Choisir une infrastructure comme outil de code pour votre organisation

AWS Conseils prescriptifs

AWS Conseils prescriptifs: Choisir une infrastructure comme outil de code pour votre organisation

Table of Contents

Choisir une infrastructure comme outil de code pour votre organisation

Amazon Web Services (contributeurs)

Février 2026 (historique du document)

L'infrastructure en tant que code (IaC) est le processus de mise en service et de gestion de l'infrastructure d'une application par le biais d'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. Il s'agit d'un élément clé de l'Agile et DevOps des pratiques, telles que le contrôle de version, l'intégration continue et le déploiement continu.

Le choix d'un outil d'infrastructure sous forme de code (IaC) est considéré comme une décision stratégique pour une organisation. Cette décision concerne toutes les équipes qui créent l'infrastructure, les applications et les services de l'entreprise. Chaque outil a ses avantages et ses inconvénients ; il n'y a donc pas de one-size-fits-all modèle.

Dans le passé, la gestion et le provisionnement de l'infrastructure étaient un processus manuel semé d'erreurs. IaC rationalise ces tâches par le biais du code et est devenu une solution fiable pour le déploiement d'infrastructures. Les outils IaC permettent aux développeurs de définir et de déployer une infrastructure à l'aide de langages de programmation. Cela améliore non seulement l'agilité de l'entreprise, mais accélère également la croissance et le rythme de l'innovation. En outre, iAC améliore considérablement la sécurité car iAC permet à votre organisation de scanner le code avant le déploiement, afin de vérifier que l'infrastructure est fiable et sécurisée. En fin de compte, le bon outil IaC n'est pas simplement une décision technique, mais une décision stratégique qui a un impact direct sur le succès global de l'entreprise.

Ce guide explore cinq outils IaC différents qui peuvent être utilisés pour provisionner AWS des ressources : AWS CloudFormation, AWS Serverless Application Model (AWS SAM) Cloud Development Kit AWS (AWS CDK), HashiCorp Terraform et Pulumi. Il compare ces outils et vous guide dans le processus de sélection de celui qui répond aux besoins de votre équipe, de votre organisation et des talents du cloud. L'essentiel est d'aligner l'outil IaC choisi sur les objectifs de votre organisation et les compétences de vos développeurs. Par exemple, si votre équipe est compétente JavaScript, vous pouvez choisir AWS CDK with TypeScript comme principal outil IaC, car il optimise votre flux de travail de développement.

Avantages de la mise en œuvre d'iAC

En choisissant l'outil IaC adapté à votre organisation, vous pouvez bénéficier des avantages suivants :

- Rapidité — L'un des objectifs d'IaC est d'accélérer les choses en éliminant les processus manuels. Une approche basée sur le code permet d'en faire plus en moins de temps. La configuration manuelle de chaque environnement informatique est très coûteuse et nécessite des ingénieurs et des architectes dédiés à la configuration du matériel et des logiciels. En augmentant la vitesse, les entreprises sont mieux à même de s'adapter rapidement à l'évolution des besoins des clients et des conditions du marché. Il vous permet également d'écrire votre code IaC une seule fois et de déployer ce même code dans des centaines d'environnements en une fraction du temps qu'il vous faudrait pour les créer manuellement.
- Évolutivité — L'IaC facilite l'ajout de ressources à l'infrastructure existante. Les mises à niveau sont fournies rapidement afin que vous puissiez vous développer rapidement pendant les périodes de pointe d'utilisation. Par exemple, les organisations qui proposent des services en ligne peuvent évoluer pour répondre aux demandes des utilisateurs pendant les périodes de forte demande, telles que le Black Friday.
- Sécurité améliorée : iAC excelle dans la mise en place de la sécurité et de la conformité pour les entreprises. Organisations sont en mesure de définir des politiques et des configurations de sécurité de base et de les appliquer par le biais de pipelines en exécutant des tests automatisés sur votre iAC. Si l'iAC enfreint les règles de conformité, le pipeline échoue et fournit automatiquement un retour d'information au développeur. Cela peut empêcher la création d'une infrastructure non sécurisée.
- Cohérence — La cohérence est un autre avantage essentiel de l'IaC. Lorsque plusieurs ingénieurs déploient manuellement des configurations, les incohérences sont inévitables. Au fil du temps, il devient difficile de suivre et de reproduire les mêmes environnements. Ces incohérences entraînent souvent des différences critiques entre les environnements de développement, d'assurance qualité et de production.
- Efficacité — L'iAC améliore l'efficacité et la productivité tout au long du cycle de développement. Les programmeurs créent des environnements sandbox à développer de manière isolée. Les opérations peuvent rapidement mettre en place une infrastructure pour les tests de sécurité. Les ingénieurs d'assurance qualité disposent de copies exactes des environnements de production pendant les tests. Lorsqu'ils sont prêts à déployer, les développeurs mettent en production l'infrastructure et le code en une seule étape. Cela peut également permettre un niveau plus élevé

de collaboration et de travail d'équipe. Les équipes peuvent créer des modèles sécurisés pour les applications, puis partager ces modèles ou structures avec d'autres équipes, ce qui réduit les doublons.

- Coûts réduits — IaC réduit les coûts de développement de logiciels. Il n'est pas nécessaire de consacrer des ressources à la configuration manuelle des environnements. Vous ne payez que pour les ressources que vous utilisez activement, il n'y a donc pas de frais supplémentaires inutiles.
- Fiabilité améliorée : si votre infrastructure est importante, il est facile de mal configurer une ressource ou de fournir des services dans le mauvais ordre. Avec IaC, les ressources sont toujours provisionnées et configurées exactement comme elles ont été déclarées. La création manuelle de ressources étant sujette aux erreurs, vous pouvez être sûr que votre infrastructure sera créée comme prévu lorsque vous utilisez IaC.
- Détection de dérive de configuration — La dérive de configuration se produit lorsque la configuration qui a approvisionné votre environnement ne correspond plus à l'environnement réel. De nombreux outils IaC vous aident à détecter les dérives et à y remédier. Par exemple, si quelqu'un modifie incorrectement vos ressources manuellement, vous pouvez utiliser l'outil IaC pour détecter ce qui a été modifié et les restaurer dans l'état prévu.
- Expérimentation — Dans la mesure où l'IaC rend le provisionnement de nouvelles infrastructures beaucoup plus rapide et plus facile, vous pouvez apporter et tester des modifications expérimentales sans investir beaucoup de temps et de ressources. Si les résultats vous plaisent, vous pouvez rapidement étendre la nouvelle infrastructure de production.

Examen des outils IaC pour AWS Cloud

Ce guide explore cinq outils IaC différents qui peuvent être utilisés pour provisionner AWS des ressources. Il compare ces outils et vous guide dans le processus de sélection de celui qui répond aux besoins de votre équipe, de votre organisation et des talents du cloud. L'essentiel est d'aligner l'outil IaC choisi sur les objectifs de votre organisation et les compétences de vos développeurs.

Dans cette section, vous en apprendrez davantage sur le fonctionnement de chaque outil ainsi que sur ses avantages et ses inconvénients. Par exemple, certains outils ne peuvent être utilisés que dans le AWS Cloud, et ils ne sont pas adaptés aux déploiements multicloud ou hybrides. D'autres nécessitent la connaissance de langages de programmation spécialisés, tandis que d'autres supportent les langages de programmation courants. Évaluez les avantages et les inconvénients de chaque outil et déterminez celui qui convient le mieux à votre organisation.

Cette section décrit les outils IaC suivants :

- AWS CloudFormation
- AWS Serverless Application Model (AWS SAM)
- Cloud Development Kit AWS (AWS CDK)
- HashiCorp Terraforme
- Pulumi

Utilisation en AWS CloudFormation tant qu'outil IaC

AWS CloudFormationest un outil Service AWS qui utilise des fichiers modèles pour automatiser le provisionnement des AWS ressources. Vous créez un modèle qui décrit toutes les AWS ressources que vous souhaitez déployer, et qui CloudFormation fournit et configure ces ressources pour vous.

CloudFormation les modèles sont écrits à l'aide de JSON ou de YAML. Une CloudFormation pile est l'implémentation des ressources définies dans votre modèle. Vous pouvez gérer vos CloudFormation piles par le biais du CloudFormation SDK Console de gestion AWS, par programmation ou par le biais du (). AWS Command Line Interface AWS CLI Pour plus d'informations sur le CloudFormation fonctionnement, consultez AWS CloudFormation les sections concepts et Comment AWS CloudFormation fonctionne dans la CloudFormation documentation.

Avantages de l'utilisation CloudFormation :

- CloudFormation les ensembles de modifications vous permettent de prévisualiser les modifications apportées à une pile en cours d'exécution avant de les déployer. Les ensembles de modifications résument les modifications proposées aux ressources en cours d'exécution dans une pile existante. Cela peut vous aider à identifier les conflits ou les conséquences imprévues avant le déploiement. Par exemple, si vous modifiez le nom d'une instance de base de données Amazon Relational Database Service (Amazon RDS) CloudFormation , vous créerez une nouvelle base de données et supprimerez l'ancienne. Vous perdriez les données de l'ancienne base de données à moins que vous ne les ayez déjà sauvegardées. Si vous générez un ensemble de modifications, vous constaterez que votre modification entraînera le remplacement de votre base de données et vous pourrez planifier en conséquence avant de mettre à jour votre pile.
- Si une erreur survient lors du déploiement d'un ensemble de modifications CloudFormation , revient automatiquement au dernier état de fonctionnement connu.
- Vous pouvez utiliser des ensembles de CloudFormation piles pour déployer des ressources sur plusieurs Comptes AWS et Régions AWS.
- L'utilisation CloudFormation avec les fournisseurs de ressources dans les espaces de noms suivants est gratuite : AWS : :*, Alexa : :* et Custom : :*. Dans ces cas, vous ne payez que pour les AWS ressources que vous fournissez, comme si vous les aviez provisionnées manuellement.
- CloudFormation gère l'état pour vous. Cela signifie que CloudFormation des appels de service sous-jacents sont AWS effectués pour provisionner et configurer vos ressources telles que définies dans vos CloudFormation modèles.
- CloudFormation fournit des outils pour détecter et corriger les dérives de configuration. Pour plus d'informations, consultez la section Détection des modifications de configuration non gérées apportées aux piles et aux ressources dans la CloudFormation documentation.
- Vous pouvez l'utiliser CloudFormation pour créer des ressources personnalisées. Vous pouvez écrire une logique de provisionnement personnalisée dans des modèles qui CloudFormation s'exécutent chaque fois que vous créez, mettez à jour ou supprimez des piles.
- CloudFormation prend en charge la modélisation, le provisionnement et la gestion des ressources d'applications tierces avec le CloudFormation registre.
- CloudFormation soutient l'importation des ressources existantes dans CloudFormation la gestion.

Inconvénients liés à l'utilisation CloudFormation :

- Si vous n'êtes pas familier avec la syntaxe JSON ou YAML, cela peut prendre un certain temps pour vous y habituer. Le JSON n'a pas été conçu pour être lisible par l'homme, et il ne vous permet

pas de faire des commentaires intégrés. YAML vous permet de faire des commentaires et est plus facile à lire. Cependant, sa syntaxe est basée sur des tabulations et des espaces, il peut donc être facile de faire des erreurs d'indentation.

- CloudFormation ne prend pas en charge les déploiements multicloud.
- Vous devez utiliser une implémentation de niveau supérieur, telle que le Cloud Development Kit AWS (AWS CDK), pour créer des constructions réutilisables et d'autres codes modularisés.

Utiliser le AWS Serverless Application Model (AWS SAM) comme outil IaC

Le AWS Serverless Application Model (AWS SAM) est une boîte à outils qui s'étend AWS CloudFormation. Il inclut des fonctionnalités supplémentaires conçues pour vous aider à créer des applications sans serveur plus rapidement. Lorsque vous déployez un AWS SAM modèle, il est converti CloudFormation afin de créer les ressources définies. AWS SAM se compose de deux parties, la spécification du AWS SAM modèle et l'interface de ligne de AWS SAM commande (AWS SAM CLI). Bien que vous puissiez utiliser la CloudFormation syntaxe directement dans le AWS SAM modèle, il AWS SAM propose sa propre syntaxe unique qui vise spécifiquement à accélérer le développement sans serveur. Cette syntaxe abrégée permet d'optimiser les définitions de l'IaC pour les ressources sans serveur, telles qu'Amazon API Gateway AWS Lambda, et AWS Step Functions les ressources. La AWS SAM CLI est un outil de développement qui inclut des fonctionnalités qui vous aident à tester des AWS Lambda fonctions localement, à créer des pipelines d'intégration continue et de livraison continue (CI/CD) et à exécuter des commandes pour déployer des applications sans serveur.

Avantages de l'utilisation AWS SAM :

- AWS SAM présente les mêmes avantages que CloudFormation.
- Par rapport à CloudFormation, vous pouvez l'utiliser plus facilement AWS SAM pour créer des applications et des ressources sans serveur, telles qu'un Amazon API Gateway soutenu par AWS Lambda.
- À l'aide de la AWS SAM CLI, vous pouvez tester AWS Lambda des fonctions localement. Lorsque vous appelez localement une fonction Lambda en mode de débogage, vous pouvez y associer un débogueur. Avec le débogueur, vous pouvez parcourir votre code ligne par ligne, voir les valeurs de diverses variables et résoudre les problèmes de la même manière que pour toute autre application.

Inconvénients liés à l'utilisation AWS SAM :

- AWS SAM présente les mêmes inconvénients que CloudFormation.
- AWS SAM ne peut pas être utilisé en dehors de AWS.

Utilisation du AWS CDK comme outil IaC

Il s'Cloud Development Kit AWS (AWS CDK)agit d'un framework de développement de logiciels open source qui vous permet de définir les ressources de vos applications cloud en utilisant des langages de programmation familiers. Les AWS CDK supports Python JavaScript TypeScript, Java, C# et Go. Ils AWS CDK approvisionnent vos ressources de manière sûre et reproductible. AWS CloudFormation Lorsque vous synthétisez votre AWS CDK code, le résultat est un CloudFormation modèle. AWS CDK fournit des abstractions de haut niveau qui simplifient le processus de définition des AWS ressources.

Il AWS CDK utilise le concept de constructions. Une construction est un composant de votre application qui représente une ou plusieurs CloudFormation ressources et leur configuration, tel qu'un bucket Amazon Simple Storage Service (Amazon S3). Les constructions peuvent être composées et personnalisées pour créer une infrastructure plus complexe. Pour plus d'informations, consultez la section Construire des niveaux dans la AWS CDK documentation. AWS CDK Génère CloudFormation des modèles basés sur le code écrit par les développeurs. Il n'est donc plus nécessaire de créer des CloudFormation modèles manuellement. De nombreuses organisations personnalisent, partagent et réutilisent des structures au sein d'une communauté, comme n'importe quelle autre bibliothèque logicielle. Le partage de structures permet aux développeurs de coder plus rapidement et d'intégrer les meilleures pratiques par défaut.

AWS CDK certains aspects peuvent aider les organisations à appliquer des normes à tous les concepts relevant d'un champ d'application donné. L'aspect peut modifier les constructions, par exemple en ajoutant des balises. Ou cela pourrait vérifier quelque chose sur l'état des constructions.

AWS CDK Cela permet aux développeurs d'utiliser leurs compétences et connaissances de programmation existantes pour définir l'infrastructure cloud. En utilisant des langages de programmation familiers, les développeurs peuvent appliquer leur expertise pour décrire les AWS ressources, ce qui facilite la transition entre le développement d'applications et le provisionnement de l'infrastructure. Ils AWS CDK peuvent également accélérer la création d' AWS infrastructures. Cela accélère le cycle de développement par rapport à l'écriture manuelle CloudFormation de modèles.

Avantages de l'utilisation de AWS CDK :

- Il AWS CDK prend en charge les langages de programmation les plus connus.
- Les langages à usage général permettent d'utiliser des constructions logiques, telles que des boucles for, des objets, des types forts et d'autres techniques de programmation. Cela permet aux développeurs de déclarer l'infrastructure de manière concise et sans erreur. Cette approche permet également d'utiliser un environnement de développement intégré (IDE) et les outils associés pour aider à gérer la complexité liée à la déclaration d'un grand nombre de ressources.
- AWS CDK les constructions sont partageables et vous aident à répondre à vos exigences de gouvernance et de conformité.
- Les AWS CDK constructions peuvent réduire le temps et les efforts de développement. Pour plus d'informations, consultez la référence de l'API Construct Library.
- Le AWS CDK est basé sur CloudFormation. Si vous connaissez ses CloudFormation concepts, les AWS CDK concepts sont plus faciles à comprendre.
- Il vous AWS CDK aide à effectuer des tests unitaires et des tests instantanés.
- Si une fonctionnalité n'est pas prise en charge nativement dans le AWS CDK, vous pouvez utiliser une construction de niveau 1 et des remplacements bruts. Vous pouvez également utiliser une ressource CloudFormation personnalisée qui appelle directement l'API.
- Vous pouvez nettoyer les ressources de manière efficace en supprimant des CloudFormation piles.

Inconvénients liés à l'utilisation de AWS CDK :

- AWS CDK Cela nécessite un environnement d'amorçage dans chacun d'entre eux. Compte AWS Le bootstrapping est une action ponctuelle que vous devez effectuer pour chaque environnement dans lequel vous déployez des ressources.
- Le AWS CDK peut être utilisé pour déployer iAc uniquement dans le AWS Cloud.

Utilisation de Terraform comme outil IaC pour AWS Cloud

HashiCorp Terraform est un outil d'infrastructure en tant que code (IaC) qui vous aide à gérer votre infrastructure cloud. À l'aide de Terraform, vous pouvez définir des ressources cloud et sur site dans des fichiers de configuration que vous pouvez versionner, réutiliser et partager. Vous pouvez ensuite utiliser un flux de travail cohérent pour provisionner et gérer l'ensemble de votre infrastructure tout au long de son cycle de vie.

Les développeurs utilisent un langage de configuration de haut niveau appelé langage Terraform. La syntaxe native de bas niveau du langage Terraform est le langage de HashiCorpconfiguration (HCL).

Le langage Terraform est conçu pour être facile à lire et à écrire pour les humains. Vous utilisez le langage Terraform pour décrire l'état final souhaité du cloud ou de l'infrastructure sur site. Terraform génère ensuite un plan pour atteindre cet état final, et vous exécutez le plan pour approvisionner l'infrastructure.

Avantages de l'utilisation de Terraform :

- Terraform est indépendant de la plateforme. Vous pouvez l'utiliser avec n'importe quel fournisseur de services cloud. Vous pouvez configurer, tester et déployer l'infrastructure au sein AWS de nombreux autres fournisseurs de cloud. Si votre entreprise utilise plusieurs fournisseurs de cloud, Terraform peut être une solution unique, unifiée et cohérente pour gérer l'infrastructure cloud. Pour plus d'informations sur le support multicloud, consultez la section Provisionnement multicloud sur le site Web de Terraform.
- Terraform est sans agent. Il ne nécessite l'installation d'aucun logiciel sur l'infrastructure gérée.
- Les modules Terraform sont un moyen puissant de réutiliser le code et de respecter le principe « Don't Repeat Yourself » (DRY). Par exemple, vous pouvez avoir une configuration spécifique pour une application qui contient une instance Amazon Elastic Compute Cloud (Amazon EC2), des volumes Amazon Elastic Block Store (Amazon EBS) et d'autres ressources regroupées de manière logique. Si vous devez créer plusieurs copies de cette configuration ou de cette application, vous pouvez regrouper les ressources dans un module Terraform et créer plusieurs instances du module plutôt que de copier le code entier plusieurs fois. Ces modules peuvent vous aider à organiser, à encapsuler et à réutiliser les configurations. Ils assurent également la cohérence et garantissent les meilleures pratiques.
- Terraform est capable de détecter et de gérer la dérive (article de blog Terraform) dans votre infrastructure. Par exemple, si les ressources gérées par Terraform sont modifiées en dehors de Terraform, vous pouvez détecter la dérive et les restaurer à l'état souhaité à l'aide de la CLI Terraform.

Inconvénients de l'utilisation de Terraform :

- Support pour les nouvelles fonctionnalités ou les nouvelles ressources liées à un fournisseur de cloud peut ne pas être disponible.
- Terraform ne gère pas automatiquement votre état comme. AWS CloudFormation Il est stocké par défaut dans un fichier local, mais vous pouvez également le stocker à distance dans un compartiment Amazon S3 ou via Terraform Enterprise.

- L'état Terraform peut contenir des données sensibles, telles que des mots de passe de base de données, qui peuvent poser des problèmes de sécurité. Il est recommandé de chiffrer votre fichier d'état, de le stocker à distance, d'activer le contrôle de version des fichiers et d'utiliser le moins de privilèges pour les opérations de lecture et d'écriture sur celui-ci. Pour plus d'informations, consultez Sécurisation des données sensibles à l'aide AWS Secrets Manager de HashiCorp Terraform.
- En août 2023, Hashicorp a annoncé qu'il ne serait plus licencié en tant que logiciel libre sous la licence publique Mozilla. Au lieu de cela, il est désormais licencié sous la licence Business Source.

Utiliser Pulumi comme outil IaC pour le AWS Cloud

Pulumi est une infrastructure open source en tant qu'outil de code. Il prend en charge les langages de programmation courants existants TypeScript JavaScript, tels que Python, Go, .NET, Java et YAML. Il utilise également son écosystème natif pour interagir avec les ressources du cloud via le SDK Pulumi. Une CLI téléchargeable, un environnement d'exécution, des bibliothèques et un service hébergé fonctionnent ensemble pour fournir, mettre à jour et gérer l'infrastructure cloud.

Vous pouvez utiliser le SDK Pulumi pour créer et déployer des logiciels cloud utilisant des conteneurs, des fonctions sans serveur, des services hébergés et une infrastructure, sur n'importe quel cloud.

Vous pouvez éventuellement associer Pulumi au Pulumi Cloud. Pulumi Cloud est un service géré qui stocke votre état et vos secrets, et qui gère les déploiements de votre infrastructure Pulumi.

Avantages de l'utilisation de Pulumi :

- Pulumi fournit l'infrastructure de plus de cinquante fournisseurs de cloud et de logiciels en tant que service (SaaS).
- Il propose une interface complète et cohérente conçue pour réduire la complexité du cloud.

Inconvénients de l'utilisation de Pulumi :

- Bien que Pulumi dispose d'une communauté active qui offre du soutien, elle est plus petite que la communauté Terraform.
- Pulumi a une courbe d'apprentissage plus abrupte.

Choisir un outil IaC

Alors, quel outil choisir ?

Avec autant d'options d'outils différentes et des exigences commerciales différentes, il n'y a pas d' one-size-fits-all approche. Outre les avantages et les inconvénients de chaque outil décrit dans ce guide, tenez compte des recommandations suivantes relatives aux exigences de votre entreprise et à votre modèle d'exploitation :

- Si vous gérez ou déployez une AWS solution sans serveur avec un minimum de dépendances, AWS Serverless Application Model (AWS SAM) peut être une bonne option pour vous. Il possède toutes les mêmes fonctionnalités que AWS CloudFormation. Il simplifie également les tests et le déploiement d'applications sans serveur sur le AWS Cloud.
- Si vous gérez entièrement votre infrastructure AWS, alors ce AWS CloudFormation Cloud Development Kit AWS (AWS CDK) sont de bonnes options. Ils assurent la gestion des out-of-the-box états et vous pouvez également utiliser de manière native de nouvelles fonctionnalités ou AWS ressources.
- Si vous recherchez un utilitaire multifournisseur, en particulier pour gérer une infrastructure multicloud ou hybride, Terraform peut être un bon choix car il est indépendant de la plate-forme. Avec Terraform, vous pouvez également utiliser une large gamme de plugins, et il dispose d'une large communauté proposant des options de support aux entreprises.
- Si vous avez une distribution descendante avec les meilleures pratiques et si vous disposez d'une orchestration dans le cadre de laquelle vous créez, publiez et distribuez des modules réutilisables à l'aide de langages de programmation courants, cette option AWS CDK peut être une bonne option.
- Si votre entreprise peut tolérer un niveau de risque élevé et doit prendre en charge des environnements multicloud ou hybrides, pensez à utiliser Pulumi.

Étapes suivantes

Ce guide décrit les avantages et les inconvénients de plusieurs outils d'infrastructure sous forme de code (IaC) que vous pouvez utiliser pour créer, déployer et gérer les AWS ressources et l'infrastructure. En fonction de l'outil que vous choisissez, nous vous recommandons de consulter les ressources suivantes pour commencer.

Cloud Development Kit AWS (AWS CDK)

- AWS CDK atelier d'introduction

AWS CloudFormation

- AWS CloudFormation labo
- Atelier AWS CloudFormation (langue française non garantie)

HashiCorp Terraforme

- AWS Atelier Terraform
- CDK pour le référentiel Terraform () GitHub

Pulumi

- Référentiel Pulumi () GitHub
- AWS modernisation avec l'atelier Pulumi

Ressources

AWS documentation

- Meilleures pratiques d'utilisation de l' AWS CDK in pour TypeScript créer des projets IaC (directives AWS prescriptives)
- Vérifiez les meilleures pratiques dans les AWS CDK applications ou les CloudFormation modèles à l'aide des packs de règles cdk-nag (AWS directives prescriptives)
- Présentation de AWS : DevOps L'infrastructure en tant que code (AWS livre blanc)
- Documentation AWS CloudFormation

Autres documentations

- Commencer à utiliser l'infrastructure en tant que code (HashiCorpsite Web)
- HashiCorpDocumentation Terraform
- Documentation Pulumi
- Pulumi : Qu'est-ce que l'infrastructure en tant que code (site Web de Pulumi)

Outils

- sac cdk () GitHub
- sac cfn-bag () GitHub
- Terratest

Collaborateurs

Conception

- Siamak Heshmati, architecte principal, DevOps AWS
- Mason Cahill, DevOps consultant principal, AWS
- Sandip Gangapadhyay, architecte principal, DevOps AWS
- Sandeep Gawande, consultant principal principal, AWS
- Rajneesh Tyagi, consultant principal, AWS

Révision

- David Howerton, responsable principal de l'engagement, AWS
- Steven Guggenheimer, architecte senior d'applications cloud, AWS

Rédaction technique

- Lilly AbouHarb, rédactrice technique senior, AWS

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
Mises à jour de Pulumi	Nous avons mis à jour le chapitre Pulumi.	17 février 2026
Publication initiale	—	23 février 2024

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.