Meilleures pratiques d'utilisation du fournisseur Terraform AWS

# AWS Conseils prescriptifs

# AWS Conseils prescriptifs: Meilleures pratiques d'utilisation du fournisseur Terraform AWS

# Table of Contents

# Bonnes pratiques d'utilisation du fournisseur Terraform AWS

Michael Begin, DevOps consultant principal, Amazon Web Services (AWS)

Mai 2024 (historique du document)

La gestion de l'infrastructure en tant que code (IaC) avec Terraform activé AWS offre des avantages importants tels que l'amélioration de la cohérence, de la sécurité et de l'agilité. Cependant, à mesure que votre configuration Terraform augmente en taille et en complexité, il devient essentiel de suivre les meilleures pratiques pour éviter les pièges.

Ce guide fournit les meilleures pratiques recommandées pour utiliser le AWS fournisseur Terraform à partir de. HashiCorp Il vous explique comment gérer correctement les versions, les contrôles de sécurité, les backends distants, la structure de base de code et les fournisseurs de communauté pour optimiser Terraform on. AWS Chaque section aborde plus en détail les spécificités de l'application de ces meilleures pratiques :

- Sécurité
- Backends
- Structure et organisation de la base de code
- AWS Gestion des versions du fournisseur
- Modules communautaires

## Objectifs

Ce guide vous aide à acquérir des connaissances opérationnelles sur le AWS fournisseur Terraform et répond aux objectifs commerciaux suivants que vous pouvez atteindre en suivant les meilleures pratiques IaC en matière de sécurité, de fiabilité, de conformité et de productivité des développeurs.

- Améliorez la qualité et la cohérence du code d'infrastructure entre les projets Terraform.
- Accélérez l'intégration des développeurs et leur capacité à contribuer au code de l'infrastructure.
- Améliorez l'agilité de l'entreprise grâce à des changements d'infrastructure plus rapides.
- Réduisez les erreurs et les temps d'arrêt liés aux modifications de l'infrastructure.
- Optimisez les coûts d'infrastructure en suivant les meilleures pratiques de l'IaC.
- Renforcez votre posture de sécurité globale grâce à la mise en œuvre des meilleures pratiques.

# Public cible

Le public cible de ce guide comprend les responsables techniques et les responsables qui supervisent les équipes qui utilisent Terraform pour iAc on. AWS Parmi les autres lecteurs potentiels figurent les ingénieurs en infrastructure, DevOps les ingénieurs, les architectes de solutions et les développeurs qui utilisent activement Terraform pour gérer AWS l'infrastructure.

Le respect de ces meilleures pratiques vous permettra de gagner du temps et de tirer parti des avantages de l'IaC pour ces rôles.

# Présentation

Les fournisseurs Terraform sont des plugins qui permettent à Terraform d'interagir avec différentes API. Le Terraform AWS Provider est le plugin officiel pour gérer l' AWS infrastructure en tant que code (iAc) avec Terraform. Il traduit la syntaxe Terraform en appels AWS d'API pour créer, lire, mettre à jour et supprimer AWS des ressources.

Le AWS fournisseur gère l'authentification, la traduction de la syntaxe Terraform en appels d' AWS API et le provisionnement des ressources dans. AWS Vous utilisez un bloc de `provider` code Terraform pour configurer le plugin du fournisseur que Terraform utilise pour interagir avec l'API. AWS Vous pouvez configurer plusieurs blocs AWS de fournisseurs pour gérer les ressources dans différentes Comptes AWS régions.

Voici un exemple de configuration Terraform qui utilise plusieurs blocs AWS Provider avec des alias pour gérer une base de données Amazon Relational Database Service (Amazon RDS) qui possède une réplique dans une région et un compte différents. Les fournisseurs principaux et secondaires assument des rôles AWS Identity and Access Management (IAM) différents :

```
# 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
}
```

Dans cet exemple :

- Le premier `provider` bloc configure le AWS fournisseur principal de la `us-west-1` région avec l'alias`primary`.
- Le deuxième `provider` bloc configure un AWS fournisseur secondaire dans la `us-east-1` région avec l'alias`replica`. Ce fournisseur est utilisé pour créer une réplique en lecture de la base de données principale dans une région et un compte différents. Le `assume_role` bloc est utilisé pour assumer un rôle IAM dans le compte de réplique. `role_arn`Spécifie le nom de ressource Amazon (ARN) du rôle IAM à assumer et `session_name` constitue un identifiant unique pour la session Terraform.
- La `aws_db_instance.primary` ressource crée la base de données Amazon RDS principale en utilisant le `primary` fournisseur de la `us-west-1` région.
- La `aws_db_instance.read_replica` ressource crée une réplique en lecture de la base de données principale de la `us-east-1` région à l'aide du `replica` fournisseur. L'`replicate_source_db`attribut fait référence à l'ID de la `primary` base de données.

# Bonnes pratiques de sécurité

La gestion correcte de l'authentification, des contrôles d'accès et de la sécurité est essentielle pour une utilisation sécurisée du fournisseur Terraform AWS . Cette section décrit les meilleures pratiques en matière de :

- Rôles et autorisations IAM pour l'accès avec le moindre privilège
- Sécurisation des informations d'identification pour empêcher tout accès non autorisé aux AWS comptes et aux ressources
- Chiffrement à distance pour protéger les données sensibles
- Analyse de l'infrastructure et du code source pour identifier les erreurs de configuration
- Contrôles d'accès pour le stockage à distance
- Application de la politique Sentinel pour mettre en place des garde-fous en matière de gouvernance

Le respect de ces meilleures pratiques permet de renforcer votre posture de sécurité lorsque vous utilisez Terraform pour gérer AWS l'infrastructure.

## Respectez le principe du moindre privilège

Le moindre privilège est un principe de sécurité fondamental qui consiste à n'accorder que les autorisations minimales requises pour qu'un utilisateur, un processus ou un système exécute les fonctions prévues. Il s'agit d'un concept fondamental du contrôle d'accès et d'une mesure préventive contre les accès non autorisés et les violations de données potentielles.

Le principe du moindre privilège est souligné à plusieurs reprises dans cette section car il est directement lié à la manière dont Terraform authentifie et exécute des actions contre des fournisseurs de cloud tels que. AWS

Lorsque vous utilisez Terraform pour provisionner et gérer AWS des ressources, il agit pour le compte d'une entité (utilisateur ou rôle) qui a besoin des autorisations appropriées pour effectuer des appels d'API. Ne pas respecter le moindre privilège présente des risques de sécurité majeurs :

- Si Terraform dispose d'autorisations excessives au-delà de ce qui est nécessaire, une mauvaise configuration involontaire peut entraîner des modifications ou des suppressions indésirables.

- Les autorisations d'accès trop permissives augmentent l'impact si les fichiers d'état ou les informations d'identification de Terraform sont compromis.
- Le fait de ne pas respecter le moindre privilège va à l'encontre des meilleures pratiques de sécurité et des exigences de conformité réglementaires relatives à l'octroi d'un accès minimal requis.

# Utilisation des rôles IAM

Utilisez des rôles IAM plutôt que des utilisateurs IAM dans la mesure du possible pour améliorer la sécurité avec le fournisseur AWS Terraform. Les rôles IAM fournissent des informations d'identification de sécurité temporaires qui changent automatiquement, ce qui élimine le besoin de gérer des clés d'accès à long terme. Les rôles offrent également des contrôles d'accès précis par le biais de politiques IAM.

## Accordez l'accès avec le moindre privilège en utilisant les politiques IAM

Élaborez soigneusement des politiques IAM pour garantir que les rôles et les utilisateurs ne disposent que du minimum d'autorisations requis pour leur charge de travail. Commencez par une politique vide et ajoutez de manière itérative les services et actions autorisés. Pour ce faire, procédez comme suit :

- Activez IAM Access Analyzer pour évaluer les politiques et mettre en évidence les autorisations non utilisées qui peuvent être supprimées.
- Passez en revue manuellement les politiques afin de supprimer toutes les fonctionnalités qui ne sont pas essentielles au regard de la responsabilité prévue du rôle.
- Utilisez les variables de politique IAM et les balises pour simplifier la gestion des autorisations.

Des politiques bien conçues accordent un accès juste suffisant pour accomplir les responsabilités liées à la charge de travail, rien de plus. Définissez des actions au niveau de l'opération et autorisez les appels uniquement aux API requises sur des ressources spécifiques.

Le respect de cette bonne pratique réduit l'ampleur de l'impact et respecte les principes de sécurité fondamentaux que sont la séparation des tâches et le moindre privilège d'accès. Commencez l'accès strict et ouvert progressivement selon les besoins, au lieu de commencer par ouvrir et d'essayer de restreindre l'accès ultérieurement.

## Assumer des rôles IAM pour l'authentification locale

Lorsque vous exécutez Terraform localement, évitez de configurer des clés d'accès statiques. Utilisez plutôt les rôles IAM pour accorder temporairement un accès privilégié sans exposer les informations d'identification à long terme.

Tout d'abord, créez un rôle IAM avec les autorisations minimales nécessaires et ajoutez une relation de confiance qui permet à votre compte utilisateur ou à votre identité fédérée d'assumer le rôle IAM. Cela autorise l'utilisation temporaire du rôle.

Exemple de politique de relation de confiance :

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

Exécutez ensuite la AWS CLI commande aws sts assume-role pour récupérer les informations d'identification de courte durée pour le rôle. Ces informations d'identification sont généralement valides pendant une heure.

AWS CLI exemple de commande :

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

La sortie de la commande contient une clé d'accès, une clé secrète et un jeton de session que vous pouvez utiliser pour vous authentifier auprès de 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"
    }
}
```

Le AWS fournisseur peut également gérer automatiquement l'acceptation du rôle.

Exemple de configuration du fournisseur pour assumer un rôle IAM :

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

Cela accorde des privilèges élevés strictement pendant la durée de la session Terraform. Les clés temporaires ne peuvent pas être divulguées car elles expirent automatiquement après la durée maximale de la session.

Les principaux avantages de cette bonne pratique incluent une sécurité améliorée par rapport aux clés d'accès à longue durée de vie, des contrôles d'accès précis sur le rôle pour le moins de privilèges et la possibilité de révoquer facilement l'accès en modifiant les autorisations du rôle. En utilisant les rôles IAM, vous évitez également d'avoir à stocker directement les secrets localement dans des scripts ou sur le disque, ce qui vous permet de partager la configuration Terraform en toute sécurité au sein d'une équipe.

## Utiliser les rôles IAM pour l'authentification Amazon EC2

Lorsque vous exécutez Terraform à partir d'instances Amazon Elastic Compute Cloud (Amazon EC2), évitez de stocker des informations d'identification à long terme localement. Utilisez plutôt

les rôles IAM et les profils d'instance pour accorder automatiquement les autorisations du moindre privilège.

Créez d'abord un rôle IAM avec les autorisations minimales et attribuez-le au profil d'instance. Le profil d'instance permet aux instances EC2 d'hériter des autorisations définies dans le rôle. Lancez ensuite les instances en spécifiant ce profil d'instance. L'instance s'authentifiera via le rôle attaché.

Avant d'exécuter des opérations Terraform, vérifiez que le rôle est présent dans les métadonnées de l'instance pour confirmer que les informations d'identification ont été héritées avec succès.

```
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/
```

Cette approche évite de coder en dur des AWS clés permanentes dans des scripts ou dans la configuration Terraform au sein de l'instance. Les informations d'identification temporaires sont mises à la disposition de Terraform de manière transparente via le rôle et le profil de l'instance.

Les principaux avantages de cette bonne pratique incluent l'amélioration de la sécurité par rapport aux informations d'identification à long terme, la réduction des frais de gestion des informations d'identification et la cohérence entre les environnements de développement, de test et de production. L'authentification des rôles IAM simplifie les exécutions de Terraform à partir d'instances EC2 tout en appliquant l'accès au moindre privilège.

## Utiliser des informations d'identification dynamiques pour les espaces de travail HCP Terraform

HCP Terraform est un service géré fourni par HashiCorp qui aide les équipes à utiliser Terraform pour fournir et gérer l'infrastructure de plusieurs projets et environnements. Lorsque vous exécutez Terraform dans HCP Terraform, utilisez des informations d'identification dynamiques pour simplifier et sécuriser l'authentification. AWS Terraform échange automatiquement des informations d'identification temporaires à chaque exécution sans avoir à assumer le rôle IAM.

Les avantages incluent une rotation plus facile des secrets, une gestion centralisée des informations d'identification dans les espaces de travail, des autorisations de moindre privilège et l'élimination des clés codées en dur. L'utilisation de clés éphémères hachées améliore la sécurité par rapport aux clés d'accès à longue durée de vie.

### Utiliser les rôles IAM dans AWS CodeBuild

Dans AWS CodeBuild, exécutez vos builds à l'aide d'un rôle IAM attribué au CodeBuild projet. Cela permet à chaque build d'hériter automatiquement des informations d'identification temporaires du rôle au lieu d'utiliser des clés à long terme.

### Exécutez GitHub des actions à distance sur HCP Terraform

Configurez GitHub les flux de travail Actions pour exécuter Terraform à distance sur les espaces de travail HCP Terraform. Misez sur des informations d'identification dynamiques et sur le verrouillage d'état à distance plutôt que sur la gestion des GitHub secrets.

### Utiliser GitHub les actions avec OIDC et configurer l'action AWS Credentials

Utilisez la norme OpenID Connect (OIDC) pour fédérer l'identité GitHub Actions via IAM. Utilisez l'action Configurer les AWS informations d'identification pour échanger le GitHub jeton contre des AWS informations d'identification temporaires sans avoir besoin de clés d'accès à long terme.

### À utiliser GitLab avec OIDC et le AWS CLI

Utilisez la norme OIDC pour fédérer les GitLab identités via IAM pour un accès temporaire. En vous appuyant sur OIDC, vous évitez d'avoir à gérer directement les clés d' AWS accès à long terme qu'il contient. GitLab Les informations d'identification sont échangées just-in-time, ce qui améliore la sécurité. Les utilisateurs obtiennent également le moindre privilège d'accès en fonction des autorisations associées au rôle IAM.

## Utilisez des utilisateurs IAM uniques grâce aux outils d'automatisation existants

Si vous disposez d'outils et de scripts d'automatisation qui ne prennent pas en charge nativement l'utilisation des rôles IAM, vous pouvez créer des utilisateurs IAM individuels pour accorder un accès programmatique. Le principe du moindre privilège s'applique toujours. Minimisez les autorisations liées aux politiques et utilisez des rôles distincts pour chaque pipeline ou script. Au fur et à mesure que vous migrez vers des outils ou des outils plus modernes, commencez à prendre en charge les rôles de manière native et passez progressivement à eux.

> **Warning**
>
> Les utilisateurs IAM disposent d'informations d'identification à long terme, ce qui présente un risque de sécurité. Pour atténuer ce risque, nous vous recommandons de n'octroyer à ces utilisateurs que les autorisations dont ils ont besoin pour effectuer la tâche et de supprimer ces utilisateurs lorsqu'ils ne sont plus nécessaires.

### Utilisez le plugin Jenkins AWS Credentials

Utilisez le plugin AWS Credentials de Jenkins pour configurer et injecter des AWS informations d'identification de manière centralisée dans les builds de manière dynamique. Cela évite de vérifier les secrets dans le contrôle de source.

## Surveillez, validez et optimisez en permanence le moindre privilège

Au fil du temps, des autorisations supplémentaires peuvent être accordées, qui peuvent dépasser les politiques minimales requises. Analysez en permanence les accès pour identifier et supprimer les droits inutiles.

### Surveillez en permanence l'utilisation des clés d'accès

Si vous ne pouvez pas éviter d'utiliser des clés d'accès, utilisez les rapports d'identification IAM pour trouver les clés d'accès non utilisées datant de plus de 90 jours, et révoquez les clés inactives à la fois sur les comptes utilisateurs et sur les rôles de machine. Demandez aux administrateurs de confirmer manuellement la suppression des clés pour les employés actifs et les systèmes.

La surveillance de l'utilisation des clés vous permet d'optimiser les autorisations, car vous pouvez identifier et supprimer les autorisations non utilisées. Lorsque vous suivez cette bonne pratique en matière de rotation des clés d'accès, cela limite la durée de vie des informations d'identification et impose l'accès avec le moindre privilège.

AWS fournit plusieurs services et fonctionnalités que vous pouvez utiliser pour configurer des alertes et des notifications pour les administrateurs. Voici quelques options :

- AWS Config: vous pouvez utiliser des AWS Config règles pour évaluer les paramètres de configuration de vos AWS ressources, notamment les clés d'accès IAM. Vous pouvez créer des règles personnalisées pour vérifier des conditions spécifiques, telles que des clés d'accès non utilisées datant de plus d'un certain nombre de jours. En cas de violation d'une règle, AWS Config

vous pouvez lancer une évaluation à des fins de correction ou envoyer des notifications à une rubrique Amazon Simple Notification Service (Amazon SNS).

- AWS Security Hub: Security Hub fournit une vue complète du niveau de sécurité de votre AWS compte et peut vous aider à détecter et à vous signaler les problèmes de sécurité potentiels, notamment les clés d'accès IAM inutilisées ou inactives. Security Hub peut s'intégrer à Amazon EventBridge et Amazon SNS ou envoyer des notifications AWS Chatbot aux administrateurs.
- AWS Lambda: Les fonctions Lambda peuvent être appelées par divers événements, notamment Amazon CloudWatch Events ou AWS Config des règles. Vous pouvez écrire des fonctions Lambda personnalisées pour évaluer l'utilisation des clés d'accès IAM, effectuer des vérifications supplémentaires et envoyer des notifications à l'aide de services tels qu'Amazon SNS ou. AWS Chatbot

### Validez en permanence les politiques IAM

Utilisez IAM Access Analyzer pour évaluer les politiques associées aux rôles et identifier les services non utilisés ou les actions excédentaires accordées. Mettez en œuvre des examens d'accès périodiques pour vérifier manuellement que les politiques correspondent aux exigences actuelles.

Comparez la politique existante avec la politique générée par IAM Access Analyzer et supprimez toutes les autorisations inutiles. Vous devez également fournir des rapports aux utilisateurs et révoquer automatiquement les autorisations non utilisées après un délai de grâce. Cela permet de garantir que les politiques minimales restent en vigueur.

La révocation proactive et fréquente des accès obsolètes minimise les informations d'identification susceptibles d'être menacées en cas de violation. L'automatisation assure une hygiène des accréditations et une optimisation des autorisations durables et à long terme. Le respect de cette bonne pratique limite la portée de l'impact en appliquant de manière proactive le moindre privilège aux AWS identités et aux ressources.

## Stockage d'état à distance sécurisé

Le stockage d'état à distance fait référence au stockage du fichier d'état Terraform à distance plutôt que localement sur la machine sur laquelle Terraform s'exécute. Le fichier d'état est crucial car il permet de suivre les ressources fournies par Terraform et leurs métadonnées.

L'incapacité à sécuriser l'état distant peut entraîner de graves problèmes tels que la perte de données d'état, l'incapacité de gérer l'infrastructure, la suppression involontaire de ressources et la divulgation

d'informations sensibles susceptibles de se trouver dans le fichier d'état. Pour cette raison, la sécurisation du stockage à distance est cruciale pour une utilisation de Terraform en production.

### Activez le chiffrement et les contrôles d'accès

Utilisez le chiffrement côté serveur (SSE) Amazon Simple Storage Service (Amazon S3) pour chiffrer l'état distant au repos.

### Limitez l'accès direct aux flux de travail collaboratifs

- Structurez les flux de travail de collaboration dans HCP Terraform ou dans un pipeline CI/CD au sein de votre référentiel Git afin de limiter l'accès direct à l'état.
- Appuyez-vous sur les pull requests, exécutez les approbations, les vérifications des politiques et les notifications pour coordonner les changements.

Le respect de ces directives permet de sécuriser les attributs sensibles des ressources et d'éviter les conflits avec les modifications apportées par les membres de l'équipe. Le chiffrement et les protections d'accès strictes aident à réduire la surface d'attaque, et les flux de travail collaboratifs favorisent la productivité.

## Utiliser AWS Secrets Manager

Terraform possède de nombreuses ressources et sources de données qui stockent des valeurs secrètes en texte clair dans le fichier d'état. Évitez de stocker des secrets dans l'état, utilisez-les plutôt AWS Secrets Manager.

Au lieu d'essayer de chiffrer manuellement des valeurs sensibles, comptez sur le support intégré de Terraform pour la gestion des états sensibles. Lorsque vous exportez des valeurs sensibles vers la sortie, assurez-vous qu'elles sont marquées comme sensibles.

## Analysez en continu l'infrastructure et le code source

Analysez de manière proactive l'infrastructure et le code source en permanence pour détecter les risques tels que des informations d'identification exposées ou des erreurs de configuration afin de renforcer votre posture de sécurité. Répondez rapidement aux résultats en reconfigurant ou en appliquant des correctifs aux ressources.

## Utiliser AWS les services pour le scan dynamique

Utilisez des outils AWS natifs tels qu'Amazon Inspector AWS Security Hub, Amazon Detective et Amazon GuardDuty pour surveiller l'infrastructure provisionnée sur l'ensemble des comptes et des régions. Planifiez des scans récurrents dans Security Hub pour suivre le déploiement et l'évolution de la configuration. Scannez les instances EC2, les fonctions Lambda, les conteneurs, les compartiments S3 et d'autres ressources.

## Réaliser une analyse statique

Intégrez des analyseurs statiques tels que Checkov directement dans les pipelines CI/CD pour scanner le code de configuration Terraform (HCL) et identifier les risques de manière préventive avant le déploiement. Cela permet de déplacer les contrôles de sécurité à un stade antérieur du processus de développement (c'est ce que l'on appelle le déplacement vers la gauche) et d'éviter toute mauvaise configuration de l'infrastructure.

## Garantir une correction rapide

Pour tous les résultats de l'analyse, veillez à une correction rapide en mettant à jour la configuration de Terraform, en appliquant des correctifs ou en reconfigurant les ressources manuellement, le cas échéant. Réduisez les niveaux de risque en vous attaquant aux causes profondes.

L'utilisation à la fois de l'analyse de l'infrastructure et de l'analyse du code fournit des informations en couches sur les configurations Terraform, les ressources allouées et le code des applications. Cela maximise la couverture des risques et la conformité grâce à des contrôles préventifs, détectifs et réactifs, tout en intégrant la sécurité plus tôt dans le cycle de vie du développement logiciel (SDLC).

# Appliquer les contrôles des politiques

Utilisez des cadres de code tels que les politiques HashiCorp Sentinel pour fournir des garanties de gouvernance et des modèles standardisés pour le provisionnement de l'infrastructure avec Terraform.

Les politiques Sentinel peuvent définir des exigences ou des restrictions relatives à la configuration de Terraform afin de s'aligner sur les normes organisationnelles et les meilleures pratiques. Par exemple, vous pouvez utiliser les politiques Sentinel pour :

- Exiger l'apposition de balises sur toutes les ressources.
- Limitez les types d'instances à une liste approuvée.

- Appliquez les variables obligatoires.
- Empêcher la destruction des ressources de production.

L'intégration de contrôles de politique dans les cycles de vie des configurations de Terraform permet une application proactive des normes et des directives d'architecture. Sentinel fournit une logique de politique partagée qui permet d'accélérer le développement tout en empêchant les pratiques non approuvées.

# Bonnes pratiques en matière de backend

L'utilisation d'un backend distant approprié pour stocker votre fichier d'état est essentielle pour permettre la collaboration, garantir l'intégrité des fichiers d'état grâce au verrouillage, fournir une sauvegarde et une restauration fiables, intégrer les flux de travail CI/CD et tirer parti des fonctionnalités avancées de sécurité, de gouvernance et de gestion proposées par les services gérés tels que HCP Terraform.

Terraform prend en charge différents types de backend tels que Kubernetes, Consul et HTTP. HashiCorp Cependant, ce guide se concentre sur Amazon S3, qui est une solution de backend optimale pour la plupart des AWS utilisateurs.

En tant que service de stockage d'objets entièrement géré offrant une durabilité et une disponibilité élevées, Amazon S3 fournit un backend sécurisé, évolutif et peu coûteux pour gérer l'état de Terraform sur. AWS L'empreinte mondiale et la résilience d'Amazon S3 dépassent ce que la plupart des équipes peuvent atteindre en gérant elles-mêmes le stockage d'état. En outre, l'intégration native AWS aux contrôles d'accès, aux options de chiffrement, aux fonctionnalités de gestion des versions et à d'autres services fait d'Amazon S3 un choix de backend pratique.

Ce guide ne fournit pas de conseils sur le backend pour d'autres solutions telles que Kubernetes ou Consul, car le public cible principal est constitué de clients. AWS Pour les équipes qui sont pleinement impliquées AWS Cloud, Amazon S3 est généralement le choix idéal par rapport aux clusters Kubernetes ou HashiCorp Consul. La simplicité, la résilience et AWS l'intégration étroite du stockage d'état Amazon S3 constituent une base optimale pour la plupart des utilisateurs qui suivent les AWS meilleures pratiques. Les équipes peuvent tirer parti de la durabilité, des protections de sauvegarde et de la disponibilité des AWS services pour garantir la haute résilience de Terraform State à distance.

Le respect des recommandations du backend de cette section permettra de créer des bases de code Terraform plus collaboratives tout en limitant l'impact des erreurs ou des modifications non autorisées. En mettant en œuvre un backend distant bien conçu, les équipes peuvent optimiser les flux de travail Terraform.

Bonnes pratiques :

- Utiliser Amazon S3 pour le stockage à distance
- Facilitez la collaboration en équipe
- Séparez les backends pour chaque environnement

- Surveillez activement l'activité des États distants

# Utiliser Amazon S3 pour le stockage à distance

Le stockage à distance de l'état Terraform dans Amazon S3 et la mise en œuvre du verrouillage d'état et du contrôle de cohérence à l'aide d'Amazon DynamoDB offrent des avantages majeurs par rapport au stockage de fichiers local. L'état à distance permet la collaboration en équipe, le suivi des modifications, les protections de sauvegarde et le verrouillage à distance pour une sécurité accrue.

L'utilisation d'Amazon S3 avec la classe de stockage S3 Standard (par défaut) au lieu d'un stockage local éphémère ou de solutions autogérées assure une durabilité de 99,999999999 % et des protections de disponibilité à 99,99 % afin d'éviter toute perte accidentelle de données d'état. AWS les services gérés tels qu'Amazon S3 et DynamoDB fournissent des accords de niveau de service (SLA) qui dépassent ce que la plupart des entreprises peuvent obtenir lorsqu'elles gèrent elles-mêmes le stockage. Appuyez-vous sur ces protections pour garder les backends distants accessibles.

## Activer le verrouillage d'état à distance

Le verrouillage DynamoDB restreint l'accès à l'état afin d'empêcher les opérations d'écriture simultanées. Cela empêche les modifications simultanées de la part de plusieurs utilisateurs et réduit les erreurs.

Exemple de configuration du backend avec verrouillage d'état :

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

## Activez le contrôle de version et les sauvegardes automatiques

Pour une protection supplémentaire, activez le contrôle automatique des versions et les sauvegardes en les utilisant AWS Backup sur les backends Amazon S3. Le versionnement préserve toutes les

versions précédentes de l'état chaque fois que des modifications sont apportées. Il vous permet également de restaurer des instantanés d'état de fonctionnement antérieurs si nécessaire pour annuler des modifications indésirables ou récupérer après un accident.

### Restaurez les versions précédentes si nécessaire

Les compartiments d'état Amazon S3 versionnés permettent d'annuler facilement les modifications en restaurant un précédent instantané d'état connu en bon état. Cela permet de se protéger contre les modifications accidentelles et fournit des fonctionnalités de sauvegarde supplémentaires.

### Utilisez HCP Terraform

HCP Terraform fournit une alternative backend entièrement gérée à la configuration de votre propre stockage d'état. HCP Terraform gère automatiquement le stockage sécurisé de l'état et du chiffrement tout en débloquant des fonctionnalités supplémentaires.

Lorsque vous utilisez HCP Terraform, l'état est stocké à distance par défaut, ce qui permet le partage et le verrouillage des états au sein de votre organisation. Des contrôles de politique détaillés vous aident à restreindre l'accès à l'État et les modifications.

Les fonctionnalités supplémentaires incluent les intégrations de contrôle de version, les garanties de politique, l'automatisation des flux de travail, la gestion des variables et les intégrations d'authentification unique avec SAML. Vous pouvez également utiliser la politique Sentinel comme code pour mettre en œuvre des contrôles de gouvernance.

Bien que HCP Terraform nécessite l'utilisation d'une plateforme logicielle en tant que service (SaaS), pour de nombreuses équipes, les avantages liés à la sécurité, aux contrôles d'accès, aux contrôles automatisés des politiques et aux fonctionnalités de collaboration en font un choix optimal par rapport au stockage d'état autogéré avec Amazon S3 ou DynamoDB.

L'intégration facile avec des services tels que GitHub et GitLab avec une configuration mineure attire également les utilisateurs qui adoptent pleinement les outils cloud et SaaS pour améliorer les flux de travail d'équipe.

## Facilitez la collaboration en équipe

Utilisez des backends distants pour partager les données d'état entre tous les membres de votre équipe Terraform. Cela facilite la collaboration car cela donne à l'ensemble de l'équipe une visibilité

sur les modifications de l'infrastructure. Les protocoles backend partagés combinés à la transparence de l'historique des états simplifient la gestion interne des modifications. Toutes les modifications de l'infrastructure passent par le pipeline établi, ce qui accroît l'agilité de l'entreprise au sein de l'entreprise.

### Améliorez la responsabilisation en utilisant AWS CloudTrail

AWS CloudTrail Intégrez le compartiment Amazon S3 pour capturer les appels d'API effectués vers le compartiment d'état. Filtrez CloudTrail les événements à suivre`PutObject`, `DeleteObject`, ainsi que les autres appels pertinents.

CloudTrail les journaux indiquent l' AWS identité du principal qui a effectué chaque appel d'API pour un changement d'état. L'identité de l'utilisateur peut être associée à un compte de machine ou aux membres de l'équipe qui interagissent avec le stockage principal.

Combinez CloudTrail les journaux avec la gestion des versions d'état d'Amazon S3 pour lier les modifications d'infrastructure au principal qui les a appliquées. En analysant plusieurs révisions, vous pouvez attribuer les mises à jour au compte de la machine ou au membre de l'équipe responsable.

En cas de modification involontaire ou perturbatrice, le contrôle des versions d'état fournit des fonctionnalités de restauration. CloudTrail retrace le changement jusqu'à l'utilisateur afin que vous puissiez discuter des améliorations préventives.

Nous vous recommandons également d'appliquer les autorisations IAM afin de limiter l'accès aux compartiments d'État. Dans l'ensemble, le contrôle des versions et la CloudTrail surveillance de S3 prennent en charge l'audit des modifications de l'infrastructure. Les équipes bénéficient de capacités de responsabilisation, de transparence et d'audit améliorées dans l'historique de l'état de Terraform.

## Séparez les backends pour chaque environnement

Utilisez des backends Terraform distincts pour chaque environnement d'application. Des backends distincts isolent l'état entre le développement, le test et la production.

### Réduire la portée de l'impact

L'isolation de l'état permet de garantir que les modifications apportées aux environnements inférieurs n'ont pas d'impact sur l'infrastructure de production. Les accidents ou les expériences dans les environnements de développement et de test ont un impact limité.

## Limiter l'accès à la production

Verrouillez les autorisations pour le backend de l'état de production afin que la plupart des utilisateurs puissent y accéder en lecture seule. Limitez le nombre de personnes autorisées à modifier l'infrastructure de production en fonction du pipeline CI/CD et à briser les rôles de verre.

## Simplifier les contrôles d'accès

La gestion des autorisations au niveau du backend simplifie le contrôle d'accès entre les environnements. L'utilisation de compartiments S3 distincts pour chaque application et environnement signifie que des autorisations de lecture ou d'écriture étendues peuvent être accordées sur l'ensemble des compartiments principaux.

## Évitez les espaces de travail partagés

Bien que vous puissiez utiliser les espaces de travail Terraform pour séparer les états entre les environnements, des backends distincts offrent une meilleure isolation. Si vous partagez des espaces de travail, les accidents peuvent toujours avoir un impact sur plusieurs environnements.

L'isolation complète des backends de l'environnement permet de minimiser l'impact d'une défaillance ou d'une violation unique. Des backends distincts alignent également les contrôles d'accès sur le niveau de sensibilité de l'environnement. Par exemple, vous pouvez fournir une protection en écriture pour l'environnement de production et un accès plus large pour les environnements de développement et de test.

# Surveillez activement l'activité des États distants

La surveillance continue de l'activité à distance est essentielle pour détecter rapidement les problèmes potentiels. Recherchez les déblocages, les modifications ou les tentatives d'accès anormaux.

## Recevez des alertes en cas de déverrouillage suspect

La plupart des changements d'état devraient passer par des pipelines CI/CD. Générez des alertes si les déblocages d'état se produisent directement via les postes de travail des développeurs, ce qui pourrait signaler des modifications non autorisées ou non testées.

# Surveiller les tentatives d'accès

Les échecs d'authentification sur les compartiments d'état peuvent indiquer une activité de reconnaissance. Notez si plusieurs comptes tentent d'accéder à l'état ou si des adresses IP inhabituelles apparaissent, ce qui indique que les informations d'identification ont été compromises.

# Bonnes pratiques pour la structure et l'organisation de la base de code

Une structure et une organisation de base de code appropriées sont essentielles car l'utilisation de Terraform augmente au sein des grandes équipes et entreprises. Une base de code bien conçue permet une collaboration à grande échelle tout en améliorant la maintenabilité.

Cette section fournit des recommandations sur la modularité de Terraform, les conventions de dénomination, la documentation et les normes de codage qui favorisent la qualité et la cohérence.

Les conseils incluent la division de la configuration en modules réutilisables par environnement et par composant, l'établissement de conventions de dénomination à l'aide de préfixes et de suffixes, la documentation des modules et l'explication claire des entrées et des sorties, et l'application de règles de formatage cohérentes à l'aide de contrôles de style automatisés.

Les meilleures pratiques supplémentaires concernent l'organisation logique des modules et des ressources dans une hiérarchie structurée, le catalogage des modules publics et privés dans la documentation et l'extraction des détails de mise en œuvre inutiles dans les modules afin de simplifier l'utilisation.

En mettant en œuvre des directives relatives à la structure de base du code relatives à la modularité, à la documentation, aux normes et à l'organisation logique, vous pouvez favoriser une large collaboration entre les équipes tout en garantissant la maintenance de Terraform à mesure que l'utilisation se répand au sein de l'organisation. En appliquant les conventions et les normes, vous pouvez éviter la complexité d'une base de code fragmentée.

Bonnes pratiques :

- Implémenter une structure de référentiel standard
- Structure pour la modularité
- Respectez les conventions de dénomination
- Utiliser les ressources relatives aux pièces jointes
- Utiliser les balises par défaut
- Répondez aux exigences du registre Terraform
- Utiliser les sources de modules recommandées
- Respectez les normes de codage

# Implémenter une structure de référentiel standard

Nous vous recommandons d'implémenter la disposition de référentiel suivante. La normalisation de ces pratiques de cohérence entre les modules améliore la découvrabilité, la transparence, l'organisation et la fiabilité tout en permettant la réutilisation dans de nombreuses configurations Terraform.

- Module ou répertoire racine : il doit être le point d'entrée principal pour les modules racine et réutilisables de Terraform et devrait être unique. Si votre architecture est plus complexe, vous pouvez utiliser des modules imbriqués pour créer des abstractions légères. Cela vous permet de décrire l'infrastructure en termes d'architecture plutôt que directement en termes d'objets physiques.
- README : le module racine et tous les modules imbriqués doivent contenir des fichiers README. Ce fichier doit être nomméREADME.md. Il doit contenir une description du module et de son utilisation. Si vous souhaitez inclure un exemple d'utilisation de ce module avec d'autres ressources, placez-le dans un `examples` répertoire. Envisagez d'inclure un diagramme qui décrit les ressources d'infrastructure que le module peut créer et leurs relations. Utilisez terraform-docs pour générer automatiquement les entrées ou les sorties du module.
- main.tf : C'est le point d'entrée principal. Pour un module simple, toutes les ressources peuvent être créées dans ce fichier. Pour un module complexe, la création de ressources peut être répartie sur plusieurs fichiers, mais tous les appels de modules imbriqués doivent figurer dans le `main.tf` fichier.
- variables.tf et outputs.tf : ces fichiers contiennent les déclarations des variables et des sorties. Toutes les variables et sorties doivent être décrites en une ou deux phrases expliquant leur objectif. Ces descriptions sont utilisées pour la documentation. Pour plus d'informations, consultez la HashiCorp documentation relative à la configuration des variables et à la configuration de sortie.
  - Toutes les variables doivent avoir un type défini.
  - La déclaration de variable peut également inclure un argument par défaut. Si la déclaration inclut un argument par défaut, la variable est considérée comme facultative et la valeur par défaut est utilisée si vous ne définissez aucune valeur lorsque vous appelez le module ou exécutez Terraform. L'argument par défaut nécessite une valeur littérale et ne peut pas faire référence à d'autres objets de la configuration. Pour rendre une variable obligatoire, omettez une valeur par défaut dans la déclaration de variable et déterminez si le réglage `nullable = false` est judicieux.
  - Pour les variables dont les valeurs sont indépendantes de l'environnement (telles que`disk_size`), fournissez des valeurs par défaut.

  - Pour les variables qui ont des valeurs spécifiques à l'environnement (telles que`project_id`), ne fournissez pas de valeurs par défaut. Dans ce cas, le module d'appel doit fournir des valeurs significatives.
  - Utiliser des valeurs par défaut vides pour des variables telles que des chaînes ou des listes vides uniquement lorsque le fait de laisser la variable vide est une préférence valide que les API sous-jacentes ne rejettent pas.
  - Utilisez judicieusement les variables. Paramétrez les valeurs uniquement si elles doivent varier pour chaque instance ou environnement. Lorsque vous décidez d'exposer une variable, assurez-vous de disposer d'un cas d'utilisation concret pour modifier cette variable. S'il n'y a qu'une faible chance qu'une variable soit nécessaire, ne l'exposez pas.
    - L'ajout d'une variable avec une valeur par défaut est rétrocompatible.
    - La suppression d'une variable est rétroincompatible.
    - Dans les cas où un littéral est réutilisé à plusieurs endroits, vous devez utiliser une valeur locale sans l'exposer en tant que variable.
  - Ne transmettez pas les sorties directement via les variables d'entrée, car cela les empêcherait d'être correctement ajoutées au graphe de dépendance. Pour garantir la création de dépendances implicites, assurez-vous que les sorties font référence aux attributs des ressources. Au lieu de référencer directement une variable d'entrée pour une instance, transmettez l'attribut.
- locals.tf : Ce fichier contient des valeurs locales qui attribuent un nom à une expression. Un nom peut donc être utilisé plusieurs fois dans un module au lieu de répéter l'expression. Les valeurs locales sont similaires aux variables locales temporaires d'une fonction. Les expressions contenues dans les valeurs locales ne se limitent pas à des constantes littérales ; elles peuvent également faire référence à d'autres valeurs du module, notamment des variables, des attributs de ressources ou d'autres valeurs locales, afin de les combiner.
- providers.tf : Ce fichier contient le bloc terraform et les blocs fournisseur. `provider`les blocs doivent être déclarés uniquement dans les modules racines par les consommateurs de modules.

  Si vous utilisez HCP Terraform, ajoutez également un bloc cloud vide. Le `cloud` bloc doit être entièrement configuré via des variables d'environnement et des informations d'identification de variables d'environnement dans le cadre d'un pipeline CI/CD.
- versions.tf : Ce fichier contient le bloc required_providers. Tous les modules Terraform doivent déclarer les fournisseurs dont ils ont besoin pour que Terraform puisse installer et utiliser ces fournisseurs.

- data.tf : Pour une configuration simple, placez les sources de données à côté des ressources qui les référencent. Par exemple, si vous récupérez une image à utiliser lors du lancement d'une instance, placez-la à côté de l'instance au lieu de collecter les ressources de données dans leur propre fichier. Si le nombre de sources de données devient trop important, envisagez de les déplacer vers un `data.tf` fichier dédié.
- Fichiers .tfvars : pour les modules root, vous pouvez fournir des variables non sensibles à l'aide d'un fichier. `.tfvars` Pour des raisons de cohérence, nommez les fichiers variables`terraform.tfvars`. Placez les valeurs communes à la racine du référentiel et les valeurs spécifiques à l'environnement dans le `envs/` dossier.
- Modules imbriqués : les modules imbriqués doivent exister dans le `modules/` sous-répertoire. Tout module imbriqué doté d'un `README.md` est considéré comme utilisable par un utilisateur externe. Si un `README.md` n'existe pas, le module est considéré pour un usage interne uniquement. Les modules imbriqués doivent être utilisés pour diviser un comportement complexe en plusieurs petits modules que les utilisateurs peuvent sélectionner avec soin.

  Si le module racine inclut des appels à des modules imbriqués, ces appels doivent utiliser des chemins relatifs, de `./modules/sample-module` manière à ce que Terraform les considère comme faisant partie du même référentiel ou package au lieu de les télécharger à nouveau séparément.

  Si un dépôt ou un package contient plusieurs modules imbriqués, ils devraient idéalement être composables par l'appelant au lieu de s'appeler directement et de créer une arborescence de modules profondément imbriquée.
- Exemples : des exemples d'utilisation d'un module réutilisable doivent se trouver dans le `examples/` sous-répertoire à la racine du dépôt. Pour chaque exemple, vous pouvez ajouter un fichier README pour expliquer l'objectif et l'utilisation de l'exemple. Les exemples de sous-modules doivent également être placés dans le `examples/` répertoire racine.

  Comme les exemples sont souvent copiés dans d'autres référentiels à des fins de personnalisation, la source des blocs de modules doit être définie sur l'adresse qu'un appelant externe utiliserait, et non sur un chemin relatif.
- Fichiers nommés par le service : les utilisateurs souhaitent souvent séparer les ressources Terraform par service dans plusieurs fichiers. Cette pratique devrait être découragée autant que possible et les ressources devraient `main.tf` plutôt être définies. Toutefois, si un ensemble de ressources (par exemple, les rôles et les politiques IAM) dépasse 150 lignes, il est raisonnable de le diviser en ses propres fichiers, tels que`iam.tf`. Dans le cas contraire, tout le code de ressource doit être défini dans le`main.tf`.

- Scripts personnalisés : utilisez des scripts uniquement lorsque cela est nécessaire. Terraform ne prend pas en compte ni ne gère l'état des ressources créées par le biais de scripts. Utilisez des scripts personnalisés uniquement lorsque les ressources Terraform ne prennent pas en charge le comportement souhaité. Placez les scripts personnalisés appelés par Terraform dans un `scripts/` répertoire.
- Scripts d'assistance : organisez les scripts d'assistance qui ne sont pas appelés par Terraform dans un répertoire. `helpers/` Documentez des scripts d'assistance dans le `README.md` fichier avec des explications et des exemples d'invocations. Si les scripts d'assistance acceptent des arguments, fournissez une vérification des arguments et une `--help` sortie.
- Fichiers statiques : les fichiers statiques auxquels Terraform fait référence mais n'exécute pas (tels que les scripts de démarrage chargés sur les instances EC2) doivent être organisés dans un répertoire. `files/` Placez les longs documents dans des fichiers externes, séparés de leur HCL. Référencez-les à l'aide de la fonction file ().
- Modèles : pour les fichiers que la fonction Templatefile de Terraform lit, utilisez l'extension de fichier. `.tftpl` Les modèles doivent être placés dans un `templates/` répertoire.

## Structure du module racine

Terraform s'exécute toujours dans le contexte d'un seul module racine. Une configuration Terraform complète comprend un module racine et l'arborescence des modules enfants (qui inclut les modules appelés par le module racine, tous les modules appelés par ces modules, etc.).

Exemple de base de la disposition du module racine Terraform :

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

```
### terraform.tfvars
### variables.tf
### versions.tf
```

## Structure de module réutilisable

Les modules réutilisables suivent les mêmes concepts que les modules racines. Pour définir un module, créez un nouveau répertoire pour celui-ci et placez-y les `.tf` fichiers, comme vous définiriez un module racine. Terraform peut charger des modules soit à partir de chemins relatifs locaux, soit à partir de référentiels distants. Si vous vous attendez à ce qu'un module soit réutilisé par de nombreuses configurations, placez-le dans son propre référentiel de contrôle de version. Il est important de garder l'arborescence des modules relativement plate pour faciliter la réutilisation des modules dans différentes combinaisons.

Exemple de base de disposition de module réutilisable Terraform :

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

```
### main.tf
### outputs.tf
### README.md
### variables.tf
### versions.tf
```

# Structure pour la modularité

En principe, vous pouvez combiner toutes les ressources et autres constructions dans un module, mais l'utilisation excessive de modules imbriqués et réutilisables peut rendre votre configuration globale de Terraform plus difficile à comprendre et à maintenir. Utilisez donc ces modules avec modération.

Lorsque cela est logique, divisez votre configuration en modules réutilisables qui augmentent le niveau d'abstraction en décrivant un nouveau concept de votre architecture construit à partir de types de ressources.

Lorsque vous modularisez votre infrastructure en définitions réutilisables, privilégiez des ensembles logiques de ressources plutôt que des composants individuels ou des collections trop complexes.

## N'encapsulez pas des ressources uniques

Vous ne devez pas créer de modules qui constituent des enveloppes fines autour d'autres types de ressources uniques. Si vous ne parvenez pas à trouver un nom différent du nom du type de ressource principal qu'il contient, votre module ne crée probablement pas de nouvelle abstraction, cela ajoute une complexité inutile. Utilisez plutôt le type de ressource directement dans le module d'appel.

## Encapsuler les relations logiques

Regroupez des ensembles de ressources connexes telles que les bases du réseau, les niveaux de données, les contrôles de sécurité et les applications. Un module réutilisable doit encapsuler des éléments d'infrastructure qui fonctionnent ensemble pour activer une fonctionnalité.

## Maintenir un héritage stable

Lorsque vous imbriquez des modules dans des sous-répertoires, évitez d'aller à plus d'un ou deux niveaux de profondeur. Les structures d'héritage profondément imbriquées compliquent les

configurations et le dépannage. Les modules doivent s'appuyer sur d'autres modules, et non créer des tunnels à travers eux.

En concentrant les modules sur des groupements de ressources logiques qui représentent des modèles d'architecture, les équipes peuvent rapidement configurer des bases d'infrastructure fiables. Équilibrez l'abstraction sans suringénierie ou simplification.

## Ressources de référence dans les sorties

Pour chaque ressource définie dans un module réutilisable, incluez au moins une sortie qui fait référence à la ressource. Les variables et les sorties vous permettent de déduire les dépendances entre les modules et les ressources. Sans aucune sortie, les utilisateurs ne peuvent pas commander correctement votre module par rapport à leurs configurations Terraform.

Des modules bien structurés qui assurent la cohérence de l'environnement, des regroupements axés sur des objectifs et des références de ressources exportées permettent une collaboration Terraform à grande échelle à l'échelle de l'organisation. Les équipes peuvent assembler l'infrastructure à partir de blocs de construction réutilisables.

## Ne configurez pas les fournisseurs

Bien que les modules partagés héritent des fournisseurs des modules appelants, les modules ne doivent pas configurer eux-mêmes les paramètres des fournisseurs. Évitez de spécifier des blocs de configuration du fournisseur dans les modules. Cette configuration ne doit être déclarée qu'une seule fois globalement.

## Déclarer les fournisseurs requis

Bien que les configurations des fournisseurs soient partagées entre les modules, les modules partagés doivent également déclarer leurs propres exigences en matière de fournisseur. Cette pratique permet à Terraform de s'assurer qu'il existe une seule version du fournisseur compatible avec tous les modules de la configuration et de spécifier l'adresse source qui sert d'identifiant global (indépendant du module) au fournisseur. Cependant, les exigences du fournisseur spécifiques au module ne spécifient aucun des paramètres de configuration qui déterminent les points de terminaison distants auxquels le fournisseur aura accès, tels qu'un. Région AWS

En déclarant les exigences de version et en évitant la configuration codée en dur des fournisseurs, les modules assurent la portabilité et la réutilisabilité entre les configurations Terraform à l'aide de fournisseurs partagés.

Pour les modules partagés, définissez les versions de fournisseur minimales requises dans un bloc required_providers dans. `versions.tf`

Pour déclarer qu'un module nécessite une version particulière du AWS fournisseur, utilisez un `required_providers` bloc à l'intérieur d'un `terraform` bloc :

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

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

Si un module partagé ne prend en charge qu'une version spécifique du AWS fournisseur, utilisez l'opérateur de contrainte pessimiste (~>), qui autorise uniquement l'incrémentation du composant de version le plus à droite :

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

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

Dans cet exemple, `~> 4.0` autorise l'installation de `4.57.1` et `4.67.0` mais non`5.0.0`. Pour plus d'informations, consultez la section Syntaxe des contraintes de version dans la HashiCorp documentation.

## Respectez les conventions de dénomination

Des noms clairs et descriptifs simplifient votre compréhension des relations entre les ressources du module et l'objectif des valeurs de configuration. La cohérence avec les directives de style améliore la lisibilité tant pour les utilisateurs du module que pour les responsables de la maintenance.

## Suivez les directives relatives à la dénomination des ressources

- Utilisez snake_case (où les termes minuscules sont séparés par des traits de soulignement) pour que tous les noms de ressources correspondent aux normes de style Terraform. Cette pratique garantit la cohérence avec la convention de dénomination des types de ressources, des types de sources de données et des autres valeurs prédéfinies. Cette convention ne s'applique pas aux arguments relatifs aux noms.
- Pour simplifier les références à une ressource unique en son genre (par exemple, un équilibreur de charge unique pour un module entier), nommez la ressource `main` ou `this` pour plus de clarté.
- Utilisez des noms significatifs qui décrivent l'objectif et le contexte de la ressource, et qui permettent de différencier les ressources similaires (par exemple, `primary` pour la base de données principale et `read_replica` pour une réplique en lecture de la base de données).
- Utilisez des noms au singulier et non au pluriel.
- Ne répétez pas le type de ressource dans le nom de la ressource.

## Suivez les directives relatives à la dénomination des variables

- Ajoutez des unités aux noms des entrées, des variables locales et des sorties qui représentent des valeurs numériques telles que la taille du disque ou la taille de la RAM (par exemple, `ram_size_gb` pour la taille de la RAM en gigaoctets). Cette pratique indique clairement l'unité d'entrée attendue pour les responsables de la configuration.
- Utilisez des unités binaires telles que MiB et GiB pour les tailles de stockage, et des unités décimales telles que Mo ou Go pour les autres métriques.
- Donnez aux variables booléennes des noms positifs tels que. `enable_external_access`

# Utiliser les ressources relatives aux pièces jointes

Certaines ressources contiennent des pseudo-ressources intégrées sous forme d'attributs. Dans la mesure du possible, évitez d'utiliser ces attributs de ressource intégrés et utilisez plutôt la ressource unique pour associer cette pseudo-ressource. Ces relations entre les ressources peuvent entraîner des cause-and-effect problèmes propres à chaque ressource.

En utilisant un attribut intégré (évitez ce modèle) :

```
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"]
  }
}
```

Utilisation des ressources de pièces jointes (de préférence) :

```
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
}
```

# Utiliser les balises par défaut

Attribuez des balises à toutes les ressources qui peuvent accepter des balises. Le AWS fournisseur Terraform dispose d'une source de données aws_default_tags que vous devez utiliser dans le module racine.

Envisagez d'ajouter les balises nécessaires à toutes les ressources créées par un module Terraform. Voici une liste des balises que vous pouvez associer :

- Nom : nom de ressource lisible par l'homme
- AppId: ID de l'application qui utilise la ressource
- AppRole: fonction technique de la ressource ; par exemple, « serveur Web » ou « base de données »
- AppPurpose: objectif commercial de la ressource ; par exemple, « interface utilisateur frontale » ou « processeur de paiement »
- Environnement : environnement logiciel, tel que dev, test ou prod
- Projet : Les projets qui utilisent la ressource
- CostCenter: Qui facturer pour l'utilisation des ressources

## Répondez aux exigences du registre Terraform

Un référentiel de modules doit répondre à toutes les exigences suivantes pour pouvoir être publié dans un registre Terraform.

Vous devez toujours respecter ces exigences, même si vous ne prévoyez pas de publier le module dans un registre à court terme. Ce faisant, vous pouvez publier le module dans un registre ultérieurement sans avoir à modifier la configuration et la structure du référentiel.

- Nom du référentiel : pour un référentiel de modules, utilisez le nom en trois parties`terraform-aws-<NAME>`, où `<NAME>` reflète le type d'infrastructure géré par le module. Le `<NAME>` segment peut contenir des traits d'union supplémentaires (par exemple,`terraform-aws-iam-terraform-roles`).
- Structure de module standard : le module doit respecter la structure de référentiel standard. Cela permet au registre d'inspecter votre module et de générer de la documentation, de suivre l'utilisation des ressources, etc.
  - Après avoir créé le dépôt Git, copiez les fichiers du module à la racine du dépôt. Nous vous recommandons de placer chaque module destiné à être réutilisable à la racine de son propre référentiel, mais vous pouvez également référencer des modules à partir de sous-répertoires.
  - Si vous utilisez HCP Terraform, publiez les modules destinés à être partagés dans le registre de votre organisation. Le registre gère les téléchargements et contrôle l'accès à l'aide de jetons

d'API HCP Terraform, de sorte que les consommateurs n'ont pas besoin d'accéder au référentiel source du module, même lorsqu'ils exécutent Terraform depuis la ligne de commande.

- Emplacement et autorisations : le référentiel doit se trouver dans l'un de vos fournisseurs de système de contrôle de version (VCS) configuré, et le compte utilisateur HCP Terraform VCS doit disposer d'un accès administrateur au référentiel. Le registre a besoin d'un accès administrateur pour créer les webhooks permettant d'importer de nouvelles versions de modules.
- balises x.y.z pour les versions : au moins une balise de version doit être présente pour que vous puissiez publier un module. Le registre utilise des balises de version pour identifier les versions des modules. Les noms des balises de version doivent utiliser le versionnement sémantique, que vous pouvez éventuellement préfixer par un v (par exemple, `v1.1.0` et). `1.1.0` Le registre ignore les balises qui ne ressemblent pas à des numéros de version. Pour plus d'informations sur les modules de publication, consultez la documentation Terraform.

Pour plus d'informations, consultez Préparation d'un référentiel de modules dans la documentation Terraform.

## Utiliser les sources de modules recommandées

Terraform utilise l'`source`argument d'un bloc de module pour rechercher et télécharger le code source d'un module enfant.

Nous vous recommandons d'utiliser des chemins locaux pour les modules étroitement liés dont l'objectif principal est de factoriser les éléments de code répétés, et d'utiliser un registre de modules Terraform natif ou un fournisseur VCS pour les modules destinés à être partagés par plusieurs configurations.

Les exemples suivants illustrent les types de sources les plus courants et les plus recommandés pour le partage de modules. Les modules de registre prennent en charge le versionnement. Vous devez toujours fournir une version spécifique, comme indiqué dans les exemples suivants.

### Registre

Registre Terraform :

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

```
  ...

}
```

En épinglant les hachages de validation, vous pouvez éviter de dériver des registres publics vulnérables aux attaques de la chaîne d'approvisionnement.

HCP Terraform :

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

  ...

  enable_karpenter = true
}
```

Terraform Entreprise :

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

  ...

  enable_karpenter = true
}
```

## Fournisseurs de VCS

Les fournisseurs de VCS soutiennent l'`ref`argument en faveur de la sélection d'une révision spécifique, comme le montrent les exemples suivants.

GitHub (HTTPS) :

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

  ...

  enable_karpenter = true
```

```
}
```

Dépôt Git générique (HTTPS) :

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

  ...

  enable_karpenter = true
}
```

Dépôt Git générique (SSH) :

> **Warning**
>
> Vous devez configurer les informations d'identification pour accéder aux référentiels privés.

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

  ...

  enable_karpenter = true
}
```

# Respectez les normes de codage

Appliquez des règles et des styles de formatage Terraform cohérents à tous les fichiers de configuration. Appliquez les normes en utilisant des contrôles de style automatisés dans les pipelines CI/CD. Lorsque vous intégrez les meilleures pratiques de codage dans les flux de travail d'équipe, les configurations restent lisibles, maintenables et collaboratives, car leur utilisation se répand largement au sein de l'organisation.

## Suivez les directives de style

- Formatez tous les fichiers Terraform (`.tf`fichiers) avec la commande terraform fmt conformément aux normes de style. HashiCorp

- Utilisez la commande terraform validate pour vérifier la syntaxe et la structure de votre configuration.
- Analysez statiquement la qualité du code à l'aide de TFlint. Ce linter vérifie les meilleures pratiques de Terraform au-delà du simple formatage et échoue aux builds lorsqu'il rencontre des erreurs.

## Configurer les hooks de pré-validation

Configurez des hooks de pré-validation côté client qui exécutent`terraform fmt`, `tflint`et `checkov`, et d'autres analyses de code et vérifications de style avant d'autoriser les validations. Cette pratique vous permet de valider la conformité aux normes plus tôt dans les flux de travail des développeurs.

Utilisez des frameworks de pré-validation tels que le pré-commit pour ajouter le linting, le formatage et le scan de code Terraform comme points d'ancrage sur votre machine locale. Les hooks s'exécutent à chaque validation Git et échouent si les vérifications ne sont pas validées.

Le transfert des contrôles de style et de qualité vers des hooks de pré-validation locaux fournit un feedback rapide aux développeurs avant l'introduction des modifications. Les normes font partie du flux de travail de codage.

# Bonnes pratiques pour la gestion des versions des AWS fournisseurs

La gestion soigneuse des versions du AWS fournisseur et des modules Terraform associés est essentielle à la stabilité. Cette section décrit les meilleures pratiques relatives aux contraintes de version et aux mises à niveau.

Bonnes pratiques :

- Ajouter des vérifications de version automatisées
- Surveillez les nouvelles versions
- Contribuez aux fournisseurs

## Ajouter des vérifications de version automatisées

Ajoutez des vérifications de version pour les fournisseurs Terraform dans vos pipelines CI/CD afin de valider l'épinglage des versions et d'échouer les builds si la version n'est pas définie.

- Ajoutez des contrôles TFlint dans les pipelines CI/CD pour rechercher les versions des fournisseurs pour lesquelles aucune contrainte de version majeure ou mineure n'est définie. Utilisez le plugin TFlint ruleset pour Terraform AWS Provider, qui fournit des règles pour détecter d'éventuelles erreurs et vérifie les meilleures pratiques en matière de ressources. AWS
- Fail CI exécute des exécutions qui détectent les versions non épinglées des fournisseurs afin d'empêcher les mises à niveau implicites d'atteindre le stade de production.

## Surveillez les nouvelles versions

- Surveillez les notes de publication et les flux du journal des modifications des fournisseurs. Recevez des notifications sur les nouvelles versions majeures/mineures.
- Évaluez les notes de publication pour détecter les modifications potentiellement importantes et évaluez leur impact sur votre infrastructure existante.
- Mettez d'abord à niveau les versions mineures dans les environnements hors production pour les valider avant de mettre à jour l'environnement de production.

En automatisant les vérifications de version dans les pipelines et en surveillant les nouvelles versions, vous pouvez détecter rapidement les mises à niveau non prises en charge et donner à vos équipes le temps d'évaluer l'impact des nouvelles versions majeures/mineures avant de mettre à jour les environnements de production.

## Contribuez aux fournisseurs

Contribuez activement au HashiCorp AWS fournisseur en signalant les défauts ou en demandant des fonctionnalités en cas de GitHub problème :

- Ouvrez les problèmes bien documentés dans le référentiel AWS Provider pour détailler les bogues que vous avez rencontrés ou les fonctionnalités manquantes. Fournissez des étapes reproductibles.
- Demandez et votez sur des améliorations visant à étendre les capacités du AWS fournisseur en matière de gestion de nouveaux services.
- Référencez les pull requests émises lorsque vous proposez des correctifs ou des améliorations à apporter aux fournisseurs. Lien vers les problèmes connexes.
- Suivez les directives de contribution du référentiel pour les conventions de codage, les normes de test et la documentation.

En redonnant aux fournisseurs auxquels vous faites appel, vous pouvez contribuer directement à leur feuille de route et contribuer à améliorer leur qualité et leurs capacités pour tous les utilisateurs.

# Bonnes pratiques pour les modules communautaires

L'utilisation efficace des modules est essentielle pour gérer les configurations complexes de Terraform et promouvoir la réutilisation. Cette section fournit les meilleures pratiques concernant les modules communautaires, les dépendances, les sources, l'abstraction et les contributions.

Bonnes pratiques :

- Découvrez les modules communautaires
- Comprendre les dépendances
- Utilisez des sources fiables
- Contribuez aux modules communautaires

## Découvrez les modules communautaires

Recherchez dans le registre Terraform et dans d'autres sources les AWS modules existants susceptibles de résoudre votre cas d'utilisation avant de créer un nouveau module. GitHub Recherchez les options populaires qui ont été mises à jour récemment et sont activement maintenues.

### Utiliser des variables pour la personnalisation

Lorsque vous utilisez des modules communautaires, transmettez les entrées par le biais de variables au lieu de forker ou de modifier directement le code source. Remplacez les valeurs par défaut si nécessaire au lieu de modifier les composants internes du module.

Le forking doit se limiter à apporter des correctifs ou des fonctionnalités au module d'origine au profit de l'ensemble de la communauté.

## Comprendre les dépendances

Avant d'utiliser le module, consultez son code source et sa documentation pour identifier les dépendances :

- Fournisseurs requis : notez les versions de AWS Kubernetes ou des autres fournisseurs requis par le module.

- Modules imbriqués : recherchez d'autres modules utilisés en interne qui introduisent des dépendances en cascade.
- Sources de données externes : notez les API, les plugins personnalisés ou les dépendances d'infrastructure sur lesquels repose le module.

En cartographiant l'arborescence complète des dépendances directes et indirectes, vous pouvez éviter les surprises lorsque vous utilisez le module.

## Utilisez des sources fiables

L'approvisionnement en modules Terraform auprès d'éditeurs non vérifiés ou inconnus présente un risque important. N'utilisez que des modules provenant de sources fiables.

- Privilégiez les modules certifiés issus du registre Terraform publiés par des créateurs vérifiés tels que nos partenaires AWS . HashiCorp
- Pour les modules personnalisés, consultez l'historique de l'éditeur, les niveaux de support et la réputation d'utilisation, même si le module provient de votre propre organisation.

En interdisant les modules provenant de sources inconnues ou non vérifiées, vous pouvez réduire le risque d'injection de vulnérabilités ou de problèmes de maintenance dans votre code.

### S’abonner aux notifications

Abonnez-vous aux notifications concernant les nouvelles versions de modules de la part d'éditeurs de confiance :

- Surveillez les référentiels de GitHub modules pour recevoir des alertes sur les nouvelles versions du module.
- Surveillez les blogs des éditeurs et les journaux des modifications pour les mises à jour.
- Recevez des notifications proactives pour les nouvelles versions provenant de sources vérifiées et bien notées au lieu d'introduire implicitement des mises à jour.

La consommation de modules provenant uniquement de sources fiables et le suivi des modifications garantissent stabilité et sécurité. Les modules approuvés améliorent la productivité tout en minimisant les risques liés à la chaîne d'approvisionnement.

# Contribuez aux modules communautaires

Soumettez des correctifs et des améliorations pour les modules communautaires hébergés dans GitHub :

- Ouvrez des pull requests sur les modules pour corriger les défauts ou les limites que vous rencontrez lors de votre utilisation.
- Demandez que de nouvelles configurations conformes aux meilleures pratiques soient ajoutées aux modules OSS existants en créant des problèmes.

La contribution aux modules communautaires améliore les modèles réutilisables et codifiés pour tous les praticiens de Terraform.

# FAQ

Q. Pourquoi se concentrer sur le AWS fournisseur ?

R. Le AWS fournisseur est l'un des fournisseurs les plus utilisés et les plus complexes pour le provisionnement de l'infrastructure dans Terraform. Le respect de ces meilleures pratiques aide les utilisateurs à optimiser leur utilisation du fournisseur pour l' AWS environnement.

Q. Je suis nouveau sur Terraform. Puis-je utiliser ce guide ?

R. Le guide s'adresse aux nouveaux utilisateurs de Terraform ainsi qu'aux praticiens plus avancés qui souhaitent améliorer leurs compétences. Les pratiques améliorent les flux de travail pour les utilisateurs à tous les stades de l'apprentissage.

Q. Quelles sont les meilleures pratiques clés couvertes ?

R. Les meilleures pratiques clés incluent l'utilisation des rôles IAM plutôt que des clés d'accès, l'épinglage des versions, l'intégration de tests automatisés, le verrouillage d'état à distance, la rotation des informations d'identification, la contribution aux fournisseurs et l'organisation logique des bases de code.

Q. Où puis-je en savoir plus sur Terraform ?

R. La section Ressources comprend des liens vers la documentation officielle de HashiCorp Terraform et les forums communautaires. Utilisez les liens pour en savoir plus sur les flux de travail Terraform avancés.

# Étapes suivantes

Voici quelques étapes potentielles à suivre après avoir lu ce guide :

- Si vous disposez d'une base de code Terraform existante, passez en revue votre configuration et identifiez les domaines susceptibles d'être améliorés en fonction des recommandations fournies dans ce guide. Par exemple, passez en revue les meilleures pratiques relatives à la mise en œuvre de backends distants, à la séparation du code en modules, à l'utilisation de l'épinglage des versions, etc., et validez-les dans votre configuration.
- Si vous ne disposez pas d'une base de code Terraform existante, utilisez ces meilleures pratiques lorsque vous structurez votre nouvelle configuration. Suivez dès le début les conseils relatifs à la gestion des états, à l'authentification, à la structure du code, etc.
- Essayez d'utiliser certains des modules HashiCorp communautaires mentionnés dans ce guide pour voir s'ils simplifient vos modèles d'architecture. Les modules permettent des niveaux d'abstraction plus élevés, vous n'avez donc pas à réécrire les ressources communes.
- Activez le linting, les scans de sécurité, les vérifications des politiques et les outils de test automatisés pour renforcer certaines des meilleures pratiques en matière de sécurité, de conformité et de qualité du code. Des outils tels que TFlint, tfsec et Checkov peuvent vous aider.
- Consultez la dernière documentation du AWS fournisseur pour voir s'il existe de nouvelles ressources ou fonctionnalités qui pourraient vous aider à optimiser votre utilisation de Terraform. Restez au courant des nouvelles versions du AWS fournisseur.
- Pour obtenir des conseils supplémentaires, consultez la documentation Terraform, le guide des meilleures pratiques et le guide de style sur le HashiCorp site Web.

# Ressources

## Références

Les liens suivants fournissent du matériel de lecture supplémentaire pour le AWS fournisseur Terraform et l'utilisation de Terraform pour iAc sur. AWS

- AWS Fournisseur Terraform (documentation) HashiCorp
- Modules Terraform pour les AWS services (Terraform Registry)
- The AWS and HashiCorp Partnership (article de HashiCorp blog)
- Identifiants dynamiques auprès du AWS fournisseur (documentation HCP Terraform)
- Verrouillage d'état DynamoDB (documentation Terraform)
- Appliquer la politique avec Sentinel (documentation Terraform)

## Outils

Les outils suivants permettent d'améliorer la qualité du code et d'automatiser les configurations Terraform AWS, comme recommandé dans ce guide des meilleures pratiques.

Qualité du code :

- Checkov : analyse le code Terraform pour identifier les erreurs de configuration avant le déploiement.
- TFlint : identifie les erreurs possibles, la syntaxe obsolète et les déclarations non utilisées. Ce linter peut également appliquer les AWS meilleures pratiques et les conventions de dénomination.
- terraform-docs : génère de la documentation à partir des modules Terraform dans différents formats de sortie.

Outils d'automatisation :

- HCP Terraform : aide les équipes à créer des versions, à collaborer et à créer des flux de travail Terraform grâce à des vérifications des politiques et à des portes d'approbation.
- Atlantis : un outil open source d'automatisation des pull requests Terraform pour valider les modifications de code.

- CDK pour Terraform : un framework qui vous permet d'utiliser des langages familiers tels que TypeScript Python, Java, C# et Go au lieu du langage de HashiCorp configuration (HCL) pour définir, provisionner et tester votre infrastructure Terraform sous forme de code.

# Historique du document

Le tableau suivant décrit les modifications importantes apportées à ce guide. Pour être averti des mises à jour à venir, abonnez-vous à un fil RSS.

| Modification | Description | Date |
| --- | --- | --- |
| Publication initiale | — | 28 mai 2024 |

# AWS Glossaire des directives prescriptives

Les termes suivants sont couramment utilisés dans les stratégies, les guides et les modèles fournis par les directives AWS prescriptives. Pour suggérer des entrées, veuillez utiliser le lien Faire un commentaire à la fin du glossaire.

## Nombres

7 R

Sept politiques de migration courantes pour transférer des applications vers le cloud. Ces politiques s'appuient sur les 5 R identifiés par Gartner en 2011 et sont les suivantes :

- Refactorisation/réarchitecture : transférez une application et modifiez son architecture en tirant pleinement parti des fonctionnalités natives cloud pour améliorer l'agilité, les performances et la capacité de mise à l'échelle. Cela implique généralement le transfert du système d'exploitation et de la base de données. Exemple : migrez votre base de données Oracle sur site vers l'édition compatible avec Amazon Aurora PostgreSQL.
- Replateformer (déplacer et remodeler) : transférez une application vers le cloud et introduisez un certain niveau d'optimisation pour tirer parti des fonctionnalités du cloud. Exemple : migrez votre base de données Oracle sur site vers Amazon Relational Database Service (Amazon RDS) pour Oracle dans le. AWS Cloud
- Racheter (rachat) : optez pour un autre produit, généralement en passant d'une licence traditionnelle à un modèle SaaS. Exemple : migrez votre système de gestion de la relation client (CRM) vers Salesforce.com.
- Réhéberger (lift and shift) : transférez une application vers le cloud sans apporter de modifications pour tirer parti des fonctionnalités du cloud. Exemple : migrez votre base de données Oracle sur site vers Oracle sur une instance EC2 dans le. AWS Cloud
- Relocaliser (lift and shift au niveau de l'hyperviseur) : transférez l'infrastructure vers le cloud sans acheter de nouveau matériel, réécrire des applications ou modifier vos opérations existantes. Vous migrez des serveurs d'une plateforme sur site vers un service cloud pour la même plateforme. Exemple : migrer une Microsoft Hyper-V application vers AWS.
- Retenir : conservez les applications dans votre environnement source. Il peut s'agir d'applications nécessitant une refactorisation majeure, que vous souhaitez retarder, et d'applications existantes que vous souhaitez retenir, car rien ne justifie leur migration sur le plan commercial.

- Retirer : mettez hors service ou supprimez les applications dont vous n'avez plus besoin dans votre environnement source.

# A

ABAC

Voir contrôle d'accès basé sur les attributs.

services abstraits

Consultez la section Services gérés.

ACIDE

Voir atomicité, consistance, isolation, durabilité.

migration active-active

Méthode de migration de base de données dans laquelle la synchronisation des bases de données source et cible est maintenue (à l'aide d'un outil de réplication bidirectionnelle ou d'opérations d'écriture double), tandis que les deux bases de données gèrent les transactions provenant de la connexion d'applications pendant la migration. Cette méthode prend en charge la migration par petits lots contrôlés au lieu d'exiger un basculement ponctuel. Elle est plus flexible mais demande plus de travail qu'une migration active-passive.

migration active-passive

Méthode de migration de base de données dans laquelle la synchronisation des bases de données source et cible est maintenue, mais seule la base de données source gère les transactions provenant de la connexion d'applications pendant que les données sont répliquées vers la base de données cible. La base de données cible n'accepte aucune transaction pendant la migration.

fonction d'agrégation

Fonction SQL qui agit sur un groupe de lignes et calcule une valeur de retour unique pour le groupe. Des exemples de fonctions d'agrégation incluent `SUM` et`MAX`.

AI

Voir intelligence artificielle.

AIOps

Voir les opérations d'intelligence artificielle.

anonymisation

Processus de suppression définitive d'informations personnelles dans un ensemble de données. L'anonymisation peut contribuer à protéger la vie privée. Les données anonymisées ne sont plus considérées comme des données personnelles.

anti-motif

Solution fréquemment utilisée pour un problème récurrent lorsque la solution est contre-productive, inefficace ou moins efficace qu'une solution alternative.

contrôle des applications

Une approche de sécurité qui permet d'utiliser uniquement des applications approuvées afin de protéger un système contre les logiciels malveillants.

portefeuille d'applications

Ensemble d'informations détaillées sur chaque application utilisée par une organisation, y compris le coût de génération et de maintenance de l'application, ainsi que sa valeur métier. Ces informations sont essentielles pour le processus de découverte et d'analyse du portefeuille et permettent d'identifier et de prioriser les applications à migrer, à moderniser et à optimiser.

intelligence artificielle (IA)

Domaine de l'informatique consacré à l'utilisation des technologies de calcul pour exécuter des fonctions cognitives généralement associées aux humains, telles que l'apprentissage, la résolution de problèmes et la reconnaissance de modèles. Pour plus d'informations, veuillez consulter Qu'est-ce que l'intelligence artificielle ?

opérations d'intelligence artificielle (AIOps)

Processus consistant à utiliser des techniques de machine learning pour résoudre les problèmes opérationnels, réduire les incidents opérationnels et les interventions humaines, mais aussi améliorer la qualité du service. Pour plus d'informations sur la façon dont les AIOps sont utilisées dans la stratégie de migration AWS , veuillez consulter le guide d'intégration des opérations.

chiffrement asymétrique

Algorithme de chiffrement qui utilise une paire de clés, une clé publique pour le chiffrement et une clé privée pour le déchiffrement. Vous pouvez partager la clé publique, car elle n'est pas utilisée pour le déchiffrement, mais l'accès à la clé privée doit être très restreint.

atomicité, cohérence, isolement, durabilité (ACID)

Ensemble de propriétés logicielles garantissant la validité des données et la fiabilité opérationnelle d'une base de données, même en cas d'erreur, de panne de courant ou d'autres problèmes.

contrôle d'accès par attributs (ABAC)

Pratique qui consiste à créer des autorisations détaillées en fonction des attributs de l'utilisateur, tels que le service, le poste et le nom de l'équipe. Pour plus d'informations, consultez ABAC pour AWS dans la documentation AWS Identity and Access Management (IAM).

source de données faisant autorité

Emplacement où vous stockez la version principale des données, considérée comme la source d'information la plus fiable. Vous pouvez copier les données de la source de données officielle vers d'autres emplacements à des fins de traitement ou de modification des données, par exemple en les anonymisant, en les expurgant ou en les pseudonymisant.

Zone de disponibilité

Un emplacement distinct au sein d'un Région AWS réseau isolé des défaillances dans d'autres zones de disponibilité et fournissant une connectivité réseau peu coûteuse et à faible latence aux autres zones de disponibilité de la même région.

AWS Cadre d'adoption du cloud (AWS CAF)

Un cadre de directives et de meilleures pratiques visant AWS à aider les entreprises à élaborer un plan efficace pour réussir leur migration vers le cloud. AWS La CAF organise ses conseils en six domaines prioritaires appelés perspectives : les affaires, les personnes, la gouvernance, les plateformes, la sécurité et les opérations. Les perspectives d'entreprise, de personnes et de gouvernance mettent l'accent sur les compétences et les processus métier, tandis que les perspectives relatives à la plateforme, à la sécurité et aux opérations se concentrent sur les compétences et les processus techniques. Par exemple, la perspective liée aux personnes cible les parties prenantes qui s'occupent des ressources humaines (RH), des fonctions de dotation en personnel et de la gestion des personnes. Dans cette perspective, la AWS CAF fournit des conseils pour le développement du personnel, la formation et les communications afin de préparer l'organisation à une adoption réussie du cloud. Pour plus d'informations, veuillez consulter le site Web AWS CAF et le livre blanc AWS CAF.

AWS Cadre de qualification de la charge de travail (AWS WQF)

Outil qui évalue les charges de travail liées à la migration des bases de données, recommande des stratégies de migration et fournit des estimations de travail. AWS Le WQF est inclus avec

AWS Schema Conversion Tool (AWS SCT). Il analyse les schémas de base de données et les objets de code, le code d'application, les dépendances et les caractéristiques de performance, et fournit des rapports d'évaluation.

# B

mauvais bot

Un bot destiné à perturber ou à nuire à des individus ou à des organisations.

BCP

Consultez la section Planification de la continuité des activités.

graphique de comportement

Vue unifiée et interactive des comportements des ressources et des interactions au fil du temps. Vous pouvez utiliser un graphique de comportement avec Amazon Detective pour examiner les tentatives de connexion infructueuses, les appels d'API suspects et les actions similaires. Pour plus d'informations, veuillez consulter Data in a behavior graph dans la documentation Detective.

système de poids fort

Système qui stocke d'abord l'octet le plus significatif. Voir aussi endianité.

classification binaire

Processus qui prédit un résultat binaire (l'une des deux classes possibles). Par exemple, votre modèle de machine learning peut avoir besoin de prévoir des problèmes tels que « Cet e-mail est-il du spam ou non ? » ou « Ce produit est-il un livre ou une voiture ? ».

filtre de Bloom

Structure de données probabiliste et efficace en termes de mémoire qui est utilisée pour tester si un élément fait partie d'un ensemble.

déploiement bleu/vert

Stratégie de déploiement dans laquelle vous créez deux environnements distincts mais identiques. Vous exécutez la version actuelle de l'application dans un environnement (bleu) et la nouvelle version de l'application dans l'autre environnement (vert). Cette stratégie vous permet de revenir rapidement en arrière avec un impact minimal.

bot

Application logicielle qui exécute des tâches automatisées sur Internet et simule l'activité ou l'interaction humaine. Certains robots sont utiles ou bénéfiques, comme les robots d'exploration Web qui indexent des informations sur Internet. D'autres robots, connus sous le nom de mauvais robots, sont destinés à perturber ou à nuire à des individus ou à des organisations.

botnet

Réseaux de robots infectés par des logiciels malveillants et contrôlés par une seule entité, connue sous le nom d'herder ou d'opérateur de bots. Les botnets sont le mécanisme le plus connu pour faire évoluer les bots et leur impact.

branche

Zone contenue d'un référentiel de code. La première branche créée dans un référentiel est la branche principale. Vous pouvez créer une branche à partir d'une branche existante, puis développer des fonctionnalités ou corriger des bogues dans la nouvelle branche. Une branche que vous créez pour générer une fonctionnalité est communément appelée branche de fonctionnalités. Lorsque la fonctionnalité est prête à être publiée, vous fusionnez à nouveau la branche de fonctionnalités dans la branche principale. Pour plus d'informations, consultez À propos des branches (GitHub documentation).

accès par brise-vitre

Dans des circonstances exceptionnelles et par le biais d'un processus approuvé, c'est un moyen rapide pour un utilisateur d'accéder à un accès auquel Compte AWS il n'est généralement pas autorisé. Pour plus d'informations, consultez l'indicateur Implementation break-glass procedures dans le guide Well-Architected AWS .

stratégie existante (brownfield)

L'infrastructure existante de votre environnement. Lorsque vous adoptez une stratégie existante pour une architecture système, vous concevez l'architecture en fonction des contraintes des systèmes et de l'infrastructure actuels. Si vous étendez l'infrastructure existante, vous pouvez combiner des politiques brownfield (existantes) et greenfield (inédites).

cache de tampon

Zone de mémoire dans laquelle sont stockées les données les plus fréquemment consultées.

capacité métier

Ce que fait une entreprise pour générer de la valeur (par exemple, les ventes, le service client ou le marketing). Les architectures de microservices et les décisions de développement

peuvent être dictées par les capacités métier. Pour plus d'informations, veuillez consulter la section Organisation en fonction des capacités métier du livre blanc Exécution de microservices conteneurisés sur AWS.

planification de la continuité des activités (BCP)

Plan qui tient compte de l'impact potentiel d'un événement perturbateur, tel qu'une migration à grande échelle, sur les opérations, et qui permet à une entreprise de reprendre ses activités rapidement.

# C

CAF

Voir le cadre d'adoption du AWS cloud.

déploiement de Canary

Diffusion lente et progressive d'une version pour les utilisateurs finaux. Lorsque vous êtes sûr, vous déployez la nouvelle version et remplacez la version actuelle dans son intégralité.

CCoE

Voir le Centre d'excellence du cloud.

CDC

Consultez la section Capture des données de modification.

capture des données de modification (CDC)

Processus de suivi des modifications apportées à une source de données, telle qu'une table de base de données, et d'enregistrement des métadonnées relatives à ces modifications. Vous pouvez utiliser la CDC à diverses fins, telles que l'audit ou la réplication des modifications dans un système cible afin de maintenir la synchronisation.

ingénierie du chaos

Introduire intentionnellement des défaillances ou des événements perturbateurs pour tester la résilience d'un système. Vous pouvez utiliser AWS Fault Injection Service (AWS FIS) pour effectuer des expériences qui stressent vos AWS charges de travail et évaluer leur réponse.

CI/CD

Découvrez l'intégration continue et la livraison continue.

classification

Processus de catégorisation qui permet de générer des prédictions. Les modèles de ML pour les problèmes de classification prédisent une valeur discrète. Les valeurs discrètes se distinguent toujours les unes des autres. Par exemple, un modèle peut avoir besoin d'évaluer la présence ou non d'une voiture sur une image.

chiffrement côté client

Chiffrement des données localement, avant que la cible ne les AWS service reçoive.

Centre d'excellence cloud (CCoE)

Une équipe multidisciplinaire qui dirige les efforts d'adoption du cloud au sein d'une organisation, notamment en développant les bonnes pratiques en matière de cloud, en mobilisant des ressources, en établissant des délais de migration et en guidant l'organisation dans le cadre de transformations à grande échelle. Pour plus d'informations, consultez les articles du CCoE sur le blog de stratégie AWS Cloud d'entreprise.

cloud computing

Technologie cloud généralement utilisée pour le stockage de données à distance et la gestion des appareils IoT. Le cloud computing est généralement associé à la technologie informatique de pointe.

modèle d'exploitation du cloud

Dans une organisation informatique, modèle d'exploitation utilisé pour créer, faire évoluer et optimiser un ou plusieurs environnements cloud. Pour plus d'informations, consultez la section Création de votre modèle d'exploitation cloud.

étapes d'adoption du cloud

Les quatre phases que les entreprises traversent généralement lorsqu'elles migrent vers AWS Cloud :

- Projet : exécution de quelques projets liés au cloud à des fins de preuve de concept et d'apprentissage
- Base : réaliser des investissements fondamentaux pour mettre à l'échelle l'adoption du cloud (par exemple, en créant une zone de destination, en définissant un CCoE ou en établissant un modèle opérationnel)
- Migration : migration d'applications individuelles

- Réinvention : optimisation des produits et services et innovation dans le cloud

Ces étapes ont été définies par Stephen Orban dans le billet de blog The Journey Toward Cloud-First & the Stages of Adoption publié sur le blog AWS Cloud Enterprise Strategy. Pour plus d'informations sur leur lien avec la stratégie de AWS migration, consultez le guide de préparation à la migration.

CMDB

Voir base de données de gestion de configuration.

référentiel de code

Emplacement où le code source et d'autres ressources, comme la documentation, les exemples et les scripts, sont stockés et mis à jour par le biais de processus de contrôle de version. Les référentiels cloud courants incluent GitHub ou AWS CodeCommit. Chaque version du code est appelée branche. Dans une structure de microservice, chaque référentiel est consacré à une seule fonctionnalité. Un seul pipeline CI/CD peut utiliser plusieurs référentiels.

cache passif

Cache tampon vide, mal rempli ou contenant des données obsolètes ou non pertinentes. Cela affecte les performances, car l'instance de base de données doit lire à partir de la mémoire principale ou du disque, ce qui est plus lent que la lecture à partir du cache tampon.

données gelées

Données rarement consultées et généralement historiques. Lorsque vous interrogez ce type de données, les requêtes lentes sont généralement acceptables. Le transfert de ces données vers des niveaux ou classes de stockage moins performants et moins coûteux peut réduire les coûts.

vision par ordinateur (CV)

Domaine de l'IA qui utilise l'apprentissage automatique pour analyser et extraire des informations à partir de formats visuels tels que des images numériques et des vidéos. Par exemple, AWS Panorama propose des appareils qui ajoutent des CV aux réseaux de caméras locaux, et Amazon SageMaker fournit des algorithmes de traitement d'image pour les CV.

dérive de configuration

Pour une charge de travail, une modification de configuration par rapport à l'état attendu. Cela peut entraîner une non-conformité de la charge de travail, et cela est généralement progressif et involontaire.

base de données de gestion des configurations (CMDB)

Référentiel qui stocke et gère les informations relatives à une base de données et à son environnement informatique, y compris les composants matériels et logiciels ainsi que leurs configurations. Vous utilisez généralement les données d'une CMDB lors de la phase de découverte et d'analyse du portefeuille de la migration.

pack de conformité

Ensemble de AWS Config règles et d'actions correctives que vous pouvez assembler pour personnaliser vos contrôles de conformité et de sécurité. Vous pouvez déployer un pack de conformité en tant qu'entité unique dans une région Compte AWS et, ou au sein d'une organisation, à l'aide d'un modèle YAML. Pour plus d'informations, consultez la section Packs de conformité dans la AWS Config documentation.

intégration continue et livraison continue (CI/CD)

Processus d'automatisation des étapes source, de génération, de test, intermédiaire et de production du processus de publication du logiciel. CI/CD est communément décrit comme un pipeline. CI/CD peut vous aider à automatiser les processus, à améliorer la productivité, à améliorer la qualité du code et à accélérer les livraisons. Pour plus d'informations, veuillez consulter Avantages de la livraison continue. CD peut également signifier déploiement continu. Pour plus d'informations, veuillez consulter Livraison continue et déploiement continu.

CV

Voir vision par ordinateur.

# D

données au repos

Données stationnaires dans votre réseau, telles que les données stockées.

classification des données

Processus permettant d'identifier et de catégoriser les données de votre réseau en fonction de leur sévérité et de leur sensibilité. Il s'agit d'un élément essentiel de toute stratégie de gestion des risques de cybersécurité, car il vous aide à déterminer les contrôles de protection et de conservation appropriés pour les données. La classification des données est une composante du pilier de sécurité du AWS Well-Architected Framework. Pour plus d'informations, veuillez consulter Classification des données.

dérive des données

Une variation significative entre les données de production et les données utilisées pour entraîner un modèle ML, ou une modification significative des données d'entrée au fil du temps. La dérive des données peut réduire la qualité, la précision et l'équité globales des prédictions des modèles ML.

données en transit

Données qui circulent activement sur votre réseau, par exemple entre les ressources du réseau.

maillage de données

Un cadre architectural qui fournit une propriété des données distribuée et décentralisée avec une gestion et une gouvernance centralisées.

minimisation des données

Le principe de collecte et de traitement des seules données strictement nécessaires. La pratique de la minimisation des données AWS Cloud peut réduire les risques liés à la confidentialité, les coûts et l'empreinte carbone de vos analyses.

périmètre de données

Ensemble de garde-fous préventifs dans votre AWS environnement qui permettent de garantir que seules les identités fiables accèdent aux ressources fiables des réseaux attendus. Pour plus d'informations, voir Création d'un périmètre de données sur AWS.

prétraitement des données

Pour transformer les données brutes en un format facile à analyser par votre modèle de ML. Le prétraitement des données peut impliquer la suppression de certaines colonnes ou lignes et le traitement des valeurs manquantes, incohérentes ou en double.

provenance des données

Le processus de suivi de l'origine et de l'historique des données tout au long de leur cycle de vie, par exemple la manière dont les données ont été générées, transmises et stockées.

sujet des données

Personne dont les données sont collectées et traitées.

entrepôt des données

Un système de gestion des données qui prend en charge les informations commerciales, telles que les analyses. Les entrepôts de données contiennent généralement de grandes quantités de données historiques et sont généralement utilisés pour les requêtes et les analyses.

langage de définition de base de données (DDL)

Instructions ou commandes permettant de créer ou de modifier la structure des tables et des objets dans une base de données.

langage de manipulation de base de données (DML)

Instructions ou commandes permettant de modifier (insérer, mettre à jour et supprimer) des informations dans une base de données.

DDL

Voir langage de définition de base de données.

ensemble profond

Sert à combiner plusieurs modèles de deep learning à des fins de prédiction. Vous pouvez utiliser des ensembles profonds pour obtenir une prévision plus précise ou pour estimer l'incertitude des prédictions.

deep learning

Un sous-champ de ML qui utilise plusieurs couches de réseaux neuronaux artificiels pour identifier le mappage entre les données d'entrée et les variables cibles d'intérêt.

defense-in-depth

Approche de la sécurité de l'information dans laquelle une série de mécanismes et de contrôles de sécurité sont judicieusement répartis sur l'ensemble d'un réseau informatique afin de protéger la confidentialité, l'intégrité et la disponibilité du réseau et des données qu'il contient. Lorsque vous adoptez cette stratégie AWS, vous ajoutez plusieurs contrôles à différentes couches de la AWS Organizations structure afin de sécuriser les ressources. Par exemple, une defense-in-depth approche peut combiner l'authentification multifactorielle, la segmentation du réseau et le chiffrement.

administrateur délégué

Dans AWS Organizations, un service compatible peut enregistrer un compte AWS membre pour administrer les comptes de l'organisation et gérer les autorisations pour ce service. Ce compte est

appelé administrateur délégué pour ce service. Pour plus d'informations et une liste des services compatibles, veuillez consulter la rubrique Services qui fonctionnent avec AWS Organizations dans la documentation AWS Organizations .

déploiement

Processus de mise à disposition d'une application, de nouvelles fonctionnalités ou de corrections de code dans l'environnement cible. Le déploiement implique la mise en œuvre de modifications dans une base de code, puis la génération et l'exécution de cette base de code dans les environnements de l'application.

environnement de développement

Voir environnement.

contrôle de détection

Contrôle de sécurité conçu pour détecter, journaliser et alerter après la survenue d'un événement. Ces contrôles constituent une deuxième ligne de défense et vous alertent en cas d'événements de sécurité qui ont contourné les contrôles préventifs en place. Pour plus d'informations, veuillez consulter la rubrique Contrôles de détection dans Implementing security controls on AWS.

cartographie de la chaîne de valeur du développement (DVSM)

Processus utilisé pour identifier et hiérarchiser les contraintes qui nuisent à la rapidité et à la qualité du cycle de vie du développement logiciel. DVSM étend le processus de cartographie de la chaîne de valeur initialement conçu pour les pratiques de production allégée. Il met l'accent sur les étapes et les équipes nécessaires pour créer et transférer de la valeur tout au long du processus de développement logiciel.

jumeau numérique

Représentation virtuelle d'un système réel, tel qu'un bâtiment, une usine, un équipement industriel ou une ligne de production. Les jumeaux numériques prennent en charge la maintenance prédictive, la surveillance à distance et l'optimisation de la production.

tableau des dimensions

Dans un schéma en étoile, table plus petite contenant les attributs de données relatifs aux données quantitatives d'une table de faits. Les attributs des tables de dimensions sont généralement des champs de texte ou des nombres discrets qui se comportent comme du texte. Ces attributs sont couramment utilisés pour la contrainte des requêtes, le filtrage et l'étiquetage des ensembles de résultats.

catastrophe

Un événement qui empêche une charge de travail ou un système d'atteindre ses objectifs commerciaux sur son site de déploiement principal. Ces événements peuvent être des catastrophes naturelles, des défaillances techniques ou le résultat d'actions humaines, telles qu'une mauvaise configuration involontaire ou une attaque de logiciel malveillant.

reprise après sinistre (DR)

La stratégie et le processus que vous utilisez pour minimiser les temps d'arrêt et les pertes de données causés par un sinistre. Pour plus d'informations, consultez Disaster Recovery of Workloads on AWS : Recovery in the Cloud in the AWS Well-Architected Framework.

DML

Voir langage de manipulation de base de données.

conception axée sur le domaine

Approche visant à développer un système logiciel complexe en connectant ses composants à des domaines évolutifs, ou objectifs métier essentiels, que sert chaque composant. Ce concept a été introduit par Eric Evans dans son ouvrage Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston : Addison-Wesley Professional, 2003). Pour plus d'informations sur l'utilisation du design piloté par domaine avec le modèle de figuier étrangleur, veuillez consulter Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

DR

Consultez la section Reprise après sinistre.

détection de dérive

Suivi des écarts par rapport à une configuration de référence. Par exemple, vous pouvez l'utiliser AWS CloudFormation pour détecter la dérive des ressources du système ou AWS Control Tower pour détecter les modifications de votre zone d'atterrissage susceptibles d'affecter le respect des exigences de gouvernance.

DVSM

Voir la cartographie de la chaîne de valeur du développement.

# E

EDA

Voir analyse exploratoire des données.

informatique de périphérie

Technologie qui augmente la puissance de calcul des appareils intelligents en périphérie d'un réseau IoT. Comparé au cloud computing, l'informatique de pointe peut réduire la latence des communications et améliorer le temps de réponse.

chiffrement

Processus informatique qui transforme des données en texte clair, lisibles par l'homme, en texte chiffré.

clé de chiffrement

Chaîne cryptographique de bits aléatoires générée par un algorithme cryptographique. La longueur des clés peut varier, et chaque clé est conçue pour être imprévisible et unique.

endianisme

Ordre selon lequel les octets sont stockés dans la mémoire de l'ordinateur. Les systèmes de poids fort stockent d'abord l'octet le plus significatif. Les systèmes de poids faible stockent d'abord l'octet le moins significatif.

point de terminaison

Voir point de terminaison de service.

service de point de terminaison

Service que vous pouvez héberger sur un cloud privé virtuel (VPC) pour le partager avec d'autres utilisateurs. Vous pouvez créer un service de point de terminaison avec AWS PrivateLink et accorder des autorisations à d'autres principaux Comptes AWS ou à AWS Identity and Access Management (IAM) principaux. Ces comptes ou principaux peuvent se connecter à votre service de point de terminaison de manière privée en créant des points de terminaison d'un VPC d'interface. Pour plus d'informations, veuillez consulter Création d'un service de point de terminaison dans la documentation Amazon Virtual Private Cloud (Amazon VPC).

planification des ressources d'entreprise (ERP)

Système qui automatise et gère les principaux processus métier (tels que la comptabilité, le MES et la gestion de projet) pour une entreprise.

chiffrement d'enveloppe

Processus de chiffrement d'une clé de chiffrement à l'aide d'une autre clé de chiffrement. Pour plus d'informations, consultez la section Chiffrement des enveloppes dans la documentation AWS Key Management Service (AWS KMS).

environment

Instance d'une application en cours d'exécution. Les types d'environnement les plus courants dans le cloud computing sont les suivants :

- Environnement de développement : instance d'une application en cours d'exécution à laquelle seule l'équipe principale chargée de la maintenance de l'application peut accéder. Les environnements de développement sont utilisés pour tester les modifications avant de les promouvoir dans les environnements supérieurs. Ce type d'environnement est parfois appelé environnement de test.
- Environnements inférieurs : tous les environnements de développement d'une application, tels que ceux utilisés pour les générations et les tests initiaux.
- Environnement de production : instance d'une application en cours d'exécution à laquelle les utilisateurs finaux peuvent accéder. Dans un pipeline CI/CD, l'environnement de production est le dernier environnement de déploiement.
- Environnements supérieurs : tous les environnements accessibles aux utilisateurs autres que l'équipe de développement principale. Ils peuvent inclure un environnement de production, des environnements de préproduction et des environnements pour les tests d'acceptation par les utilisateurs.

épopée

Dans les méthodologies agiles, catégories fonctionnelles qui aident à organiser et à prioriser votre travail. Les épopées fournissent une description détaillée des exigences et des tâches d'implémentation. Par exemple, les points forts de la AWS CAF en matière de sécurité incluent la gestion des identités et des accès, les contrôles de détection, la sécurité des infrastructures, la protection des données et la réponse aux incidents. Pour plus d'informations sur les épopées dans la stratégie de migration AWS , veuillez consulter le guide d'implémentation du programme.

ERP

Voir Planification des ressources d'entreprise.

analyse exploratoire des données (EDA)

Processus d'analyse d'un jeu de données pour comprendre ses principales caractéristiques. Vous collectez ou agrégez des données, puis vous effectuez des enquêtes initiales pour trouver des modèles, détecter des anomalies et vérifier les hypothèses. L'EDA est réalisée en calculant des statistiques récapitulatives et en créant des visualisations de données.

# F

tableau des faits

La table centrale dans un schéma en étoile. Il stocke des données quantitatives sur les opérations commerciales. Généralement, une table de faits contient deux types de colonnes : celles qui contiennent des mesures et celles qui contiennent une clé étrangère pour une table de dimensions.

échouer rapidement

Une philosophie qui utilise des tests fréquents et progressifs pour réduire le cycle de vie du développement. C'est un élément essentiel d'une approche agile.

limite d'isolation des défauts

Dans le AWS Cloud, une limite telle qu'une zone de disponibilité Région AWS, un plan de contrôle ou un plan de données qui limite l'effet d'une panne et contribue à améliorer la résilience des charges de travail. Pour plus d'informations, consultez la section Limites d'isolation des AWS pannes.

branche de fonctionnalités

Voir la succursale.

fonctionnalités

Les données d'entrée que vous utilisez pour faire une prédiction. Par exemple, dans un contexte de fabrication, les fonctionnalités peuvent être des images capturées périodiquement à partir de la ligne de fabrication.

importance des fonctionnalités

Le niveau d'importance d'une fonctionnalité pour les prédictions d'un modèle. Il s'exprime généralement sous la forme d'un score numérique qui peut être calculé à l'aide de différentes

techniques, telles que la méthode Shapley Additive Explanations (SHAP) et les gradients intégrés. Pour plus d'informations, voir Interprétabilité du modèle d'apprentissage automatique avec :AWS.

transformation de fonctionnalité

Optimiser les données pour le processus de ML, notamment en enrichissant les données avec des sources supplémentaires, en mettant à l'échelle les valeurs ou en extrayant plusieurs ensembles d'informations à partir d'un seul champ de données. Cela permet au modèle de ML de tirer parti des données. Par exemple, si vous décomposez la date « 2021-05-27 00:15:37 » en « 2021 », « mai », « jeudi » et « 15 », vous pouvez aider l'algorithme d'apprentissage à apprendre des modèles nuancés associés à différents composants de données.

FGAC

Découvrez le contrôle d'accès détaillé.

contrôle d'accès détaillé (FGAC)

Utilisation de plusieurs conditions pour autoriser ou refuser une demande d'accès.

migration instantanée (flash-cut)

Méthode de migration de base de données qui utilise la réplication continue des données via la capture des données de modification afin de migrer les données dans les plus brefs délais, au lieu d'utiliser une approche progressive. L'objectif est de réduire au maximum les temps d'arrêt.

# G

blocage géographique

Voir les restrictions géographiques.

restrictions géographiques (blocage géographique)

Sur Amazon CloudFront, option permettant d'empêcher les utilisateurs de certains pays d'accéder aux distributions de contenu. Vous pouvez utiliser une liste d'autorisation ou une liste de blocage pour spécifier les pays approuvés et interdits. Pour plus d'informations, consultez la section Restreindre la distribution géographique de votre contenu dans la CloudFront documentation.

Flux de travail Gitflow

Approche dans laquelle les environnements inférieurs et supérieurs utilisent différentes branches dans un référentiel de code source. Le flux de travail Gitflow est considéré comme existant, et le flux de travail basé sur les troncs est l'approche moderne préférée.

stratégie inédite

L'absence d'infrastructures existantes dans un nouvel environnement. Lorsque vous adoptez une stratégie inédite pour une architecture système, vous pouvez sélectionner toutes les nouvelles technologies sans restriction de compatibilité avec l'infrastructure existante, également appelée brownfield. Si vous étendez l'infrastructure existante, vous pouvez combiner des politiques brownfield (existantes) et greenfield (inédites).

barrière de protection

Règle de haut niveau qui permet de régir les ressources, les politiques et la conformité au sein des unités d'organisation (UO). Les barrières de protection préventives appliquent des politiques pour garantir l'alignement sur les normes de conformité. Elles sont mises en œuvre à l'aide de politiques de contrôle des services et de limites des autorisations IAM. Les barrières de protection de détection détectent les violations des politiques et les problèmes de conformité, et génèrent des alertes pour y remédier. Ils sont implémentés à l'aide d'Amazon AWS Config AWS Security Hub GuardDuty AWS Trusted Advisor, d'Amazon Inspector et de AWS Lambda contrôles personnalisés.

# H

HA

Découvrez la haute disponibilité.

migration de base de données hétérogène

Migration de votre base de données source vers une base de données cible qui utilise un moteur de base de données différent (par exemple, Oracle vers Amazon Aurora). La migration hétérogène fait généralement partie d'un effort de réarchitecture, et la conversion du schéma peut s'avérer une tâche complexe. AWS propose AWS SCT qui facilite les conversions de schémas.

haute disponibilité (HA)

Capacité d'une charge de travail à fonctionner en continu, sans intervention, en cas de difficultés ou de catastrophes. Les systèmes HA sont conçus pour basculer automatiquement, fournir constamment des performances de haute qualité et gérer différentes charges et défaillances avec un impact minimal sur les performances.

modernisation de l'historien

Approche utilisée pour moderniser et mettre à niveau les systèmes de technologie opérationnelle (OT) afin de mieux répondre aux besoins de l'industrie manufacturière. Un historien est un type de base de données utilisé pour collecter et stocker des données provenant de diverses sources dans une usine.

migration de base de données homogène

Migration de votre base de données source vers une base de données cible qui partage le même moteur de base de données (par exemple, Microsoft SQL Server vers Amazon RDS for SQL Server). La migration homogène s'inscrit généralement dans le cadre d'un effort de réhébergement ou de replateforme. Vous pouvez utiliser les utilitaires de base de données natifs pour migrer le schéma.

données chaudes

Données fréquemment consultées, telles que les données en temps réel ou les données translationnelles récentes. Ces données nécessitent généralement un niveau ou une classe de stockage à hautes performances pour fournir des réponses rapides aux requêtes.

correctif

Solution d'urgence à un problème critique dans un environnement de production. En raison de son urgence, un correctif est généralement créé en dehors du flux de travail de DevOps publication habituel.

période de soins intensifs

Immédiatement après le basculement, période pendant laquelle une équipe de migration gère et surveille les applications migrées dans le cloud afin de résoudre les problèmes éventuels. En règle générale, cette période dure de 1 à 4 jours. À la fin de la période de soins intensifs, l'équipe de migration transfère généralement la responsabilité des applications à l'équipe des opérations cloud.

## I

IaC

Considérez l'infrastructure comme un code.

politique basée sur l’identité

Politique attachée à un ou plusieurs principaux IAM qui définit leurs autorisations au sein de l' AWS Cloud environnement.

application inactive

Application dont l'utilisation moyenne du processeur et de la mémoire se situe entre 5 et 20 % sur une période de 90 jours. Dans un projet de migration, il est courant de retirer ces applications ou de les retenir sur site.

IIoT

Voir Internet industriel des objets.

infrastructure immuable

Modèle qui déploie une nouvelle infrastructure pour les charges de travail de production au lieu de mettre à jour, d'appliquer des correctifs ou de modifier l'infrastructure existante. Les infrastructures immuables sont intrinsèquement plus cohérentes, fiables et prévisibles que les infrastructures mutables. Pour plus d'informations, consultez les meilleures pratiques de déploiement à l'aide d'une infrastructure immuable dans le AWS Well-Architected Framework.

VPC entrant (d'entrée)

Dans une architecture AWS multi-comptes, un VPC qui accepte, inspecte et achemine les connexions réseau depuis l'extérieur d'une application. L'architecture de référence de sécuritéAWS recommande de configurer votre compte réseau avec des VPC entrants, sortants et d'inspection afin de protéger l'interface bidirectionnelle entre votre application et Internet en général.

migration incrémentielle

Stratégie de basculement dans le cadre de laquelle vous migrez votre application par petites parties au lieu d'effectuer un basculement complet unique. Par exemple, il se peut que vous ne transfériez que quelques microservices ou utilisateurs vers le nouveau système dans un premier temps. Après avoir vérifié que tout fonctionne correctement, vous pouvez transférer progressivement des microservices ou des utilisateurs supplémentaires jusqu'à ce que vous puissiez mettre hors service votre système hérité. Cette stratégie réduit les risques associés aux migrations de grande ampleur.

Industry 4.0

Terme introduit par Klaus Schwab en 2016 pour désigner la modernisation des processus de fabrication grâce aux avancées en matière de connectivité, de données en temps réel, d'automatisation, d'analyse et d'IA/ML.

infrastructure

Ensemble des ressources et des actifs contenus dans l'environnement d'une application.

infrastructure en tant que code (IaC)

Processus de mise en service et de gestion de l'infrastructure d'une application via un ensemble de fichiers de configuration. IaC est conçue pour vous aider à centraliser la gestion de l'infrastructure, à normaliser les ressources et à mettre à l'échelle rapidement afin que les nouveaux environnements soient reproductibles, fiables et cohérents.

internet industriel des objets (IIoT)

L'utilisation de capteurs et d'appareils connectés à Internet dans les secteurs industriels tels que la fabrication, l'énergie, l'automobile, les soins de santé, les sciences de la vie et l'agriculture. Pour plus d'informations, veuillez consulter Building an industrial Internet of Things (IIoT) digital transformation strategy.

VPC d'inspection

Dans une architecture AWS multi-comptes, un VPC centralisé qui gère les inspections du trafic réseau entre les VPC (identiques ou Régions AWS différents), Internet et les réseaux sur site. L'architecture de référence de sécuritéAWS recommande de configurer votre compte réseau avec des VPC entrants, sortants et d'inspection afin de protéger l'interface bidirectionnelle entre votre application et Internet en général.

Internet des objets (IoT)

Réseau d'objets physiques connectés dotés de capteurs ou de processeurs intégrés qui communiquent avec d'autres appareils et systèmes via Internet ou via un réseau de communication local. Pour plus d'informations, veuillez consulter la section Qu'est-ce que l'IoT ?.

interprétabilité

Caractéristique d'un modèle de machine learning qui décrit dans quelle mesure un être humain peut comprendre comment les prédictions du modèle dépendent de ses entrées. Pour plus d'informations, veuillez consulter Machine learning model interpretability with AWS.

IoT

Voir Internet des objets.

Bibliothèque d'informations informatiques (ITIL)

Ensemble de bonnes pratiques pour proposer des services informatiques et les aligner sur les exigences métier. L'ITIL constitue la base de l'ITSM.

gestion des services informatiques (ITSM)

Activités associées à la conception, à la mise en œuvre, à la gestion et à la prise en charge de services informatiques d'une organisation. Pour plus d'informations sur l'intégration des opérations cloud aux outils ITSM, veuillez consulter le guide d'intégration des opérations.

ITIL

Consultez la bibliothèque d'informations informatiques.

ITSM

Consultez la section Gestion des services informatiques.

# L

contrôle d'accès basé sur des étiquettes (LBAC)

Une implémentation du contrôle d'accès obligatoire (MAC) dans laquelle une valeur d'étiquette de sécurité est explicitement attribuée aux utilisateurs et aux données elles-mêmes. L'intersection entre l'étiquette de sécurité utilisateur et l'étiquette de sécurité des données détermine les lignes et les colonnes visibles par l'utilisateur.

zone de destination

Une zone d'atterrissage est un AWS environnement multi-comptes bien conçu, évolutif et sécurisé. Il s'agit d'un point de départ à partir duquel vos entreprises peuvent rapidement lancer et déployer des charges de travail et des applications en toute confiance dans leur environnement de sécurité et d'infrastructure. Pour plus d'informations sur les zones de destination, veuillez consulter Setting up a secure and scalable multi-account AWS environment.

migration de grande envergure

Migration de 300 serveurs ou plus.

LBAC

Voir contrôle d'accès basé sur des étiquettes.

principe de moindre privilège

Bonne pratique de sécurité qui consiste à accorder les autorisations minimales nécessaires à l'exécution d'une tâche. Pour plus d'informations, veuillez consulter la rubrique Accorder les autorisations de moindre privilège dans la documentation IAM.

lift and shift

Voir 7 Rs.

système de poids faible

Système qui stocke d'abord l'octet le moins significatif. Voir aussi endianité.

environnements inférieurs

Voir environnement.

# M

machine learning (ML)

Type d'intelligence artificielle qui utilise des algorithmes et des techniques pour la reconnaissance et l'apprentissage de modèles. Le ML analyse et apprend à partir de données enregistrées, telles que les données de l'Internet des objets (IoT), pour générer un modèle statistique basé sur des modèles. Pour plus d'informations, veuillez consulter Machine Learning.

branche principale

Voir la succursale.

malware

Logiciel conçu pour compromettre la sécurité ou la confidentialité de l'ordinateur. Les logiciels malveillants peuvent perturber les systèmes informatiques, divulguer des informations sensibles ou obtenir un accès non autorisé. Parmi les malwares, on peut citer les virus, les vers, les rançongiciels, les chevaux de Troie, les logiciels espions et les enregistreurs de frappe.

services gérés

AWS services qui AWS gère la couche d'infrastructure, le système d'exploitation et les plateformes, et vous accédez aux points de terminaison pour stocker et récupérer des données.

Amazon Simple Storage Service (Amazon S3) et Amazon DynamoDB sont des exemples de services gérés. Ils sont également connus sous le nom de services abstraits.

système d'exécution de la fabrication (MES)

Un système logiciel pour le suivi, la surveillance, la documentation et le contrôle des processus de production qui convertissent les matières premières en produits finis dans l'atelier.

MAP

Voir Migration Acceleration Program.

mécanisme

Processus complet au cours duquel vous créez un outil, favorisez son adoption, puis inspectez les résultats afin de procéder aux ajustements nécessaires. Un mécanisme est un cycle qui se renforce et s'améliore lorsqu'il fonctionne. Pour plus d'informations, voir Création de mécanismes dans le cadre AWS Well-Architected.

compte membre

Tous, à l' Comptes AWS exception du compte de gestion, qui font partie d'une organisation dans AWS Organizations. Un compte ne peut être membre que d'une seule organisation à la fois.

MAILLES

Voir le système d'exécution de la fabrication.

Transport télémétrique en file d'attente de messages (MQTT)

Protocole de communication léger machine-to-machine (M2M), basé sur le modèle de publication/d'abonnement, pour les appareils IoT aux ressources limitées.

microservice

Petit service indépendant qui communique via des API bien définies et qui est généralement détenu par de petites équipes autonomes. Par exemple, un système d'assurance peut inclure des microservices qui mappent à des capacités métier, telles que les ventes ou le marketing, ou à des sous-domaines, tels que les achats, les réclamations ou l'analytique. Les avantages des microservices incluent l'agilité, la flexibilité de la mise à l'échelle, la facilité de déploiement, la réutilisation du code et la résilience. Pour plus d'informations, consultez la section Intégration de microservices à l'aide de services AWS sans serveur.

architecture de microservices

Approche de création d'une application avec des composants indépendants qui exécutent chaque processus d'application en tant que microservice. Ces microservices communiquent via une

interface bien définie à l'aide d'API légères. Chaque microservice de cette architecture peut être mis à jour, déployé et mis à l'échelle pour répondre à la demande de fonctions spécifiques d'une application. Pour plus d'informations, consultez la section Implémentation de microservices sur AWS.

Programme d'accélération des migrations (MAP)

Un AWS programme qui fournit un support de conseil, des formations et des services pour aider les entreprises à établir une base opérationnelle solide pour passer au cloud, et pour aider à compenser le coût initial des migrations. MAP inclut une méthodologie de migration pour exécuter les migrations héritées de manière méthodique, ainsi qu'un ensemble d'outils pour automatiser et accélérer les scénarios de migration courants.

migration à grande échelle

Processus consistant à transférer la majeure partie du portefeuille d'applications vers le cloud par vagues, un plus grand nombre d'applications étant déplacées plus rapidement à chaque vague. Cette phase utilise les bonnes pratiques et les enseignements tirés des phases précédentes pour implémenter une usine de migration d'équipes, d'outils et de processus en vue de rationaliser la migration des charges de travail grâce à l'automatisation et à la livraison agile. Il s'agit de la troisième phase de la stratégie de migration AWS.

usine de migration

Équipes interfonctionnelles qui rationalisent la migration des charges de travail grâce à des approches automatisées et agiles. Les équipes de Migration Factory comprennent généralement les opérations, les analystes commerciaux et les propriétaires, les ingénieurs de migration, les développeurs et les DevOps professionnels travaillant dans le cadre de sprints. Entre 20 et 50 % du portefeuille d'applications d'entreprise est constitué de modèles répétés qui peuvent être optimisés par une approche d'usine. Pour plus d'informations, veuillez consulter la rubrique discussion of migration factories et le guide Cloud Migration Factory dans cet ensemble de contenus.

métadonnées de migration

Informations relatives à l'application et au serveur nécessaires pour finaliser la migration. Chaque modèle de migration nécessite un ensemble de métadonnées de migration différent. Les exemples de métadonnées de migration incluent le sous-réseau cible, le groupe de sécurité et le AWS compte.

modèle de migration

Tâche de migration reproductible qui détaille la stratégie de migration, la destination de la migration et l'application ou le service de migration utilisé. Exemple : réorganisez la migration vers Amazon EC2 AWS avec le service de migration d'applications.

Évaluation du portefeuille de migration (MPA)

Outil en ligne qui fournit des informations pour valider l'analyse de rentabilisation en faveur de la migration vers le. AWS Cloud La MPA propose une évaluation détaillée du portefeuille (dimensionnement approprié des serveurs, tarification, comparaison du coût total de possession, analyse des coûts de migration), ainsi que la planification de la migration (analyse et collecte des données d'applications, regroupement des applications, priorisation des migrations et planification des vagues). L'outil MPA (connexion requise) est disponible gratuitement pour tous les AWS consultants et consultants APN Partner.

Évaluation de la préparation à la migration (MRA)

Processus qui consiste à obtenir des informations sur l'état de préparation d'une organisation au cloud, à identifier les forces et les faiblesses et à élaborer un plan d'action pour combler les lacunes identifiées, à l'aide du AWS CAF. Pour plus d'informations, veuillez consulter le guide de préparation à la migration. La MRA est la première phase de la stratégie de migration AWS.

stratégie de migration

L'approche utilisée pour migrer une charge de travail vers le AWS Cloud. Pour plus d'informations, reportez-vous aux 7 R de ce glossaire et à Mobiliser votre organisation pour accélérer les migrations à grande échelle.

ML

Voir apprentissage automatique.

modernisation

Transformation d'une application obsolète (héritée ou monolithique) et de son infrastructure en un système agile, élastique et hautement disponible dans le cloud afin de réduire les coûts, de gagner en efficacité et de tirer parti des innovations. Pour plus d'informations, consultez la section Stratégie de modernisation des applications dans le AWS Cloud.

évaluation de la préparation à la modernisation

Évaluation qui permet de déterminer si les applications d'une organisation sont prêtes à être modernisées, d'identifier les avantages, les risques et les dépendances, et qui détermine dans quelle mesure l'organisation peut prendre en charge l'état futur de ces applications. Le résultat

de l'évaluation est un plan de l'architecture cible, une feuille de route détaillant les phases de développement et les étapes du processus de modernisation, ainsi qu'un plan d'action pour combler les lacunes identifiées. Pour plus d'informations, consultez la section Évaluation de l'état de préparation à la modernisation des applications dans le AWS Cloud.

applications monolithiques (monolithes)

Applications qui s'exécutent en tant que service unique avec des processus étroitement couplés. Les applications monolithiques ont plusieurs inconvénients. Si une fonctionnalité de l'application connaît un pic de demande, l'architecture entière doit être mise à l'échelle. L'ajout ou l'amélioration des fonctionnalités d'une application monolithique devient également plus complexe lorsque la base de code s'élargit. Pour résoudre ces problèmes, vous pouvez utiliser une architecture de microservices. Pour plus d'informations, veuillez consulter Decomposing monoliths into microservices.

MPA

Voir Évaluation du portefeuille de migration.

MQTT

Voir Message Queuing Telemetry Transport.

classification multi-classes

Processus qui permet de générer des prédictions pour plusieurs classes (prédiction d'un résultat parmi plus de deux). Par exemple, un modèle de ML peut demander « Ce produit est-il un livre, une voiture ou un téléphone ? » ou « Quelle catégorie de produits intéresse le plus ce client ? ».

infrastructure mutable

Modèle qui met à jour et modifie l'infrastructure existante pour les charges de travail de production. Pour améliorer la cohérence, la fiabilité et la prévisibilité, le AWS Well-Architected Framework recommande l'utilisation d'une infrastructure immuable comme meilleure pratique.

# O

OAC

Voir Contrôle d'accès à l'origine.

OAI

Voir l'identité d'accès à l'origine.

OCM

Voir gestion du changement organisationnel.

migration hors ligne

Méthode de migration dans laquelle la charge de travail source est supprimée au cours du processus de migration. Cette méthode implique un temps d'arrêt prolongé et est généralement utilisée pour de petites charges de travail non critiques.

OI

Consultez la section Intégration des opérations.

OLA

Voir l'accord au niveau opérationnel.

migration en ligne

Méthode de migration dans laquelle la charge de travail source est copiée sur le système cible sans être mise hors ligne. Les applications connectées à la charge de travail peuvent continuer à fonctionner pendant la migration. Cette méthode implique un temps d'arrêt nul ou minimal et est généralement utilisée pour les charges de travail de production critiques.

OPC-UA

Voir Open Process Communications - Architecture unifiée.

Communications par processus ouvert - Architecture unifiée (OPC-UA)

Un protocole de communication machine-to-machine (M2M) pour l'automatisation industrielle. L'OPC-UA fournit une norme d'interopérabilité avec des schémas de cryptage, d'authentification et d'autorisation des données.

accord au niveau opérationnel (OLA)

Accord qui précise ce que les groupes informatiques fonctionnels s'engagent à fournir les uns aux autres, afin de prendre en charge un contrat de niveau de service (SLA).

examen de l'état de préparation opérationnelle (ORR)

Une liste de questions et de bonnes pratiques associées qui vous aident à comprendre, évaluer, prévenir ou réduire l'ampleur des incidents et des défaillances possibles. Pour plus d'informations, voir Operational Readiness Reviews (ORR) dans le AWS Well-Architected Framework.

technologie opérationnelle (OT)

Systèmes matériels et logiciels qui fonctionnent avec l'environnement physique pour contrôler les opérations, les équipements et les infrastructures industriels. Dans le secteur manufacturier, l'intégration des systèmes OT et des technologies de l'information (IT) est au cœur des transformations de l'industrie 4.0.

intégration des opérations (OI)

Processus de modernisation des opérations dans le cloud, qui implique la planification de la préparation, l'automatisation et l'intégration. Pour en savoir plus, veuillez consulter le guide d'intégration des opérations.

journal de suivi d'organisation

Un parcours créé par AWS CloudTrail qui enregistre tous les événements pour tous les membres Comptes AWS d'une organisation dans AWS Organizations. Ce journal de suivi est créé dans chaque Compte AWS qui fait partie de l'organisation et suit l'activité de chaque compte. Pour plus d'informations, consultez la section Création d'un suivi pour une organisation dans la CloudTrail documentation.

gestion du changement organisationnel (OCM)

Cadre pour gérer les transformations métier majeures et perturbatrices du point de vue des personnes, de la culture et du leadership. L'OCM aide les organisations à se préparer et à effectuer la transition vers de nouveaux systèmes et de nouvelles politiques en accélérant l'adoption des changements, en abordant les problèmes de transition et en favorisant des changements culturels et organisationnels. Dans la stratégie de AWS migration, ce cadre est appelé accélération du personnel, en raison de la rapidité du changement requise dans les projets d'adoption du cloud. Pour plus d'informations, veuillez consulter le guide OCM.

contrôle d'accès d'origine (OAC)

Dans CloudFront, une option améliorée pour restreindre l'accès afin de sécuriser votre contenu Amazon Simple Storage Service (Amazon S3). L'OAC prend en charge tous les compartiments S3 dans leur ensemble Régions AWS, le chiffrement côté serveur avec AWS KMS (SSE-KMS) et les requêtes dynamiques `PUT` adressées au compartiment S3. `DELETE`

identité d'accès d'origine (OAI)

Dans CloudFront, une option permettant de restreindre l'accès afin de sécuriser votre contenu Amazon S3. Lorsque vous utilisez OAI, il CloudFront crée un principal auprès duquel Amazon S3 peut s'authentifier. Les principaux authentifiés ne peuvent accéder au contenu d'un compartiment

S3 que par le biais d'une distribution spécifique CloudFront . Voir également OAC, qui fournit un contrôle d'accès plus précis et amélioré.

OU

Voir l'examen de l'état de préparation opérationnelle.

DE

Voir technologie opérationnelle.

VPC sortant (de sortie)

Dans une architecture AWS multi-comptes, un VPC qui gère les connexions réseau initiées depuis une application. L'architecture de référence de sécuritéAWS recommande de configurer votre compte réseau avec des VPC entrants, sortants et d'inspection afin de protéger l'interface bidirectionnelle entre votre application et Internet en général.

# P

limite des autorisations

Politique de gestion IAM attachée aux principaux IAM pour définir les autorisations maximales que peut avoir l'utilisateur ou le rôle. Pour plus d'informations, veuillez consulter la rubrique Limites des autorisations dans la documentation IAM.

informations personnelles identifiables (PII)

Informations qui, lorsqu'elles sont consultées directement ou associées à d'autres données connexes, peuvent être utilisées pour déduire raisonnablement l'identité d'une personne. Les exemples d'informations personnelles incluent les noms, les adresses et les informations de contact.

PII

Voir les informations personnelles identifiables.

manuel stratégique

Ensemble d'étapes prédéfinies qui capturent le travail associé aux migrations, comme la fourniture de fonctions d'opérations de base dans le cloud. Un manuel stratégique peut revêtir la forme de scripts, de runbooks automatisés ou d'un résumé des processus ou des étapes nécessaires au fonctionnement de votre environnement modernisé.

PLC

Voir contrôleur logique programmable.

PLM

Consultez la section Gestion du cycle de vie des produits.

politique

Objet capable de définir les autorisations (voir la politique basée sur l'identité), de spécifier les conditions d'accès (voir la politique basée sur les ressources) ou de définir les autorisations maximales pour tous les comptes d'une organisation dans AWS Organizations (voir la politique de contrôle des services).

persistance polyglotte

Choix indépendant de la technologie de stockage de données d'un microservice en fonction des modèles d'accès aux données et d'autres exigences. Si vos microservices utilisent la même technologie de stockage de données, ils peuvent rencontrer des difficultés d'implémentation ou présenter des performances médiocres. Les microservices sont plus faciles à mettre en œuvre, atteignent de meilleures performances, ainsi qu'une meilleure capacité de mise à l'échelle s'ils utilisent l'entrepôt de données le mieux adapté à leurs besoins. Pour plus d'informations, veuillez consulter Enabling data persistence in microservices.

évaluation du portefeuille

Processus de découverte, d'analyse et de priorisation du portefeuille d'applications afin de planifier la migration. Pour plus d'informations, veuillez consulter Evaluating migration readiness.

predicate

Une condition de requête qui renvoie `true` ou`false`, généralement située dans une `WHERE` clause.

prédicat pushdown

Technique d'optimisation des requêtes de base de données qui filtre les données de la requête avant le transfert. Cela réduit la quantité de données qui doivent être extraites et traitées à partir de la base de données relationnelle et améliore les performances des requêtes.

contrôle préventif

Contrôle de sécurité conçu pour empêcher qu'un événement ne se produise. Ces contrôles constituent une première ligne de défense pour empêcher tout accès non autorisé ou toute

modification indésirable de votre réseau. Pour plus d'informations, veuillez consulter Preventative controls dans Implementing security controls on AWS.

principal

Entité AWS capable d'effectuer des actions et d'accéder aux ressources. Cette entité est généralement un utilisateur root pour un Compte AWS rôle IAM ou un utilisateur. Pour plus d'informations, veuillez consulter la rubrique Principal dans Termes et concepts relatifs aux rôles, dans la documentation IAM.

Confidentialité dès la conception

Une approche de l'ingénierie des systèmes qui prend en compte la confidentialité tout au long du processus d'ingénierie.

zones hébergées privées

Conteneur qui contient des informations concernant la façon dont vous souhaitez qu'Amazon Route 53 réponde aux requêtes DNS pour un domaine et ses sous-domaines dans un ou plusieurs VPC. Pour plus d'informations, veuillez consulter Working with private hosted zones dans la documentation Route 53.

contrôle proactif

Contrôle de sécurité conçu pour empêcher le déploiement de ressources non conformes. Ces contrôles analysent les ressources avant qu'elles ne soient provisionnées. Si la ressource n'est pas conforme au contrôle, elle n'est pas provisionnée. Pour plus d'informations, consultez le guide de référence sur les contrôles dans la AWS Control Tower documentation et consultez la section Contrôles proactifs dans Implémentation des contrôles de sécurité sur AWS.

gestion du cycle de vie des produits (PLM)

Gestion des données et des processus d'un produit tout au long de son cycle de vie, depuis la conception, le développement et le lancement, en passant par la croissance et la maturité, jusqu'au déclin et au retrait.

environnement de production

Voir environnement.

contrôleur logique programmable (PLC)

Dans le secteur manufacturier, un ordinateur hautement fiable et adaptable qui surveille les machines et automatise les processus de fabrication.

pseudonymisation

Processus de remplacement des identifiants personnels dans un ensemble de données par des valeurs fictives. La pseudonymisation peut contribuer à protéger la vie privée. Les données pseudonymisées sont toujours considérées comme des données personnelles.

publier/souscrire (pub/sub)

Modèle qui permet des communications asynchrones entre les microservices afin d'améliorer l'évolutivité et la réactivité. Par exemple, dans un MES basé sur des microservices, un microservice peut publier des messages d'événements sur un canal auquel d'autres microservices peuvent s'abonner. Le système peut ajouter de nouveaux microservices sans modifier le service de publication.

# Q

plan de requête

Série d'étapes, telles que des instructions, utilisées pour accéder aux données d'un système de base de données relationnelle SQL.

régression du plan de requêtes

Le cas où un optimiseur de service de base de données choisit un plan moins optimal qu'avant une modification donnée de l'environnement de base de données. Cela peut être dû à des changements en termes de statistiques, de contraintes, de paramètres d'environnement, de liaisons de paramètres de requêtes et de mises à jour du moteur de base de données.

# R

Matrice RACI

Voir responsable, responsable, consulté, informé (RACI).

rançongiciel

Logiciel malveillant conçu pour bloquer l'accès à un système informatique ou à des données jusqu'à ce qu'un paiement soit effectué.

Matrice RASCI

Voir responsable, responsable, consulté, informé (RACI).

RCAC

Voir contrôle d'accès aux lignes et aux colonnes.

réplica en lecture

Copie d'une base de données utilisée en lecture seule. Vous pouvez acheminer les requêtes vers le réplica de lecture pour réduire la charge sur votre base de données principale.

réarchitecte

Voir 7 Rs.

objectif de point de récupération (RPO)

Durée maximale acceptable depuis le dernier point de récupération des données. Cela permet de déterminer ce qui est considéré comme une perte de données acceptable entre le dernier point de restauration et l'interruption du service.

objectif de temps de récupération (RTO)

Le délai maximum acceptable entre l'interruption du service et le rétablissement du service.

refactoriser

Voir 7 Rs.

Région

Un ensemble de AWS ressources dans une zone géographique. Chacun Région AWS est isolé et indépendant des autres pour garantir tolérance aux pannes, stabilité et résilience. Pour plus d'informations, voir Spécifier ce que Régions AWS votre compte peut utiliser.

régression

Technique de ML qui prédit une valeur numérique. Par exemple, pour résoudre le problème « Quel sera le prix de vente de cette maison ? », un modèle de ML pourrait utiliser un modèle de régression linéaire pour prédire le prix de vente d'une maison sur la base de faits connus à son sujet (par exemple, la superficie en mètres carrés).

réhéberger

Voir 7 Rs.

version

Dans un processus de déploiement, action visant à promouvoir les modifications apportées à un environnement de production.

déplacer

Voir 7 Rs.

replateforme

Voir 7 Rs.

rachat

Voir 7 Rs.

résilience

La capacité d'une application à résister aux perturbations ou à s'en remettre. La haute disponibilité et la reprise après sinistre sont des considérations courantes lors de la planification de la résilience dans le AWS Cloud. Pour plus d'informations, consultez AWS Cloud Résilience.

politique basée sur les ressources

Politique attachée à une ressource, comme un compartiment Amazon S3, un point de terminaison ou une clé de chiffrement. Ce type de politique précise les principaux auxquels l'accès est autorisé, les actions prises en charge et toutes les autres conditions qui doivent être remplies.

matrice responsable, redevable, consulté et informé (RACI)

Une matrice qui définit les rôles et les responsabilités de toutes les parties impliquées dans les activités de migration et les opérations cloud. Le nom de la matrice est dérivé des types de responsabilité définis dans la matrice : responsable (R), responsable (A), consulté (C) et informé (I). Le type de support (S) est facultatif. Si vous incluez le support, la matrice est appelée matrice RASCI, et si vous l'excluez, elle est appelée matrice RACI.

contrôle réactif

Contrôle de sécurité conçu pour permettre de remédier aux événements indésirables ou aux écarts par rapport à votre référence de sécurité. Pour plus d'informations, veuillez consulter la rubrique Responsive controls dans Implementing security controls on AWS.

retain

Voir 7 Rs.

se retirer

Voir 7 Rs.

rotation

Processus de mise à jour périodique d'un secret pour empêcher un attaquant d'accéder aux informations d'identification.

contrôle d'accès aux lignes et aux colonnes (RCAC)

Utilisation d'expressions SQL simples et flexibles dotées de règles d'accès définies. Le RCAC comprend des autorisations de ligne et des masques de colonnes.

RPO

Voir l'objectif du point de récupération.

RTO

Voir l'objectif en matière de temps de rétablissement.

runbook

Ensemble de procédures manuelles ou automatisées nécessaires à l'exécution d'une tâche spécifique. Elles visent généralement à rationaliser les opérations ou les procédures répétitives présentant des taux d'erreur élevés.

# S

SAML 2.0

Un standard ouvert utilisé par de nombreux fournisseurs d'identité (IdPs). Cette fonctionnalité permet l'authentification unique fédérée (SSO), afin que les utilisateurs puissent se connecter AWS Management Console ou appeler les opérations d' AWS API sans que vous ayez à créer un utilisateur dans IAM pour tous les membres de votre organisation. Pour plus d'informations sur la fédération SAML 2.0, veuillez consulter À propos de la fédération SAML 2.0 dans la documentation IAM.

SCADA

Voir Contrôle de supervision et acquisition de données.

SCP

Voir la politique de contrôle des services.

secret

Dans AWS Secrets Manager des informations confidentielles ou restreintes, telles qu'un mot de passe ou des informations d'identification utilisateur, que vous stockez sous forme cryptée. Il comprend la valeur secrète et ses métadonnées. La valeur secrète peut être binaire, une chaîne unique ou plusieurs chaînes. Pour plus d'informations, voir Que contient le secret d'un Secrets Manager ? dans la documentation de Secrets Manager.

contrôle de sécurité

Barrière de protection technique ou administrative qui empêche, détecte ou réduit la capacité d'un assaillant d'exploiter une vulnérabilité de sécurité. Il existe quatre principaux types de contrôles de sécurité : préventifs, détectifs, réactifs et proactifs.

renforcement de la sécurité

Processus qui consiste à réduire la surface d'attaque pour la rendre plus résistante aux attaques. Cela peut inclure des actions telles que la suppression de ressources qui ne sont plus requises, la mise en œuvre des bonnes pratiques de sécurité consistant à accorder le moindre privilège ou la désactivation de fonctionnalités inutiles dans les fichiers de configuration.

système de gestion des informations et des événements de sécurité (SIEM)

Outils et services qui associent les systèmes de gestion des informations de sécurité (SIM) et de gestion des événements de sécurité (SEM). Un système SIEM collecte, surveille et analyse les données provenant de serveurs, de réseaux, d'appareils et d'autres sources afin de détecter les menaces et les failles de sécurité, mais aussi de générer des alertes.

automatisation des réponses de sécurité

Action prédéfinie et programmée conçue pour répondre automatiquement à un événement de sécurité ou y remédier. Ces automatisations servent de contrôles de sécurité détectifs ou réactifs qui vous aident à mettre en œuvre les meilleures pratiques AWS de sécurité. Parmi les actions de réponse automatique, citons la modification d'un groupe de sécurité VPC, l'application de correctifs à une instance Amazon EC2 ou la rotation des informations d'identification.

chiffrement côté serveur

Chiffrement des données à destination, par celui AWS service qui les reçoit.

Politique de contrôle des services (SCP)

Politique qui propose un contrôle centralisé des autorisations pour tous les comptes d'une organisation dans AWS Organizations. Les SCP définissent des barrières de protection ou des

limites aux actions qu'un administrateur peut déléguer à des utilisateurs ou à des rôles. Vous pouvez utiliser les SCP comme listes d'autorisation ou de refus, pour indiquer les services ou les actions autorisés ou interdits. Pour plus d'informations, consultez la section Politiques de contrôle des services dans la AWS Organizations documentation.

point de terminaison du service

URL du point d'entrée pour un AWS service. Pour vous connecter par programmation au service cible, vous pouvez utiliser un point de terminaison. Pour plus d'informations, veuillez consulter la rubrique AWS service endpoints dans Références générales AWS.

contrat de niveau de service (SLA)

Accord qui précise ce qu'une équipe informatique promet de fournir à ses clients, comme le temps de disponibilité et les performances des services.

indicateur de niveau de service (SLI)

Mesure d'un aspect des performances d'un service, tel que son taux d'erreur, sa disponibilité ou son débit.

objectif de niveau de service (SLO)

Mesure cible qui représente l'état d'un service, tel que mesuré par un indicateur de niveau de service.

modèle de responsabilité partagée

Un modèle décrivant la responsabilité que vous partagez en matière AWS de sécurité et de conformité dans le cloud. AWS est responsable de la sécurité du cloud, alors que vous êtes responsable de la sécurité dans le cloud. Pour de plus amples informations, veuillez consulter Modèle de responsabilité partagée.

SIEM

Consultez les informations de sécurité et le système de gestion des événements.

point de défaillance unique (SPOF)

Défaillance d'un seul composant critique d'une application susceptible de perturber le système.

SLA

Voir le contrat de niveau de service.

SLI

Voir l'indicateur de niveau de service.

SLO

Voir l'objectif de niveau de service.

split-and-seed modèle

Modèle permettant de mettre à l'échelle et d'accélérer les projets de modernisation. Au fur et à mesure que les nouvelles fonctionnalités et les nouvelles versions de produits sont définies, l'équipe principale se divise pour créer des équipes de produit. Cela permet de mettre à l'échelle les capacités et les services de votre organisation, d'améliorer la productivité des développeurs et de favoriser une innovation rapide. Pour plus d'informations, consultez la section Approche progressive de la modernisation des applications dans le. AWS Cloud

SPOF

Voir point de défaillance unique.

schéma en étoile

Structure organisationnelle de base de données qui utilise une grande table de faits pour stocker les données transactionnelles ou mesurées et utilise une ou plusieurs tables dimensionnelles plus petites pour stocker les attributs des données. Cette structure est conçue pour être utilisée dans un entrepôt de données ou à des fins de business intelligence.

modèle de figuier étrangleur

Approche de modernisation des systèmes monolithiques en réécrivant et en remplaçant progressivement les fonctionnalités du système jusqu'à ce que le système hérité puisse être mis hors service. Ce modèle utilise l'analogie d'un figuier de vigne qui se développe dans un arbre existant et qui finit par supplanter son hôte. Le schéma a été présenté par Martin Fowler comme un moyen de gérer les risques lors de la réécriture de systèmes monolithiques. Pour obtenir un exemple d'application de ce modèle, veuillez consulter Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

sous-réseau

Plage d'adresses IP dans votre VPC. Un sous-réseau doit se trouver dans une seule zone de disponibilité.

contrôle de supervision et acquisition de données (SCADA)

Dans le secteur manufacturier, un système qui utilise du matériel et des logiciels pour surveiller les actifs physiques et les opérations de production.

chiffrement symétrique

Algorithme de chiffrement qui utilise la même clé pour chiffrer et déchiffrer les données.

tests synthétiques

Tester un système de manière à simuler les interactions des utilisateurs afin de détecter les problèmes potentiels ou de surveiller les performances. Vous pouvez utiliser Amazon CloudWatch Synthetics pour créer ces tests.

# T

balises

Des paires clé-valeur qui agissent comme des métadonnées pour organiser vos AWS ressources. Les balises peuvent vous aider à gérer, identifier, organiser, rechercher et filtrer des ressources. Pour plus d'informations, veuillez consulter la rubrique Balisage de vos AWS ressources.

variable cible

La valeur que vous essayez de prédire dans le cadre du ML supervisé. Elle est également qualifiée de variable de résultat. Par exemple, dans un environnement de fabrication, la variable cible peut être un défaut du produit.

liste de tâches

Outil utilisé pour suivre les progrès dans un runbook. Liste de tâches qui contient une vue d'ensemble du runbook et une liste des tâches générales à effectuer. Pour chaque tâche générale, elle inclut le temps estimé nécessaire, le propriétaire et l'avancement.

environnement de test

Voir environnement.

entraînement

Pour fournir des données à partir desquelles votre modèle de ML peut apprendre. Les données d'entraînement doivent contenir la bonne réponse. L'algorithme d'apprentissage identifie des modèles dans les données d'entraînement, qui mettent en correspondance les attributs des données d'entrée avec la cible (la réponse que vous souhaitez prédire). Il fournit un modèle de ML qui capture ces modèles. Vous pouvez alors utiliser le modèle de ML pour obtenir des prédictions sur de nouvelles données pour lesquelles vous ne connaissez pas la cible.

passerelle de transit

Hub de transit de réseau que vous pouvez utiliser pour relier vos VPC et vos réseaux sur site. Pour plus d'informations, voir Qu'est-ce qu'une passerelle de transit dans la AWS Transit Gateway documentation.

flux de travail basé sur jonction

Approche selon laquelle les développeurs génèrent et testent des fonctionnalités localement dans une branche de fonctionnalités, puis fusionnent ces modifications dans la branche principale. La branche principale est ensuite intégrée aux environnements de développement, de préproduction et de production, de manière séquentielle.

accès sécurisé

Accorder des autorisations à un service que vous spécifiez pour effectuer des tâches au sein de votre organisation AWS Organizations et dans ses comptes en votre nom. Le service de confiance crée un rôle lié au service dans chaque compte, lorsque ce rôle est nécessaire, pour effectuer des tâches de gestion à votre place. Pour plus d'informations, consultez la section Utilisation AWS Organizations avec d'autres AWS services dans la AWS Organizations documentation.

réglage

Pour modifier certains aspects de votre processus d'entraînement afin d'améliorer la précision du modèle de ML. Par exemple, vous pouvez entraîner le modèle de ML en générant un ensemble d'étiquetage, en ajoutant des étiquettes, puis en répétant ces étapes plusieurs fois avec différents paramètres pour optimiser le modèle.

équipe de deux pizzas

Une petite DevOps équipe que vous pouvez nourrir avec deux pizzas. Une équipe de deux pizzas garantit les meilleures opportunités de collaboration possible dans le développement de logiciels.

# U

incertitude

Un concept qui fait référence à des informations imprécises, incomplètes ou inconnues susceptibles de compromettre la fiabilité des modèles de ML prédictifs. Il existe deux types d'incertitude : l'incertitude épistémique est causée par des données limitées et incomplètes, alors que l'incertitude aléatoire est causée par le bruit et le caractère aléatoire inhérents aux données.

Pour plus d'informations, veuillez consulter le guide Quantifying uncertainty in deep learning systems.

tâches indifférenciées

Également connu sous le nom de « levage de charges lourdes », ce travail est nécessaire pour créer et exploiter une application, mais qui n'apporte pas de valeur directe à l'utilisateur final ni d'avantage concurrentiel. Les exemples de tâches indifférenciées incluent l'approvisionnement, la maintenance et la planification des capacités.

environnements supérieurs

Voir environnement.

# V

mise à vide

Opération de maintenance de base de données qui implique un nettoyage après des mises à jour incrémentielles afin de récupérer de l'espace de stockage et d'améliorer les performances.

contrôle de version

Processus et outils permettant de suivre les modifications, telles que les modifications apportées au code source dans un référentiel.

Appairage de VPC

Connexion entre deux VPC qui vous permet d'acheminer le trafic à l'aide d'adresses IP privées. Pour plus d'informations, veuillez consulter la rubrique Qu'est-ce que l'appairage de VPC ? dans la documentation Amazon VPC.

vulnérabilités

Défaut logiciel ou matériel qui compromet la sécurité du système.

# W

cache actif

Cache tampon qui contient les données actuelles et pertinentes fréquemment consultées. L'instance de base de données peut lire à partir du cache tampon, ce qui est plus rapide que la lecture à partir de la mémoire principale ou du disque.

données chaudes

Données rarement consultées. Lorsque vous interrogez ce type de données, des requêtes modérément lentes sont généralement acceptables.

fonction de fenêtre

Fonction SQL qui effectue un calcul sur un groupe de lignes liées d'une manière ou d'une autre à l'enregistrement en cours. Les fonctions de fenêtre sont utiles pour traiter des tâches, telles que le calcul d'une moyenne mobile ou l'accès à la valeur des lignes en fonction de la position relative de la ligne en cours.

charge de travail

Ensemble de ressources et de code qui fournit une valeur métier, par exemple une application destinée au client ou un processus de backend.

flux de travail

Groupes fonctionnels d'un projet de migration chargés d'un ensemble de tâches spécifique. Chaque flux de travail est indépendant, mais prend en charge les autres flux de travail du projet. Par exemple, le flux de travail du portefeuille est chargé de prioriser les applications, de planifier les vagues et de collecter les métadonnées de migration. Le flux de travail du portefeuille fournit ces actifs au flux de travail de migration, qui migre ensuite les serveurs et les applications.

VER

Voir écrire une fois, lire plusieurs.

WQF

Consultez le cadre de qualification des charges de travail AWS.

écrire une fois, lire plusieurs (WORM)

Modèle de stockage qui écrit les données une seule fois et empêche leur suppression ou leur modification. Les utilisateurs autorisés peuvent lire les données autant de fois que nécessaire, mais ils ne peuvent pas les modifier. Cette infrastructure de stockage de données est considérée comme immuable.

# Z

exploit Zero-Day

Une attaque, généralement un logiciel malveillant, qui tire parti d'une vulnérabilité de type « jour zéro ».

vulnérabilité de type « jour zéro »

Une faille ou une vulnérabilité non atténuée dans un système de production. Les acteurs malveillants peuvent utiliser ce type de vulnérabilité pour attaquer le système. Les développeurs prennent souvent conscience de la vulnérabilité à la suite de l'attaque.

application zombie

Application dont l'utilisation moyenne du processeur et de la mémoire est inférieure à 5 %. Dans un projet de migration, il est courant de retirer ces applications.

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.