# Guide Complet : Sécurisation IAM AWS pour Applications SaaS

**Version:** 1.0
**Date:** Novembre 2025
**Destiné à:** Équipes de sécurité et DevSecOps

---

## Table des Matières



---

## Vue d'Ensemble

### Contexte de Sécurité 2025

Selon le rapport Verizon 2024 sur les violations de données, **plus de 80% des incidents de sécurité cloud** sont liés à des configurations incorrectes, souvent dues à des règles d'accès trop permissives. En 2025, IAM constitue l'épine dorsale de la sécurité cloud, avec l'utilisation croissante d'outils IAM pilotés par IA pour la détection de menaces, la classification des données et la gouvernance.

### Statistiques Clés

- **65% des violations de données** proviennent de contrôles d'accès trop permissifs (CISA 2024)
- **57% des escalades de privilèges** résultent d'autorisations excessives (Flexera 2024)
- **50% des violations d'identité** exploitent l'absence de vérifications contextuelles (Gartner 2024)

---

# Principe du Moindre Privilège (PoLP)

## 1. Définition et Importance

Le principe du moindre privilège consiste à accorder aux utilisateurs et rôles **uniquement les actions spécifiques requises**, en évitant les permissions larges comme `"Action": "*"`.

## 2. Stratégies de Mise en Œuvre

### 2.1 Utilisation d'IAM Access Analyzer

```
# Analyser les permissions non utilisées
aws accessanalyzer list-analyzers

# Générer une politique de moindre privilège basée sur l'activité
aws accessanalyzer generate-finding-recommendations \
    --analyzer-arn arn:aws:access-analyzer:region:account-id:analyzer/analyzer-name \
    --resource-arn arn:aws:iam::account-id:role/role-name
```

**IAM Access Analyzer** analyse les services et actions que vos rôles IAM utilisent réellement, puis génère une politique de moindre privilège que vous pouvez utiliser.

### 2.2 Éviter les Politiques Trop Larges

❌ **Mauvaise Pratique:**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```

✅ **Bonne Pratique:**

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::my-saas-bucket/${aws:userid}/*"
    }
  ]
}
```

### 2.3 Politiques de Pré-Production

Avant de lancer votre application en production, vous pouvez **générer une politique IAM basée sur l'activité d'accès** d'un rôle IAM pendant la phase de développement.

```
# Générer une politique basée sur CloudTrail
aws iam generate-service-last-accessed-details \
    --arn arn:aws:iam::123456789012:role/MyRole
```

## 3. Audits Réguliers

Les politiques IAM doivent être auditées :
- **À chaque changement majeur d'infrastructure**
- **Au moins une fois tous les 3 mois**
- **Après tout incident de sécurité**

AWS Security Hub et CloudTrail fournissent une visibilité sur 47% des violations résultant d'une visibilité insuffisante des changements de configuration (Gartner 2024).

---

# Authentification Multi-Facteurs (MFA)

## 1. Importance Critique

L'authentification multi-facteurs (MFA) est **essentielle** pour protéger les comptes privilégiés. 50% des violations d'identité en 2024 ont exploité l'absence de vérifications d'accès contextuel (Gartner).

## 2. Types de MFA Acceptables

| Type de MFA | Sécurité | Cas d'Usage |
|---|---|---|
| **Hardware MFA** (YubiKey, Gemalto) | ⭐⭐⭐⭐⭐ | Comptes root, administrateurs |
| **Virtual MFA** (Google Authenticator, Authy) | ⭐⭐⭐⭐ | Utilisateurs réguliers |
| **U2F Security Keys** | ⭐⭐⭐⭐⭐ | Développeurs, ops |

## 3. Stratégies d'Application MFA

### 3.1 MFA pour tous les Utilisateurs Console

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAllExceptListedIfNoMFA",
      "Effect": "Deny",
      "NotAction": [
        "iam:CreateVirtualMFADevice",
        "iam:EnableMFADevice",
        "iam:GetUser",
        "iam:ListMFADevices",
        "iam:ListVirtualMFADevices",
        "iam:ResyncMFADevice",
        "sts:GetSessionToken"
      ],
      "Resource": "*",
      "Condition": {
        "BoolIfExists": {
          "aws:MultiFactorAuthPresent": "false"
        }
      }
    }
```

```
  ]
}
```

Cette politique **bloque toutes les actions** sauf la configuration MFA si l'utilisateur n'est pas authentifié avec MFA.

### 3.2 MFA via Service Control Policies (SCP)

Pour une application au niveau organisationnel :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "BoolIfExists": {
          "aws:MultiFactorAuthPresent": "false"
        }
      }
    }
  ]
}
```

### 3.3 MFA pour AWS CLI

Les utilisateurs IAM doivent d'abord récupérer leur token MFA avec l'opération AWS STS `GetSessionToken` :

```
# Obtenir un token de session avec MFA
aws sts get-session-token \
    --serial-number arn:aws:iam::123456789012:mfa/user \
    --token-code 123456 \
    --duration-seconds 129600

# Utiliser les credentials temporaires
export AWS_ACCESS_KEY_ID=ASIAIOSFODNN7EXAMPLE
export AWS_SECRET_ACCESS_KEY=wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
export AWS_SESSION_TOKEN=AQoDYXdzEJr...
```

## 4. Auto-Enregistrement MFA

Stratégie permettant aux utilisateurs qui n'ont pas encore de dispositif MFA enregistré de s'auto-inscrire lors de la connexion, permettant de sécuriser les environnements AWS avec MFA sans distribuer individuellement les dispositifs.

# Gestion des Rôles IAM

## 1. Rôles vs Utilisateurs

### Pourquoi Privilégier les Rôles ?

| Critère | Rôles IAM | Utilisateurs IAM |
|---|---|---|
| **Credentials** | Temporaires | Permanents |
| **Rotation** | Automatique | Manuelle |
| **Sécurité** | ⭐⭐⭐⭐⭐ | ⭐⭐⭐ |
| **Cas d'usage** | Applications, services | Humains (limité) |

**Recommandation AWS:** Utilisez des rôles IAM pour les utilisateurs humains et les charges de travail accédant à vos ressources AWS afin qu'ils s'appuient sur des **credentials temporaires**.

## 2. Credentials Temporaires

```
# Assumer un rôle pour obtenir des credentials temporaires
aws sts assume-role \
    --role-arn arn:aws:iam::123456789012:role/MyRole \
    --role-session-name MySession \
    --duration-seconds 3600
```

Les credentials temporaires :
- ✅ Se désactivent automatiquement après expiration
- ✅ Réduisent le risque de compromission
- ✅ Sont conformes aux meilleures pratiques 2025

## 3. IAM Roles for Service Accounts (IRSA) - Kubernetes/EKS

Pour les charges de travail Kubernetes sur EKS :

```
apiVersion: v1
kind: ServiceAccount
metadata:
```

```
  name: my-service-account
  annotations:
    eks.amazonaws.com/role-arn: arn:aws:iam::123456789012:role/my-role
```

**Avantages:**
- **Moindre privilège** : Permissions spécifiques à un service account
- **Auditabilité** : Logs disponibles via CloudTrail
- **Isolation** : Chaque pod peut avoir ses propres permissions

### 4. Trust Policies et External ID

Pour les rôles cross-account ou partagés avec des tiers :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::EXTERNAL-ACCOUNT-ID:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "sts:ExternalId": "unique-external-id-12345"
        },
        "IpAddress": {
          "aws:SourceIp": "203.0.113.0/24"
        }
      }
    }
  ]
}
```

**External ID** protège contre le "confused deputy problem" où un attaquant pourrait tromper votre service pour utiliser ses permissions.

---

## Multi-Tenant et Isolation

### 1. Isolation des Tenants avec ABAC

**Attribute-Based Access Control (ABAC)** permet de gérer la complexité des politiques IAM dans les environnements multi-tenants.

### Principe:

Les attributs de session et de ressource doivent correspondre (ex: `TenantID: yellow` peut uniquement accéder aux ressources taggées `TenantID: yellow`).

### Exemple de Politique ABAC:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:PutObject",
        "s3:DeleteObject"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "s3:ExistingObjectTag/TenantID": "${aws:PrincipalTag/TenantID}"
        }
      }
    }
  ]
}
```

Cette politique permet aux utilisateurs d'accéder uniquement aux objets S3 dont le tag `TenantID` correspond à leur propre `TenantID`.

## 2. Politiques Générées Dynamiquement

Pour Lambda et EC2 dans des architectures multi-tenants :

```
import json
import boto3

def generate_tenant_policy(tenant_id):
    """Génère une politique IAM dynamique pour un tenant"""
    policy = {
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "dynamodb:GetItem",
                    "dynamodb:PutItem",
                    "dynamodb:Query"
                ],
                "Resource": f"arn:aws:dynamodb:region:account:table/TenantData",
```

```
                "Condition": {
                    "ForAllValues:StringEquals": {
                        "dynamodb:LeadingKeys": [tenant_id]
                    }
                }
            }
        ]
    }
    return json.dumps(policy)
```

### 3. Amazon Verified Permissions

Service permettant de gérer les politiques d'accès avec un **policy store par tenant** :

```
# Créer un policy store pour un tenant
aws verifiedpermissions create-policy-store \
    --validation-settings mode=STRICT \
    --description "Policy store for tenant-123"
```

---

## Audit et Surveillance

### 1. Services de Surveillance IAM

| Service | Fonction | Fréquence |
|---|---|---|
| **AWS CloudTrail** | Audit des appels API IAM | Temps réel |
| **IAM Access Analyzer** | Détection des accès externes | Continue |
| **AWS Security Hub** | Conformité et recommandations | Quotidienne |
| **GuardDuty** | Détection de menaces IAM | Temps réel |

### 2. Alertes CloudWatch pour IAM

```
# Créer une alarme pour les changements de politique IAM
aws cloudwatch put-metric-alarm \
    --alarm-name IAM-Policy-Changes \
    --alarm-description "Alert on IAM policy changes" \
    --metric-name PolicyChanges \
    --namespace AWS/IAM \
    --statistic Sum \
    --period 300 \
```

```
    --threshold 1 \
    --comparison-operator GreaterThanThreshold \
    --evaluation-periods 1
```

### 3. Filtres CloudWatch Logs pour Événements IAM

```
{
  "$.eventName": "AttachRolePolicy",
  "$.eventName": "DetachRolePolicy",
  "$.eventName": "PutRolePolicy",
  "$.eventName": "DeleteRolePolicy",
  "$.eventName": "CreateAccessKey",
  "$.eventName": "DeleteAccessKey"
}
```

### 4. Surveillance des Credentials Compromis

```
# Vérifier le rapport de credentials IAM
aws iam generate-credential-report
aws iam get-credential-report --output text --query 'Content' | base64 -d > credentials.csv

# Analyser les clés non utilisées depuis 90+ jours
awk -F, '$4 == "true" && $11 != "N/A" {
    cmd = "date -d "$11" +%s"
    cmd | getline last_used
    close(cmd)
    cmd = "date +%s"
    cmd | getline now
    close(cmd)
    if ((now - last_used) / 86400 > 90) print $1
}' credentials.csv
```

---

## Automatisation et Conformité

### 1. Service Control Policies (SCP)

Les SCP permettent d'établir des **guardrails de permissions** au niveau organisationnel :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
```

```
        "iam:DeleteUser",
        "iam:DeleteRole"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:PrincipalOrgID": "${aws:PrincipalOrgID}"
        }
      }
    }
  ]
}
```

## 2. AWS Organizations - Comptes Séparés

**Recommandation:** Utilisez des comptes séparés pour isoler les environnements de développement et de test des environnements de production, réduisant les risques de perturbations ou de conflits.

## 3. Identity Federation avec IAM Identity Center

Pour une gestion centralisée des accès :

```
# Configurer IAM Identity Center pour la fédération
aws sso-admin create-permission-set \
    --instance-arn arn:aws:sso:::instance/ssoins-1234567890abcdef \
    --name PowerUserAccess \
    --description "Power user access for developers"
```

## 4. Outils d'Automatisation

```
import boto3

def enforce_mfa_for_console_users():
    """Vérifie et applique MFA pour tous les utilisateurs console"""
    iam = boto3.client('iam')

    users = iam.list_users()['Users']

    for user in users:
        username = user['UserName']

        # Vérifier si l'utilisateur a un mot de passe console
        try:
            iam.get_login_profile(UserName=username)
            has_console = True
        except:
            has_console = False
```

```
        if has_console:
            # Vérifier MFA
            mfa_devices = iam.list_mfa_devices(UserName=username)['MFADevices']

            if not mfa_devices:
                print(f"WARNING: User {username} has console access without MFA!")
                # Ici, vous pourriez attacher une politique de refus
```

---

# Checklist de Sécurité IAM

## ✅ Niveau Critique (Priorité 1)

- [ ] **Root Account MFA activé**
- [ ] **Pas d'access keys sur le compte root**
- [ ] **MFA appliqué pour tous les utilisateurs privilégiés**
- [ ] **Politiques basées sur le moindre privilège**
- [ ] **CloudTrail activé et logs conservés**
- [ ] **IAM Access Analyzer activé**
- [ ] **Pas de credentials codés en dur dans le code**
- [ ] **Rotation automatique des secrets (Secrets Manager)**

## ✅ Niveau Important (Priorité 2)

- [ ] **Service Control Policies (SCP) configurés**
- [ ] **Stratégie multi-comptes implémentée**
- [ ] **Identity Federation avec IAM Identity Center**
- [ ] **Politiques d'isolation multi-tenant (ABAC)**
- [ ] **Alarmes CloudWatch pour changements IAM**
- [ ] **Audit IAM trimestriel**
- [ ] **GuardDuty activé pour détection de menaces**
- [ ] **Credentials temporaires uniquement pour applications**

## ✅ Niveau Recommandé (Priorité 3)

- [ ] **Politiques générées par IAM Access Analyzer**
- [ ] **Tags de ressources appliqués systématiquement**

- [ ] **Conditions IP dans les trust policies**
- [ ] **External ID pour rôles tiers**
- [ ] **Session duration limitée (< 12h)**
- [ ] **Suppression automatique des clés inactives (> 90 jours)**
- [ ] **Security Hub pour conformité continue**
- [ ] **Documentation des rôles et permissions**

---

# Références et Ressources

## Documentation Officielle AWS

- IAM Best Practices
- IAM Access Analyzer
- SaaS Tenant Isolation with ABAC
- IAM Roles for Service Accounts

## Rapports de Sécurité 2024-2025

- Verizon Data Breach Investigations Report 2024
- Gartner Identity Security Survey 2024
- CISA Cloud Security Guidelines 2024
- SailPoint State of Identity Security 2025

## Outils et Services

- **AWS Security Hub** - Conformité et alertes
- **AWS CloudTrail** - Audit des API
- **AWS GuardDuty** - Détection de menaces
- **AWS Config** - Suivi des configurations
- **IAM Access Analyzer** - Analyse des politiques

---

# Conclusion

La sécurisation IAM est un **processus continu** qui nécessite :
- Une application stricte du principe du moindre privilège
- Une authentification forte avec MFA
- Des audits réguliers et automatisés
- Une surveillance proactive des menaces

En 2025, avec l'augmentation des attaques ciblant les identités cloud, IAM représente la première ligne de défense de votre infrastructure AWS. L'implémentation de ce guide permettra de réduire significativement la surface d'attaque et de garantir la conformité avec les standards de sécurité actuels.

---

**Document préparé pour:** [Nom du Client]
**Contact support:** [Email de l'équipe sécurité]
**Dernière mise à jour:** Novembre 2025