den zugrunde liegenden APIs nicht abgelehnt wird.
  - Gehen Sie bei der Verwendung von Variablen mit Bedacht vor. Parametrisieren Sie Werte nur, wenn sie für jede Instanz oder Umgebung variieren müssen. Wenn Sie entscheiden, ob eine Variable verfügbar gemacht werden soll, stellen Sie sicher, dass Sie einen konkreten Anwendungsfall für die Änderung dieser Variablen haben. Wenn nur eine geringe Wahrscheinlichkeit besteht, dass eine Variable benötigt wird, machen Sie sie nicht verfügbar.
    - Das Hinzufügen einer Variablen mit einem Standardwert ist abwärtskompatibel.
    - Das Entfernen einer Variablen ist abwärtsinkompatibel.
    - In Fällen, in denen ein Literal an mehreren Stellen wiederverwendet wird, sollten Sie einen lokalen Wert verwenden, ohne ihn als Variable verfügbar zu machen.
  - Übergeben Sie Ausgaben nicht direkt über Eingabevariablen, da sie dadurch nicht ordnungsgemäß zum Abhängigkeitsdiagramm hinzugefügt werden können. Um sicherzustellen, dass implizite Abhängigkeiten erstellt werden, stellen Sie sicher, dass Referenzattribute von Ressourcen ausgegeben werden. Anstatt direkt auf eine Eingabevariable für eine Instanz zu verweisen, übergeben Sie das Attribut.
- locals.tf: Diese Datei enthält lokale Werte, die einem Ausdruck einen Namen zuweisen, sodass ein Name innerhalb eines Moduls mehrfach verwendet werden kann, anstatt den Ausdruck zu wiederholen. Lokale Werte sind wie die temporären lokalen Variablen einer Funktion. Die Ausdrücke in lokalen Werten sind nicht auf Literalkonstanten beschränkt. Sie können auch auf andere Werte im Modul verweisen, darunter Variablen, Ressourcenattribute oder andere lokale Werte, um sie zu kombinieren.
- providers.tf: Diese Datei enthält den Terraform-Block und die Provider-Blöcke. `provider`Blöcke dürfen nur in Root-Modulen von Modulkonsumenten deklariert werden.

  Wenn Sie HCP Terraform verwenden, fügen Sie auch einen leeren Cloud-Block hinzu. Der `cloud` Block sollte als Teil einer CI/CD-Pipeline vollständig über Umgebungsvariablen und Anmeldeinformationen für Umgebungsvariablen konfiguriert werden.

- versions.tf: Diese Datei enthält den Block required_providers. Alle Terraform-Module müssen deklarieren, welche Anbieter sie benötigen, damit Terraform diese Anbieter installieren und verwenden kann.
- data.tf: Platzieren Sie zur einfachen Konfiguration Datenquellen neben den Ressourcen, die auf sie verweisen. Wenn Sie beispielsweise ein Bild abrufen, das beim Starten einer Instanz verwendet werden soll, platzieren Sie es neben der Instanz, anstatt Datenressourcen in einer eigenen Datei zu sammeln. Wenn die Anzahl der Datenquellen zu groß wird, sollten Sie erwägen, sie in eine spezielle `data.tf` Datei zu verschieben.
- .tfvars-Dateien: Für Root-Module können Sie nicht sensible Variablen mithilfe einer Datei bereitstellen. `.tfvars` Benennen Sie die Variablendateien aus Konsistenzgründen. `terraform.tfvars` Platzieren Sie allgemeine Werte im Stammverzeichnis des Repositorys und umgebungsspezifische Werte innerhalb des `envs/` Ordners.
- Verschachtelte Module: Verschachtelte Module sollten im Unterverzeichnis existieren. `modules/` Jedes verschachtelte Modul, das über eine verfügt, `README.md` wird von einem externen Benutzer als nutzbar angesehen. Wenn a nicht `README.md` existiert, wird das Modul nur für den internen Gebrauch in Betracht gezogen. Verschachtelte Module sollten verwendet werden, um komplexes Verhalten in mehrere kleine Module aufzuteilen, die Benutzer sorgfältig auswählen können.

  Wenn das Root-Modul Aufrufe verschachtelter Module enthält, sollten diese Aufrufe relative Pfade verwenden, `./modules/sample-module` sodass Terraform sie beispielsweise als Teil desselben Repositorys oder Pakets betrachtet, anstatt sie erneut separat herunterzuladen.

  Wenn ein Repository oder Paket mehrere verschachtelte Module enthält, sollten sie idealerweise vom Aufrufer zusammengesetzt werden können, anstatt sich gegenseitig direkt aufzurufen und einen tief verschachtelten Modulbaum zu erstellen.
- Beispiele: Beispiele für die Verwendung eines wiederverwendbaren Moduls sollten sich im `examples/` Unterverzeichnis im Stammverzeichnis des Repositorys befinden. Für jedes Beispiel können Sie eine README-Datei hinzufügen, um das Ziel und die Verwendung des Beispiels zu erläutern. Beispiele für Submodule sollten ebenfalls im Stammverzeichnis `examples/` abgelegt werden.

  Da Beispiele häufig zur Anpassung in andere Repositorien kopiert werden, sollte der Quelltext von Modulblöcken auf die Adresse gesetzt werden, die ein externer Aufrufer verwenden würde, und nicht auf einen relativen Pfad.
- Dateien mit Dienstnamen: Benutzer möchten Terraform-Ressourcen häufig nach Diensten in mehreren Dateien trennen. Von dieser Praxis sollte so weit wie möglich abgeraten werden, und

Ressourcen sollten stattdessen definiert werden. `main.tf` Wenn eine Sammlung von Ressourcen (z. B. IAM-Rollen und -Richtlinien) jedoch mehr als 150 Zeilen umfasst, ist es sinnvoll, sie in eigene Dateien aufzuteilen, wie z. `iam.tf` Andernfalls sollte der gesamte Ressourcencode in der `main.tf` definiert werden.

- Benutzerdefinierte Skripte: Verwenden Sie Skripten nur bei Bedarf. Terraform berücksichtigt oder verwaltet den Status von Ressourcen, die mithilfe von Skripten erstellt werden, nicht. Verwenden Sie benutzerdefinierte Skripts nur, wenn Terraform-Ressourcen das gewünschte Verhalten nicht unterstützen. Platzieren Sie von Terraform aufgerufene benutzerdefinierte Skripte in einem Verzeichnis. `scripts/`
- Hilfsskripte: Organisieren Sie Hilfsskripte, die nicht von Terraform aufgerufen werden, in einem Verzeichnis. `helpers/` Dokumentieren Sie Hilfsskripte in der `README.md` Datei mit Erklärungen und Beispielaufrufen. Wenn Hilfsskripten Argumente akzeptieren, bieten sie eine Argumentüberprüfung und `--help` Ausgabe an.
- Statische Dateien: Statische Dateien, auf die Terraform verweist, die aber nicht ausgeführt werden (z. B. auf EC2-Instances geladene Startskripte), müssen in einem Verzeichnis organisiert werden. `files/` Platzieren Sie umfangreiche Dokumente getrennt von ihrer HCL in externen Dateien. Verweisen Sie mit der Funktion file () auf sie.
- Vorlagen: Verwenden Sie für Dateien, die die Terraform-Funktion templatefile einliest, die Dateierweiterung. `.tftpl` Vorlagen müssen in einem Verzeichnis abgelegt werden. `templates/`

## Struktur des Stammmoduls

Terraform läuft immer im Kontext eines einzelnen Root-Moduls. Eine vollständige Terraform-Konfiguration besteht aus einem Root-Modul und dem Baum der untergeordneten Module (einschließlich der Module, die vom Root-Modul aufgerufen werden, aller Module, die von diesen Modulen aufgerufen werden, usw.).

Grundlegendes Beispiel für das Layout des Terraform-Root-Moduls:

```
.
### 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
```

## Wiederverwendbare Modulstruktur

Wiederverwendbare Module folgen den gleichen Konzepten wie Root-Module. Um ein Modul zu definieren, erstellen Sie ein neues Verzeichnis dafür und platzieren Sie die `.tf` Dateien darin, genau wie Sie ein Root-Modul definieren würden. Terraform kann Module entweder aus lokalen relativen Pfaden oder aus Remote-Repositorys laden. Wenn Sie erwarten, dass ein Modul von vielen Konfigurationen wiederverwendet wird, platzieren Sie es in einem eigenen Versionskontroll-Repository. Es ist wichtig, den Modulbaum relativ flach zu halten, um die Wiederverwendung der Module in verschiedenen Kombinationen zu erleichtern.

Grundlegendes Beispiel für das Layout eines wiederverwendbaren Terraform-Moduls:

```
.
### 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
```

# Struktur für Modularität

Im Prinzip können Sie beliebige Ressourcen und andere Konstrukte zu einem Modul kombinieren. Wenn Sie jedoch zu viele verschachtelte und wiederverwendbare Module verwenden, kann dies das Verständnis und die Wartung Ihrer gesamten Terraform-Konfiguration erschweren. Verwenden Sie diese Module daher in Maßen.

Wenn es sinnvoll ist, teilen Sie Ihre Konfiguration in wiederverwendbare Module auf, die die Abstraktionsebene erhöhen, indem Sie ein neues Konzept in Ihrer Architektur beschreiben, das aus Ressourcentypen aufgebaut ist.

Wenn Sie Ihre Infrastruktur in wiederverwendbare Definitionen modularisieren, sollten Sie logische Ressourcensätze anstelle von einzelnen Komponenten oder übermäßig komplexen Sammlungen anstreben.

## Verpacken Sie keine einzelnen Ressourcen

Sie sollten keine Module erstellen, die nur eine dünne Hülle für andere einzelne Ressourcentypen bilden. Wenn Sie Probleme haben, einen Namen für Ihr Modul zu finden, der sich vom Namen des darin enthaltenen Hauptressourcentyps unterscheidet, erstellt Ihr Modul wahrscheinlich keine neue Abstraktion — es fügt unnötige Komplexität hinzu. Verwenden Sie stattdessen den Ressourcentyp direkt im aufrufenden Modul.

## Kapseln Sie logische Beziehungen

Gruppieren Sie Gruppen verwandter Ressourcen wie Netzwerkgrundlagen, Datenebenen, Sicherheitskontrollen und Anwendungen. Ein wiederverwendbares Modul sollte Infrastrukturkomponenten kapseln, die zusammenarbeiten, um eine Funktion zu aktivieren.

## Halten Sie die Erbschaft flach

Wenn Sie Module in Unterverzeichnissen verschachteln, vermeiden Sie es, mehr als ein oder zwei Ebenen tief zu gehen. Tief verschachtelte Vererbungsstrukturen erschweren Konfigurationen und Problembehebungen. Module sollten auf anderen Modulen aufbauen und keine Tunnel durch sie bauen.

Durch die Fokussierung der Module auf logische Ressourcengruppierungen, die Architekturmuster darstellen, können Teams schnell zuverlässige Infrastrukturgrundlagen konfigurieren. Ausgewogene Abstraktion ohne übertriebene Technik oder zu starke Vereinfachung.

## Referenzressourcen in den Ergebnissen

Fügen Sie für jede Ressource, die in einem wiederverwendbaren Modul definiert ist, mindestens eine Ausgabe hinzu, die auf die Ressource verweist. Mit Variablen und Ausgaben können Sie Abhängigkeiten zwischen Modulen und Ressourcen ableiten. Ohne Ausgaben können Benutzer Ihr Modul nicht richtig in Bezug auf ihre Terraform-Konfigurationen bestellen.

Gut strukturierte Module, die eine konsistente Umgebung, zweckorientierte Gruppierungen und exportierte Ressourcenreferenzen bieten, ermöglichen eine unternehmensweite Terraform-Zusammenarbeit in großem Maßstab. Teams können die Infrastruktur aus wiederverwendbaren Bausteinen zusammenstellen.

## Konfigurieren Sie keine Anbieter

Obwohl gemeinsam genutzte Module Anbieter von aufrufenden Modulen erben, sollten Module die Anbietereinstellungen nicht selbst konfigurieren. Vermeiden Sie die Angabe von Provider-Konfigurationsblöcken in Modulen. Diese Konfiguration sollte nur einmal global deklariert werden.

## Deklarieren Sie die erforderlichen Anbieter

Obwohl die Anbieterkonfigurationen von allen Modulen gemeinsam genutzt werden, müssen gemeinsam genutzte Module auch ihre eigenen Anbieteranforderungen deklarieren. Diese Vorgehensweise ermöglicht es Terraform, sicherzustellen, dass es eine einzige Version des Anbieters gibt, die mit allen Modulen in der Konfiguration kompatibel ist, und die Quelladresse anzugeben, die als globale (modulunabhängige) Kennung für den Anbieter dient. Die modulspezifischen Anbieteranforderungen spezifizieren jedoch keine der Konfigurationseinstellungen, die bestimmen, auf welche Remote-Endpunkte der Anbieter zugreift, z. B. auf. AWS-Region

Durch die Deklaration von Versionsanforderungen und die Vermeidung einer fest codierten Anbieterkonfiguration bieten Module Portabilität und Wiederverwendbarkeit in allen Terraform-Konfigurationen mithilfe gemeinsam genutzter Anbieter.

Definieren Sie für gemeinsam genutzte Module die mindestens erforderlichen Anbieterversionen in einem required_providers-Block in. `versions.tf`

Um zu deklarieren, dass ein Modul eine bestimmte Version des AWS Anbieters benötigt, verwenden Sie einen `required_providers` Block innerhalb eines `terraform` Blocks:

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

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

Wenn ein gemeinsam genutztes Modul nur eine bestimmte Version des AWS Anbieters unterstützt, verwenden Sie den pessimistischen Einschränkungsoperator (~>), mit dem nur die Versionskomponente ganz rechts inkrementiert werden kann:

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

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

`~> 4.0`Erlaubt in diesem Beispiel die Installation von und, aber nicht. `4.57.1` `4.67.0` `5.0.0` Weitere Informationen finden Sie in der HashiCorp Dokumentation unter Syntax von Versionseinschränkungen.

# Beachten Sie die Namenskonventionen

Klare, aussagekräftige Namen erleichtern Ihnen das Verständnis der Beziehungen zwischen Ressourcen im Modul und des Zwecks von Konfigurationswerten. Die Einhaltung der Stilrichtlinien verbessert die Lesbarkeit sowohl für Benutzer als auch für Entwickler des Moduls.

## Folgen Sie den Richtlinien für die Benennung von Ressourcen

- Verwenden Sie snake_case (wobei Begriffe in Kleinbuchstaben durch Unterstriche getrennt werden) für alle Ressourcennamen, um den Standards im Terraform-Stil zu entsprechen. Diese Vorgehensweise gewährleistet die Konsistenz mit der Namenskonvention für Ressourcentypen, Datenquellentypen und andere vordefinierte Werte. Diese Konvention gilt nicht für Namensargumente.
- Um Verweise auf eine Ressource zu vereinfachen, die die einzige Ressource ihres Typs ist (z. B. ein einzelner Load Balancer für ein ganzes Modul), geben Sie der Ressource einen Namen `main` oder der Übersichtlichkeit `this` halber.
- Verwenden Sie aussagekräftige Namen, die den Zweck und den Kontext der Ressource beschreiben und dabei helfen, zwischen ähnlichen Ressourcen zu unterscheiden (z. B. `primary` für die Hauptdatenbank und `read_replica` für eine Read Replica der Datenbank).
- Verwenden Sie Namen im Singular, nicht im Plural.
- Wiederholen Sie den Ressourcentyp nicht im Ressourcennamen.

## Folgen Sie den Richtlinien für die Benennung von Variablen

- Fügen Sie den Namen von Eingaben, lokalen Variablen und Ausgaben Einheiten hinzu, die numerische Werte wie Festplattengröße oder RAM-Größe darstellen (z. B. `ram_size_gb` für die RAM-Größe in Gigabyte). Diese Vorgehensweise macht die zu erwartende Eingabeeinheit für Konfigurationsbetreuer deutlich.
- Verwenden Sie binäre Einheiten wie MiB und GiB für Speichergrößen und Dezimaleinheiten wie MB oder GB für andere Metriken.
- Geben Sie booleschen Variablen positive Namen wie. `enable_external_access`

# Verwenden Sie Ressourcen für Dateianhänge

In einigen Ressourcen sind Pseudoressourcen als Attribute eingebettet. Wenn möglich, sollten Sie die Verwendung dieser eingebetteten Ressourcenattribute vermeiden und stattdessen die eindeutige Ressource verwenden, um diese Pseudoressource anzuhängen. Diese Ressourcenbeziehungen können zu cause-and-effect Problemen führen, die für jede Ressource einzigartig sind.

Verwenden eines eingebetteten Attributs (vermeiden Sie dieses Muster):

```
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"]
  }
}
```

Verwenden von Ressourcen für Anlagen (bevorzugt):

```
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
}
```

## Verwenden Sie Standard-Tags

Weisen Sie allen Ressourcen, die Tags akzeptieren können, Tags zu. Der Terraform AWS Provider verfügt über eine aws_default_tags-Datenquelle, die Sie im Root-Modul verwenden sollten.

Erwägen Sie, allen Ressourcen, die von einem Terraform-Modul erstellt wurden, die erforderlichen Tags hinzuzufügen. Hier ist eine Liste möglicher Tags, die angehängt werden können:

- Name: Für Menschen lesbarer Ressourcenname
- AppId: Die ID für die Anwendung, die die Ressource verwendet
- AppRole: Die technische Funktion der Ressource, zum Beispiel „Webserver“ oder „Datenbank“
- AppPurpose: Der Geschäftszweck der Ressource, zum Beispiel „Frontend-UI“ oder „Zahlungsabwickler“
- Umgebung: Die Softwareumgebung, z. B. Entwicklung, Test oder Produktion
- Projekt: Die Projekte, die die Ressource verwenden
- CostCenter: Wem muss der Ressourcenverbrauch in Rechnung gestellt werden

## Erfüllen Sie die Terraform-Registrierungsanforderungen

Ein Modul-Repository muss alle folgenden Anforderungen erfüllen, damit es in einer Terraform-Registry veröffentlicht werden kann.

Sie sollten diese Anforderungen immer einhalten, auch wenn Sie nicht vorhaben, das Modul kurzfristig in einer Registry zu veröffentlichen. Auf diese Weise können Sie das Modul später in einer Registrierung veröffentlichen, ohne die Konfiguration und Struktur des Repositorys ändern zu müssen.

- Repository-Name: Verwenden Sie für ein Modul-Repository den dreiteiligen Namen`terraform-aws-<NAME>`, der die Art der Infrastruktur `<NAME>` widerspiegelt, die das Modul verwaltet. Das `<NAME>` Segment kann zusätzliche Bindestriche enthalten (z. B.). `terraform-aws-iam-terraform-roles`

- Standardmodulstruktur: Das Modul muss der Standard-Repository-Struktur entsprechen. Auf diese Weise kann die Registry Ihr Modul überprüfen und Dokumentation erstellen, die Ressourcennutzung verfolgen und vieles mehr.
  - Nachdem Sie das Git-Repository erstellt haben, kopieren Sie die Moduldateien in das Stammverzeichnis des Repositorys. Wir empfehlen, dass Sie jedes Modul, das wiederverwendet werden soll, im Stammverzeichnis eines eigenen Repositorys ablegen. Sie können aber auch Module aus Unterverzeichnissen referenzieren.
  - Wenn Sie HCP Terraform verwenden, veröffentlichen Sie die Module, die gemeinsam genutzt werden sollen, in Ihrer Organisationsregistrierung. Die Registrierung verarbeitet Downloads und steuert den Zugriff mit HCP Terraform API-Token, sodass Verbraucher keinen Zugriff auf das Quell-Repository des Moduls benötigen, selbst wenn sie Terraform über die Befehlszeile ausführen.
- Standort und Berechtigungen: Das Repository muss sich bei einem Ihrer konfigurierten Anbieter für Versionskontrollsysteme (VCS) befinden, und das HCP Terraform VCS-Benutzerkonto muss Administratorzugriff auf das Repository haben. Die Registrierung benötigt Administratorzugriff, um die Webhooks zum Importieren neuer Modulversionen zu erstellen.
- x.y.z-Tags für Releases: Es muss mindestens ein Release-Tag vorhanden sein, damit Sie ein Modul veröffentlichen können. Die Registrierung verwendet Release-Tags, um Modulversionen zu identifizieren. Namen von Release-Tags müssen eine semantische Versionierung verwenden, der Sie optional ein Präfix `v` (z. B. `v1.1.0` und`1.1.0`) voranstellen können. Die Registrierung ignoriert Tags, die nicht wie Versionsnummern aussehen. Weitere Informationen zum Veröffentlichen von Modulen finden Sie in der Terraform-Dokumentation.

Weitere Informationen finden Sie unter Vorbereiten eines Modul-Repositorys in der Terraform-Dokumentation.

## Verwenden Sie die empfohlenen Modulquellen

Terraform verwendet das `source` Argument in einem Modulblock, um den Quellcode für ein untergeordnetes Modul zu finden und herunterzuladen.

Wir empfehlen, lokale Pfade für eng verwandte Module zu verwenden, die den Hauptzweck haben, wiederholte Codeelemente auszuschließen, und eine native Terraform-Modulregistrierung oder einen VCS-Anbieter für Module zu verwenden, die von mehreren Konfigurationen gemeinsam genutzt werden sollen.

Die folgenden Beispiele veranschaulichen die gängigsten und empfohlenen Quelltypen für die gemeinsame Nutzung von Modulen. Registrierungsmodule unterstützen die Versionierung. Sie sollten immer eine bestimmte Version angeben, wie in den folgenden Beispielen gezeigt.

## Registrierung

Terraform-Registrierung:

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

  ...

}
```

Indem Sie Commit-Hashes anheften, können Sie verhindern, dass öffentliche Register verlassen, die anfällig für Angriffe auf die Lieferkette sind.

HCP Terraform:

```
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
}
```

# VCS-Anbieter

VCS-Anbieter unterstützen das `ref` Argument für die Auswahl einer bestimmten Revision, wie in den folgenden Beispielen gezeigt.

GitHub (HTTPS):

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

  ...

  enable_karpenter = true
}
```

Generisches Git-Repository (HTTPS):

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

  ...

  enable_karpenter = true
}
```

Generisches Git-Repository (SSH):

> **Warning**
>
> Sie müssen Anmeldeinformationen konfigurieren, um auf private Repositorys zugreifen zu können.

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

  ...

  enable_karpenter = true
}
```

# Befolgen Sie die Codierungsstandards

Wenden Sie konsistente Terraform-Formatierungsregeln und -Stile für alle Konfigurationsdateien an. Setzen Sie Standards durch, indem Sie automatisierte Stilprüfungen in CI/CD-Pipelines verwenden. Wenn Sie bewährte Programmierpraktiken in Team-Workflows integrieren, bleiben die Konfigurationen lesbar, wartbar und kollaborativ, da sich die Nutzung im gesamten Unternehmen verbreitet.

## Halten Sie sich an die Stilrichtlinien

- Formatieren Sie alle Terraform-Dateien (`.tf`Dateien) mit dem Befehl terraform fmt so, dass sie den Stilstandards entsprechen. HashiCorp
- Verwenden Sie den Befehl terraform validate, um die Syntax und Struktur Ihrer Konfiguration zu überprüfen.
- Analysieren Sie die Codequalität statisch mithilfe von TFLint. Dieser Linter sucht nach bewährten Methoden von Terraform, die über die bloße Formatierung hinausgehen, und schlägt beim Erstellen fehl, wenn Fehler auftreten.

## Konfigurieren Sie Pre-Commit-Hooks

Konfigurieren Sie clientseitige Pre-Commit-Hooks, die`terraform fmt`, `tflint`check`ov`, und andere Codescans und Stilprüfungen ausführen, bevor Sie Commits zulassen. Diese Vorgehensweise hilft Ihnen, die Einhaltung von Standards zu einem früheren Zeitpunkt in Entwickler-Workflows zu überprüfen.

Verwenden Sie Pre-Commit-Frameworks wie Pre-Commit, um Terraform-Linting, Formatierung und Code-Scanning als Hooks auf Ihrem lokalen Computer hinzuzufügen. Hooks werden bei jedem Git-Commit ausgeführt und schlagen beim Commit fehl, wenn die Prüfungen nicht bestanden werden.

Durch die Verlagerung von Stil- und Qualitätsprüfungen auf lokale Pre-Commit-Hooks erhalten Entwickler schnelles Feedback, bevor Änderungen eingeführt werden. Standards werden Teil des Codierungs-Workflows.

# Bewährte Methoden für die AWS Provider-Versionsverwaltung

Die sorgfältige Verwaltung der Versionen des AWS Providers und der zugehörigen Terraform-Module ist für die Stabilität von entscheidender Bedeutung. In diesem Abschnitt werden bewährte Methoden im Zusammenhang mit Versionseinschränkungen und Upgrades beschrieben.

Bewährte Methoden:

- Fügen Sie automatisierte Versionsprüfungen hinzu
- Überwachen Sie neue Versionen
- Tragen Sie zu Anbietern bei

## Fügen Sie automatisierte Versionsprüfungen hinzu

Fügen Sie Versionsprüfungen für Terraform-Anbieter in Ihre CI/CD-Pipelines ein, um das Versions-Pinning zu überprüfen, und schlagen Sie Builds fehl, wenn die Version undefiniert ist.

- Fügen Sie TFLint-Prüfungen in CI/CD-Pipelines hinzu, um nach Anbieterversionen zu suchen, für die keine festgelegten Einschränkungen für Haupt-/Nebenversionen definiert sind. Verwenden Sie das TFLint-Regelsatz-Plugin für Terraform AWS Provider, das Regeln zur Erkennung möglicher Fehler bereitstellt und nach bewährten Methoden für Ressourcen sucht. AWS
- Fehlerhafte CI-Läufe, die nicht angeheftete Anbieterversionen erkennen, um zu verhindern, dass implizite Upgrades in die Produktionsumgebung gelangen.

## Überwachen Sie neue Versionen

- Überwachen Sie die Versionshinweise und Changelog-Feeds der Anbieter. Erhalten Sie Benachrichtigungen über neue Haupt-/Nebenversionen.
- Untersuchen Sie die Versionshinweise auf potenziell wichtige Änderungen und bewerten Sie deren Auswirkungen auf Ihre bestehende Infrastruktur.
- Führen Sie zunächst ein Upgrade von Nebenversionen in Umgebungen außerhalb der Produktionsumgebung durch, um sie zu validieren, bevor Sie die Produktionsumgebung aktualisieren.

Durch die Automatisierung von Versionsprüfungen in Pipelines und die Überwachung neuer Versionen können Sie nicht unterstützte Upgrades frühzeitig catch und Ihren Teams Zeit geben, die Auswirkungen neuer Haupt-/Nebenversionen zu bewerten, bevor Sie Produktionsumgebungen aktualisieren.

## Tragen Sie zu Anbietern bei

Tragen Sie aktiv zu HashiCorp AWS Provider bei, indem Sie Fehler melden oder Funktionen bei GitHub Problemen anfordern:

- Öffne gut dokumentierte Probleme im AWS Provider-Repository, um alle Fehler, auf die du gestoßen bist, oder fehlende Funktionen detailliert zu beschreiben. Stellen Sie reproduzierbare Schritte bereit.
- Fordern Sie Verbesserungen an und stimmen Sie darüber ab, um die Funktionen des AWS Anbieters für die Verwaltung neuer Dienste zu erweitern.
- Verweisen Sie auf ausgegebene Pull-Requests, wenn Sie Vorschläge zur Behebung von Fehlern oder Verbesserungen des Anbieters einreichen. Link zu verwandten Problemen.
- Folgen Sie den Richtlinien für Beiträge im Repositorium für Kodierungskonventionen, Teststandards und Dokumentation.

Indem Sie den Anbietern, die Sie verwenden, etwas zurückgeben, können Sie direkt zu deren Roadmap beitragen und dazu beitragen, deren Qualität und Funktionen für alle Benutzer zu verbessern.

# Bewährte Methoden für Community-Module

Der effektive Einsatz von Modulen ist der Schlüssel zur Verwaltung komplexer Terraform-Konfigurationen und zur Förderung der Wiederverwendung. Dieser Abschnitt enthält bewährte Methoden zu Community-Modulen, Abhängigkeiten, Quellen, Abstraktionen und Beiträgen.

Bewährte Verfahren:

- Entdecken Sie Community-Module
- Verstehen Sie Abhängigkeiten
- Verwenden Sie vertrauenswürdige Quellen
- Tragen Sie zu Community-Modulen bei

## Entdecken Sie Community-Module

Durchsuchen Sie die Terraform Registry und andere Quellen nach vorhandenen AWS Modulen GitHub, die Ihren Anwendungsfall lösen könnten, bevor Sie ein neues Modul erstellen. Suchen Sie nach beliebten Optionen, die kürzlich aktualisiert wurden und aktiv gewartet werden.

### Verwenden Sie Variablen zur Anpassung

Wenn Sie Community-Module verwenden, geben Sie Eingaben über Variablen weiter, anstatt den Quellcode zu forken oder direkt zu ändern. Überschreiben Sie bei Bedarf die Standardeinstellungen, anstatt die internen Funktionen des Moduls zu ändern.

Das Forking sollte sich darauf beschränken, Korrekturen oder Funktionen zum ursprünglichen Modul beizutragen, um der breiteren Community zu helfen.

## Verstehen Sie Abhängigkeiten

Bevor Sie das Modul verwenden, lesen Sie den Quellcode und die Dokumentation, um Abhängigkeiten zu identifizieren:

- Erforderliche Anbieter: Notieren Sie sich die Versionen von AWS Kubernetes oder anderen Anbietern, die das Modul benötigt.
- Verschachtelte Module: Suchen Sie nach anderen intern verwendeten Modulen, die kaskadierende Abhängigkeiten einführen.

- Externe Datenquellen: Notieren Sie sich die APIs, benutzerdefinierten Plugins oder Infrastrukturabhängigkeiten, auf die das Modul angewiesen ist.

Indem Sie den vollständigen Baum der direkten und indirekten Abhängigkeiten abbilden, können Sie Überraschungen bei der Verwendung des Moduls vermeiden.

# Verwenden Sie vertrauenswürdige Quellen

Die Beschaffung von Terraform-Modulen von nicht verifizierten oder unbekannten Herausgebern birgt ein erhebliches Risiko. Verwenden Sie Module nur aus vertrauenswürdigen Quellen.

- Bevorzugen Sie zertifizierte Module aus der Terraform Registry, die von verifizierten Entwicklern wie AWS unseren Partnern veröffentlicht wurden. HashiCorp
- Überprüfen Sie bei benutzerdefinierten Modulen den Verlauf des Herausgebers, die Support-Stufen und den Ruf der Nutzung, auch wenn das Modul von Ihrer eigenen Organisation stammt.

Indem Sie Module aus unbekannten oder ungeprüften Quellen nicht zulassen, können Sie das Risiko verringern, dass Sicherheitslücken oder Wartungsprobleme in Ihren Code eingebracht werden.

## Abonnieren von -Benachrichtigungen

Abonnieren Sie Benachrichtigungen über neue Modulversionen von vertrauenswürdigen Herausgebern:

- Schauen Sie sich die GitHub Modul-Repositorys an, um Benachrichtigungen über neue Versionen des Moduls zu erhalten.
- Überwachen Sie die Blogs und Changelogs der Herausgeber auf Aktualisierungen.
- Lassen Sie sich proaktiv über neue Versionen von verifizierten, hoch bewerteten Quellen benachrichtigen, anstatt implizit Updates abzurufen.

Die ausschließliche Nutzung von Modulen aus vertrauenswürdigen Quellen und die Überwachung von Änderungen sorgen für Stabilität und Sicherheit. Geprüfte Module steigern die Produktivität und minimieren gleichzeitig das Risiko in der Lieferkette.

# Tragen Sie zu Community-Modulen bei

Reichen Sie Korrekturen und Verbesserungen für Community-Module ein, die gehostet werden in GitHub:

- Öffnen Sie Pull-Requests für Module, um Fehler oder Einschränkungen zu beheben, auf die Sie bei Ihrer Nutzung stoßen.
- Fordern Sie an, dass neue Best-Practice-Konfigurationen zu bestehenden OSS-Modulen hinzugefügt werden, indem Sie Probleme erstellen.

Durch Beiträge zu Community-Modulen werden wiederverwendbare, kodifizierte Muster für alle Terraform-Praktiker verbessert.

# Häufig gestellte Fragen

F: Warum sollten Sie sich auf den Anbieter konzentrieren? AWS

Antwort: Der AWS Anbieter ist einer der am häufigsten verwendeten und komplexesten Anbieter für die Bereitstellung von Infrastruktur in Terraform. Die Einhaltung dieser bewährten Methoden hilft Benutzern dabei, ihre Nutzung des Anbieters für die jeweilige Umgebung zu optimieren. AWS

F: Ich bin neu bei Terraform. Kann ich diesen Leitfaden verwenden?

A. Der Leitfaden richtet sich sowohl an Personen, die neu bei Terraform sind, als auch an fortgeschrittene Praktiker, die ihre Fähigkeiten verbessern möchten. Die Praktiken verbessern die Arbeitsabläufe für Benutzer in jeder Lernphase.

F: Welche wichtigen bewährten Methoden werden behandelt?

Antwort: Zu den wichtigsten bewährten Methoden gehören die Verwendung von IAM-Rollen anstelle von Zugriffsschlüsseln, das Fixieren von Versionen, die Integration automatisierter Tests, das Sperren von Fernzuständen, die Rotation von Anmeldeinformationen, die Bereitstellung von Beiträgen an Anbieter und die logische Organisation von Codebasen.

F: Wo kann ich mehr über Terraform erfahren?

Antwort: Der Bereich Ressourcen enthält Links zur offiziellen HashiCorp Terraform-Dokumentation und zu den Community-Foren. Verwenden Sie die Links, um mehr über erweiterte Terraform-Workflows zu erfahren.

# Nächste Schritte

Hier sind einige mögliche nächste Schritte nach dem Lesen dieses Handbuchs:

- Wenn Sie bereits über eine Terraform-Codebasis verfügen, überprüfen Sie Ihre Konfiguration und identifizieren Sie Bereiche, die anhand der Empfehlungen in diesem Handbuch verbessert werden könnten. Informieren Sie sich beispielsweise über bewährte Methoden für die Implementierung von Remote-Backends, die Aufteilung von Code in Module, die Verwendung von Versions-Pinning usw. und validieren Sie diese in Ihrer Konfiguration.
- Wenn Sie noch nicht über eine bestehende Terraform-Codebasis verfügen, verwenden Sie diese bewährten Methoden, wenn Sie Ihre neue Konfiguration strukturieren. Befolgen Sie von Anfang an die Hinweise zur Zustandsverwaltung, Authentifizierung, Codestruktur usw.
- Versuchen Sie, einige der HashiCorp Community-Module, auf die in diesem Handbuch verwiesen wird, zu verwenden, um festzustellen, ob sie Ihre Architekturmuster vereinfachen. Die Module ermöglichen höhere Abstraktionsebenen, sodass Sie gemeinsame Ressourcen nicht neu schreiben müssen.
- Aktivieren Sie Linting, Sicherheitsscans, Richtlinienprüfungen und automatisierte Testtools, um einige der bewährten Methoden in den Bereichen Sicherheit, Compliance und Codequalität zu stärken. Tools wie TFlint, tfsec und Checkov können helfen.
- Lesen Sie in der neuesten AWS Provider-Dokumentation nach, ob es neue Ressourcen oder Funktionen gibt, die Ihnen helfen könnten, Ihre Terraform-Nutzung zu optimieren. Bleiben Sie über neue Versionen des Anbieters auf dem AWS Laufenden.
- Weitere Hinweise finden Sie in der Terraform-Dokumentation, im Best-Practice-Leitfaden und im Styleguide auf der HashiCorp Website.

# Ressourcen

## Referenzen

Die folgenden Links bieten zusätzliches Lesematerial für den Terraform AWS Provider und die Verwendung von Terraform for IaC auf. AWS

- Terraform Provider (Dokumentation) AWS HashiCorp
- Terraform-Module für AWS Dienste (Terraform Registry)
- Die AWS und die HashiCorp Partnerschaft (Blogbeitrag) HashiCorp
- Dynamische Anmeldeinformationen beim AWS Anbieter (HCP Terraform-Dokumentation)
- DynamoDB-Statussperre (Terraform-Dokumentation)
- Richtlinien mit Sentinel durchsetzen (Terraform-Dokumentation)

## Tools

Die folgenden Tools tragen dazu bei, die Codequalität und die Automatisierung von Terraform-Konfigurationen zu verbessern AWS, wie in diesem Best-Practice-Leitfaden empfohlen.

Codequalität:

- Checkov: Scannt Terraform-Code, um Fehlkonfigurationen vor der Bereitstellung zu identifizieren.
- TFLint: Identifiziert mögliche Fehler, veraltete Syntax und unbenutzte Deklarationen. Dieser Linter kann auch AWS bewährte Verfahren und Namenskonventionen durchsetzen.
- terraform-docs: Generiert Dokumentation aus Terraform-Modulen in verschiedenen Ausgabeformaten.

Automatisierungstools:

- HCP Terraform: Unterstützt Teams bei der Versionierung, Zusammenarbeit und Erstellung von Terraform-Workflows mit Richtlinienprüfungen und Genehmigungen.
- Atlantis: Ein Open-Source-Tool zur Automatisierung von Terraform-Pull-Requests zur Validierung von Codeänderungen.

- CDK for Terraform: Ein Framework, mit dem Sie vertraute Sprachen wie Python TypeScript, Java, C# und Go anstelle von HashiCorp Configuration Language (HCL) verwenden können, um Ihre Terraform-Infrastruktur als Code zu definieren, bereitzustellen und zu testen.

# Dokumentverlauf

In der folgenden Tabelle werden wichtige Änderungen in diesem Leitfaden beschrieben. Um Benachrichtigungen über zukünftige Aktualisierungen zu erhalten, können Sie einen RSS-Feed abonnieren.

| Änderung | Beschreibung | Datum |
| --- | --- | --- |
| Erste Veröffentlichung | — | 28. Mai 2024 |

# AWS Glossar zu präskriptiven Leitlinien

Die folgenden Begriffe werden häufig in Strategien, Leitfäden und Mustern von AWS Prescriptive Guidance verwendet. Um Einträge vorzuschlagen, verwenden Sie bitte den Link Feedback geben am Ende des Glossars.

## Zahlen

7 Rs

Sieben gängige Migrationsstrategien für die Verlagerung von Anwendungen in die Cloud. Diese Strategien bauen auf den 5 Rs auf, die Gartner 2011 identifiziert hat, und bestehen aus folgenden Elementen:

- Faktorwechsel/Architekturwechsel – Verschieben Sie eine Anwendung und ändern Sie ihre Architektur, indem Sie alle Vorteile cloudnativer Feature nutzen, um Agilität, Leistung und Skalierbarkeit zu verbessern. Dies beinhaltet in der Regel die Portierung des Betriebssystems und der Datenbank. Beispiel: Migrieren Sie Ihre lokale Oracle-Datenbank auf die Amazon Aurora PostgreSQL-kompatible Edition.
- Plattformwechsel (Lift and Reshape) – Verschieben Sie eine Anwendung in die Cloud und führen Sie ein gewisses Maß an Optimierung ein, um die Cloud-Funktionen zu nutzen. Beispiel: Migrieren Sie Ihre lokale Oracle-Datenbank zu Amazon Relational Database Service (Amazon RDS) für Oracle in der. AWS Cloud
- Neukauf (Drop and Shop) – Wechseln Sie zu einem anderen Produkt, indem Sie typischerweise von einer herkömmlichen Lizenz zu einem SaaS-Modell wechseln. Beispiel: Migrieren Sie Ihr CRM-System (Customer Relationship Management) zu Salesforce.com.
- Hostwechsel (Lift and Shift) – Verschieben Sie eine Anwendung in die Cloud, ohne Änderungen vorzunehmen, um die Cloud-Funktionen zu nutzen. Beispiel: Migrieren Sie Ihre lokale Oracle-Datenbank zu Oracle auf einer EC2 Instanz in der. AWS Cloud
- Verschieben (Lift and Shift auf Hypervisor-Ebene) – Verlagern Sie die Infrastruktur in die Cloud, ohne neue Hardware kaufen, Anwendungen umschreiben oder Ihre bestehenden Abläufe ändern zu müssen. Sie migrieren Server von einer lokalen Plattform zu einem Cloud-Dienst für dieselbe Plattform. Beispiel: Migrieren Sie eine Microsoft Hyper-V Anwendung zu AWS.
- Beibehaltung (Wiederaufgreifen) – Bewahren Sie Anwendungen in Ihrer Quellumgebung auf. Dazu können Anwendungen gehören, die einen umfangreichen Faktorwechsel erfordern und

die Sie auf einen späteren Zeitpunkt verschieben möchten, sowie ältere Anwendungen, die Sie beibehalten möchten, da es keine geschäftliche Rechtfertigung für ihre Migration gibt.

- Außerbetriebnahme – Dekommissionierung oder Entfernung von Anwendungen, die in Ihrer Quellumgebung nicht mehr benötigt werden.

# A

ABAC

Siehe attributbasierte Zugriffskontrolle.

abstrahierte Dienste

Weitere Informationen finden Sie unter Managed Services.

ACID

Siehe Atomarität, Konsistenz, Isolierung und Haltbarkeit.

Aktiv-Aktiv-Migration

Eine Datenbankmigrationsmethode, bei der die Quell- und Zieldatenbanken synchron gehalten werden (mithilfe eines bidirektionalen Replikationstools oder dualer Schreibvorgänge) und beide Datenbanken Transaktionen von miteinander verbundenen Anwendungen während der Migration verarbeiten. Diese Methode unterstützt die Migration in kleinen, kontrollierten Batches, anstatt einen einmaligen Cutover zu erfordern. Es ist flexibler, erfordert aber mehr Arbeit als eine aktiv-passive Migration.

Aktiv-Passiv-Migration

Eine Datenbankmigrationsmethode, bei der die Quell- und Zieldatenbanken synchron gehalten werden, aber nur die Quelldatenbank Transaktionen von verbindenden Anwendungen verarbeitet, während Daten in die Zieldatenbank repliziert werden. Die Zieldatenbank akzeptiert während der Migration keine Transaktionen.

Aggregatfunktion

Eine SQL-Funktion, die mit einer Gruppe von Zeilen arbeitet und einen einzelnen Rückgabewert für die Gruppe berechnet. Beispiele für Aggregatfunktionen sind `SUM` und`MAX`.

AI

Siehe künstliche Intelligenz.

AIOps

Siehe Operationen im Bereich künstliche Intelligenz.

Anonymisierung

Der Prozess des dauerhaften Löschens personenbezogener Daten in einem Datensatz. Anonymisierung kann zum Schutz der Privatsphäre beitragen. Anonymisierte Daten gelten nicht mehr als personenbezogene Daten.

Anti-Muster

Eine häufig verwendete Lösung für ein wiederkehrendes Problem, bei dem die Lösung kontraproduktiv, ineffektiv oder weniger wirksam als eine Alternative ist.

Anwendungssteuerung

Ein Sicherheitsansatz, bei dem nur zugelassene Anwendungen verwendet werden können, um ein System vor Schadsoftware zu schützen.

Anwendungsportfolio

Eine Sammlung detaillierter Informationen zu jeder Anwendung, die von einer Organisation verwendet wird, einschließlich der Kosten für die Erstellung und Wartung der Anwendung und ihres Geschäftswerts. Diese Informationen sind entscheidend für den Prozess der Portfoliofindung und -analyse und hilft bei der Identifizierung und Priorisierung der Anwendungen, die migriert, modernisiert und optimiert werden sollen.

künstliche Intelligenz (KI)

Das Gebiet der Datenverarbeitungswissenschaft, das sich der Nutzung von Computertechnologien zur Ausführung kognitiver Funktionen widmet, die typischerweise mit Menschen in Verbindung gebracht werden, wie Lernen, Problemlösen und Erkennen von Mustern. Weitere Informationen finden Sie unter Was ist künstliche Intelligenz?

Operationen mit künstlicher Intelligenz (AIOps)

Der Prozess des Einsatzes von Techniken des Machine Learning zur Lösung betrieblicher Probleme, zur Reduzierung betrieblicher Zwischenfälle und menschlicher Eingriffe sowie zur Steigerung der Servicequalität. Weitere Informationen zur Verwendung in der AWS Migrationsstrategie finden Sie im Operations Integration Guide. AIOps

Asymmetrische Verschlüsselung

Ein Verschlüsselungsalgorithmus, der ein Schlüsselpaar, einen öffentlichen Schlüssel für die Verschlüsselung und einen privaten Schlüssel für die Entschlüsselung verwendet. Sie können den

öffentlichen Schlüssel teilen, da er nicht für die Entschlüsselung verwendet wird. Der Zugriff auf den privaten Schlüssel sollte jedoch stark eingeschränkt sein.

Atomizität, Konsistenz, Isolierung, Haltbarkeit (ACID)

Eine Reihe von Softwareeigenschaften, die die Datenvalidität und betriebliche Zuverlässigkeit einer Datenbank auch bei Fehlern, Stromausfällen oder anderen Problemen gewährleisten.

Attributbasierte Zugriffskontrolle (ABAC)

Die Praxis, detaillierte Berechtigungen auf der Grundlage von Benutzerattributen wie Abteilung, Aufgabenrolle und Teamname zu erstellen. Weitere Informationen finden Sie unter ABAC AWS in der AWS Identity and Access Management (IAM-) Dokumentation.

autoritative Datenquelle

Ein Ort, an dem Sie die primäre Version der Daten speichern, die als die zuverlässigste Informationsquelle angesehen wird. Sie können Daten aus der maßgeblichen Datenquelle an andere Speicherorte kopieren, um die Daten zu verarbeiten oder zu ändern, z. B. zu anonymisieren, zu redigieren oder zu pseudonymisieren.

Availability Zone

Ein bestimmter Standort innerhalb einer AWS-Region , der vor Ausfällen in anderen Availability Zones geschützt ist und kostengünstige Netzwerkkonnektivität mit niedriger Latenz zu anderen Availability Zones in derselben Region bietet.

AWS Framework für die Cloud-Einführung (AWS CAF)

Ein Framework mit Richtlinien und bewährten Verfahren, das Unternehmen bei der Entwicklung eines effizienten und effektiven Plans für den erfolgreichen Umstieg auf die Cloud unterstützt. AWS AWS CAF unterteilt die Leitlinien in sechs Schwerpunktbereiche, die als Perspektiven bezeichnet werden: Unternehmen, Mitarbeiter, Unternehmensführung, Plattform, Sicherheit und Betrieb. Die Perspektiven Geschäft, Mitarbeiter und Unternehmensführung konzentrieren sich auf Geschäftskompetenzen und -prozesse, während sich die Perspektiven Plattform, Sicherheit und Betriebsabläufe auf technische Fähigkeiten und Prozesse konzentrieren. Die Personalperspektive zielt beispielsweise auf Stakeholder ab, die sich mit Personalwesen (HR), Personalfunktionen und Personalmanagement befassen. Aus dieser Perspektive bietet AWS CAF Leitlinien für Personalentwicklung, Schulung und Kommunikation, um das Unternehmen auf eine erfolgreiche Cloud-Einführung vorzubereiten. Weitere Informationen finden Sie auf der AWS -CAF-Webseite und dem AWS -CAF-Whitepaper.

AWS Workload-Qualifizierungsrahmen (AWS WQF)

Ein Tool, das Workloads bei der Datenbankmigration bewertet, Migrationsstrategien empfiehlt und Arbeitsschätzungen bereitstellt. AWS WQF ist in () enthalten. AWS Schema Conversion Tool AWS SCT Es analysiert Datenbankschemas und Codeobjekte, Anwendungscode, Abhängigkeiten und Leistungsmerkmale und stellt Bewertungsberichte bereit.

# B

schlechter Bot

Ein Bot, der Einzelpersonen oder Organisationen stören oder ihnen Schaden zufügen soll.

BCP

Siehe Planung der Geschäftskontinuität.

Verhaltensdiagramm

Eine einheitliche, interaktive Ansicht des Ressourcenverhaltens und der Interaktionen im Laufe der Zeit. Sie können ein Verhaltensdiagramm mit Amazon Detective verwenden, um fehlgeschlagene Anmeldeversuche, verdächtige API-Aufrufe und ähnliche Vorgänge zu untersuchen. Weitere Informationen finden Sie unter Daten in einem Verhaltensdiagramm in der Detective-Dokumentation.

Big-Endian-System

Ein System, welches das höchstwertige Byte zuerst speichert. Siehe auch Endianness.

Binäre Klassifikation

Ein Prozess, der ein binäres Ergebnis vorhersagt (eine von zwei möglichen Klassen). Beispielsweise könnte Ihr ML-Modell möglicherweise Probleme wie „Handelt es sich bei dieser E-Mail um Spam oder nicht?“ vorhersagen müssen oder „Ist dieses Produkt ein Buch oder ein Auto?“

Bloom-Filter

Eine probabilistische, speichereffiziente Datenstruktur, mit der getestet wird, ob ein Element Teil einer Menge ist.

Blau/Grün-Bereitstellung

Eine Bereitstellungsstrategie, bei der Sie zwei separate, aber identische Umgebungen erstellen. Sie führen die aktuelle Anwendungsversion in einer Umgebung (blau) und die neue

Anwendungsversion in der anderen Umgebung (grün) aus. Mit dieser Strategie können Sie schnell und mit minimalen Auswirkungen ein Rollback durchführen.

Bot

Eine Softwareanwendung, die automatisierte Aufgaben über das Internet ausführt und menschliche Aktivitäten oder Interaktionen simuliert. Manche Bots sind nützlich oder nützlich, wie z. B. Webcrawler, die Informationen im Internet indexieren. Einige andere Bots, sogenannte bösartige Bots, sollen Einzelpersonen oder Organisationen stören oder ihnen Schaden zufügen.

Botnetz

Netzwerke von Bots, die mit Malware infiziert sind und unter der Kontrolle einer einzigen Partei stehen, die als Bot-Herder oder Bot-Operator bezeichnet wird. Botnetze sind der bekannteste Mechanismus zur Skalierung von Bots und ihrer Wirkung.

branch

Ein containerisierter Bereich eines Code-Repositorys. Der erste Zweig, der in einem Repository erstellt wurde, ist der Hauptzweig. Sie können einen neuen Zweig aus einem vorhandenen Zweig erstellen und dann Feature entwickeln oder Fehler in dem neuen Zweig beheben. Ein Zweig, den Sie erstellen, um ein Feature zu erstellen, wird allgemein als Feature-Zweig bezeichnet. Wenn das Feature zur Veröffentlichung bereit ist, führen Sie den Feature-Zweig wieder mit dem Hauptzweig zusammen. Weitere Informationen finden Sie unter Über Branches (GitHub Dokumentation).

Zugang durch Glasbruch

Unter außergewöhnlichen Umständen und im Rahmen eines genehmigten Verfahrens ist dies eine schnelle Methode für einen Benutzer, auf einen Bereich zuzugreifen AWS-Konto , für den er in der Regel keine Zugriffsrechte besitzt. Weitere Informationen finden Sie unter dem Indikator Implementation break-glass procedures in den AWS Well-Architected-Leitlinien.

Brownfield-Strategie

Die bestehende Infrastruktur in Ihrer Umgebung. Wenn Sie eine Brownfield-Strategie für eine Systemarchitektur anwenden, richten Sie sich bei der Gestaltung der Architektur nach den Einschränkungen der aktuellen Systeme und Infrastruktur. Wenn Sie die bestehende Infrastruktur erweitern, könnten Sie Brownfield- und Greenfield-Strategien mischen.

Puffer-Cache

Der Speicherbereich, in dem die am häufigsten abgerufenen Daten gespeichert werden.

Geschäftsfähigkeit

Was ein Unternehmen tut, um Wert zu generieren (z. B. Vertrieb, Kundenservice oder Marketing). Microservices-Architekturen und Entwicklungsentscheidungen können von den Geschäftskapazitäten beeinflusst werden. Weitere Informationen finden Sie im Abschnitt Organisiert nach Geschäftskapazitäten des Whitepapers Ausführen von containerisierten Microservices in AWS.

Planung der Geschäftskontinuität (BCP)

Ein Plan, der die potenziellen Auswirkungen eines störenden Ereignisses, wie z. B. einer groß angelegten Migration, auf den Betrieb berücksichtigt und es einem Unternehmen ermöglicht, den Betrieb schnell wieder aufzunehmen.

# C

CAF

Weitere Informationen finden Sie unter Framework für die AWS Cloud-Einführung.

Bereitstellung auf Kanaren

Die langsame und schrittweise Veröffentlichung einer Version für Endbenutzer. Wenn Sie sich sicher sind, stellen Sie die neue Version bereit und ersetzen die aktuelle Version vollständig.

CCoE

Weitere Informationen finden Sie im Cloud Center of Excellence.

CDC

Siehe Erfassung von Änderungsdaten.

Erfassung von Datenänderungen (CDC)

Der Prozess der Nachverfolgung von Änderungen an einer Datenquelle, z. B. einer Datenbanktabelle, und der Aufzeichnung von Metadaten zu der Änderung. Sie können CDC für verschiedene Zwecke verwenden, z. B. für die Prüfung oder Replikation von Änderungen in einem Zielsystem, um die Synchronisation aufrechtzuerhalten.

Chaos-Technik

Absichtliches Einführen von Ausfällen oder Störungsereignissen, um die Widerstandsfähigkeit eines Systems zu testen. Sie können AWS Fault Injection Service (AWS FIS) verwenden, um Experimente durchzuführen, die Ihre AWS Workloads stress, und deren Reaktion zu bewerten.

CI/CD

Siehe Continuous Integration und Continuous Delivery.

Klassifizierung

Ein Kategorisierungsprozess, der bei der Erstellung von Vorhersagen hilft. ML-Modelle für Klassifikationsprobleme sagen einen diskreten Wert voraus. Diskrete Werte unterscheiden sich immer voneinander. Beispielsweise muss ein Modell möglicherweise auswerten, ob auf einem Bild ein Auto zu sehen ist oder nicht.

clientseitige Verschlüsselung

Lokale Verschlüsselung von Daten, bevor das Ziel sie AWS-Service empfängt.

Cloud-Exzellenzzentrum (CCoE)

Ein multidisziplinäres Team, das die Cloud-Einführung in der gesamten Organisation vorantreibt, einschließlich der Entwicklung bewährter Cloud-Methoden, der Mobilisierung von Ressourcen, der Festlegung von Migrationszeitplänen und der Begleitung der Organisation durch groß angelegte Transformationen. Weitere Informationen finden Sie in den CCoE-Beiträgen im AWS Cloud Enterprise Strategy Blog.

Cloud Computing

Die Cloud-Technologie, die typischerweise für die Ferndatenspeicherung und das IoT-Gerätemanagement verwendet wird. Cloud Computing ist häufig mit Edge-Computing-Technologie verbunden.

Cloud-Betriebsmodell

In einer IT-Organisation das Betriebsmodell, das zum Aufbau, zur Weiterentwicklung und Optimierung einer oder mehrerer Cloud-Umgebungen verwendet wird. Weitere Informationen finden Sie unter Aufbau Ihres Cloud-Betriebsmodells.

Phasen der Einführung der Cloud

Die vier Phasen, die Unternehmen bei der Migration in der Regel durchlaufen AWS Cloud:

- Projekt – Durchführung einiger Cloud-bezogener Projekte zu Machbarkeitsnachweisen und zu Lernzwecken
- Fundament — Tätigen Sie grundlegende Investitionen, um Ihre Cloud-Einführung zu skalieren (z. B. Einrichtung einer landing zone, Definition eines CCo E, Einrichtung eines Betriebsmodells)

- Migration – Migrieren einzelner Anwendungen
- Neuentwicklung – Optimierung von Produkten und Services und Innovation in der Cloud

Diese Phasen wurden von Stephen Orban im Blogbeitrag The Journey Toward Cloud-First & the Stages of Adoption im AWS Cloud Enterprise Strategy-Blog definiert. Informationen darüber, wie sie mit der AWS Migrationsstrategie zusammenhängen, finden Sie im Leitfaden zur Vorbereitung der Migration.

CMDB

Siehe Datenbank für das Konfigurationsmanagement.

Code-Repository

Ein Ort, an dem Quellcode und andere Komponenten wie Dokumentation, Beispiele und Skripts gespeichert und im Rahmen von Versionskontrollprozessen aktualisiert werden. Zu den gängigen Cloud-Repositorys gehören GitHub oderBitbucket Cloud. Jede Version des Codes wird Zweig genannt. In einer Microservice-Struktur ist jedes Repository einer einzelnen Funktionalität gewidmet. Eine einzelne CI/CD-Pipeline kann mehrere Repositorien verwenden.

Kalter Cache

Ein Puffer-Cache, der leer oder nicht gut gefüllt ist oder veraltete oder irrelevante Daten enthält. Dies beeinträchtigt die Leistung, da die Datenbank-Instance aus dem Hauptspeicher oder der Festplatte lesen muss, was langsamer ist als das Lesen aus dem Puffercache.

Kalte Daten

Daten, auf die selten zugegriffen wird und die in der Regel historisch sind. Bei der Abfrage dieser Art von Daten sind langsame Abfragen in der Regel akzeptabel. Durch die Verlagerung dieser Daten auf leistungsschwächere und kostengünstigere Speicherstufen oder -klassen können Kosten gesenkt werden.

Computer Vision (CV)

Ein Bereich der KI, der maschinelles Lernen nutzt, um Informationen aus visuellen Formaten wie digitalen Bildern und Videos zu analysieren und zu extrahieren. Amazon SageMaker AI bietet beispielsweise Bildverarbeitungsalgorithmen für CV.

Drift in der Konfiguration

Bei einer Arbeitslast eine Änderung der Konfiguration gegenüber dem erwarteten Zustand. Dies kann dazu führen, dass der Workload nicht mehr richtlinienkonform wird, und zwar in der Regel schrittweise und unbeabsichtigt.

Verwaltung der Datenbankkonfiguration (CMDB)

Ein Repository, das Informationen über eine Datenbank und ihre IT-Umgebung speichert und verwaltet, inklusive Hardware- und Softwarekomponenten und deren Konfigurationen. In der Regel verwenden Sie Daten aus einer CMDB in der Phase der Portfolioerkennung und -analyse der Migration.

Konformitätspaket

Eine Sammlung von AWS Config Regeln und Abhilfemaßnahmen, die Sie zusammenstellen können, um Ihre Konformitäts- und Sicherheitsprüfungen individuell anzupassen. Mithilfe einer YAML-Vorlage können Sie ein Conformance Pack als einzelne Entität in einer AWS-Konto AND-Region oder unternehmensweit bereitstellen. Weitere Informationen finden Sie in der Dokumentation unter Conformance Packs. AWS Config

Kontinuierliche Bereitstellung und kontinuierliche Integration (CI/CD)

Der Prozess der Automatisierung der Quell-, Build-, Test-, Staging- und Produktionsphasen des Softwareveröffentlichungsprozesses. CI/CD is commonly described as a pipeline. CI/CDkann Ihnen helfen, Prozesse zu automatisieren, die Produktivität zu steigern, die Codequalität zu verbessern und schneller zu liefern. Weitere Informationen finden Sie unter Vorteile der kontinuierlichen Auslieferung. CD kann auch für kontinuierliche Bereitstellung stehen. Weitere Informationen finden Sie unterKontinuierliche Auslieferung im Vergleich zu kontinuierlicher Bereitstellung.

CV

Siehe Computer Vision.

# D

Daten im Ruhezustand

Daten, die in Ihrem Netzwerk stationär sind, z. B. Daten, die sich im Speicher befinden.

Datenklassifizierung

Ein Prozess zur Identifizierung und Kategorisierung der Daten in Ihrem Netzwerk auf der Grundlage ihrer Kritikalität und Sensitivität. Sie ist eine wichtige Komponente jeder Strategie für das Management von Cybersecurity-Risiken, da sie Ihnen hilft, die geeigneten Schutz- und Aufbewahrungskontrollen für die Daten zu bestimmen. Die Datenklassifizierung ist ein Bestandteil

der Sicherheitssäule im AWS Well-Architected Framework. Weitere Informationen finden Sie unter Datenklassifizierung.

Datendrift

Eine signifikante Variation zwischen den Produktionsdaten und den Daten, die zum Trainieren eines ML-Modells verwendet wurden, oder eine signifikante Änderung der Eingabedaten im Laufe der Zeit. Datendrift kann die Gesamtqualität, Genauigkeit und Fairness von ML-Modellvorhersagen beeinträchtigen.

Daten während der Übertragung

Daten, die sich aktiv durch Ihr Netzwerk bewegen, z. B. zwischen Netzwerkressourcen.

Datennetz

Ein architektonisches Framework, das verteilte, dezentrale Dateneigentum mit zentraler Verwaltung und Steuerung ermöglicht.

Datenminimierung

Das Prinzip, nur die Daten zu sammeln und zu verarbeiten, die unbedingt erforderlich sind. Durch Datenminimierung im AWS Cloud können Datenschutzrisiken, Kosten und der CO2-Fußabdruck Ihrer Analysen reduziert werden.

Datenperimeter

Eine Reihe präventiver Schutzmaßnahmen in Ihrer AWS Umgebung, die sicherstellen, dass nur vertrauenswürdige Identitäten auf vertrauenswürdige Ressourcen von erwarteten Netzwerken zugreifen. Weitere Informationen finden Sie unter Aufbau eines Datenperimeters auf. AWS

Vorverarbeitung der Daten

Rohdaten in ein Format umzuwandeln, das von Ihrem ML-Modell problemlos verarbeitet werden kann. Die Vorverarbeitung von Daten kann bedeuten, dass bestimmte Spalten oder Zeilen entfernt und fehlende, inkonsistente oder doppelte Werte behoben werden.

Herkunft der Daten

Der Prozess der Nachverfolgung des Ursprungs und der Geschichte von Daten während ihres gesamten Lebenszyklus, z. B. wie die Daten generiert, übertragen und gespeichert wurden.

betroffene Person

Eine Person, deren Daten gesammelt und verarbeitet werden.

Data Warehouse

Ein Datenverwaltungssystem, das Business Intelligence wie Analysen unterstützt. Data Warehouses enthalten in der Regel große Mengen historischer Daten und werden in der Regel für Abfragen und Analysen verwendet.

Datenbankdefinitionssprache (DDL)

Anweisungen oder Befehle zum Erstellen oder Ändern der Struktur von Tabellen und Objekten in einer Datenbank.

Datenbankmanipulationssprache (DML)

Anweisungen oder Befehle zum Ändern (Einfügen, Aktualisieren und Löschen) von Informationen in einer Datenbank.

DDL

Siehe Datenbankdefinitionssprache.

Deep-Ensemble

Mehrere Deep-Learning-Modelle zur Vorhersage kombinieren. Sie können Deep-Ensembles verwenden, um eine genauere Vorhersage zu erhalten oder um die Unsicherheit von Vorhersagen abzuschätzen.

Deep Learning

Ein ML-Teilbereich, der mehrere Schichten künstlicher neuronaler Netzwerke verwendet, um die Zuordnung zwischen Eingabedaten und Zielvariablen von Interesse zu ermitteln.

defense-in-depth

Ein Ansatz zur Informationssicherheit, bei dem eine Reihe von Sicherheitsmechanismen und -kontrollen sorgfältig in einem Computernetzwerk verteilt werden, um die Vertraulichkeit, Integrität und Verfügbarkeit des Netzwerks und der darin enthaltenen Daten zu schützen. Wenn Sie diese Strategie anwenden AWS, fügen Sie mehrere Steuerelemente auf verschiedenen Ebenen der AWS Organizations Struktur hinzu, um die Ressourcen zu schützen. Ein defense-in-depth Ansatz könnte beispielsweise Multi-Faktor-Authentifizierung, Netzwerksegmentierung und Verschlüsselung kombinieren.

delegierter Administrator

In AWS Organizations kann ein kompatibler Dienst ein AWS Mitgliedskonto registrieren, um die Konten der Organisation und die Berechtigungen für diesen Dienst zu verwalten. Dieses Konto

wird als delegierter Administrator für diesen Service bezeichnet. Weitere Informationen und eine Liste kompatibler Services finden Sie unter Services, die mit AWS Organizations funktionieren in der AWS Organizations -Dokumentation.

Bereitstellung

Der Prozess, bei dem eine Anwendung, neue Feature oder Codekorrekturen in der Zielumgebung verfügbar gemacht werden. Die Bereitstellung umfasst das Implementieren von Änderungen an einer Codebasis und das anschließende Erstellen und Ausführen dieser Codebasis in den Anwendungsumgebungen.

Entwicklungsumgebung

Siehe Umgebung.

Detektivische Kontrolle

Eine Sicherheitskontrolle, die darauf ausgelegt ist, ein Ereignis zu erkennen, zu protokollieren und zu warnen, nachdem ein Ereignis eingetreten ist. Diese Kontrollen stellen eine zweite Verteidigungslinie dar und warnen Sie vor Sicherheitsereignissen, bei denen die vorhandenen präventiven Kontrollen umgangen wurden. Weitere Informationen finden Sie unter Detektivische Kontrolle in Implementierung von Sicherheitskontrollen in AWS.

Abbildung des Wertstroms in der Entwicklung (DVSM)

Ein Prozess zur Identifizierung und Priorisierung von Einschränkungen, die sich negativ auf Geschwindigkeit und Qualität im Lebenszyklus der Softwareentwicklung auswirken. DVSM erweitert den Prozess der Wertstromanalyse, der ursprünglich für Lean-Manufacturing-Praktiken konzipiert wurde. Es konzentriert sich auf die Schritte und Teams, die erforderlich sind, um durch den Softwareentwicklungsprozess Mehrwert zu schaffen und zu steigern.

digitaler Zwilling

Eine virtuelle Darstellung eines realen Systems, z. B. eines Gebäudes, einer Fabrik, einer Industrieanlage oder einer Produktionslinie. Digitale Zwillinge unterstützen vorausschauende Wartung, Fernüberwachung und Produktionsoptimierung.

Maßtabelle

In einem Sternschema eine kleinere Tabelle, die Datenattribute zu quantitativen Daten in einer Faktentabelle enthält. Bei Attributen von Dimensionstabellen handelt es sich in der Regel um Textfelder oder diskrete Zahlen, die sich wie Text verhalten. Diese Attribute werden häufig zum Einschränken von Abfragen, zum Filtern und zur Kennzeichnung von Ergebnismengen verwendet.

Katastrophe

Ein Ereignis, das verhindert, dass ein Workload oder ein System seine Geschäftsziele an seinem primären Einsatzort erfüllt. Diese Ereignisse können Naturkatastrophen, technische Ausfälle oder das Ergebnis menschlichen Handelns sein, z. B. unbeabsichtigte Fehlkonfigurationen oder ein Malware-Angriff.

Disaster Recovery (DR)

Die Strategie und der Prozess, die Sie verwenden, um Ausfallzeiten und Datenverluste aufgrund einer Katastrophe zu minimieren. Weitere Informationen finden Sie unter Disaster Recovery von Workloads unter AWS: Wiederherstellung in der Cloud im AWS Well-Architected Framework.

DML

Siehe Sprache zur Datenbankmanipulation.

Domainorientiertes Design

Ein Ansatz zur Entwicklung eines komplexen Softwaresystems, bei dem seine Komponenten mit sich entwickelnden Domains oder Kerngeschäftszielen verknüpft werden, denen jede Komponente dient. Dieses Konzept wurde von Eric Evans in seinem Buch Domaingesteuertes Design: Bewältigen der Komplexität im Herzen der Software (Boston: Addison-Wesley Professional, 2003) vorgestellt. Informationen darüber, wie Sie domaingesteuertes Design mit dem Strangler-Fig-Muster verwenden können, finden Sie unter Schrittweises Modernisieren älterer Microsoft ASP.NET (ASMX)-Webservices mithilfe von Containern und Amazon API Gateway.

DR

Siehe Disaster Recovery.

Erkennung von Driften

Verfolgung von Abweichungen von einer Basiskonfiguration. Sie können es beispielsweise verwenden, AWS CloudFormation um Abweichungen bei den Systemressourcen zu erkennen, oder Sie können AWS Control Tower damit Änderungen in Ihrer landing zone erkennen, die sich auf die Einhaltung von Governance-Anforderungen auswirken könnten.

DVSM

Siehe Abbildung des Wertstroms in der Entwicklung.

# E

EDA

Siehe explorative Datenanalyse.

EDI

Siehe elektronischer Datenaustausch.

Edge-Computing

Die Technologie, die die Rechenleistung für intelligente Geräte an den Rändern eines IoT-Netzwerks erhöht. Im Vergleich zu Cloud Computing kann Edge Computing die Kommunikationslatenz reduzieren und die Reaktionszeit verbessern.

elektronischer Datenaustausch (EDI)

Der automatisierte Austausch von Geschäftsdokumenten zwischen Organisationen. Weitere Informationen finden Sie unter Was ist elektronischer Datenaustausch.

Verschlüsselung

Ein Rechenprozess, der Klartextdaten, die für Menschen lesbar sind, in Chiffretext umwandelt.

Verschlüsselungsschlüssel

Eine kryptografische Zeichenfolge aus zufälligen Bits, die von einem Verschlüsselungsalgorithmus generiert wird. Schlüssel können unterschiedlich lang sein, und jeder Schlüssel ist so konzipiert, dass er unvorhersehbar und einzigartig ist.

Endianismus

Die Reihenfolge, in der Bytes im Computerspeicher gespeichert werden. Big-Endian-Systeme speichern das höchstwertige Byte zuerst. Little-Endian-Systeme speichern das niedrigwertigste Byte zuerst.

Endpunkt

Siehe Service-Endpunkt.

Endpunkt-Services

Ein Service, den Sie in einer Virtual Private Cloud (VPC) hosten können, um ihn mit anderen Benutzern zu teilen. Sie können einen Endpunktdienst mit anderen AWS-Konten oder AWS Identity and Access Management (IAM AWS PrivateLink -) Prinzipalen erstellen und diesen

Berechtigungen gewähren. Diese Konten oder Prinzipale können sich privat mit Ihrem Endpunktservice verbinden, indem sie Schnittstellen-VPC-Endpunkte erstellen. Weitere Informationen finden Sie unter Einen Endpunkt-Service erstellen in der Amazon Virtual Private Cloud (Amazon VPC)-Dokumentation.

Unternehmensressourcenplanung (ERP)

Ein System, das wichtige Geschäftsprozesse (wie Buchhaltung, MES und Projektmanagement) für ein Unternehmen automatisiert und verwaltet.

Envelope-Verschlüsselung

Der Prozess der Verschlüsselung eines Verschlüsselungsschlüssels mit einem anderen Verschlüsselungsschlüssel. Weitere Informationen finden Sie unter Envelope-Verschlüsselung in der AWS Key Management Service (AWS KMS) -Dokumentation.

Umgebung

Eine Instance einer laufenden Anwendung. Die folgenden Arten von Umgebungen sind beim Cloud-Computing üblich:

- Entwicklungsumgebung – Eine Instance einer laufenden Anwendung, die nur dem Kernteam zur Verfügung steht, das für die Wartung der Anwendung verantwortlich ist. Entwicklungsumgebungen werden verwendet, um Änderungen zu testen, bevor sie in höhere Umgebungen übertragen werden. Diese Art von Umgebung wird manchmal als Testumgebung bezeichnet.
- Niedrigere Umgebungen – Alle Entwicklungsumgebungen für eine Anwendung, z. B. solche, die für erste Builds und Tests verwendet wurden.
- Produktionsumgebung – Eine Instance einer laufenden Anwendung, auf die Endbenutzer zugreifen können. In einer CI/CD-Pipeline ist die Produktionsumgebung die letzte Bereitstellungsumgebung.
- Höhere Umgebungen – Alle Umgebungen, auf die auch andere Benutzer als das Kernentwicklungsteam zugreifen können. Dies kann eine Produktionsumgebung, Vorproduktionsumgebungen und Umgebungen für Benutzerakzeptanztests umfassen.

Epics

In der agilen Methodik sind dies funktionale Kategorien, die Ihnen helfen, Ihre Arbeit zu organisieren und zu priorisieren. Epics bieten eine allgemeine Beschreibung der Anforderungen und Implementierungsaufgaben. Zu den Sicherheitsthemen AWS von CAF gehören beispielsweise Identitäts- und Zugriffsmanagement, Detektivkontrollen, Infrastruktursicherheit,

Datenschutz und Reaktion auf Vorfälle. Weitere Informationen zu Epics in der AWS - Migrationsstrategie finden Sie im Leitfaden zur Programm-Implementierung.

ERP

Siehe Enterprise Resource Planning.

Explorative Datenanalyse (EDA)

Der Prozess der Analyse eines Datensatzes, um seine Hauptmerkmale zu verstehen. Sie sammeln oder aggregieren Daten und führen dann erste Untersuchungen durch, um Muster zu finden, Anomalien zu erkennen und Annahmen zu überprüfen. EDA wird durchgeführt, indem zusammenfassende Statistiken berechnet und Datenvisualisierungen erstellt werden.

# F

Faktentabelle

Die zentrale Tabelle in einem Sternschema. Sie speichert quantitative Daten über den Geschäftsbetrieb. In der Regel enthält eine Faktentabelle zwei Arten von Spalten: Spalten, die Kennzahlen enthalten, und Spalten, die einen Fremdschlüssel für eine Dimensionstabelle enthalten.

schnell scheitern

Eine Philosophie, die häufige und inkrementelle Tests verwendet, um den Entwicklungslebenszyklus zu verkürzen. Dies ist ein wichtiger Bestandteil eines agilen Ansatzes.

Grenze zur Fehlerisolierung

Dabei handelt es sich um eine Grenze AWS Cloud, z. B. eine Availability Zone AWS-Region, eine Steuerungsebene oder eine Datenebene, die die Auswirkungen eines Fehlers begrenzt und die Widerstandsfähigkeit von Workloads verbessert. Weitere Informationen finden Sie unter Grenzen zur AWS Fehlerisolierung.

Feature-Zweig

Siehe Zweig.

Features

Die Eingabedaten, die Sie verwenden, um eine Vorhersage zu treffen. In einem Fertigungskontext könnten Feature beispielsweise Bilder sein, die regelmäßig von der Fertigungslinie aus aufgenommen werden.

Bedeutung der Feature

Wie wichtig ein Feature für die Vorhersagen eines Modells ist. Dies wird in der Regel als numerischer Wert ausgedrückt, der mit verschiedenen Techniken wie Shapley Additive Explanations (SHAP) und integrierten Gradienten berechnet werden kann. Weitere Informationen finden Sie unter Interpretierbarkeit von Modellen für maschinelles Lernen mit AWS.

Featuretransformation

Daten für den ML-Prozess optimieren, einschließlich der Anreicherung von Daten mit zusätzlichen Quellen, der Skalierung von Werten oder der Extraktion mehrerer Informationssätze aus einem einzigen Datenfeld. Das ermöglicht dem ML-Modell, von den Daten profitieren. Wenn Sie beispielsweise das Datum „27.05.2021 00:15:37“ in „2021“, „Mai“, „Donnerstag“ und „15“ aufschlüsseln, können Sie dem Lernalgorithmus helfen, nuancierte Muster zu erlernen, die mit verschiedenen Datenkomponenten verknüpft sind.

Eingabeaufforderung mit wenigen Klicks

Bereitstellung einer kleinen Anzahl von Beispielen, die die Aufgabe und das gewünschte Ergebnis veranschaulichen, bevor das LLM aufgefordert wird, eine ähnliche Aufgabe auszuführen. Bei dieser Technik handelt es sich um eine Anwendung des kontextbezogenen Lernens, bei der Modelle anhand von Beispielen (Aufnahmen) lernen, die in Eingabeaufforderungen eingebettet sind. Bei Aufgaben, die spezifische Formatierungs-, Argumentations- oder Fachkenntnisse erfordern, kann die Eingabeaufforderung mit wenigen Handgriffen effektiv sein. Siehe auch Zero-Shot Prompting.

FGAC

Siehe detaillierte Zugriffskontrolle.

Feinkörnige Zugriffskontrolle (FGAC)

Die Verwendung mehrerer Bedingungen, um eine Zugriffsanfrage zuzulassen oder abzulehnen.

Flash-Cut-Migration

Eine Datenbankmigrationsmethode, bei der eine kontinuierliche Datenreplikation durch Erfassung von Änderungsdaten verwendet wird, um Daten in kürzester Zeit zu migrieren, anstatt einen schrittweisen Ansatz zu verwenden. Ziel ist es, Ausfallzeiten auf ein Minimum zu beschränken.

FM

Siehe Fundamentmodell.

Fundamentmodell (FM)

Ein großes neuronales Deep-Learning-Netzwerk, das mit riesigen Datensätzen generalisierter und unbeschrifteter Daten trainiert wurde. FMs sind in der Lage, eine Vielzahl allgemeiner Aufgaben zu erfüllen, z. B. Sprache zu verstehen, Text und Bilder zu generieren und Konversationen in natürlicher Sprache zu führen. Weitere Informationen finden Sie unter Was sind Foundation-Modelle.

# G

generative KI

Eine Untergruppe von KI-Modellen, die mit großen Datenmengen trainiert wurden und mit einer einfachen Textaufforderung neue Inhalte und Artefakte wie Bilder, Videos, Text und Audio erstellen können. Weitere Informationen finden Sie unter Was ist Generative KI.

Geoblocking

Siehe geografische Einschränkungen.

Geografische Einschränkungen (Geoblocking)

Bei Amazon eine Option CloudFront, um zu verhindern, dass Benutzer in bestimmten Ländern auf Inhaltsverteilungen zugreifen. Sie können eine Zulassungsliste oder eine Sperrliste verwenden, um zugelassene und gesperrte Länder anzugeben. Weitere Informationen finden Sie in der Dokumentation unter Beschränkung der geografischen Verteilung Ihrer Inhalte. CloudFront

Gitflow-Workflow

Ein Ansatz, bei dem niedrigere und höhere Umgebungen unterschiedliche Zweige in einem Quellcode-Repository verwenden. Der Gitflow-Workflow gilt als veraltet, und der Trunk-basierte Workflow ist der moderne, bevorzugte Ansatz.

goldenes Bild

Ein Snapshot eines Systems oder einer Software, der als Vorlage für die Bereitstellung neuer Instanzen dieses Systems oder dieser Software verwendet wird. In der Fertigung kann ein Golden Image beispielsweise zur Bereitstellung von Software auf mehreren Geräten verwendet werden und trägt zur Verbesserung der Geschwindigkeit, Skalierbarkeit und Produktivität bei der Geräteherstellung bei.

Greenfield-Strategie

Das Fehlen vorhandener Infrastruktur in einer neuen Umgebung. Bei der Einführung einer Neuausrichtung einer Systemarchitektur können Sie alle neuen Technologien ohne Einschränkung der Kompatibilität mit der vorhandenen Infrastruktur auswählen, auch bekannt als Brownfield. Wenn Sie die bestehende Infrastruktur erweitern, könnten Sie Brownfield- und Greenfield-Strategien mischen.

Integritätsschutz

Eine allgemeine Regel, die dazu beiträgt, Ressourcen, Richtlinien und die Einhaltung von Vorschriften in allen Unternehmenseinheiten zu regeln (OUs). Präventiver Integritätsschutz setzt Richtlinien durch, um die Einhaltung von Standards zu gewährleisten. Sie werden mithilfe von Service-Kontrollrichtlinien und IAM-Berechtigungsgrenzen implementiert. Detektivischer Integritätsschutz erkennt Richtlinienverstöße und Compliance-Probleme und generiert Warnmeldungen zur Abhilfe. Sie werden mithilfe von AWS Config, AWS Security Hub, Amazon GuardDuty AWS Trusted Advisor, Amazon Inspector und benutzerdefinierten AWS Lambda Prüfungen implementiert.

# H

HEKTAR

Siehe Hochverfügbarkeit.

Heterogene Datenbankmigration

Migrieren Sie Ihre Quelldatenbank in eine Zieldatenbank, die eine andere Datenbank-Engine verwendet (z. B. Oracle zu Amazon Aurora). Eine heterogene Migration ist in der Regel Teil einer Neuarchitektur, und die Konvertierung des Schemas kann eine komplexe Aufgabe sein. AWS bietet AWS SCT, welches bei Schemakonvertierungen hilft.

hohe Verfügbarkeit (HA)

Die Fähigkeit eines Workloads, im Falle von Herausforderungen oder Katastrophen kontinuierlich und ohne Eingreifen zu arbeiten. HA-Systeme sind so konzipiert, dass sie automatisch ein Failover durchführen, gleichbleibend hohe Leistung bieten und unterschiedliche Lasten und Ausfälle mit minimalen Leistungseinbußen bewältigen.

historische Modernisierung

Ein Ansatz zur Modernisierung und Aufrüstung von Betriebstechnologiesystemen (OT), um den Bedürfnissen der Fertigungsindustrie besser gerecht zu werden. Ein Historian ist eine Art von Datenbank, die verwendet wird, um Daten aus verschiedenen Quellen in einer Fabrik zu sammeln und zu speichern.

Holdout-Daten

Ein Teil historischer, beschrifteter Daten, der aus einem Datensatz zurückgehalten wird, der zum Trainieren eines Modells für maschinelles Lernen verwendet wird. Sie können Holdout-Daten verwenden, um die Modellleistung zu bewerten, indem Sie die Modellvorhersagen mit den Holdout-Daten vergleichen.

Homogene Datenbankmigration

Migrieren Sie Ihre Quelldatenbank zu einer Zieldatenbank, die dieselbe Datenbank-Engine verwendet (z. B. Microsoft SQL Server zu Amazon RDS für SQL Server). Eine homogene Migration ist in der Regel Teil eines Hostwechsels oder eines Plattformwechsels. Sie können native Datenbankserviceprogramme verwenden, um das Schema zu migrieren.

heiße Daten

Daten, auf die häufig zugegriffen wird, z. B. Echtzeitdaten oder aktuelle Translationsdaten. Für diese Daten ist in der Regel eine leistungsstarke Speicherebene oder -klasse erforderlich, um schnelle Abfrageantworten zu ermöglichen.

Hotfix

Eine dringende Lösung für ein kritisches Problem in einer Produktionsumgebung. Aufgrund seiner Dringlichkeit wird ein Hotfix normalerweise außerhalb des typischen DevOps Release-Workflows erstellt.

Hypercare-Phase

Unmittelbar nach dem Cutover, der Zeitraum, in dem ein Migrationsteam die migrierten Anwendungen in der Cloud verwaltet und überwacht, um etwaige Probleme zu beheben. In der Regel dauert dieser Zeitraum 1–4 Tage. Am Ende der Hypercare-Phase überträgt das Migrationsteam in der Regel die Verantwortung für die Anwendungen an das Cloud-Betriebsteam.

# I

IaC

Sehen Sie sich Infrastruktur als Code an.

Identitätsbasierte Richtlinie

Eine Richtlinie, die einem oder mehreren IAM-Prinzipalen zugeordnet ist und deren Berechtigungen innerhalb der AWS Cloud Umgebung definiert.

Leerlaufanwendung

Eine Anwendung mit einer durchschnittlichen CPU- und Arbeitsspeicherauslastung zwischen 5 und 20 Prozent über einen Zeitraum von 90 Tagen. In einem Migrationsprojekt ist es üblich, diese Anwendungen außer Betrieb zu nehmen oder sie On-Premises beizubehalten.

IIoT

Siehe Industrielles Internet der Dinge.

unveränderliche Infrastruktur

Ein Modell, das eine neue Infrastruktur für Produktionsworkloads bereitstellt, anstatt die bestehende Infrastruktur zu aktualisieren, zu patchen oder zu modifizieren. Unveränderliche Infrastrukturen sind von Natur aus konsistenter, zuverlässiger und vorhersehbarer als veränderliche Infrastrukturen. Weitere Informationen finden Sie in der Best Practice Deploy using immutable infrastructure im AWS Well-Architected Framework.

Eingehende (ingress) VPC

In einer Architektur AWS mit mehreren Konten ist dies eine VPC, die Netzwerkverbindungen von außerhalb einer Anwendung akzeptiert, überprüft und weiterleitet. Die AWS Security Reference Architecture empfiehlt, Ihr Netzwerkkonto mit eingehendem und ausgehendem Datenverkehr und Inspektion einzurichten, VPCs um die bidirektionale Schnittstelle zwischen Ihrer Anwendung und dem Internet im weiteren Sinne zu schützen.

Inkrementelle Migration

Eine Cutover-Strategie, bei der Sie Ihre Anwendung in kleinen Teilen migrieren, anstatt eine einziges vollständiges Cutover durchzuführen. Beispielsweise könnten Sie zunächst nur einige Microservices oder Benutzer auf das neue System umstellen. Nachdem Sie sich vergewissert haben, dass alles ordnungsgemäß funktioniert, können Sie weitere Microservices oder Benutzer

schrittweise verschieben, bis Sie Ihr Legacy-System außer Betrieb nehmen können. Diese Strategie reduziert die mit großen Migrationen verbundenen Risiken.

Industrie 4.0

Ein Begriff, der 2016 von Klaus Schwab eingeführt wurde und sich auf die Modernisierung von Fertigungsprozessen durch Fortschritte in den Bereichen Konnektivität, Echtzeitdaten, Automatisierung, Analytik und KI/ML bezieht.

Infrastruktur

Alle Ressourcen und Komponenten, die in der Umgebung einer Anwendung enthalten sind.

Infrastructure as Code (IaC)

Der Prozess der Bereitstellung und Verwaltung der Infrastruktur einer Anwendung mithilfe einer Reihe von Konfigurationsdateien. IaC soll Ihnen helfen, das Infrastrukturmanagement zu zentralisieren, Ressourcen zu standardisieren und schnell zu skalieren, sodass neue Umgebungen wiederholbar, zuverlässig und konsistent sind.

industrielles Internet der Dinge (T) IIo

Einsatz von mit dem Internet verbundenen Sensoren und Geräten in Industriesektoren wie Fertigung, Energie, Automobilindustrie, Gesundheitswesen, Biowissenschaften und Landwirtschaft. Weitere Informationen finden Sie unter Aufbau einer digitalen Transformationsstrategie für das industrielle Internet der Dinge (IIoT).

Inspektions-VPC

In einer Architektur AWS mit mehreren Konten eine zentralisierte VPC, die Inspektionen des Netzwerkverkehrs zwischen VPCs (in demselben oder unterschiedlichen AWS-Regionen), dem Internet und lokalen Netzwerken verwaltet. In der AWS Security Reference Architecture wird empfohlen, Ihr Netzwerkkonto mit eingehendem und ausgehendem Datenverkehr sowie Inspektionen einzurichten, VPCs um die bidirektionale Schnittstelle zwischen Ihrer Anwendung und dem Internet im weiteren Sinne zu schützen.

Internet of Things (IoT)

Das Netzwerk verbundener physischer Objekte mit eingebetteten Sensoren oder Prozessoren, das über das Internet oder über ein lokales Kommunikationsnetzwerk mit anderen Geräten und Systemen kommuniziert. Weitere Informationen finden Sie unter Was ist IoT?

Interpretierbarkeit

Ein Merkmal eines Modells für Machine Learning, das beschreibt, inwieweit ein Mensch verstehen kann, wie die Vorhersagen des Modells von seinen Eingaben abhängen. Weitere Informationen finden Sie unter Interpretierbarkeit von Modellen für maschinelles Lernen mit. AWS

IoT

Siehe Internet der Dinge.

IT information library (ITIL, IT-Informationsbibliothek)

Eine Reihe von bewährten Methoden für die Bereitstellung von IT-Services und die Abstimmung dieser Services auf die Geschäftsanforderungen. ITIL bietet die Grundlage für ITSM.

T service management (ITSM, IT-Servicemanagement)

Aktivitäten im Zusammenhang mit der Gestaltung, Implementierung, Verwaltung und Unterstützung von IT-Services für eine Organisation. Informationen zur Integration von Cloud-Vorgängen mit ITSM-Tools finden Sie im Leitfaden zur Betriebsintegration.

BIS

Weitere Informationen finden Sie in der IT-Informationsbibliothek.

ITSM

Siehe IT-Servicemanagement.

# L

Labelbasierte Zugangskontrolle (LBAC)

Eine Implementierung der Mandatory Access Control (MAC), bei der den Benutzern und den Daten selbst jeweils explizit ein Sicherheitslabelwert zugewiesen wird. Die Schnittmenge zwischen der Benutzersicherheitsbeschriftung und der Datensicherheitsbeschriftung bestimmt, welche Zeilen und Spalten für den Benutzer sichtbar sind.

Landing Zone

Eine landing zone ist eine gut strukturierte AWS Umgebung mit mehreren Konten, die skalierbar und sicher ist. Dies ist ein Ausgangspunkt, von dem aus Ihre Organisationen Workloads und Anwendungen schnell und mit Vertrauen in ihre Sicherheits- und Infrastrukturumgebung starten

und bereitstellen können. Weitere Informationen zu Landing Zones finden Sie unter Einrichtung einer sicheren und skalierbaren AWS -Umgebung mit mehreren Konten..

großes Sprachmodell (LLM)

Ein Deep-Learning-KI-Modell, das anhand einer riesigen Datenmenge vorab trainiert wurde. Ein LLM kann mehrere Aufgaben ausführen, z. B. Fragen beantworten, Dokumente zusammenfassen, Text in andere Sprachen übersetzen und Sätze vervollständigen. Weitere Informationen finden Sie unter Was sind. LLMs

Große Migration

Eine Migration von 300 oder mehr Servern.

SCHWARZ

Weitere Informationen finden Sie unter Label-basierte Zugriffskontrolle.

Geringste Berechtigung

Die bewährte Sicherheitsmethode, bei der nur die für die Durchführung einer Aufgabe erforderlichen Mindestberechtigungen erteilt werden. Weitere Informationen finden Sie unter Geringste Berechtigungen anwenden in der IAM-Dokumentation.

Lift and Shift

Siehe 7 Rs.

Little-Endian-System

Ein System, welches das niedrigwertigste Byte zuerst speichert. Siehe auch Endianness.

LLM

Siehe großes Sprachmodell.

Niedrigere Umgebungen

Siehe Umgebung.

# M

Machine Learning (ML)

Eine Art künstlicher Intelligenz, die Algorithmen und Techniken zur Mustererkennung und zum Lernen verwendet. ML analysiert aufgezeichnete Daten, wie z. B. Daten aus dem Internet der

Dinge (IoT), und lernt daraus, um ein statistisches Modell auf der Grundlage von Mustern zu erstellen. Weitere Informationen finden Sie unter Machine Learning.

Hauptzweig

Siehe Filiale.

Malware

Software, die entwickelt wurde, um die Computersicherheit oder den Datenschutz zu gefährden. Malware kann Computersysteme stören, vertrauliche Informationen durchsickern lassen oder sich unbefugten Zugriff verschaffen. Beispiele für Malware sind Viren, Würmer, Ransomware, Trojaner, Spyware und Keylogger.

verwaltete Dienste

AWS-Services für die die Infrastrukturebene, das Betriebssystem und die Plattformen AWS betrieben werden, und Sie greifen auf die Endgeräte zu, um Daten zu speichern und abzurufen. Amazon Simple Storage Service (Amazon S3) und Amazon DynamoDB sind Beispiele für Managed Services. Diese werden auch als abstrakte Dienste bezeichnet.

Manufacturing Execution System (MES)

Ein Softwaresystem zur Nachverfolgung, Überwachung, Dokumentation und Steuerung von Produktionsprozessen, bei denen Rohstoffe in der Fertigung zu fertigen Produkten umgewandelt werden.

MAP

Siehe Migration Acceleration Program.

Mechanismus

Ein vollständiger Prozess, bei dem Sie ein Tool erstellen, die Akzeptanz des Tools vorantreiben und anschließend die Ergebnisse überprüfen, um Anpassungen vorzunehmen. Ein Mechanismus ist ein Zyklus, der sich im Laufe seiner Tätigkeit selbst verstärkt und verbessert. Weitere Informationen finden Sie unter Aufbau von Mechanismen im AWS Well-Architected Framework.

Mitgliedskonto

Alle AWS-Konten außer dem Verwaltungskonto, die Teil einer Organisation in sind. AWS Organizations Ein Konto kann jeweils nur einer Organisation angehören.

DURCHEINANDER

Siehe Manufacturing Execution System.

Message Queuing-Telemetrietransport (MQTT)

Ein leichtes machine-to-machine (M2M) -Kommunikationsprotokoll, das auf dem Publish/Subscribe-Muster für IoT-Geräte mit beschränkten Ressourcen basiert.

Microservice

Ein kleiner, unabhängiger Dienst, der über genau definierte Kanäle kommuniziert APIs und in der Regel kleinen, eigenständigen Teams gehört. Ein Versicherungssystem kann beispielsweise Microservices beinhalten, die Geschäftsfunktionen wie Vertrieb oder Marketing oder Subdomains wie Einkauf, Schadenersatz oder Analytik zugeordnet sind. Zu den Vorteilen von Microservices gehören Agilität, flexible Skalierung, einfache Bereitstellung, wiederverwendbarer Code und Ausfallsicherheit. Weitere Informationen finden Sie unter Integration von Microservices mithilfe serverloser Dienste. AWS

Microservices-Architekturen

Ein Ansatz zur Erstellung einer Anwendung mit unabhängigen Komponenten, die jeden Anwendungsprozess als Microservice ausführen. Diese Microservices kommunizieren mithilfe von Lightweight über eine klar definierte Schnittstelle. APIs Jeder Microservice in dieser Architektur kann aktualisiert, bereitgestellt und skaliert werden, um den Bedarf an bestimmten Funktionen einer Anwendung zu decken. Weitere Informationen finden Sie unter Implementierung von Microservices auf. AWS

Migration Acceleration Program (MAP)

Ein AWS Programm, das Beratung, Unterstützung, Schulungen und Services bietet, um Unternehmen dabei zu unterstützen, eine solide betriebliche Grundlage für die Umstellung auf die Cloud zu schaffen und die anfänglichen Kosten von Migrationen auszugleichen. MAP umfasst eine Migrationsmethode für die methodische Durchführung von Legacy-Migrationen sowie eine Reihe von Tools zur Automatisierung und Beschleunigung gängiger Migrationsszenarien.

Migration in großem Maßstab

Der Prozess, bei dem der Großteil des Anwendungsportfolios in Wellen in die Cloud verlagert wird, wobei in jeder Welle mehr Anwendungen schneller migriert werden. In dieser Phase werden die bewährten Verfahren und Erkenntnisse aus den früheren Phasen zur Implementierung einer Migrationsfabrik von Teams, Tools und Prozessen zur Optimierung der Migration von Workloads durch Automatisierung und agile Bereitstellung verwendet. Dies ist die dritte Phase der AWS - Migrationsstrategie.

Migrationsfabrik

Funktionsübergreifende Teams, die die Migration von Workloads durch automatisierte, agile Ansätze optimieren. Zu den Teams in der Migrationsabteilung gehören in der Regel Betriebsabläufe, Geschäftsanalysten und Eigentümer, Migrationsingenieure, Entwickler und DevOps Experten, die in Sprints arbeiten. Zwischen 20 und 50 Prozent eines Unternehmensanwendungsportfolios bestehen aus sich wiederholenden Mustern, die durch einen Fabrik-Ansatz optimiert werden können. Weitere Informationen finden Sie in Diskussion über Migrationsfabriken und den Leitfaden zur Cloud-Migration-Fabrik in diesem Inhaltssatz.

Migrationsmetadaten

Die Informationen über die Anwendung und den Server, die für den Abschluss der Migration benötigt werden. Für jedes Migrationsmuster ist ein anderer Satz von Migrationsmetadaten erforderlich. Beispiele für Migrationsmetadaten sind das Zielsubnetz, die Sicherheitsgruppe und AWS das Konto.

Migrationsmuster

Eine wiederholbare Migrationsaufgabe, in der die Migrationsstrategie, das Migrationsziel und die verwendete Migrationsanwendung oder der verwendete Migrationsservice detailliert beschrieben werden. Beispiel: Rehost-Migration zu Amazon EC2 mit AWS Application Migration Service.

Migration Portfolio Assessment (MPA)

Ein Online-Tool, das Informationen zur Validierung des Geschäftsszenarios für die Migration auf das bereitstellt. AWS Cloud MPA bietet eine detaillierte Portfoliobewertung (richtige Servergröße, Preisgestaltung, Gesamtbetriebskostenanalyse, Migrationskostenanalyse) sowie Migrationsplanung (Anwendungsdatenanalyse und Datenerfassung, Anwendungsgruppierung, Migrationspriorisierung und Wellenplanung). Das MPA-Tool (Anmeldung erforderlich) steht allen AWS Beratern und APN-Partnerberatern kostenlos zur Verfügung.

Migration Readiness Assessment (MRA)

Der Prozess, bei dem mithilfe des AWS CAF Erkenntnisse über den Cloud-Bereitschaftsstatus eines Unternehmens gewonnen, Stärken und Schwächen identifiziert und ein Aktionsplan zur Schließung festgestellter Lücken erstellt wird. Weitere Informationen finden Sie im Benutzerhandbuch für Migration Readiness. MRA ist die erste Phase der AWS - Migrationsstrategie.

Migrationsstrategie

Der Ansatz, der verwendet wurde, um einen Workload auf den AWS Cloud zu migrieren. Weitere Informationen finden Sie im Eintrag 7 Rs in diesem Glossar und unter Mobilisieren Sie Ihr Unternehmen, um umfangreiche Migrationen zu beschleunigen.

ML

Siehe maschinelles Lernen.

Modernisierung

Umwandlung einer veralteten (veralteten oder monolithischen) Anwendung und ihrer Infrastruktur in ein agiles, elastisches und hochverfügbares System in der Cloud, um Kosten zu senken, die Effizienz zu steigern und Innovationen zu nutzen. Weitere Informationen finden Sie unter Strategie zur Modernisierung von Anwendungen in der AWS Cloud.

Bewertung der Modernisierungsfähigkeit

Eine Bewertung, anhand derer festgestellt werden kann, ob die Anwendungen einer Organisation für die Modernisierung bereit sind, Vorteile, Risiken und Abhängigkeiten identifiziert und ermittelt wird, wie gut die Organisation den zukünftigen Status dieser Anwendungen unterstützen kann. Das Ergebnis der Bewertung ist eine Vorlage der Zielarchitektur, eine Roadmap, in der die Entwicklungsphasen und Meilensteine des Modernisierungsprozesses detailliert beschrieben werden, sowie ein Aktionsplan zur Behebung festgestellter Lücken. Weitere Informationen finden Sie unter Evaluierung der Modernisierungsbereitschaft von Anwendungen in der AWS Cloud.

Monolithische Anwendungen (Monolithen)

Anwendungen, die als ein einziger Service mit eng gekoppelten Prozessen ausgeführt werden. Monolithische Anwendungen haben verschiedene Nachteile. Wenn ein Anwendungs-Feature stark nachgefragt wird, muss die gesamte Architektur skaliert werden. Das Hinzufügen oder Verbessern der Feature einer monolithischen Anwendung wird ebenfalls komplexer, wenn die Codebasis wächst. Um diese Probleme zu beheben, können Sie eine Microservices-Architektur verwenden. Weitere Informationen finden Sie unter Zerlegen von Monolithen in Microservices.

MPA

Siehe Bewertung des Migrationsportfolios.

MQTT

Siehe Message Queuing-Telemetrietransport.

Mehrklassen-Klassifizierung

Ein Prozess, der dabei hilft, Vorhersagen für mehrere Klassen zu generieren (wobei eines von mehr als zwei Ergebnissen vorhergesagt wird). Ein ML-Modell könnte beispielsweise fragen: „Ist dieses Produkt ein Buch, ein Auto oder ein Telefon?“ oder „Welche Kategorie von Produkten ist für diesen Kunden am interessantesten?“

veränderbare Infrastruktur

Ein Modell, das die bestehende Infrastruktur für Produktionsworkloads aktualisiert und modifiziert. Für eine verbesserte Konsistenz, Zuverlässigkeit und Vorhersagbarkeit empfiehlt das AWS Well-Architected Framework die Verwendung einer unveränderlichen Infrastruktur als bewährte Methode.

# O

OAC

Siehe Origin Access Control.

OAI

Siehe Zugriffsidentität von Origin.

COM

Siehe organisatorisches Change-Management.

Offline-Migration

Eine Migrationsmethode, bei der der Quell-Workload während des Migrationsprozesses heruntergefahren wird. Diese Methode ist mit längeren Ausfallzeiten verbunden und wird in der Regel für kleine, unkritische Workloads verwendet.

OI

Siehe Betriebsintegration.

OLA

Siehe Vereinbarung auf operativer Ebene.

Online-Migration

Eine Migrationsmethode, bei der der Quell-Workload auf das Zielsystem kopiert wird, ohne offline genommen zu werden. Anwendungen, die mit dem Workload verbunden sind, können während

der Migration weiterhin funktionieren. Diese Methode beinhaltet keine bis minimale Ausfallzeit und wird in der Regel für kritische Produktionsworkloads verwendet.

OPC-UA

Siehe Open Process Communications — Unified Architecture.

Offene Prozesskommunikation — Einheitliche Architektur (OPC-UA)

Ein machine-to-machine (M2M) -Kommunikationsprotokoll für die industrielle Automatisierung. OPC-UA bietet einen Interoperabilitätsstandard mit Datenverschlüsselungs-, Authentifizierungs- und Autorisierungsschemata.

Vereinbarung auf Betriebsebene (OLA)

Eine Vereinbarung, in der klargestellt wird, welche funktionalen IT-Gruppen sich gegenseitig versprechen zu liefern, um ein Service Level Agreement (SLA) zu unterstützen.

Überprüfung der Betriebsbereitschaft (ORR)

Eine Checkliste mit Fragen und zugehörigen bewährten Methoden, die Ihnen helfen, Vorfälle und mögliche Ausfälle zu verstehen, zu bewerten, zu verhindern oder deren Umfang zu reduzieren. Weitere Informationen finden Sie unter Operational Readiness Reviews (ORR) im AWS Well-Architected Framework.

Betriebstechnologie (OT)

Hardware- und Softwaresysteme, die mit der physischen Umgebung zusammenarbeiten, um industrielle Abläufe, Ausrüstung und Infrastruktur zu steuern. In der Fertigung ist die Integration von OT- und Informationstechnologie (IT) -Systemen ein zentraler Schwerpunkt der Industrie 4.0-Transformationen.

Betriebsintegration (OI)

Der Prozess der Modernisierung von Abläufen in der Cloud, der Bereitschaftsplanung, Automatisierung und Integration umfasst. Weitere Informationen finden Sie im Leitfaden zur Betriebsintegration.

Organisationspfad

Ein Pfad, der von erstellt wird und in AWS CloudTrail dem alle Ereignisse für alle AWS-Konten in einer Organisation protokolliert werden. AWS Organizations Diese Spur wird in jedem AWS-Konto , der Teil der Organisation ist, erstellt und verfolgt die Aktivität in jedem Konto. Weitere Informationen finden Sie in der CloudTrail Dokumentation unter Einen Trail für eine Organisation erstellen.

Organisatorisches Veränderungsmanagement (OCM)

Ein Framework für das Management wichtiger, disruptiver Geschäftstransformationen aus Sicht der Mitarbeiter, der Kultur und der Führung. OCM hilft Organisationen dabei, sich auf neue Systeme und Strategien vorzubereiten und auf diese umzustellen, indem es die Akzeptanz von Veränderungen beschleunigt, Übergangsprobleme angeht und kulturelle und organisatorische Veränderungen vorantreibt. In der AWS Migrationsstrategie wird dieses Framework aufgrund der Geschwindigkeit des Wandels, der bei Projekten zur Cloud-Einführung erforderlich ist, als Mitarbeiterbeschleunigung bezeichnet. Weitere Informationen finden Sie im OCM-Handbuch.

Ursprungszugriffskontrolle (OAC)

In CloudFront, eine erweiterte Option zur Zugriffsbeschränkung, um Ihre Amazon Simple Storage Service (Amazon S3) -Inhalte zu sichern. OAC unterstützt alle S3-Buckets insgesamt AWS-Regionen, serverseitige Verschlüsselung mit AWS KMS (SSE-KMS) sowie dynamische `PUT` und `DELETE` Anfragen an den S3-Bucket.

Ursprungszugriffsidentität (OAI)

In CloudFront, eine Option zur Zugriffsbeschränkung, um Ihre Amazon S3 S3-Inhalte zu sichern. Wenn Sie OAI verwenden, CloudFront erstellt es einen Principal, mit dem sich Amazon S3 authentifizieren kann. Authentifizierte Principals können nur über eine bestimmte Distribution auf Inhalte in einem S3-Bucket zugreifen. CloudFront Siehe auch OAC, das eine detailliertere und verbesserte Zugriffskontrolle bietet.

ORR

Weitere Informationen finden Sie unter Überprüfung der Betriebsbereitschaft.

NICHT

Siehe Betriebstechnologie.

Ausgehende (egress) VPC

In einer Architektur AWS mit mehreren Konten eine VPC, die Netzwerkverbindungen verarbeitet, die von einer Anwendung aus initiiert werden. Die AWS Security Reference Architecture empfiehlt die Einrichtung Ihres Netzwerkkontos mit eingehendem und ausgehendem Datenverkehr sowie Inspektion, VPCs um die bidirektionale Schnittstelle zwischen Ihrer Anwendung und dem Internet im weiteren Sinne zu schützen.

# P

Berechtigungsgrenze

Eine IAM-Verwaltungsrichtlinie, die den IAM-Prinzipalen zugeordnet ist, um die maximalen Berechtigungen festzulegen, die der Benutzer oder die Rolle haben kann. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitys in der IAM-Dokumentation.

persönlich identifizierbare Informationen (PII)

Informationen, die, wenn sie direkt betrachtet oder mit anderen verwandten Daten kombiniert werden, verwendet werden können, um vernünftige Rückschlüsse auf die Identität einer Person zu ziehen. Beispiele für personenbezogene Daten sind Namen, Adressen und Kontaktinformationen.

Personenbezogene Daten

Siehe persönlich identifizierbare Informationen.

Playbook

Eine Reihe vordefinierter Schritte, die die mit Migrationen verbundenen Aufgaben erfassen, z. B. die Bereitstellung zentraler Betriebsfunktionen in der Cloud. Ein Playbook kann die Form von Skripten, automatisierten Runbooks oder einer Zusammenfassung der Prozesse oder Schritte annehmen, die für den Betrieb Ihrer modernisierten Umgebung erforderlich sind.

PLC

Siehe programmierbare Logiksteuerung.

PLM

Siehe Produktlebenszyklusmanagement.

policy

Ein Objekt, das Berechtigungen definieren (siehe identitätsbasierte Richtlinie), Zugriffsbedingungen spezifizieren (siehe ressourcenbasierte Richtlinie) oder die maximalen Berechtigungen für alle Konten in einer Organisation definieren kann AWS Organizations (siehe Dienststeuerungsrichtlinie).

Polyglotte Beharrlichkeit

Unabhängige Auswahl der Datenspeichertechnologie eines Microservices auf der Grundlage von Datenzugriffsmustern und anderen Anforderungen. Wenn Ihre Microservices über dieselbe

Datenspeichertechnologie verfügen, kann dies zu Implementierungsproblemen oder zu Leistungseinbußen führen. Microservices lassen sich leichter implementieren und erzielen eine bessere Leistung und Skalierbarkeit, wenn sie den Datenspeicher verwenden, der ihren Anforderungen am besten entspricht. Weitere Informationen finden Sie unter Datenpersistenz in Microservices aktivieren.

Portfoliobewertung

Ein Prozess, bei dem das Anwendungsportfolio ermittelt, analysiert und priorisiert wird, um die Migration zu planen. Weitere Informationen finden Sie in Bewerten der Migrationsbereitschaft.

predicate

Eine Abfragebedingung, die `true` oder zurückgibt`false`, was üblicherweise in einer Klausel vorkommt. `WHERE`

Prädikat Pushdown

Eine Technik zur Optimierung von Datenbankabfragen, bei der die Daten in der Abfrage vor der Übertragung gefiltert werden. Dadurch wird die Datenmenge reduziert, die aus der relationalen Datenbank abgerufen und verarbeitet werden muss, und die Abfrageleistung wird verbessert.

Präventive Kontrolle

Eine Sicherheitskontrolle, die verhindern soll, dass ein Ereignis eintritt. Diese Kontrollen stellen eine erste Verteidigungslinie dar, um unbefugten Zugriff oder unerwünschte Änderungen an Ihrem Netzwerk zu verhindern. Weitere Informationen finden Sie unter Präventive Kontrolle in Implementierung von Sicherheitskontrollen in AWS.

Prinzipal

Eine Entität AWS , die Aktionen ausführen und auf Ressourcen zugreifen kann. Bei dieser Entität handelt es sich in der Regel um einen Root-Benutzer für eine AWS-Konto, eine IAM-Rolle oder einen Benutzer. Weitere Informationen finden Sie unter Prinzipal in Rollenbegriffe und -konzepte in der IAM-Dokumentation.

Datenschutz von Natur aus

Ein systemtechnischer Ansatz, der den Datenschutz während des gesamten Entwicklungsprozesses berücksichtigt.

Privat gehostete Zonen

Ein Container, der Informationen darüber enthält, wie Amazon Route 53 auf DNS-Abfragen für eine Domain und deren Subdomains innerhalb einer oder mehrerer VPCs Domains antworten

soll. Weitere Informationen finden Sie unter Arbeiten mit privat gehosteten Zonen in der Route-53-Dokumentation.

proaktive Steuerung

Eine Sicherheitskontrolle, die den Einsatz nicht richtlinienkonformer Ressourcen verhindern soll. Diese Steuerelemente scannen Ressourcen, bevor sie bereitgestellt werden. Wenn die Ressource nicht mit der Steuerung konform ist, wird sie nicht bereitgestellt. Weitere Informationen finden Sie im Referenzhandbuch zu Kontrollen in der AWS Control Tower Dokumentation und unter Proaktive Kontrollen unter Implementierung von Sicherheitskontrollen am AWS.

Produktlebenszyklusmanagement (PLM)

Das Management von Daten und Prozessen für ein Produkt während seines gesamten Lebenszyklus, vom Design, der Entwicklung und Markteinführung über Wachstum und Reife bis hin zur Markteinführung und Markteinführung.

Produktionsumgebung

Siehe Umgebung.

Speicherprogrammierbare Steuerung (SPS)

In der Fertigung ein äußerst zuverlässiger, anpassungsfähiger Computer, der Maschinen überwacht und Fertigungsprozesse automatisiert.

schnelle Verkettung

Verwendung der Ausgabe einer LLM-Eingabeaufforderung als Eingabe für die nächste Aufforderung, um bessere Antworten zu generieren. Diese Technik wird verwendet, um eine komplexe Aufgabe in Unteraufgaben zu unterteilen oder um eine vorläufige Antwort iterativ zu verfeinern oder zu erweitern. Sie trägt dazu bei, die Genauigkeit und Relevanz der Antworten eines Modells zu verbessern und ermöglicht detailliertere, personalisierte Ergebnisse.

Pseudonymisierung

Der Prozess, bei dem persönliche Identifikatoren in einem Datensatz durch Platzhalterwerte ersetzt werden. Pseudonymisierung kann zum Schutz der Privatsphäre beitragen. Pseudonymisierte Daten gelten weiterhin als personenbezogene Daten.

publish/subscribe (pub/sub)

Ein Muster, das asynchrone Kommunikation zwischen Microservices ermöglicht, um die Skalierbarkeit und Reaktionsfähigkeit zu verbessern. In einem auf Microservices basierenden MES kann ein Microservice beispielsweise Ereignismeldungen in einem Kanal veröffentlichen,

den andere Microservices abonnieren können. Das System kann neue Microservices hinzufügen, ohne den Veröffentlichungsservice zu ändern.

# Q

Abfrageplan

Eine Reihe von Schritten, wie Anweisungen, die für den Zugriff auf die Daten in einem relationalen SQL-Datenbanksystem verwendet werden.

Abfrageplanregression

Wenn ein Datenbankserviceoptimierer einen weniger optimalen Plan wählt als vor einer bestimmten Änderung der Datenbankumgebung. Dies kann durch Änderungen an Statistiken, Beschränkungen, Umgebungseinstellungen, Abfrageparameter-Bindungen und Aktualisierungen der Datenbank-Engine verursacht werden.

# R

RACI-Matrix

Siehe verantwortlich, rechenschaftspflichtig, konsultiert, informiert (RACI).

LAPPEN

Siehe Erweiterte Generierung beim Abrufen.

Ransomware

Eine bösartige Software, die entwickelt wurde, um den Zugriff auf ein Computersystem oder Daten zu blockieren, bis eine Zahlung erfolgt ist.

RASCI-Matrix

Siehe verantwortlich, rechenschaftspflichtig, konsultiert, informiert (RACI).

RCAC

Siehe Zugriffskontrolle für Zeilen und Spalten.

Read Replica

Eine Kopie einer Datenbank, die nur für Lesezwecke verwendet wird. Sie können Abfragen an das Lesereplikat weiterleiten, um die Belastung auf Ihrer Primärdatenbank zu reduzieren.

neu strukturieren

Siehe 7 Rs.

Recovery Point Objective (RPO)

Die maximal zulässige Zeitspanne seit dem letzten Datenwiederherstellungspunkt. Damit wird festgelegt, was als akzeptabler Datenverlust zwischen dem letzten Wiederherstellungspunkt und der Serviceunterbrechung gilt.

Wiederherstellungszeitziel (RTO)

Die maximal zulässige Verzögerung zwischen der Betriebsunterbrechung und der Wiederherstellung des Dienstes.

Refaktorierung

Siehe 7 Rs.

Region

Eine Sammlung von AWS Ressourcen in einem geografischen Gebiet. Jeder AWS-Region ist isoliert und unabhängig von den anderen, um Fehlertoleranz, Stabilität und Belastbarkeit zu gewährleisten. Weitere Informationen finden Sie unter Geben Sie an, was AWS-Regionen Ihr Konto verwenden kann.

Regression

Eine ML-Technik, die einen numerischen Wert vorhersagt. Zum Beispiel, um das Problem „Zu welchem Preis wird dieses Haus verkauft werden?“ zu lösen Ein ML-Modell könnte ein lineares Regressionsmodell verwenden, um den Verkaufspreis eines Hauses auf der Grundlage bekannter Fakten über das Haus (z. B. die Quadratmeterzahl) vorherzusagen.

rehosten

Siehe 7 Rs.

Veröffentlichung

In einem Bereitstellungsprozess der Akt der Förderung von Änderungen an einer Produktionsumgebung.

umziehen

Siehe 7 Rs.

neue Plattform

Siehe 7 Rs.

Rückkauf

Siehe 7 Rs.

Ausfallsicherheit

Die Fähigkeit einer Anwendung, Störungen zu widerstehen oder sich von ihnen zu erholen. Hochverfügbarkeit und Notfallwiederherstellung sind häufig Überlegungen bei der Planung der Ausfallsicherheit in der. AWS Cloud Weitere Informationen finden Sie unter AWS Cloud Resilienz.

Ressourcenbasierte Richtlinie

Eine mit einer Ressource verknüpfte Richtlinie, z. B. ein Amazon-S3-Bucket, ein Endpunkt oder ein Verschlüsselungsschlüssel. Diese Art von Richtlinie legt fest, welchen Prinzipalen der Zugriff gewährt wird, welche Aktionen unterstützt werden und welche anderen Bedingungen erfüllt sein müssen.

RACI-Matrix (verantwortlich, rechenschaftspflichtig, konsultiert, informiert)

Eine Matrix, die die Rollen und Verantwortlichkeiten aller an Migrationsaktivitäten und Cloud-Operationen beteiligten Parteien definiert. Der Matrixname leitet sich von den in der Matrix definierten Zuständigkeitstypen ab: verantwortlich (R), rechenschaftspflichtig (A), konsultiert (C) und informiert (I). Der Unterstützungstyp (S) ist optional. Wenn Sie Unterstützung einbeziehen, wird die Matrix als RASCI-Matrix bezeichnet, und wenn Sie sie ausschließen, wird sie als RACI-Matrix bezeichnet.

Reaktive Kontrolle

Eine Sicherheitskontrolle, die darauf ausgelegt ist, die Behebung unerwünschter Ereignisse oder Abweichungen von Ihren Sicherheitsstandards voranzutreiben. Weitere Informationen finden Sie unter Reaktive Kontrolle in Implementieren von Sicherheitskontrollen in AWS.

Beibehaltung

Siehe 7 Rs.

zurückziehen

Siehe 7 Rs.

Retrieval Augmented Generation (RAG)

Eine generative KI-Technologie, bei der ein LLM auf eine maßgebliche Datenquelle verweist, die sich außerhalb seiner Trainingsdatenquellen befindet, bevor eine Antwort generiert wird. Ein RAG-Modell könnte beispielsweise eine semantische Suche in der Wissensdatenbank oder in benutzerdefinierten Daten einer Organisation durchführen. Weitere Informationen finden Sie unter Was ist RAG.

Drehung

Der Vorgang, bei dem ein Geheimnis regelmäßig aktualisiert wird, um es einem Angreifer zu erschweren, auf die Anmeldeinformationen zuzugreifen.

Zugriffskontrolle für Zeilen und Spalten (RCAC)

Die Verwendung einfacher, flexibler SQL-Ausdrücke mit definierten Zugriffsregeln. RCAC besteht aus Zeilenberechtigungen und Spaltenmasken.

RPO

Siehe Recovery Point Objective.

RTO

Siehe Ziel der Wiederherstellungszeit.

Runbook

Eine Reihe manueller oder automatisierter Verfahren, die zur Ausführung einer bestimmten Aufgabe erforderlich sind. Diese sind in der Regel darauf ausgelegt, sich wiederholende Operationen oder Verfahren mit hohen Fehlerquoten zu rationalisieren.

# S

SAML 2.0

Ein offener Standard, den viele Identitätsanbieter (IdPs) verwenden. Diese Funktion ermöglicht föderiertes Single Sign-On (SSO), sodass sich Benutzer bei den API-Vorgängen anmelden AWS Management Console oder die AWS API-Operationen aufrufen können, ohne dass Sie einen Benutzer in IAM für alle in Ihrer Organisation erstellen müssen. Weitere Informationen zum SAML-2.0.-basierten Verbund finden Sie unter Über den SAML-2.0-basierten Verbund in der IAM-Dokumentation.

SCADA

Siehe Aufsichtskontrolle und Datenerfassung.

SCP

Siehe Richtlinie zur Dienstkontrolle.

Secret

Interne AWS Secrets Manager, vertrauliche oder eingeschränkte Informationen, wie z. B. ein Passwort oder Benutzeranmeldedaten, die Sie in verschlüsselter Form speichern. Es besteht aus dem geheimen Wert und seinen Metadaten. Der geheime Wert kann binär, eine einzelne Zeichenfolge oder mehrere Zeichenketten sein. Weitere Informationen finden Sie unter Was ist in einem Secrets Manager Manager-Geheimnis? in der Secrets Manager Manager-Dokumentation.

Sicherheit durch Design

Ein systemtechnischer Ansatz, der die Sicherheit während des gesamten Entwicklungsprozesses berücksichtigt.

Sicherheitskontrolle

Ein technischer oder administrativer Integritätsschutz, der die Fähigkeit eines Bedrohungsakteurs, eine Schwachstelle auszunutzen, verhindert, erkennt oder einschränkt. Es gibt vier Haupttypen von Sicherheitskontrollen: präventiv, detektiv, reaktionsschnell und proaktiv.

Härtung der Sicherheit

Der Prozess, bei dem die Angriffsfläche reduziert wird, um sie widerstandsfähiger gegen Angriffe zu machen. Dies kann Aktionen wie das Entfernen von Ressourcen, die nicht mehr benötigt werden, die Implementierung der bewährten Sicherheitsmethode der Gewährung geringster Berechtigungen oder die Deaktivierung unnötiger Feature in Konfigurationsdateien umfassen.

System zur Verwaltung von Sicherheitsinformationen und Ereignissen (security information and event management – SIEM)

Tools und Services, die Systeme für das Sicherheitsinformationsmanagement (SIM) und das Management von Sicherheitsereignissen (SEM) kombinieren. Ein SIEM-System sammelt, überwacht und analysiert Daten von Servern, Netzwerken, Geräten und anderen Quellen, um Bedrohungen und Sicherheitsverletzungen zu erkennen und Warnmeldungen zu generieren.

Automatisierung von Sicherheitsreaktionen

Eine vordefinierte und programmierte Aktion, die darauf ausgelegt ist, automatisch auf ein Sicherheitsereignis zu reagieren oder es zu beheben. Diese Automatisierungen dienen als

detektive oder reaktionsschnelle Sicherheitskontrollen, die Sie bei der Implementierung bewährter AWS Sicherheitsmethoden unterstützen. Beispiele für automatisierte Antwortaktionen sind das Ändern einer VPC-Sicherheitsgruppe, das Patchen einer EC2 Amazon-Instance oder das Rotieren von Anmeldeinformationen.

Serverseitige Verschlüsselung

Verschlüsselung von Daten am Zielort durch denjenigen AWS-Service , der sie empfängt.

Service-Kontrollrichtlinie (SCP)

Eine Richtlinie, die eine zentrale Steuerung der Berechtigungen für alle Konten in einer Organisation in ermöglicht AWS Organizations. SCPs Definieren Sie Leitplanken oder legen Sie Grenzwerte für Aktionen fest, die ein Administrator an Benutzer oder Rollen delegieren kann. Sie können sie SCPs als Zulassungs- oder Ablehnungslisten verwenden, um festzulegen, welche Dienste oder Aktionen zulässig oder verboten sind. Weitere Informationen finden Sie in der AWS Organizations Dokumentation unter Richtlinien zur Dienststeuerung.

Service-Endpunkt

Die URL des Einstiegspunkts für einen AWS-Service. Sie können den Endpunkt verwenden, um programmgesteuert eine Verbindung zum Zielservice herzustellen. Weitere Informationen finden Sie unter AWS-Service -Endpunkte in der Allgemeine AWS-Referenz.

Service Level Agreement (SLA)

Eine Vereinbarung, in der klargestellt wird, was ein IT-Team seinen Kunden zu bieten verspricht, z. B. in Bezug auf Verfügbarkeit und Leistung der Services.

Service-Level-Indikator (SLI)

Eine Messung eines Leistungsaspekts eines Dienstes, z. B. seiner Fehlerrate, Verfügbarkeit oder Durchsatz.

Service-Level-Ziel (SLO)

Eine Zielkennzahl, die den Zustand eines Dienstes darstellt, gemessen anhand eines Service-Level-Indikators.

Modell der geteilten Verantwortung

Ein Modell, das die Verantwortung beschreibt, mit der Sie gemeinsam AWS für Cloud-Sicherheit und Compliance verantwortlich sind. AWS ist für die Sicherheit der Cloud verantwortlich, während Sie für die Sicherheit in der Cloud verantwortlich sind. Weitere Informationen finden Sie unter Modell der geteilten Verantwortung.

SIEM

Siehe Sicherheitsinformations- und Event-Management-System.

Single Point of Failure (SPOF)

Ein Fehler in einer einzelnen, kritischen Komponente einer Anwendung, der das System stören kann.

SLA

Siehe Service Level Agreement.

SLI

Siehe Service-Level-Indikator.

ALSO

Siehe Service-Level-Ziel.

split-and-seed Modell

Ein Muster für die Skalierung und Beschleunigung von Modernisierungsprojekten. Sobald neue Features und Produktversionen definiert werden, teilt sich das Kernteam auf, um neue Produktteams zu bilden. Dies trägt zur Skalierung der Fähigkeiten und Services Ihrer Organisation bei, verbessert die Produktivität der Entwickler und unterstützt schnelle Innovationen. Weitere Informationen finden Sie unter Schrittweiser Ansatz zur Modernisierung von Anwendungen in der AWS Cloud.

SPOTTEN

Siehe Single Point of Failure.

Sternschema

Eine Datenbank-Organisationsstruktur, die eine große Faktentabelle zum Speichern von Transaktions- oder Messdaten und eine oder mehrere kleinere dimensionale Tabellen zum Speichern von Datenattributen verwendet. Diese Struktur ist für die Verwendung in einem Data Warehouse oder für Business Intelligence-Zwecke konzipiert.

Strangler-Fig-Muster

Ein Ansatz zur Modernisierung monolithischer Systeme, bei dem die Systemfunktionen schrittweise umgeschrieben und ersetzt werden, bis das Legacy-System außer Betrieb

genommen werden kann. Dieses Muster verwendet die Analogie einer Feigenrebe, die zu einem etablierten Baum heranwächst und schließlich ihren Wirt überwindet und ersetzt. Das Muster wurde eingeführt von Martin Fowler als Möglichkeit, Risiken beim Umschreiben monolithischer Systeme zu managen. Ein Beispiel für die Anwendung dieses Musters finden Sie unter Schrittweises Modernisieren älterer Microsoft ASP.NET (ASMX)-Webservices mithilfe von Containern und Amazon API Gateway.

Subnetz

Ein Bereich von IP-Adressen in Ihrer VPC. Ein Subnetz muss sich in einer einzigen Availability Zone befinden.

Aufsichtskontrolle und Datenerfassung (SCADA)

In der Fertigung ein System, das Hardware und Software zur Überwachung von Sachanlagen und Produktionsabläufen verwendet.

Symmetrische Verschlüsselung

Ein Verschlüsselungsalgorithmus, der denselben Schlüssel zum Verschlüsseln und Entschlüsseln der Daten verwendet.

synthetisches Testen

Testen eines Systems auf eine Weise, die Benutzerinteraktionen simuliert, um potenzielle Probleme zu erkennen oder die Leistung zu überwachen. Sie können Amazon CloudWatch Synthetics verwenden, um diese Tests zu erstellen.

Systemaufforderung

Eine Technik, mit der einem LLM Kontext, Anweisungen oder Richtlinien zur Verfügung gestellt werden, um sein Verhalten zu steuern. Systemaufforderungen helfen dabei, den Kontext festzulegen und Regeln für Interaktionen mit Benutzern festzulegen.

# T

tags

Schlüssel-Wert-Paare, die als Metadaten für die Organisation Ihrer Ressourcen dienen. AWS Mit Tags können Sie Ressourcen verwalten, identifizieren, organisieren, suchen und filtern. Weitere Informationen finden Sie unter Markieren Ihrer AWS -Ressourcen.

Zielvariable

Der Wert, den Sie in überwachtem ML vorhersagen möchten. Dies wird auch als Ergebnisvariable bezeichnet. In einer Fertigungsumgebung könnte die Zielvariable beispielsweise ein Produktfehler sein.

Aufgabenliste

Ein Tool, das verwendet wird, um den Fortschritt anhand eines Runbooks zu verfolgen. Eine Aufgabenliste enthält eine Übersicht über das Runbook und eine Liste mit allgemeinen Aufgaben, die erledigt werden müssen. Für jede allgemeine Aufgabe werden der geschätzte Zeitaufwand, der Eigentümer und der Fortschritt angegeben.

Testumgebungen

Siehe Umgebung.

Training

Daten für Ihr ML-Modell bereitstellen, aus denen es lernen kann. Die Trainingsdaten müssen die richtige Antwort enthalten. Der Lernalgorithmus findet Muster in den Trainingsdaten, die die Attribute der Input-Daten dem Ziel (die Antwort, die Sie voraussagen möchten) zuordnen. Es gibt ein ML-Modell aus, das diese Muster erfasst. Sie können dann das ML-Modell verwenden, um Voraussagen für neue Daten zu erhalten, bei denen Sie das Ziel nicht kennen.

Transit-Gateway

Ein Netzwerk-Transit-Hub, über den Sie Ihre Netzwerke VPCs und Ihre lokalen Netzwerke miteinander verbinden können. Weitere Informationen finden Sie in der Dokumentation unter Was ist ein Transit-Gateway. AWS Transit Gateway

Stammbasierter Workflow

Ein Ansatz, bei dem Entwickler Feature lokal in einem Feature-Zweig erstellen und testen und diese Änderungen dann im Hauptzweig zusammenführen. Der Hauptzweig wird dann sequentiell für die Entwicklungs-, Vorproduktions- und Produktionsumgebungen erstellt.

Vertrauenswürdiger Zugriff

Gewährung von Berechtigungen für einen Dienst, den Sie angeben, um Aufgaben in Ihrer Organisation AWS Organizations und in deren Konten in Ihrem Namen auszuführen. Der vertrauenswürdige Service erstellt in jedem Konto eine mit dem Service verknüpfte Rolle, wenn diese Rolle benötigt wird, um Verwaltungsaufgaben für Sie auszuführen. Weitere Informationen

finden Sie in der AWS Organizations Dokumentation unter Verwendung AWS Organizations mit anderen AWS Diensten.

Optimieren

Aspekte Ihres Trainingsprozesses ändern, um die Genauigkeit des ML-Modells zu verbessern. Sie können das ML-Modell z. B. trainieren, indem Sie einen Beschriftungssatz generieren, Beschriftungen hinzufügen und diese Schritte dann mehrmals unter verschiedenen Einstellungen wiederholen, um das Modell zu optimieren.

Zwei-Pizzen-Team

Ein kleines DevOps Team, das Sie mit zwei Pizzen ernähren können. Eine Teamgröße von zwei Pizzen gewährleistet die bestmögliche Gelegenheit zur Zusammenarbeit bei der Softwareentwicklung.

# U

Unsicherheit

Ein Konzept, das sich auf ungenaue, unvollständige oder unbekannte Informationen bezieht, die die Zuverlässigkeit von prädiktiven ML-Modellen untergraben können. Es gibt zwei Arten von Unsicherheit: Epistemische Unsicherheit wird durch begrenzte, unvollständige Daten verursacht, wohingegen aleatorische Unsicherheit durch Rauschen und Randomisierung verursacht wird, die in den Daten liegt. Weitere Informationen finden Sie im Leitfaden Quantifizieren der Unsicherheit in Deep-Learning-Systemen.

undifferenzierte Aufgaben

Diese Arbeit wird auch als Schwerstarbeit bezeichnet. Dabei handelt es sich um Arbeiten, die zwar für die Erstellung und den Betrieb einer Anwendung erforderlich sind, aber dem Endbenutzer keinen direkten Mehrwert bieten oder keinen Wettbewerbsvorteil bieten. Beispiele für undifferenzierte Aufgaben sind Beschaffung, Wartung und Kapazitätsplanung.

höhere Umgebungen

Siehe Umgebung.

# V

Vacuuming

Ein Vorgang zur Datenbankwartung, bei dem die Datenbank nach inkrementellen Aktualisierungen bereinigt wird, um Speicherplatz zurückzugewinnen und die Leistung zu verbessern.

Versionskontrolle

Prozesse und Tools zur Nachverfolgung von Änderungen, z. B. Änderungen am Quellcode in einem Repository.

VPC-Peering

Eine Verbindung zwischen zwei VPCs , die es Ihnen ermöglicht, den Verkehr mithilfe privater IP-Adressen weiterzuleiten. Weitere Informationen finden Sie unter Was ist VPC-Peering? in der Amazon-VPC-Dokumentation.

Schwachstelle

Ein Software- oder Hardwarefehler, der die Sicherheit des Systems beeinträchtigt.

# W

Warmer Cache

Ein Puffer-Cache, der aktuelle, relevante Daten enthält, auf die häufig zugegriffen wird. Die Datenbank-Instance kann aus dem Puffer-Cache lesen, was schneller ist als das Lesen aus dem Hauptspeicher oder von der Festplatte.

warme Daten

Daten, auf die selten zugegriffen wird. Bei der Abfrage dieser Art von Daten sind mäßig langsame Abfragen in der Regel akzeptabel.

Fensterfunktion

Eine SQL-Funktion, die eine Berechnung für eine Gruppe von Zeilen durchführt, die sich in irgendeiner Weise auf den aktuellen Datensatz beziehen. Fensterfunktionen sind nützlich für die Verarbeitung von Aufgaben wie die Berechnung eines gleitenden Durchschnitts oder für den Zugriff auf den Wert von Zeilen auf der Grundlage der relativen Position der aktuellen Zeile.

Workload

Ein Workload ist eine Sammlung von Ressourcen und Code, die einen Unternehmenswert bietet, wie z. B. eine kundenorientierte Anwendung oder ein Backend-Prozess.

Workstream

Funktionsgruppen in einem Migrationsprojekt, die für eine bestimmte Reihe von Aufgaben verantwortlich sind. Jeder Workstream ist unabhängig, unterstützt aber die anderen Workstreams im Projekt. Der Portfolio-Workstream ist beispielsweise für die Priorisierung von Anwendungen, die Wellenplanung und die Erfassung von Migrationsmetadaten verantwortlich. Der Portfolio-Workstream liefert diese Komponenten an den Migrations-Workstream, der dann die Server und Anwendungen migriert.

WURM

Sehen Sie einmal schreiben, viele lesen.

WQF

Siehe AWS Workload-Qualifizierungsrahmen.

einmal schreiben, viele lesen (WORM)

Ein Speichermodell, das Daten ein einziges Mal schreibt und verhindert, dass die Daten gelöscht oder geändert werden. Autorisierte Benutzer können die Daten so oft wie nötig lesen, aber sie können sie nicht ändern. Diese Datenspeicherinfrastruktur gilt als unveränderlich.

# Z

Zero-Day-Exploit

Ein Angriff, in der Regel Malware, der eine Zero-Day-Sicherheitslücke ausnutzt.

Zero-Day-Sicherheitslücke

Ein unfehlbarer Fehler oder eine Sicherheitslücke in einem Produktionssystem. Bedrohungsakteure können diese Art von Sicherheitslücke nutzen, um das System anzugreifen. Entwickler werden aufgrund des Angriffs häufig auf die Sicherheitsanfälligkeit aufmerksam.

Zero-Shot-Aufforderung

Bereitstellung von Anweisungen für die Ausführung einer Aufgabe an einen LLM, jedoch ohne Beispiele (Schnappschüsse), die ihm als Orientierungshilfe dienen könnten. Der LLM muss sein

vortrainiertes Wissen einsetzen, um die Aufgabe zu bewältigen. Die Effektivität von Zero-Shot Prompting hängt von der Komplexität der Aufgabe und der Qualität der Aufforderung ab. Siehe auch Few-Shot-Prompting.

Zombie-Anwendung

Eine Anwendung, deren durchschnittliche CPU- und Arbeitsspeichernutzung unter 5 Prozent liegt. In einem Migrationsprojekt ist es üblich, diese Anwendungen außer Betrieb zu nehmen.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.