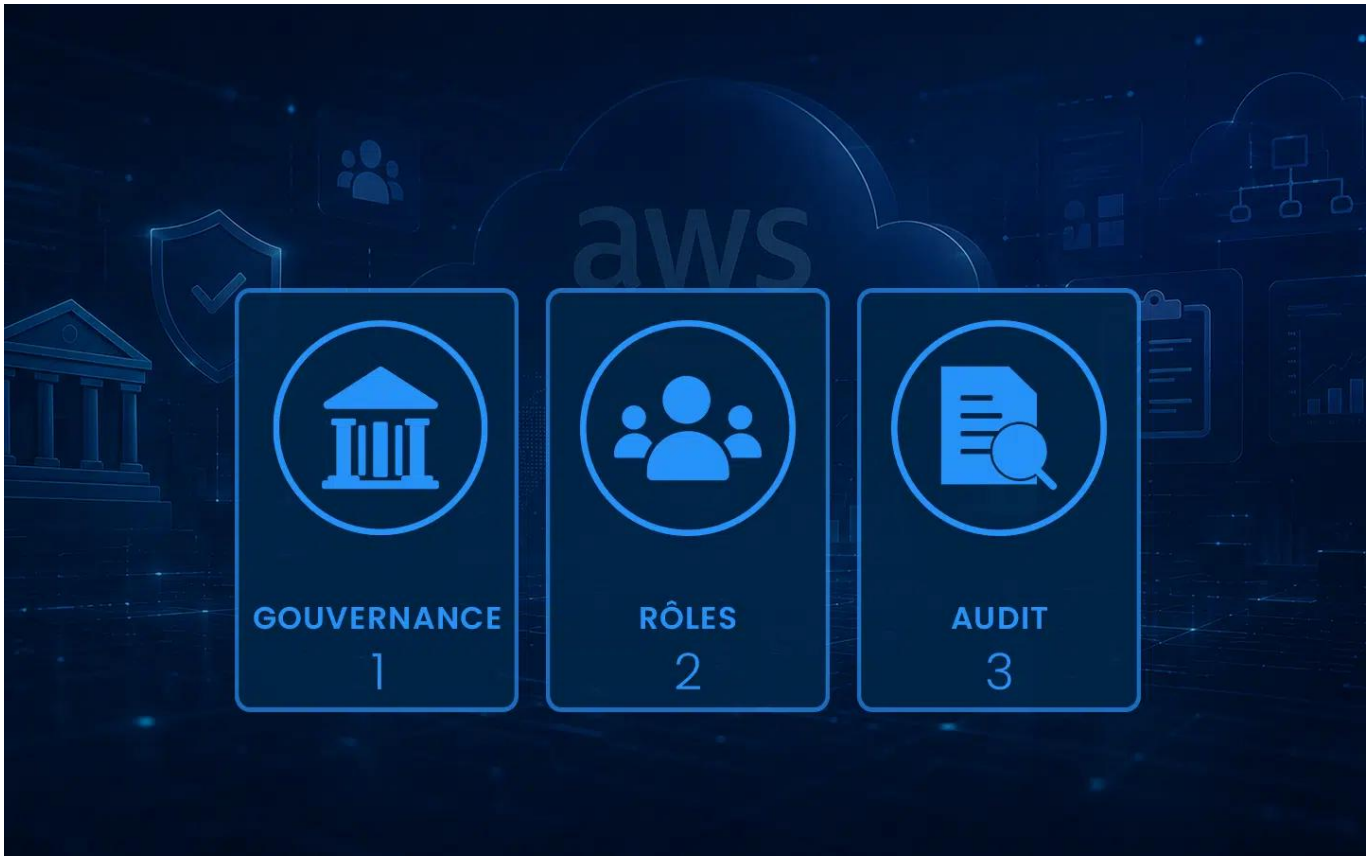
IAM avancé sur AWS : gouvernance, rôles et audit avec CloudTrail



La gestion des accès dans AWS est l'un des sujets les plus importants en entreprise. Une mauvaise configuration IAM peut exposer des données sensibles, créer des accèsnon autorisés ou provoquer des erreurs coûteuses en production.

Dans cet article, nous allons voir comment utiliser AWS IAM de manière professionnelle : rôles IAM, séparation des environnements, principe du moindre privilège, gouvernance et audit des accès avec CloudTrail.

Vous souhaitez apprendre AWS de manière professionnelle ?
Découvrez nos formations :

Formation AWS Certified Cloud Practitioner (CLF-C02)

Formation AWS Solutions Architect Associate (SAA-C03)

Pourquoi IAM est essentiel dans AWS

IAM ne sert pas uniquement à créer des utilisateurs. Il permet aussi de gérer les permissions des services AWS, des applications et des accès temporaires.

- Les utilisateurs IAM servent surtout aux personnes ou aux tests.
- Les rôles IAM servent aux services AWS et aux applications.
- Les policies IAM définissent les actions autorisées.

Les rôles IAM : une bonne pratique moderne

Un rôle IAM est une identité temporaire qu'un service AWS peut utiliser pour exécuter des actions. Par exemple, une fonction Lambda peut utiliser un rôle IAM pour lire un fichier S3, écrire dans DynamoDB ou envoyer des logs vers CloudWatch.

```
Lambda
  ↓ assume role
IAM Role
  ↓ permissions
S3 / DynamoDB / Secrets Manager
```

Un rôle IAM contient deux éléments importants :

- **Trust policy :** définit qui peut utiliser le rôle.
- **Permissions policy :** définit ce que le rôle peut faire.

Exemple de trust policy pour Lambda

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "lambda.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

La trust policy ne donne pas accès à S3 ou DynamoDB. Elle indique seulement quel service peut utiliser le rôle.

Pourquoi éviter les clés AWS dans le code

Stocker des clés AWS directement dans le code, dans un fichier local ou dans

un dépôt Git représente un risque de sécurité important. La bonne pratique consiste à utiliser des rôles IAM attachés aux services AWS.

Approche	Niveau de risque	Recommandation
Clés AWS dans le code	Élevé	À éviter
Clés AWS dans un fichier local	Moyen	Tests contrôlés uniquement
Rôle IAM attaché au service	Faible	Recommandé

Séparer les environnements : dev, test et production

En entreprise, il est essentiel de séparer les environnements afin d'éviter qu'une erreur en développement ait un impact sur la production.

Environnement	Exemple de bucket	Exemple de rôle	Niveau d'accès
dev	app-dev-data	role-lambda-dev	Plus flexible
test	app-test-data	role-lambda-test	Contrôlé
prod	app-prod-data	role-lambda-prod	Très restrictif

> ***Règle importante :*** *un rôle dev ne doit pas accéder aux ressources prod.*

Gouvernance AWS : nommage et tags

Le nommage et les tags permettent de mieux organiser, contrôler et auditer les ressources AWS.

Exemples de conventions de nommage

```
role-lambda-dev-traitement
role-lambda-prod-traitement
bucket-app-dev-data
bucket-app-prod-data
```

Tag	Exemple	Utilité
Environment	dev, test, prod	Identifier l'environnement
Owner	equipe-data	Identifier le responsable
Application	portail-client	Identifier l'application

Pour approfondir les bonnes pratiques AWS et préparer une certification, consultez également notre :

Formation AWS Solutions Architect Associate (SAA-C03)

CloudTrail : comprendre les permissions IAM nécessaires

Une bonne pratique consiste à ne pas deviner les permissions IAM. Il vaut mieux tester, provoquer une erreur, lire CloudTrail, consulter la documentation AWS, puis corriger la policy avec uniquement la permission nécessaire.

1. Provoquer volontairement une action.
2. Observer l'événement dans CloudTrail.
3. Identifier le champ eventName.
4. Repérer l'erreur éventuelle AccessDenied.
5. Ajouter uniquement la permission nécessaire.

Conclusion

La sécurité IAM ne doit pas être improvisée. Pour sécuriser efficacement un environnement AWS, il faut utiliser des rôles IAM, éviter les clés d'accès permanentes, séparer les environnements et auditer régulièrement les permissions.

La meilleure méthode reste simple :

> Tester, observer, comprendre, puis corriger.

Aller plus loin avec nos formations AWS

Formation AWS Certified Cloud Practitioner (CLF-C02)

Formation AWS Solutions Architect Associate (SAA-C03)

Ces formations permettent de maîtriser les services AWS, les architectures cloud, la sécurité IAM, les environnements serverless ainsi que les bonnes pratiques d’administration et de gouvernance cloud.