Infrastructure en tant que code

Juillet 2017

Mentions légales

Ce document est fourni à titre informatif uniquement. Il présente l'offre de produits et les pratiques actuelles d'AWS à la date de publication de ce document, des informations qui sont susceptibles d'être modifiées sans préavis. Il incombe aux clients de procéder à leur propre évaluation indépendante des informations contenues dans ce document et chaque client est responsable de son utilisation des produits ou services AWS, chacun étant fourni « en l'état », sans garantie d'aucune sorte, qu'elle soit explicite ou implicite. Ce document n'offre pas de garantie, représentation, engagement contractuel, condition ou assurance de la part d'AWS, de ses sociétés apparentées, fournisseurs ou concédants de licence. Les responsabilités et obligations d'AWS vis-à-vis de ses clients sont régies par les contrats AWS. Le présent document ne fait partie d'aucun contrat et ne modifie aucun contrat entre AWS et ses clients.

Table des matières

Résumé

L'Infrastructure en tant que code s'est imposée comme pratique idéale pour automatiser la mise en service de services d'infrastructure. Ce livre blanc décrit les avantages de l'infrastructure en tant que code et la façon d'exploiter les fonctionnalités d'Amazon Web Services dans ce domaine pour prendre en charge les projets DevOps.

Le DevOps est une combinaison de philosophies culturelles, de pratiques et d'outils qui améliorent la capacité de votre organisation à livrer des applications et des services à haute vitesse. Cela permet à votre entreprise de mieux répondre aux besoins de vos clients. La pratique de l'infrastructure en tant que code peut être un déclencheur qui permet d'atteindre une telle vitesse.

Présentation de l'infrastructure en tant que code

La gestion de l'infrastructure est un processus associé à l'ingénierie logicielle. Traditionnellement, les organisations « empilaient et intégraient » du matériel avant d'installer et configurer des systèmes d'exploitation et des applications prenant en charge leurs besoins technologiques. Le cloud computing tire parti de la virtualisation pour activer la mise en service à la demande de calcul, de réseau et de ressources de stockage qui constituent les infrastructures technologiques.

Les gestionnaires de l'infrastructure ont souvent effectué cette mise en service manuellement. Les processus manuels présentent certains inconvénients, y compris :

- Coût plus élevé car ils monopolisent des ressources humaines qui auraient pu remplir des tâches commerciales plus importantes.
- Incohérences en raison d'erreurs humaines, susceptibles d'entraîner des écarts par rapport aux normes de configuration.
- Manque d'agilité liée à la vitesse limitée à laquelle votre organisation peut publier de nouvelles versions des services en réponse aux besoins des clients et des moteurs du marché.
- Difficultés pour atteindre et conserver la conformité aux normes du secteur ou de l'entreprise en raison de l'absence de processus reproductible.

L'infrastructure en tant que code répond à ces défaillances en introduisant l'automatisation du processus de mise en service. Plutôt que de s'appuyer sur des étapes réalisées manuellement, les administrateurs et les développeurs peuvent instancier l'infrastructure à l'aide de fichiers de configuration. L'infrastructure en tant que code traite ces fichiers de configuration comme du code logiciel. Ces fichiers peuvent être utilisés pour produire un ensemble d'artefacts, à savoir les capacités de calcul, de stockage, de réseau, et d'applications qui comprennent un environnement d'exécution. L'infrastructure en tant que code élimine les risques de changement de configuration grâce à l'automatisation, augmentant ainsi la vitesse et la flexibilité des déploiements d'infrastructure.

Cycle de vie des ressources d'infrastructure

Dans la section précédente, nous avons présenté Infrastructure en tant que code comme un moyen de mettre en service des ressources de manière cohérente et reproductible. Les concepts sous-jacents sont également pertinents pour l'ensemble des rôles des opérations technologiques de l'infrastructure. Examinez le diagramme suivant.

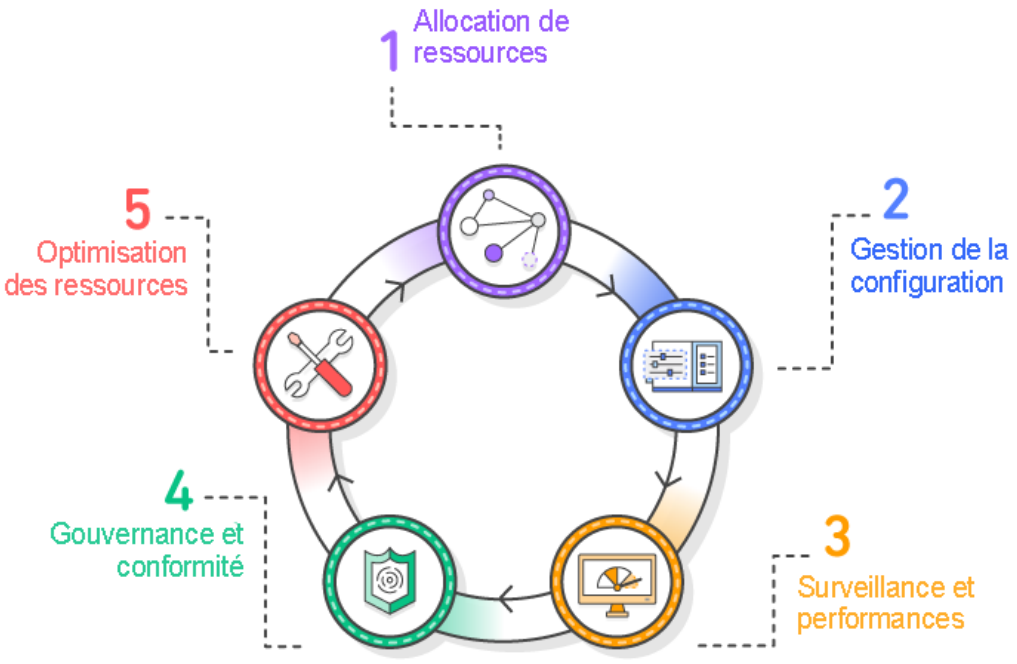

Figure 1 : Cycle de vie des ressources d'infrastructure

La figure 1 illustre une vision courante du cycle de vie des ressources d'infrastructure au sein d'une organisation. Les étapes du cycle de vie sont les suivantes :

1. **Mise en service de ressources.** Les administrateurs mettent en service les ressources selon les spécifications voulues.
2. **Gérez de la configuration.** Les ressources deviennent les composants d'un système de gestion de la configuration qui prend en charge les activités telles que le réglage et l'application des correctifs.

3. **Surveillance et performances.** Les outils de surveillance et performances valident l'état opérationnel des ressources en examinant des éléments tels que les métriques, les transactions synthétiques et les fichiers journaux.
4. **Conformité et gouvernance.** Les infrastructures de conformité et de gouvernance apportent une validation supplémentaire pour garantir l'alignement avec les normes du secteur et de l'entreprise ainsi que les exigences réglementaires.
5. **Optimisation des ressources.** Les administrateurs examinent les données de performances et identifient les modifications nécessaires pour optimiser l'environnement autour des critères tels que les performances et la gestion des coûts.

Chaque étape implique des procédures qui peuvent exploiter le code. Cela décuple les avantages de l'infrastructure en tant que code depuis son rôle traditionnel de mise en service de ressources jusqu'à l'ensemble du cycle de vie. Chaque cycle de vie bénéficie ensuite de la cohérence et la répétabilité offertes par l'infrastructure en tant que code. Cette vision élargie de l'infrastructure en tant que code se traduit par un plus haut niveau de maturité dans l'organisation des services informatiques (IT) dans son ensemble.

Dans les sections suivantes, nous allons explorer chaque phase du cycle de vie - la mise en service, la gestion de la configuration, la surveillance et les performances, la gouvernance et la conformité, et l'optimisation. Nous allons examiner les différentes tâches associées à chaque étape et expliquer comment effectuer ces tâches à l'aide de toutes les fonctionnalités d'Amazon Web Services (AWS).

Mise en service de ressources

Le cycle de vie des ressources d'informations commence par la mise en service des ressources. Les administrateurs peuvent utiliser le principe de l'infrastructure en tant que code pour simplifier le processus de mise en service. Envisagez les situations suivantes :

- Un gestionnaire de publication doit créer un réplica d'un environnement de production dans le cloud, à des fins de reprise après sinistre. L'administrateur élabore un modèle de l'environnement de production qui met en service une infrastructure identique à celle de l'emplacement de reprise après sinistre.

- Un professeur veut mettre en service des ressources pour les cours chaque semestre. Les étudiants de la classe ont besoin d'un environnement qui contient les outils appropriés pour leurs études. Le professeur crée un modèle avec les composants d'infrastructure, puis instancie le modèle de ressources pour chaque étudiant en fonction des besoins.
- Un service qui souhaite respecter certaines normes de protection de l'industrie nécessite une infrastructure avec un ensemble de contrôles de sécurité chaque fois que le service est installé. L'administrateur de sécurité intègre les contrôles de sécurité dans le modèle de configuration de façon que les contrôles de sécurité soient instanciés avec l'infrastructure.
- Le gestionnaire de l'équipe de projet du logiciel a besoin de fournir des environnements de développement pour les programmeurs qui comportent les outils nécessaires et la possibilité d'agir en interface avec une plateforme d'intégration continue. Le gestionnaire crée un modèle des ressources et publie le modèle dans un catalogue de ressources. Cela permet aux membres de l'équipe mettre en service leurs propres environnements selon vos besoins.

Ces situations ont un point commun : le besoin d'un processus reproductible pour l'instanciation des ressources de manière cohérente. L'infrastructure en tant que code fournit le cadre de ce processus. Pour répondre à ce besoin, AWS propose AWS CloudFormation.[1]

AWS CloudFormation

AWS CloudFormation permet aux développeurs et aux administrateurs système de créer, gérer, mettre en service et mettre à jour un ensemble de ressources AWS liées de manière ordonnée et prévisible. AWS CloudFormation utilise des modèles écrits au format JSON ou YAML pour décrire la collection de ressources AWS (appelée une pile), leurs dépendances associées et tous les paramètres d'exécution requis. Vous pouvez utiliser un modèle à plusieurs reprises pour créer des copies identiques de la même pile de manière cohérente dans toutes les régions AWS. Après avoir déployé les ressources, vous pouvez les modifier et les mettre à jour de manière contrôlée et prévisible. En effet, vous appliquez un contrôle de version de votre infrastructure AWS semblable à celui que vous appliquez pour votre code d'application.

Anatomie du modèle

La figure 2 montre un fragment de modèle de base AWS CloudFormation au format YAML. Les modèles contiennent des paramètres, la déclaration des ressources et les sorties. Les modèles peuvent référencer les sorties d'autres modèles, ce qui permet une modularisation.

```
---
AWSTemplateFormatVersion : "version date"

Description:
  String

Parameters:
  set of parameters

Mappings :
  set of mappings

Conditions:
  set of conditions

Transform:
  set of transforms

Resources:
  set of resources

Outputs:
  set of outputs
```

Figure 2 : Structure d'un modèle YAML AWS CloudFormation

La figure 3 est un exemple de modèle AWS CloudFormation. Le modèle demande le nom d'une paire de clés Amazon Elastic Compute Cloud (EC2) auprès de l'utilisateur dans la section des paramètres.[2] La section des ressources du modèle crée ensuite une instance EC2 à l'aide de cette paire de clés, avec un groupe de sécurité EC2 qui autorise l'accès au port TCP 80 (HTTP).

```
Parameters:
  KeyName:
    Description: The EC2 key pair to allow SSH access to the
instance
    Type: AWS::EC2::KeyPair::KeyName
Resources:
  Ec2Instance:
    Type: AWS::EC2::Instance
    Properties:
      SecurityGroups: ! Ref InstanceSecurityGroup
      KeyName: ! Ref KeyName
      ImageId: ami-70065467
  InstanceSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Enable HTTP access via port 80
      SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: '80'
        ToPort: '80'
        CidrIp: 0.0.0.0/0
```

Figure 3 : Exemple d'un modèle YAML AWS CloudFormation

Jeux de modifications

Vous pouvez mettre à jour des modèles AWS CloudFormation avec le code source de l'application pour ajouter, modifier ou supprimer des ressources de pile. La fonctionnalité jeux de modifications vous permet d'afficher des modifications proposées pour une pile sans effectuer les mises à jour associées.[3] Vous pouvez contrôler la possibilité de créer et d'afficher des jeux de modifications à l'aide d'AWS Identity and Access Management (IAM).[4] Vous pouvez autoriser certains développeurs à créer et prévisualiser les jeux de modifications, tout en réservant la possibilité de mettre à jour des piles ou exécuter des jeux de modifications pour quelques-unes d'entre elles. Par exemple, vous pouvez autoriser un développeur à voir l'impact d'un modèle avant de modifier cette modification à l'étape de test.

Il existe trois étapes associées à l'utilisation de jeux de modifications.

1. Création du jeu de modifications.

   Pour créer un jeu de modifications pour une pile, envoyez les modifications apportées à un modèle ou à des paramètres à AWS CloudFormation. AWS CloudFormation génère un jeu de modifications en comparant la pile avec vos modifications.

2. Affichage du jeu de modifications.

   Vous pouvez utiliser la console AWS CloudFormation, l'interface de ligne de commande AWS ou l'API AWS CloudFormation pour afficher les jeux de modifications. La console AWS CloudFormation propose un récapitulatif des modifications, ainsi qu'une liste détaillée des modifications au format JSON. L'interface de ligne de commande AWS et les API AWS CloudFormation renvoient une liste détaillée des modifications au format JSON.

3. Exécution du jeu de modifications.

   Vous pouvez sélectionner et exécuter le jeu de modifications dans la console AWS CloudFormation, utilisez la commande `aws cloudformation execute-change-set` dans l'interface de ligne de commande AWS ou l' API `ExecuteChangeSet`.

La capacité des jeux de modifications vous permet d'aller au-delà du contrôle de version dans AWS CloudFormation, grâce à elle vous pouvez suivre ce qui va réellement changer d'une version à l'autre. Les développeurs et les administrateurs disposent d'une plus grande visibilité concernant l'impact des modifications avant leur promotion, ce qui limite le risque d'introduire des erreurs.

Modèles réutilisables

De nombreux langages de programmation proposent des moyens de modulariser le code avec des concepts tels que des fonctions et des sous-programmes. De même, AWS CloudFormation propose plusieurs méthodes pour gérer et organiser vos piles. Même si vous pouvez conserver toutes vos ressources dans une même pile, les modèles volumineux à pile unique peuvent devenir difficile à gérer. Les possibilités d'être confronté à certaines limites sont également plus nombreuses qu'avec AWS CloudFormation.[5]

Lors de la conception de l'architecture de vos piles AWS CloudFormation, vous pouvez regrouper les piles logiquement par fonction. Au lieu de créer un seul modèle qui inclut toutes les ressources dont vous avez besoin, telles que des clouds privés virtuels (Virtual Private Cloud, VPC), des sous-réseaux et des groupes de sécurité, vous pouvez utiliser des piles imbriquées ou des références entre piles. [6,7]

La fonctionnalité de pile imbriquée vous permet de créer une nouvelle ressource de pile AWS CloudFormation au sein d'un modèle AWS CloudFormation et d'établir une relation parent-enfant entre les deux piles. Chaque fois que vous créez une pile AWS CloudFormation à l'aide du modèle parent, AWS CloudFormation crée également une nouvelle pile enfant. Cette approche vous permet de partager le code d'infrastructure entre les projets tout entretenant des piles complètement distinctes pour chaque projet.

Les références entre piles permettent à une pile AWS CloudFormation d'exporter des valeurs que les autres piles AWS CloudFormation peuvent ensuite importer. Les références entre piles contribuent à un modèle orienté services avec couplage faible qui vous permet de partager un seul ensemble de ressources sur plusieurs projets.

Commande lint sur modèle

Comme avec le code d'application, les modèles AWS CloudFormation doivent passer par une forme d'analyse statique, également connu sous le nom de commande lint. L'objectif de la commande lint consiste à déterminer si le code est correct sur le plan syntaxiquement, d'identifier les erreurs potentielles, et d'évaluer le respect de ces instructions. Dans AWS CloudFormation, la commande lint valide qu'un modèle est correctement écrit au format JSON ou YAML.

AWS CloudFormation fournit l'API `ValidateTemplate` qui vérifie que le JSON ou le YAML sont corrects. [8] Si la vérification échoue, AWS CloudFormation renvoie une erreur de validation de modèle. Par exemple, vous pouvez exécuter la commande suivante pour valider un modèle stocké dans Amazon Simple Storage Service (Amazon S3):[9]

```
aws cloudformation validate-template --template-url \
    s3://examplebucket/example_template.template
```

Vous pouvez également utiliser la validation par des outils tiers. Par exemple, la commande cfn-tag effectue des évaluations sur les modèles supplémentaires pour rechercher des problèmes de sécurité potentiels. Un autre outil, cfn-check, effectue des contrôles plus en détail des spécifications de ressources pour identifier de potentielles erreurs avant leur apparition au cours la création de piles.[10,11]

Bonnes pratiques

Le manuel AWS CloudFormation User Guide fournit une liste de bonnes pratiques pour concevoir et mettre en œuvre les modèles AWS CloudFormation.[12] Nous fournissons des liens vers ces pratiques ci-dessous.

Planification et organisation

- Organisez vos piles par cycle de vie et propriété[13]
- Utilisation d'IAM pour contrôler l'accès[14]
- Modèles de réutilisation pour répliquer les piles dans des environnements multiples[15]
- Utilisez les piles imbriquées pour réutiliser les schémas de modèles courants[16]
- Utilisation de références entre piles pour exporter des ressources partagées[17]

Création de modèles

- N'incorporez pas d'informations d'identification dans vos modèles[18]
- Utilisez des types de paramètres AWS spécifiques[19]
- Utilisez des contraintes de paramètres[20]
- Utilisez AWS::CloudFormation::Init pour déployer des applications logicielles sur des instances EC2 Amazon[21]
- Utilisation des derniers scripts[22]d'assistant
- Validation des modèles avant leur utilisation[23]
- Utilisation du Parameter Store pour une gestion centralisée des paramètres dans vos modèles[24]

Gestion des piles

- Gestion de toutes les ressources d'une pile via AWS CloudFormation[25]
- Création de jeux de modification avant la mise à jour des piles[26]
- Utilisation des stratégies de pile[27]
- Utilisation d'AWS CloudTrail pour journaliser les appels AWS CloudFormation[28]
- Utilisation des révisions de code et des contrôles de révision pour gérer les modèles[29]
- Mise à jour régulière de vos instances Amazon EC2 Linux[30]

Résumé

Le cycle de vie des ressources d'information commence par la mise en service de ressources. AWS CloudFormation propose une méthode basée sur un modèle de création d'infrastructure et de gestion des dépendances entre les ressources au cours du processus de création. Avec AWS CloudFormation, vous pouvez entretenir votre infrastructure exactement comme le code source de l'application.

Gestion de configuration

Une fois que vous mettez en service vos ressources d'infrastructure et que l'infrastructure est opérationnelle, vous devez prendre en compte les besoins de gestion de configuration en cours de l'environnement. Envisagez les situations suivantes :

- Une gestionnaire de version souhaite déployer une version d'une application dans un groupe de serveurs et effectuer une restauration en cas de problèmes.
- Un administrateur système reçoit une demande d'installation d'un nouveau package de système d'exploitation dans des environnements de développeurs, tout en conservant les autres environnements intacts.
- Une application administrateur doit régulièrement mettre à jour un fichier de configuration sur l'ensemble des serveurs hébergeant une application.

Une manière de traiter ces situations consiste à revenir à la phase de mise en service, mettre en service de nouvelles ressources avec les modifications

requises, et supprimer les anciennes ressources. Cette approche, également connue sous le nom d'immuabilité d'infrastructure, permet de s'assurer que les ressources allouées sont générées à nouveau selon la base de code chaque fois qu'une modification est apportée. Cela élimine les écarts de configuration.

Il peut arriver, toutefois, que vous souhaitiez adopter une approche différente. Dans les environnements présentant des niveaux élevés de durabilité, il peut être préférable de disposer des méthodes permettant d'apporter des modifications incrémentielles aux ressources actuelles au lieu de les remettre en service. Pour répondre à ce besoin, AWS propose Amazon EC2 Systems Manager et AWS OpsWorks for Chef Automate.[31,32]

Amazon EC2 Systems Manager

Amazon EC2 Systems Manager est un ensemble de fonctionnalités courantes, qui simplifie la maintenance, la gestion, le déploiement et l'exécution de tâches opérationnelles sur des instances EC2 et des serveurs ou machines virtuelles (VM) dans les environnements sur site. Systems Manager vous permet de facilement comprendre et contrôler l'état actuel de votre instance EC2 et les configurations du système d'exploitation. Vous pouvez suivre et gérer à distance la configuration système, les niveaux de correctif du système d'exploitation, les configurations d'application et d'autres détails sur les déploiements au fur et à mesure qu'ils se produisent. Ces fonctionnalités contribuent à l'automatisation des tâches répétitives et complexes, la définition de configurations système, ce qui empêche les écarts de conformité et de gestion des logiciels Amazon EC2 et les configurations sur site.

Le tableau 1 présente les tâches que Systems Manager simplifie.

Tâches	Détails
Exécuter une commande[33]	Gérer la configuration des instances gérées à grande échelle en répartissant des commandes sur une flotte.
Inventaire[34]	Automatiser la collecte de l'inventaire logiciel à partir d'instances gérées.
State Manager[35]	Conserver les instances gérées dans un état défini et cohérent.
Fenêtre de maintenance[36]	Définir une fenêtre de maintenance pour l'exécution des tâches administratives.
Gestionnaire de correctifs[37]	Déployer automatiquement des correctifs logiciels à partir de groupes d'instances.

Tâches	Détails
Automatisation[38]	Simplifier les tâches courantes de maintenance et de déploiement, telles que la mise à jour des Amazon Machine Images (AMI).
Parameter Store[39]	Stocker, contrôler, consulter et récupérer les données de configuration, qu'il s'agisse de données en texte brut (comme des chaînes de base de données) ou de codes secrets comme des mots chiffrés avec AWS Key Management System (KMS).

Tableau 1 : Tâches Amazon EC2 Systems Manager

Structure de document

Un document Systems Manager définit les actions exécutées par Systems Manager sur vos instances gérées. Systems Manager inclut plus d'une dizaine de documents préconfigurés pour prendre en charge les fonctionnalités répertoriées dans le tableau 1. Vous pouvez également créer des documents contrôlés par versions personnalisées pour augmenter les capacités de Systems Manager. Vous pouvez définir une version par défaut et la partager entre plusieurs comptes AWS. Les étapes du document précisent l'ordre d'exécution. Tous les documents sont écrits au format JSON et comprennent à la fois les paramètres et les actions. Comme pour AWS OpsWorks for Chef Automate, les documents de Systems Manager deviennent une partie du code, intégrant l'infrastructure en tant que code dans la gestion de la configuration.

Voici un exemple de document personnalisé pour un hôte Windows. Le document utilise la commande ipconfig pour collecter la configuration réseau du nœud, puis installe MySQL.

```
{
   "schemaVersion": "2.0",
   "description": "Sample version 2.0 document v2",
   "parameters": {},
   "mainSteps": [
      {
         "action": "aws:runPowerShellScript",
         "name": "runShellScript",
         "inputs": {
            "runCommand": ["ipconfig"]
         }
      },
      {
         "action": "aws:applications",
```

```
            "name": "installapp",
            "inputs": {
               "action": "Install",
               "source":
"http://dev.mysql.com/get/Downloads/MySQLInstaller/mysql-
installer-community-5.6.22.0.msi"
            }
         }
      ]
}
```

Figure 4 : Exemple de document Systems Manager

Bonnes pratiques

Les bonnes pratiques pour chacune des fonctionnalités Systems Manager s'affichent ci-dessous.

Exécuter la commande

- Améliorez vos niveaux de sécurité en tirant parti de la fonctionnalité Exécuter la commande pour accéder à vos instances EC2, au lieu de SSH ou RDP.[40]
- Contrôlez tous les appels d'API effectués par ou au nom de la fonctionnalité Exécuter la commande à l'aide d'AWS CloudTrail.
- Utilisez la fonction de contrôle du débit dans la fonctionnalité Exécuter la commande pour effectuer une exécution des commandes indexées.[41]
- Utilisez les autorisations d'accès plus détaillées pour la fonctionnalité Exécuter la commande (et l'ensemble des fonctions Systems Manager) à l'aide des stratégies AWS Identity and Access Management (IAM).[42]

Inventaire

- Utilisez Inventory avec AWS Config pour contrôler les configurations des applications supplémentaires.

State Manager

- Mettre à jour l'agent SSM régulièrement (au moins une fois par mois) en utilisant le document AWS-UpdateSSMAgent préconfiguré.[43]
- Amorcez les instances EC2 au lancement en utilisant EC2Config pour Windows.[44]
- (Windows) Téléchargez le PowerShell ou DSC (Desired State Configuration) sur Amazon S3, et utilisez AWS-InstallPowerShellModule.
- Utilisez des balises pour créer des groupes d'applications. Ensuite, ciblez les instances qui utilisent le paramètre `Targets`, au lieu de spécifier des ID d'instance individuelle.
- Corrigez automatiquement les résultats générés par Amazon Inspector à l'aide de Systems Manager.[45]
- Utilisez un référentiel de configuration centralisée pour l'ensemble de vos documents Systems Manager, et partagez les documents dans l'ensemble de votre organisation.[46]

Fenêtres de maintenance

- Définissez une planification pour exécuter des actions perturbatrices, comme le correctif du système d'exploitation, les mises à jour du pilote ou les installations de logiciels.

Gestionnaire de correctifs

- Utilisez le gestionnaire de correctifs pour lancer les correctifs à grande échelle et augmenter la visibilité de la conformité de la flotte dans toutes vos instances EC2.

Automatisation

- Créez des runbooks en libre-service pour l'infrastructure sous forme de documents d'Automation.

- Utilisez Automation pour simplifier la création d'AMI à partir de AWS Marketplace ou personnaliser des AMI, en utilisation des documents publics ou en autorisant vos propres flux de travail.
- Utilisez les documents AWS-UpdateLinuxAmi ou AWS-UpdateWindowsAmi ou créez un document d'automatisation personnalisé pour créer et gérer des images.

Parameter Store

- Utiliser le Parameter Store pour gérer les paramètres de configuration généraux de manière centralisée.[47]
- Utiliser le Parameter Store pour des gestions secrètes, chiffrées par AWS KMS.[48]
- Utiliser le Parameter Store avec Amazon EC2 Container Service (ECS) des définitions de tâches pour stocker des codes secrets.[49]

AWS OpsWorks for Chef Automate

AWS OpsWorks for Chef Automate transmet les fonctionnalités de Chef, une plate-forme de gestion de la configuration, dans AWS. OpsWorks pour Chef Automate s'appuie sur des fonctionnalités Chef en y ajoutant des fonctionnalités supplémentaires qui prennent en charge les fonctionnalités DevOps à l'échelle. Chef repose sur le concept de *recettes*, des scripts de configuration écrits en langage Ruby qui effectuent des tâches telles que l'installation de services. Les « recettes » (recipes) Chef, telles que des modèles AWS CloudFormation, sont une forme de code source pouvant contrôlé, ce qui étend le principe de l'infrastructure en tant que code à l'étape de gestion de la configuration du cycle de vie de la ressource.

OpsWorks pour Chef Automate étend les fonctionnalités de Chef pour permettre à votre organisation de mettre en œuvre DevOps à l'échelle. OpsWorks pour Chef Automate fournit trois fonctionnalités clés que vous pouvez configurer pour prendre en charge les pratiques de DevOps : flux de travail, conformité et visibilité.

Flux de travail

Vous pouvez utiliser un flux de travail dans OpsWorks pour Chef Automate pour coordonner le développement, le test et le déploiement. Le flux de travail comprend des portails de qualité qui accorde aux utilisateurs les privilèges

appropriés pour promouvoir le code entre les différentes phases du processus de gestion des versions. Cette fonctionnalité peut être très utile pour favoriser la collaboration entre équipes. Chaque équipe peut mettre en œuvre ses propres portails afin de garantir la compatibilité entre les projets de chaque équipe.

Conformité

OpsWorks pour Chef Automate fournit des fonctionnalités qui peuvent vous aider à la conformité organisationnelle dans le cadre de la gestion de la configuration. Chef Automate peut fournir des rapports qui mettent en avant l'importance de la conformité et des risques associés. Vous pouvez également exploiter les profils de groupes connus tels que le Center for Internet Security (CIS).

Visibilité

OpsWorks pour Chef Automate fournit une meilleure visibilité sur l'état du flux de travail et la conformité au sein des projets. Un utilisateur de Chef peut créer et afficher des tableaux de bord qui fournissent des informations sur les événements associés et interroger les événements par le biais d'une interface utilisateur.

Anatomie d'une recette

Une recette Chef est constituée d'un ensemble de définitions de ressources. Les définitions décrivent l'état souhaité pour les ressources et la façon dont Chef peut les conduire à cet état. Chef prend en charge plus de 60 types de ressources. Une liste des types de ressources courants s'affiche ci-dessous.

Nom de la ressource	Objectif
Bash	Exécuter un script à l'aide de l'interpréteur bash
Directory	Gestion des annuaires
Execute	Exécuter une commande unique
File	Gestion des fichiers
Git	Gérer les ressources source dans les référentiels Git
Group	Gérer les groupes
Package	Gestion des packages
Route	Gérer une entrée de table de routage Linux
Service	Gérer un service
User	Gestion des utilisateurs

Tableau 2: Ressources Chef courantes

Voici un exemple de recette Chef. Cet exemple définit une ressource en fonction de l'installation du serveur Web Apache. La définition de ressource inclut une vérification pour le système d'exploitation sous-jacent. Il utilise l'opérateur de `cas` pour examiner la valeur de `node[:platform]` et vérifier le système d'exploitation sous-jacent. La directive `action: install` conduit la ressource à l'état désiré (autrement dit, il installe le package).

```
package 'apache2' do
  case node[:platform]
  when 'centos','redhat','fedora','amazon'
    package_name 'httpd'
  when 'debian','ubuntu'
    package_name 'apache2'
  end
  action :install
end
```

Figure 5 : Exemple de recette Chef

Linting et test de recette

Des outils variés disponibles à la fois auprès de Chef et de la communauté d'utilisateurs de Chef prennent en charge la commande Lint de Chef (vérification de syntaxe) et le test d'unité et d'intégration. Nous mettons en lumière certaines des plateformes les plus courantes dans les sections suivantes.

Linting avec Rubocop et Foodcritic

Le Linting est également possible sur du code d'infrastructure tel que des recettes Chef, comme Rubocop et Foodcritic.[50,51,52] Rubocop procède à une analyse statique des « recettes » (recipes) Chef en se basant sur le guide de style Ruby. (Ruby est le langage utilisé pour créer les recettes Chef.) Cet outil fait partie du kit de développement Chef et peut être intégré dans le flux de travail de développement de logiciels. Foodcritic vérifie les recettes Chef à la rechercher d'erreurs de syntaxe en s'appuyant sur un ensemble de règles préintégrées, pouvant être enrichi par les contributions de la communauté.

Test d'unité avec ChefSpec

ChefSpec peut fournir des tests d'unités sur les livres de cuisine Chef.[53] Ces tests peuvent déterminer si Chef doit effectuer des tâches appropriées pour atteindre

les objectifs. ChefSpec nécessite une spécification de test de configuration qui est ensuite évaluée par rapport à une recette.

Par exemple, ChefSpec ne vérifie pas réellement si Chef a installé le package Apache,il vérifie plutôt si une recette Chef a demandé l'installation d'Apache. L'objectif du test consiste à valider si la recette reflète les intentions du programmeur.

Test d'intégration avec Test Kitchen

Test Kitchen est une plateforme de test qui crée des environnements de test et utilise ensuite des *bussers*, qui sont des systèmes de test, pour valider la création des ressources spécifiées dans les recettes Chef.[54]

En tirant profit des outils de test précédents conjointement avec les fonctionnalités de flux de travail d'OpsWorks pour Chef Automate, les développeurs peuvent automatiser le contrôle de leurs infrastructures au cours du cycle de développement. Ces tests sont une forme de code eux-mêmes. Ils sont un autre élément clé de l'approche d'infrastructure en tant que code pour les déploiements.

Bonnes pratiques

Les stratégies, techniques et suggestions présentées ici vous aideront à optimiser les avantages et les résultats obtenus avec AWS OpsWorks for Chef Automate :

- Imaginez que vous stockez vos recettes Chef dans une archive Amazon S3. Amazon S3 est extrêmement fiable et durable. Versionnez chaque fichier d'archive de façon explicite en suivant une convention d'appellation. Vous pouvez également utiliser le système de gestion des versions d'Amazon S3, qui fournit un journal de suivi et un moyen simple de revenir à une version antérieure.
- Établissez un calendrier de sauvegarde qui répond à vos exigences en matière de gouvernance d'organisation.
- Utilisez IAM pour limiter l'accès aux appels d'API OpsWorks pour Chef Automate.

Résumé

Amazon EC2 Systems Manager vous permet de déployer, personnaliser, appliquer et contrôler l'état attendu d'une configuration pour vos instances EC2 et des serveurs ou machines virtuelles de votre environnement sur site. AWS OpsWorks for Chef Automate vous permet d'utiliser les recettes Chef pour prendre en charge la configuration d'un environnement. Vous pouvez utiliser OpsWorks pour Chef Automate indépendamment ou en plus d'un environnement mis en service par AWS CloudFormation. L'exécution des documents et des stratégies associées à Systems Manager et les recettes associées à OpsWorks pour Chef Automate peuvent devenir une partie du code de l'infrastructure de base et être contrôlées comme le code source de l'application.

Surveillance et performances

Après avoir examiné le rôle de l'infrastructure en tant que code dans la mise en service de ressources d'infrastructure et de gestion de la configuration, observons maintenant l'infrastructure. Songez à la façon dont les événements suivants peuvent affecter le fonctionnement d'un site web pendant les périodes de pic de demande :

- Les utilisateurs d'une application Web présentant des délais d'attente en raison de la latence de l'équilibreur de charge, il est difficile de parcourir les catalogues de produits.
- Un serveur d'application voit ses performances dégradées en raison d'un manque de capacité d'UC et ne peut plus traiter de nouvelles commandes.
- Une base de données qui permet de suivre l'état des sessions ne dispose pas de suffisamment de débit. Cela entraîne des retards tandis que les utilisateurs passent par les différentes étapes d'une application.

Ces situations décrivent les problèmes opérationnels résultant des ressources d'infrastructure qui ne répondent pas aux attentes en matière de performances. Il est important de capturer des mesures clés pour évaluer l'état de santé de l'environnement et prendre des mesures correctives en cas de problèmes. Les métriques offrent de la visibilité. Avec les métriques, votre organisation peut réagir automatiquement aux événements. Sans les métriques, votre organisation est aveugle à ce qui se passe dans son infrastructure, ce qui nécessite une intervention humaine pour résoudre tous les problèmes. Avec les

systèmes scalables et à couplage faible, écrits dans plusieurs langages et infrastructures, il peut être difficile de capturer des métriques et journaux pertinents et de réagir en conséquence. Pour répondre à ce besoin, AWS propose la solution Amazon CloudWatch .[55]

Amazon CloudWatch

Amazon CloudWatch est un ensemble de services qui intègre, interprète et répond aux métriques d'exécution, aux journaux et aux événements. CloudWatch recueille automatiquement les mesures de nombreux services AWS, tels qu'Amazon EC2, Elastic Load Balancing (ELB) et Amazon DynamoDB[56],[57],[58] les réponses peuvent inclure des actions intégrées telles que l'envoi de notifications ou les actions personnalisées gérées par AWS Lambda, une plateforme de calcul orientée événement sans serveur.[59] Le code des fonctions Lambda fait partie du code d'infrastructure de base, ce qui élève l'infrastructure en tant que code au niveau opérationnel. CloudWatch se compose de trois services : le principal service CloudWatch, Amazon CloudWatch Logs et Amazon CloudWatch Events. Nous allons maintenant aborder chacun de ces éléments plus en détail.

Amazon CloudWatch

Le principal service Amazon CloudWatch collecte et suit les métriques pour de nombreux services AWS tels qu'Amazon EC2, ELB, DynamoDB et Amazon Relational Database Service (RDS). Vous pouvez également créer des métriques personnalisées pour les services que vous développez, telles que les applications. CloudWatch émet des alarmes lorsque les métriques atteignent un seuil donné au cours d'une période.

Voici quelques exemples des métriques et de solutions possibles pouvant s'appliquer aux situations mentionnées au début de cette section :

- Si la latence de ELB dépasse 5 secondes de plus de deux minutes, envoyer une notification par e-mail aux administrateurs système.
- Lorsque l'utilisation moyenne de l'UC de l'instance EC2 dépasse 60 % pendant trois minutes, lancer une autre instance EC2.
- Augmenter la capacité d'une table DynamoDB en cas de limitation excessive.

Vous pouvez mettre en œuvre des réponses aux alarmes basées sur les métriques à l'aide de notifications intégrées, ou en écrivant des fonctions Lambda personnalisées dans Python, Node.js, Java ou C #. La figure 6 illustre un exemple de la façon dont une alarme CloudWatch utilise Amazon Simple Notification Service (Amazon SNS) pour déclencher une capacité DynamoDB mise à jour.

Figure 6 : Exemple de flux d'alarme CloudWatch

Amazon CloudWatch Logs

Amazon CloudWatch Logs surveille et stocke les journaux d'Amazon EC2, d'AWS CloudTrail ou d'autres sources. Les instances EC2 peuvent envoyer des informations de journalisation à l'aide de l'agent CloudWatch Logs et des outils de journalisation tels que Logstash, Graylog et Fluentd.[60] Les journaux stockés dans Amazon S3 peuvent être envoyés à CloudWatch Logs par la configuration d'un événement Amazon S3 pour déclencher une fonction Lambda.

Les données de journaux intégrées peuvent servir de base aux nouvelles mesures CloudWatch qui peuvent, à leur tour, déclencher des alarmes CloudWatch. Vous pouvez utiliser cette fonctionnalité pour surveiller une ressource qui génère des journaux sans écrire de code. Si vous avez besoin d'une procédure de réponse plus avancée, vous pouvez créer une fonction Lambda pour effectuer les actions appropriées. Par exemple, une fonction Lambda peut utiliser les API SNS.Publish ou SES.SendEmail pour publier des informations vers un canal Slack à chaque erreur `NullPointerException` apparaissant dans les journaux de production.[61,62]

Le traitement et la mise en corrélation des fichiers journaux permettent une analyse plus approfondie des comportements d'applications et peuvent mettre en lumière des détails qui sont difficiles à repérer à partir de métriques. CloudWatch Logs fournit à la fois le stockage et l'analyse des journaux, et le traitement des données permettant des réponses basées sur les données aux problèmes opérationnels.

Amazon CloudWatch Events

Amazon CloudWatch Events fournit un flux d'événements à partir de modifications apportées aux environnements AWS, applique un moteur de règles, et fournit des événements correspondant à des cibles spécifiés. Parmi les exemples d'événements pouvant être diffusés en continu figurent les modifications d'état d'instance EC2, les actions Auto Scaling, les appels d'API publiés par CloudTrail, les connexions à la console AWS, les notifications d'optimisation AWS Trusted Advisor, les événements de niveau application personnalisés et les actions planifiées. Les cibles peuvent comprendre des actions intégrées telles que les notifications SNS ou les réponses personnalisées à l'aide de fonctions Lambda.

La capacité d'une infrastructure à répondre à certains événements présente des avantages à la fois dans les opérations et dans la sécurité. Du point de vue des opérations, les événements peuvent automatiser les opérations de maintenance, sans qu'il soit nécessaire de gérer un système de planification distinct. En ce qui concerne la sécurité des informations, les événements peuvent fournir des notifications aux identifiants de la console, des échecs d'authentification, et des appels d'API dangereux enregistrés par CloudTrail. Dans les deux domaines, l'intégration de réponses à des événements dans le code d'infrastructure contribue à un degré de réparation automatique plus élevé ainsi qu'à une plus grande maturité opérationnelle.

Bonnes pratiques

Voici quelques recommandations pour les meilleures pratiques liées à la surveillance :

- Assurez-vous que toutes les ressources AWS produisent des métriques.
- Créez des alarmes CloudWatch pour les métriques qui fournissent des réponses appropriées en cas d'événements liés à la métrique.
- Envoyez les journaux de ressources AWS, y compris Amazon S3 et Amazon EC2, à CloudWatch Logs pour analyse à l'aide de déclencheurs de flux de journaux et de fonctions Lambda.
- Planifiez des tâches de maintenance continue avec CloudWatch et Lambda.
- Utilisez des événements personnalisés CloudWatch pour réagir aux problèmes au niveau de l'application.

Résumé

La surveillance est essentielle pour comprendre le comportement des systèmes et pour automatiser les réactions guidées par les données. CloudWatch collecte les observations des environnements d'exécution, sous la forme de métriques et de journaux et permet de les activer par des alarmes, des flux et des événements. Les fonctions Lambda écrites en Python, Node.js, Java ou C # peuvent répondre à des événements, ce qui élargit le rôle de l'infrastructure en tant que code au domaine opérationnel et améliore la résilience des systèmes d'exploitation.

Gouvernance et conformité

Après avoir examiné la façon dont vous pouvez utiliser l'infrastructure en tant que code pour surveiller l'état des environnements de votre organisation, nous allons maintenant nous concentrer sur la technologie de gouvernance et de conformité. De nombreuses organisations nécessitent une visibilité sur leurs infrastructures pour répondre aux exigences réglementaires ou du secteur. Les capacités de dimensionnement dynamique du cloud posent des problèmes particuliers dans la mesure où la visibilité et la gouvernance doivent être conservées à mesure que des ressources sont ajoutées, supprimées ou mises à jour. Envisagez les situations suivantes :

- Un utilisateur est ajouté à un groupe d'administration privilégié, et le service informatique n'est pas en mesure d'expliquer à quel moment il l'a été.
- Les règles d'accès au réseau qui limite la gestion à distance à un ensemble défini d'adresses IP sont modifiées pour autoriser l'accès à partir d'emplacements supplémentaires.
- Les configurations de la mémoire vive et de l'UC pour plusieurs serveurs ont doublé de manière inattendue, ce qui se traduit par une facture beaucoup plus importante que les mois précédents.

Même si vous avez une meilleure visibilité sur l'état actuel de vos configurations de ressources AWS grâce à l'interface de ligne de commande AWS et aux appels d'API, pour réagir devant ces situations, il est impératif de pouvoir observer de quelle manière ces ressources ont évolué dans le temps. Pour répondre à ce besoin, AWS propose le service AWS Config.[63]

AWS Config

AWS Config vous permet d'estimer, auditer et évaluer les configurations de vos ressources AWS. AWS Config crée automatiquement un inventaire de vos ressources et effectue le suivi des modifications qui leur sont apportées.
La figure 7 illustre un exemple d'inventaire d'instances EC2 AWS Config.

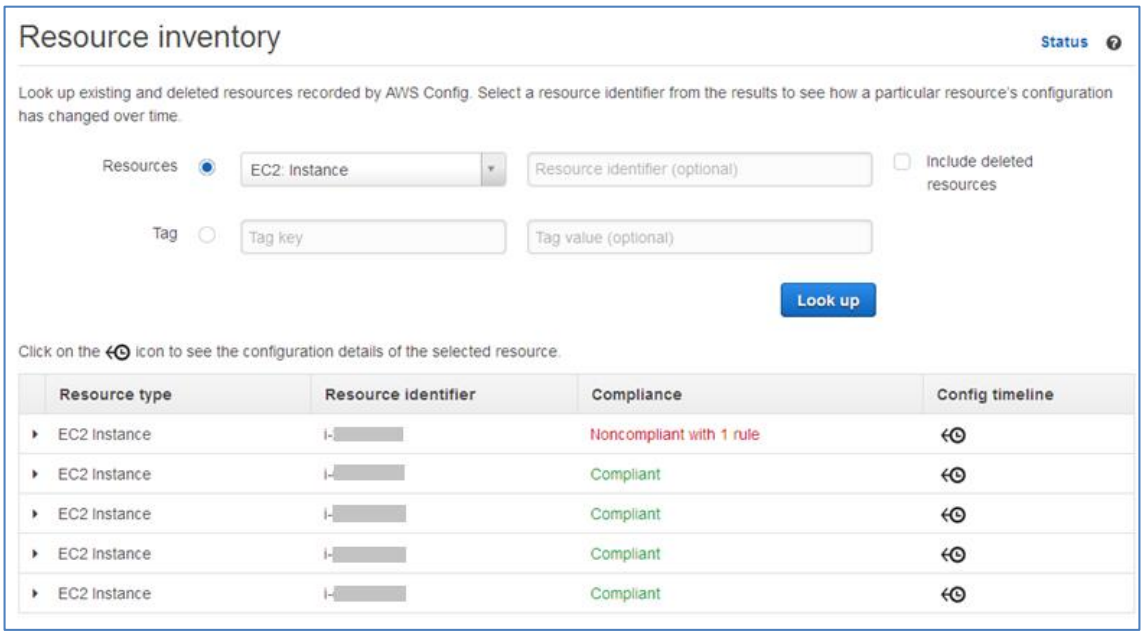

Figure 7 : Exemple d'inventaire de ressources AWS Config

AWS Config fournit également une bonne visibilité de la chronologie des modifications au niveau des ressources, y compris les modifications dans les configurations de ressources et les associations de ces ressources à d'autres ressources AWS. Figure 8 affiche les informations conservées par AWS Config pour une ressource VPC.

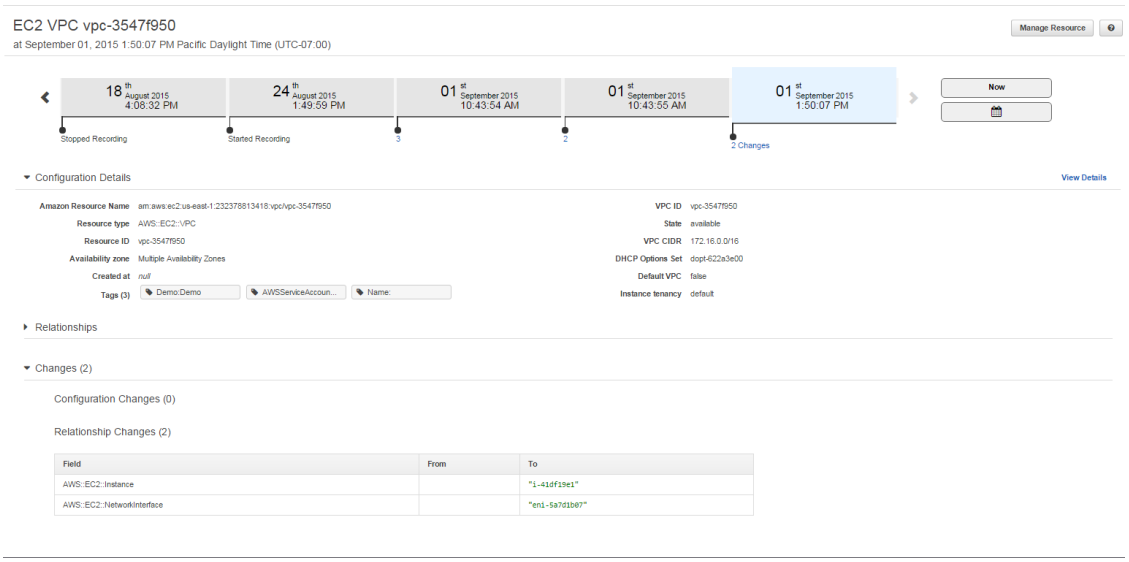

Figure 8 : Exemple de chronologie de ressources AWS Config

Lorsque plusieurs ressources changent fréquemment et automatiquement, l'automatisation de la mise en conformité peut devenir aussi importante que l'automatisation du pipeline de livraison. Pour répondre à l'évolution de l'environnement, vous pouvez utiliser les règles AWS Config.

AWS Config Rules

Avec les règles AWS Config, chaque modification doit déclencher une évaluation par rapport aux règles associées aux ressources. AWS fournit un ensemble de règles gérées pour les exigences communes telles que des utilisateurs IAM ayant les bons mots de passe, les groupes et les politiques, ou pour déterminer si les instances EC2 sont correctes sur le VPC et les groupes de sécurité. Les règles AWS Config peuvent identifier rapidement les ressources non conformes et aider à leur signalement et aux mesures correctives. Pour les validations au-delà de celles fournies par les règles gérées, AWS Config Rules prend également en charge la création de règles personnalisées à l'aide de fonctions Lambda.[64] Ces règles deviennent partie de la base de code d'infrastructure, transmettant le concept de l'infrastructure en tant que code aux étapes de gouvernance et de conformité du cycle de vie des ressources d'informations.

Structure de règle

Lorsqu'une règle personnalisée est appelée via les règles AWS Config, la fonction Lambda associée reçoit les événements de configuration, les traite et renvoie les résultats. La fonction suivante détermine si les journaux de flux Amazon Virtual Private Cloud (Amazon VPC) sont activés sur un VPC Amazon.

```
import boto3
import json

def evaluate_compliance (config_item, vpc_id):
    if (config_item['resourceType'] != 'AWS::EC2::VPC'):
        return 'NOT_APPLICABLE'
    elif is_flow_logs_enabled(vpc_id):
        return 'COMPLIANT'
    else:
        return 'NON_COMPLIANT'

def is_flow_logs_enabled(vpc_id):
    ec2 = boto3.client('ec2')
    response = ec2.describe_flow_logs(
        Filter=[{'Name': 'resource-id','Values': [vpc_id]},],
    )
    if len(response[u'FlowLogs']) != 0: return True

def lambda_handler(event, context):
    invoking_event = json.loads(event['invokingEvent'])
    compliance_value = 'NOT_APPLICABLE'
    vpc_id = invoking_event['configurationItem']['resourceId']
    compliance_value = evaluate_compliance(
        invoking_event['configurationItem'], vpc_id)

    config = boto3.client('config')
    response = config.put_evaluations(
       Evaluations=[
           {
               'ComplianceResourceType':
invoking_event['configurationItem']['resourceType'],
               'ComplianceResourceId': vpc_id,
               'ComplianceType': compliance_value,
               'OrderingTimestamp':
invoking_event['configurationItem']['configurationItemCaptureTim
e']
           },
       ],
       ResultToken=event['resultToken'])
```

Figure 9 : Exemple d'une fonction Lambda pour prendre en charge AWS Config Rules

Dans cet exemple, lorsqu'un événement de configuration sur un VPC Amazon se produit, l'événement est transmis à la fonction lambda_handler. Ce code extrait l'ID du VPC Amazon et utilise l'appel d'API describe_flow_logs pour déterminer si les journaux de flux sont activés. La fonction Lambda renvoie une valeur COMPLIANT si les journaux de flux sont activés et NON_COMPLIANT dans le cas contraire.

Bonnes pratiques

Voici quelques recommandations pour mettre en œuvre les solutions AWS Config dans vos environnements :

- Permettre à AWS Config pour toutes les régions d'enregistrer l'historique des éléments de configuration, afin de faciliter l'audit et le suivi de conformité.
- Mettre en place un processus de réaction aux modifications détectées par AWS Config. Cela peut comprendre des notifications par e-mail et l'utilisation de règles AWS Config pour répondre à des modifications par programmation.

Résumé

AWS Config étend le concept de code d'infrastructure au domaine de la gouvernance et de la conformité. AWS Config peut enregistrer en continu la configuration des ressources tandis que les règles AWS Config prévoient des réponses basées sur les événements en réaction aux modifications de la configuration de ressources. Vous pouvez utiliser cette fonctionnalité pour assister votre organisation dans la surveillance des contrôles de conformité.

Optimisation des ressources

Concentrons-nous maintenant sur la dernière étape du cycle de vie des ressources d'informations, l'optimisation de vos ressources. Au cours de cette étape, les administrateurs passent en revue les données de performances et identifient les modifications nécessaires pour optimiser l'environnement autour de critères tels que la sécurité, les performances et la gestion des coûts. Il est important pour les parties impliquées dans toutes les applications d'évaluer régulièrement l'infrastructure afin de déterminer si elle est utilisée de manière optimale.

Tenez compte des questions suivantes :

- Certaines ressources mises en place sont-elles sous-utilisées ?
- Existe-t-il des moyens de réduire les frais associés à l'environnement d'exécution ?
- Des suggestions ont-elles été apportées pour améliorer les performances des ressources mises en ressources ?
- Des limites de service s'appliquent-elles aux ressources utilisées dans l'environnement et, si tel est le cas, l'utilisation actuelle de ressources est-elle sur le point de dépasser ces limites ?

Pour répondre à ces questions, nous avons besoin d'un moyen d'interroger l'environnement d'exécution, d'extraire les données relatives à l'optimisation, et d'utiliser ces données pour prendre des décisions pertinentes. Pour ce faire, AWS propose AWS Trusted Advisor.[65]

AWS Trusted Advisor

AWS Trusted Advisor vous permet de suivre les bonnes pratiques en matière de sécurité en analysant vos ressources AWS et en comparant leur utilisation par rapport aux bonnes pratiques selon quatre catégories : optimisation des coûts, performances, sécurité et tolérance aux pannes. Dans le cadre de l'amélioration continue de votre infrastructure et de vos applications, Trusted Advisor peut vous aider à faire en sorte que vos ressources soient en permanence mises en service de façon optimale. La figure 10 illustre un exemple de tableau de bord de Trusted Advisor.

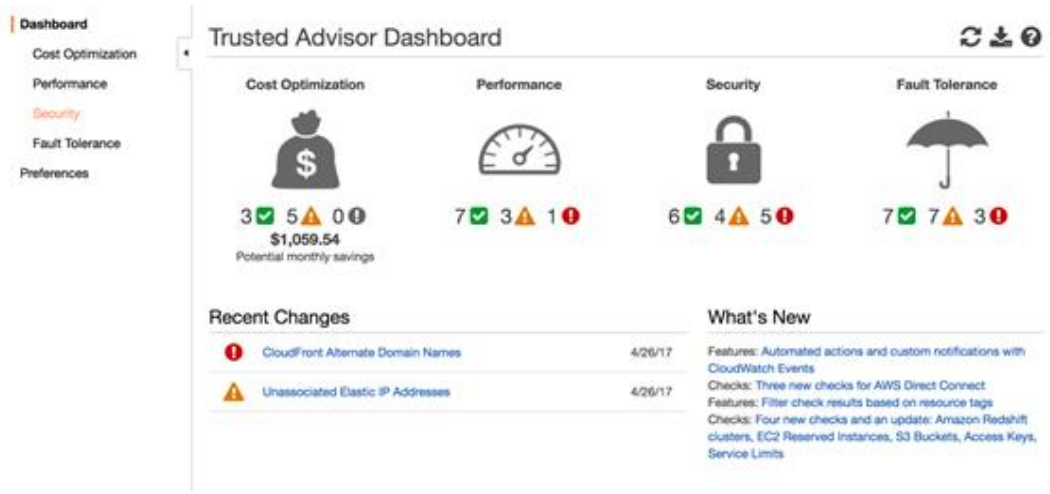

Figure 10 : Exemple de tableau de bord d'AWS Trusted Advisor

Vérifications

Trusted Advisor fournit une grande variété de vérifications visant à déterminer si l'infrastructure suit les bonnes pratiques. Les contrôles comprennent des descriptions détaillées de bonnes pratiques recommandées, des critères d'alerte, des recommandations d'action et une liste des ressources utiles concernant la rubrique. Trusted Advisor fournit les résultats des vérifications et peut également fournir des notifications hebdomadaires en continu pour les mises à jour de statut et des économies de coûts.

Tous les clients ont accès à un ensemble de bases de vérifications Trusted Advisor. Les clients bénéficiant d'un support AWS Business ou Enterprise peuvent accéder à toutes les vérifications Trusted Advisor et aux API Trusted Advisor. À l'aide des API, vous pouvez obtenir des informations provenant de Trusted Advisor et prendre des mesures correctives. Par exemple, un programme peut exploiter Trusted Advisor pour examiner les limites de service du compte actuel. Si l'utilisation des ressources actuelles approche des limites, vous pouvez créer automatiquement une demande de support pour augmenter les limites.

En outre, Trusted Advisor est désormais intégré à Amazon CloudWatch Events. Vous pouvez concevoir une fonction Lambda pour notifier un canal Slack quand le statut des vérifications Trusted Advisor change. Ces exemples illustrent la façon dont le concept de l'infrastructure en tant que code peut être étendu à l'optimisation des ressources au niveau du cycle de vie des ressources d'informations.

Bonnes pratiques

Les bonnes pratiques pour AWS Trusted Advisor sont répertoriées ci-dessous.

- Abonnez-vous aux notifications Trusted Advisor par e-mail ou un autre système de diffusion.
- Utilisez des listes de distribution et assurez-vous que les destinataires concernés sont inclus pour toutes ces notifications.
- Si vous bénéficiez d'un support AWS Business ou Enterprise, utilisez l'API AWS Support conjointement avec les notifications Trusted Advisor pour créer des demandes d'actions correctives auprès d'AWS Support.

Résumé

Vous devez continuellement surveiller votre infrastructure pour optimiser les ressources d'infrastructure concernant les performances, la sécurité et les coûts. AWS Trusted Advisor offre la possibilité d'utiliser des API pour interroger votre infrastructure AWS pour les recommandations, par conséquent, l'extension de l'infrastructure en tant que code à la phase d'optimisation du cycle de vie des informations de ressources.

Étapes suivantes

Au moment d'adopter l'infrastructure en tant que code dans votre organisation, vous pouvez commencer par consulter les spécifications de votre infrastructure de la même manière que vous consultez votre code produit. AWS propose un large éventail d'outils qui vous donnent plus de contrôle et de flexibilité sur la façon de mettre en service, gérer et rendre opérationnelle votre infrastructure dans le cloud.

Voici quelques actions clés que vous pouvez exécuter lorsque vous mettez en œuvre l'infrastructure en tant que code dans votre organisation :

- Démarrez à l'aide d'un service de contrôle de source gérée, tel que AWS CodeCommit, pour votre code d'infrastructure.
- Intégrez un processus de contrôle qualité via des tests unitaires et l'analyse du code statique avant le déploiement.

- Supprimez l'élément humain et automatisez la mise en service de l'infrastructure, y compris les stratégies d'autorisation d'infrastructure.
- Créez un code d'infrastructure idempotent que vous pouvez facilement redéployer.
- Déployer chaque nouvelle mise à jour dans votre infrastructure via le code en mettant à jour vos piles idempotentes. Évitez d'effectuer des modifications ponctuelles manuellement.
- Adoptez l'automatisation de bout en bout.
- Intégrez le travail d'automatisation de l'infrastructure dans la dernière ligne droite classique des produits.
- Effectuez des modifications contrôlables, et rendez la journalisation obligatoire.
- Définissez des normes courantes au sein de votre organisation et optimisez en continu.

En adoptant ces principes, votre infrastructure peut évoluer de manière dynamique et accélérer au rythme de vos besoins métiers en constante évolution.

Conclusion

L'infrastructure en tant que code vous permet d'encoder la définition de ressources d'infrastructure dans des fichiers de configuration et des versions de contrôle, tout comme le logiciel d'application. Nous pouvons désormais mettre à jour notre schéma de cycle de vie et illustrer de quelle manière AWS prend en charge chaque étape à l'aide du code.

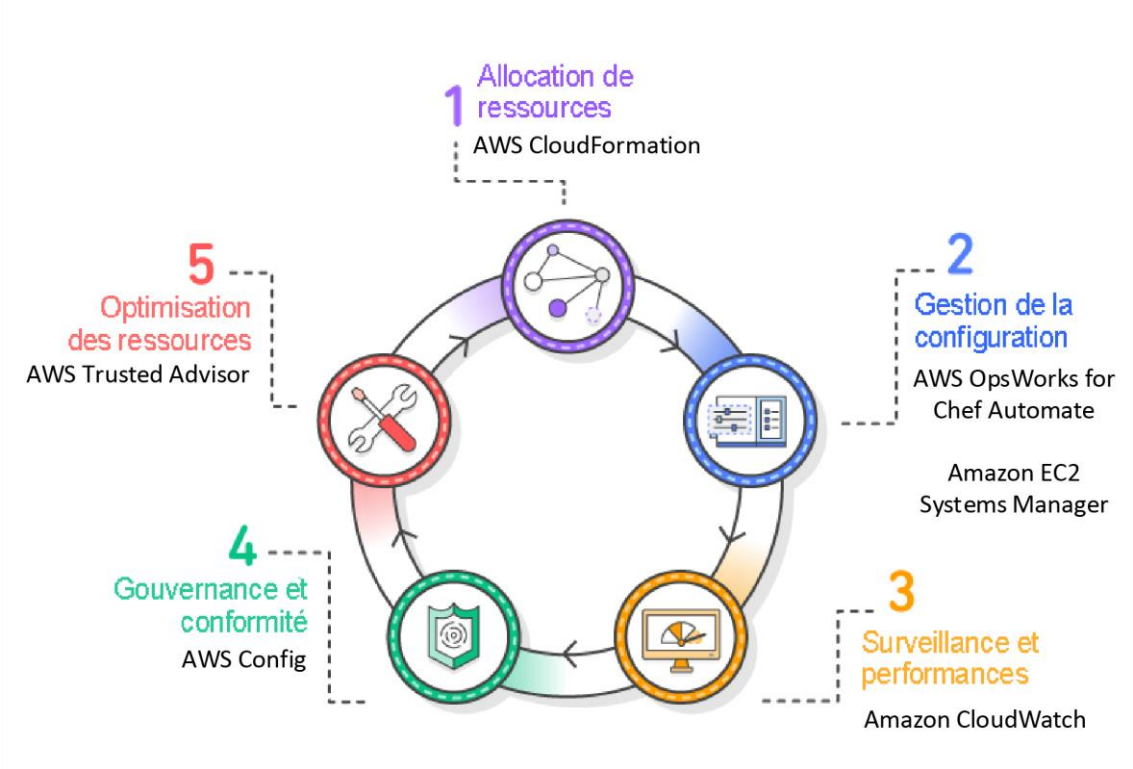

Figure 11 : Informations du cycle de vie des ressources avec AWS

AWS CloudFormation, AWS OpsWorks for Chef Automate, Amazon EC2 Systems Manager, Amazon CloudWatch, AWS Config et AWS Trusted Advisor vous permettent d'intégrer le concept de l'infrastructure en tant que code dans toutes les phases du cycle de vie des projets. En utilisant l'infrastructure en tant que code, votre entreprise peut déployer automatiquement et de manière cohérente des environnements intégrés, ce qui peut aider votre entreprise à améliorer sa maturité globale.

Participants

Les personnes et organisations suivantes ont participé à l'élaboration de ce document :

- Hubert Cheung, architecte de solutions, Amazon Web Services
- Julio Faerman, évangéliste technologique, Amazon Web Services
- Balaji Iyer, consultant en services professionnels Balaji, Amazon Web Services
- Jeffrey S. Levine, architecte de solutions, Amazon Web Services

Ressources

Consultez les ressources suivantes pour en savoir plus sur nos bonnes pratiques en matière d'infrastructure en tant que code.

Vidéos

- AWS re:Invent 2015 – DevOps at Amazon[66]
- AWS Summit 2016 - DevOps, Continuous Integration and Deployment on AWS[67]

Documentation et blogs

- DevOps et AWS[68]
- Qu' est-ce que l'intégration continue[69]
- Qu' est-ce que la livraison continue[70]
- AWS DevOps Blog[71]

Livres blancs

- Introduction to DevOps on AWS[72]
- AWS Operational Checklist[73]
- Bonnes pratiques de sécurité AWS[74]
- Risques et conformité d'AWS :[75]

AWS Support

- AWS Premium Support[76]
- AWS Trusted Advisor[77]

Remarques

[1] https://aws.amazon.com/cloudformation/

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

[3] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/using-cfn-updating-stacks-changesets.html

[4] http://aws.amazon.com/iam

[5] https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cloudformation-limits.html

[6] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-properties-stack.html

[7] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/walkthrough-crossstackref.html

[8] http://docs.aws.amazon.com/AWSCloudFormation/latest/APIReference/API_ValidateTemplate.html

[9] http://aws.amazon.com/s3

[10] https://stelligent.com/2016/04/07/finding-security-problems-early-in-the-development-process-of-a-cloudformation-template-with-cfn-nag/

[11] https://www.npmjs.com/package/cfn-check

[12] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html

[13] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#organizingstacks

[14] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#use-iam-to-control-access

[15] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#reuse

[16] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#nested

[17] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#cross-stack

[18] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#creds

[19] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#parmtypes

[20] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#parmconstraints

[21] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#cfninit

[22] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#helper-scripts

[23] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#validate

[24] https://aws.amazon.com/ec2/systems-manager/parameter-store/

[25] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#donttouch

[26] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#cfn-best-practices-changesets

[27] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#stackpolicy

[28] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#cloudtrail

[29] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#code

[30] http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/best-practices.html#update-ec2-linux

[31] https://aws.amazon.com/ec2/systems-manager/

[32] https://aws.amazon.com/opsworks/chefautomate/

[33] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/execute-remote-commands.html

[34] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/systems-manager-inventory.html

[35] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/systems-manager-state.html

[36] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/systems-manager-ami.html

[37] https://aws.amazon.com/ec2/systems-manager/patch-manager/

[38] https://aws.amazon.com/ec2/systems-manager/automation/

[39] https://aws.amazon.com/ec2/systems-manager/parameter-store/

[40] https://aws.amazon.com/blogs/mt/replacing-a-bastion-host-with-amazon-ec2-systems-manager/

[41] http://docs.aws.amazon.com/systems-manager/latest/userguide/send-commands-multiple.html

[42] http://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-configuring-access-iam-create.html

[43] https://aws.amazon.com/blogs/mt/replacing-a-bastion-host-with-amazon-ec2-systems-manager/

[44] http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/ec2-configuration-manage.html

[45] https://aws.amazon.com/blogs/security/how-to-remediate-amazon-inspector-security-findings-automatically/

[46] http://docs.aws.amazon.com/systems-manager/latest/userguide/ssm-sharing.html

[47] http://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-paramstore.html

[48] http://docs.aws.amazon.com/systems-manager/latest/userguide/sysman-paramstore-walk.html

[49] https://aws.amazon.com/blogs/compute/managing-secrets-for-amazon-ecs-applications-using-parameter-store-and-iam-roles-for-tasks/

[50] https://en.wikipedia.org/wiki/Lint_(software)

[51] https://docs.chef.io/rubocop.html

[52] https://docs.chef.io/foodcritic.html

[53] https://docs.chef.io/chefspec.html

[54] https://docs.chef.io/kitchen.html

[55] https://aws.amazon.com/cloudwatch/

[56] https://aws.amazon.com/dynamodb/

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

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

[59] https://aws.amazon.com/lambda/

[60] http://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/QuickStartEC2Instance.html

[61] http://docs.aws.amazon.com/ses/latest/APIReference/API_SendEmail.html

[62] http://docs.aws.amazon.com/sns/latest/api/API_Publish.html

[63] https://aws.amazon.com/config/

[64] http://docs.aws.amazon.com/config/latest/developerguide/evaluate-config_develop-rules.html

[65] https://aws.amazon.com/premiumsupport/trustedadvisor/

[66] https://www.youtube.com/watch?v=esEFaYoFDKc

[67] https://www.youtube.com/watch?v=DurzNeBQ-WU

[68] https://aws.amazon.com/devops/

[69] https://aws.amazon.com/devops/continuous-integration/

[70] https://aws.amazon.com/devops/continuous-delivery/

[71] https://aws.amazon.com/blogs/devops/

[72] https://do.awsstatic.com/whitepapers/AWS_DevOps.pdf

[73] https://media.amazonwebservices.com/AWS_Operational_Checklists.pdf

[74] https://do.awsstatic.com/whitepapers/Security/AWS_Security_Best_Practices.pdf

[75] https://do.awsstatic.com/whitepapers/compliance/AWS_Risk_and_Compliance_Whitepaper.pdf

[76] https://aws.amazon.com/premiumsupport/

[77] https://aws.amazon.com/premiumsupport/trustedadvisor/