



Worketyamo

AWS Identity and Access Management (IAM)



@worketyamo



www.worketyamo.com



worketyamo.dev@gmail.com



+237 655 19 35 30

Le Principe du Moindre Privilège

Le principe du moindre privilège consiste à donner à une personne ou à un système uniquement les **permissions nécessaires** pour accomplir une tâche spécifique, et rien de plus. Par exemple, si tu travailles à l'accueil, tu n'as besoin que des accès pour les outils que tu utilises, pas pour tout le système informatique de l'entreprise. Ce principe limite les risques en empêchant les accès non autorisés ou les erreurs qui pourraient compromettre la sécurité des informations sensibles. En d'autres termes, on ne donne que ce qui est nécessaire pour faire le travail, afin de protéger tout le reste.

Modèle de responsabilité partagé

Le modèle de responsabilité partagée d'AWS définit clairement qui est responsable de la sécurité dans le cloud. AWS se charge de sécuriser **l'infrastructure physique** qui héberge ses services, comme les centres de données, le matériel, et les réseaux. En tant que client, vous êtes responsable de sécuriser tout ce que vous créez et stockez dans le cloud, y compris vos données, la configuration de vos services, et la gestion des accès. Ce modèle permet de comprendre que la sécurité est une responsabilité partagée : AWS protège le cloud, tandis que vous devez protéger ce que vous y faites.

Sécurité chez AWS

La sécurité consiste à protéger votre propriété intellectuelle contre l'accès, l'utilisation ou la modification non autorisés. Un aspect crucial de la sécurité de l'information dans toute organisation est la **confidentialité**—s'assurer que seules les bonnes personnes ont accès aux informations, tout en empêchant l'accès aux personnes non autorisées. Pour construire une culture de sécurité solide, les organisations doivent suivre le **principe du moindre privilège** et appliquer des pratiques d'authentification robustes.

Concepts clés :

Accorder l'Accès au Besoin

Ne donnez aux utilisateurs ou aux systèmes que l'accès minimum nécessaire pour accomplir leurs tâches. Cela réduit le risque d'actions non autorisées et de potentielles failles de sécurité.

Séparer les Tâches

Les rôles différents doivent avoir des responsabilités distinctes, pour s'assurer qu'une seule personne n'a pas le contrôle sur tous les aspects d'un système, ce qui aide à prévenir les abus de pouvoir et les erreurs.

Éviter les Identifiants à Long Terme

Utilisez des identifiants temporaires autant que possible, comme ceux fournis par les rôles IAM, pour minimiser le risque d'exposition à long terme d'informations sensibles.

Utilisateurs et Groupes

AWS Identity and Access Management (IAM) vous aide à décider **qui peut accéder à quoi** dans votre espace cloud. Quand vous créez un compte AWS, vous commencez avec un "**utilisateur racine**" qui peut tout faire. Mais pour des raisons de sécurité, il vaut mieux utiliser cet utilisateur uniquement pour créer d'autres comptes avec moins de pouvoirs, adaptés à des tâches précises.

Avec IAM, vous pouvez créer des utilisateurs (**qui peuvent être des personnes ou des applications**) et les regrouper en groupes. Par exemple, vous pouvez créer un groupe "**Développeurs**" pour donner à tous vos développeurs les **autorisations** dont ils ont besoin. Cela rend les choses plus faciles : au lieu de configurer chaque utilisateur séparément, vous les ajoutez simplement au bon groupe.

Les rôles

Les rôles IAM sont comme des **costumes** que les utilisateurs ou applications peuvent enfiler pour obtenir temporairement les autorisations nécessaires à une tâche spécifique. Par exemple, si un utilisateur doit faire quelque chose pour laquelle il n'a pas normalement les droits, il peut "**endosser**" un rôle pour obtenir l'accès nécessaire, un peu comme emprunter une clé pour ouvrir une porte.

Dans quel situation utiliser les rôles ?

Les rôles sont pratiques dans plusieurs situations :

- Quand un utilisateur a besoin d'un accès particulier.
- Quand une application AWS (comme un serveur) doit accéder à d'autres services AWS.
- Quand vous voulez permettre à quelqu'un de l'extérieur d'accéder à vos ressources avec ses propres identifiants.

Types d'identifiants AWS :

Quand vous créez un **utilisateur** AWS, il obtient un nom d'utilisateur et un mot de passe pour se connecter à AWS. Il peut aussi avoir des **clés d'accès** pour interagir avec AWS via des commandes ou des scripts.

Pour plus de sécurité, AWS propose l'authentification **multiproducteurelle (MFA)**. Cela signifie qu'en plus du mot de passe, l'utilisateur doit entrer un code unique provenant d'un appareil MFA (comme une appli sur son téléphone). Cela rend l'accès non autorisé beaucoup plus difficile.

Concepts clés :

- **Groupes IAM** : Utilisez des groupes pour gérer facilement les autorisations de plusieurs utilisateurs. Créez des groupes selon les rôles, comme "Développeurs" ou "Managers", et attribuez les permissions au groupe plutôt qu'à chaque utilisateur.
- **Rôles IAM** : Les rôles permettent de donner un accès temporaire sans partager de mots de passe ou d'informations sensibles.
- **Contrôle Centralisé** : IAM vous permet de gérer tous les accès à vos ressources AWS depuis un seul endroit, ce qui simplifie la sécurité et l'organisation de vos ressources.

Comprendre les politiques IAM (IAM policy)

Pour bien utiliser les politiques IAM (Identity and Access Management) d'AWS, il faut d'abord comprendre trois éléments clés, un peu comme les parties d'une phrase : le principal (qui fait l'action), l'action (ce qui est fait), et la ressource (sur quoi l'action est faite).

- **Principal** : C'est la personne, le rôle, ou l'application qui essaie d'accéder à quelque chose sur AWS. Imaginez que c'est comme un employé avec une clé qui essaie d'entrer dans un bureau.
- **Action** : C'est ce que cette personne, rôle, ou application veut faire (comme lire des données, modifier un fichier, etc.). Ici, l'action est ce que cet employé veut faire une fois dans le bureau : par exemple, consulter un dossier.
- **Ressource** : C'est l'objet AWS sur lequel l'action est effectuée (comme un fichier dans un bucket S3 ou une machine virtuelle EC2). La ressource, c'est le dossier que l'employé essaie de consulter.

Comment fonctionnent les politiques IAM

Dans AWS, vous gérez l'accès aux ressources en créant des politiques. Ces politiques sont comme des règles de sécurité qui disent qui peut faire quoi. Vous attachez ces règles à des utilisateurs, des rôles ou directement à des ressources. Quand quelqu'un fait une demande pour accéder à quelque chose sur AWS, AWS vérifie les règles (les politiques) pour voir si cette action est autorisée ou non. C'est comme un agent de sécurité qui vérifie si l'employé a la bonne autorisation pour entrer et consulter ce dossier.

Types d'identifiants AWS :

- **Politiques gérées par AWS** : Ce sont des politiques créées par AWS pour vous aider à démarrer. Elles sont prêtes à l'emploi et peuvent être attachées à plusieurs utilisateurs ou rôles. Pensez à ces politiques comme des clés standards fournies par le propriétaire de l'immeuble. Si vous débutez, c'est un bon point de départ.
- **Politiques gérées par vous** : Vous pouvez aussi créer vos propres politiques pour un contrôle plus précis. Ici, c'est comme si vous fabriquiez vous-même des clés spéciales pour chaque porte, en décidant exactement qui peut les utiliser.
- **Politiques en ligne** : Celles-ci sont attachées directement à un utilisateur ou un rôle. Elles sont utiles si vous voulez qu'une politique s'applique uniquement à une personne spécifique, comme une clé unique que seul un employé peut utiliser. Mais elles ne sont pas souvent recommandées car elles sont plus difficiles à gérer.

Comment AWS traite une requête

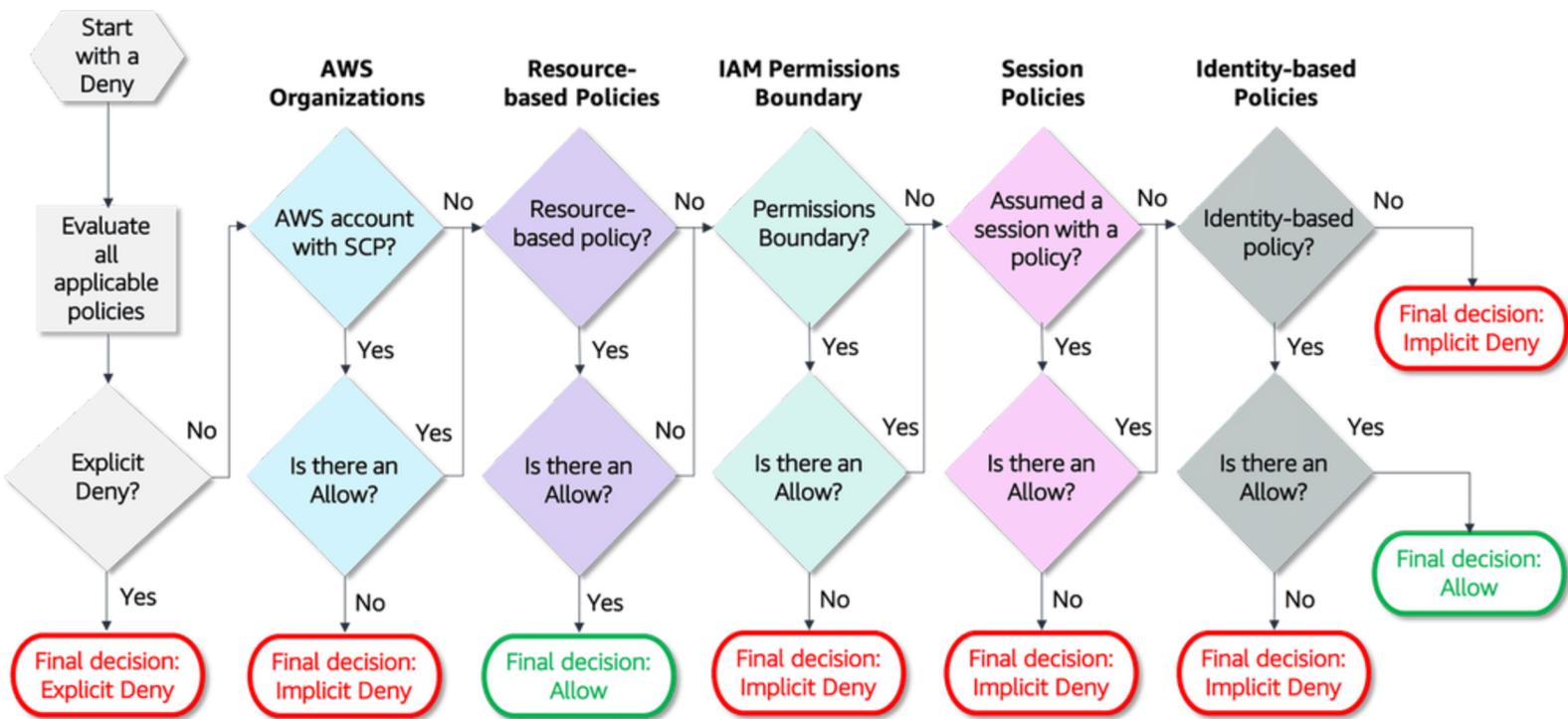
Prenons un exemple simple : un utilisateur essaie d'accéder à un fichier dans un bucket S3. Voici ce qui se passe :

1. **Authentification** : AWS vérifie d'abord que l'utilisateur est bien connecté. C'est comme si l'employé présentait son badge d'identification à la porte.
2. **Autorisation** : AWS regarde ensuite si cet utilisateur a le droit de faire ce qu'il demande (comme accéder au fichier). L'agent de sécurité vérifie si l'employé a bien les autorisations pour entrer dans ce bureau.
3. **Vérification des Actions** : AWS s'assure que l'action demandée est bien autorisée par les règles. L'agent s'assure que l'employé peut effectivement consulter le dossier.
4. **Vérification de l'Accès à la Ressource** : AWS regarde aussi les règles attachées au fichier (la ressource) pour voir si l'accès est autorisé ou non. Même si l'employé a la clé, certaines portes pourraient avoir des restrictions supplémentaires.
5. **Décision Finale** : Si tout est en ordre, AWS autorise l'accès. Sinon, la demande est refusée. C'est comme si l'agent laissait entrer l'employé ou refusait l'accès.

Ajouter des conditions :

Vous pouvez ajouter des conditions à vos politiques pour rendre les règles encore plus spécifiques. Par exemple, vous pouvez dire qu'un utilisateur ne peut accéder à une ressource que depuis une certaine adresse IP ou à une certaine heure de la journée. C'est comme dire que l'employé ne peut entrer dans le bureau que pendant les heures de bureau ou seulement depuis un certain bâtiment.

Logique d'évaluation



Points Essentiels

- Un autorisation explicite dans une politique basée sur l'identité ou sur les ressources remplace le refus par défaut.
- Si une limite d'autorisations, une politique SCP d'AWS Organizations, ou une politique de session est présente, elle peut annuler l'autorisation explicite avec un refus implicite.
- Un refus explicite dans n'importe quelle politique annule toutes les autorisations.
- Si la ressource demandée a une politique basée sur la ressource qui autorise l'action demandée, alors AWS renvoie une décision finale d'autorisation. S'il n'y a pas de politique basée sur la ressource ou si la politique n'inclut pas d'instruction "Allow", l'évaluation continue.
- Si une politique de session est présente et qu'elle n'autorise pas l'action demandée, la requête est implicitement refusée.

Exemple de politique IAM

```
"Version": "2012-10-17",
"Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "s3:GetObject",
            "s3>ListBucket"
        ],
        "Resource": [
            "arn:aws:s3:::mon-bucket-exemple",
            "arn:aws:s3:::mon-bucket-exemple/*"
        ]
    }
]
```

Explication de la politique :

Version : Cette version spécifie le format de la politique. La version "2012-10-17" est la plus couramment utilisée.

Statement : C'est l'élément principal de la politique qui contient les instructions.

Effect : Ici, "Allow" signifie que l'accès est autorisé.

Action : Cette section définit les actions que l'utilisateur est autorisé à effectuer. Dans ce cas, "s3:GetObject" permet de lire des objets dans le bucket, et "s3>ListBucket" permet de lister les objets dans le bucket.

Resource : Cette section spécifie les ressources sur lesquelles les actions peuvent être effectuées. "arn:aws:s3:::mon-bucket-exemple" fait référence au bucket lui-même, et "arn:aws:s3:::mon-bucket-exemple/*" fait référence à tous les objets dans ce bucket.

Exemple plus complet

```
"Version": "2012-10-17",
"Statement": [
    {
        "Effect": "Allow",
        "Action": "s3>ListBucket",
        "Resource": "arn:aws:s3:::mon-bucket-exemple"
    },
    {
        "Effect": "Allow",
        "Action": [
            "s3:GetObject",
            "s3:PutObject",
            "s3>DeleteObject"
        ],
        "Resource": "arn:aws:s3:::mon-bucket-exemple/*"
    },
    {
        "Effect": "Allow",
        "Action": "s3:PutObject",
        "Resource": "arn:aws:s3:::mon-bucket-exemple/*",
        "Condition": {
            "DateGreaterThan": {
                "aws:CurrentTime": "2024-08-01T00:00:00Z"
            },
            "DateLessThan": {
                "aws:CurrentTime": "2024-08-31T23:59:59Z"
            }
        }
    }
]
```

Explication de la politique

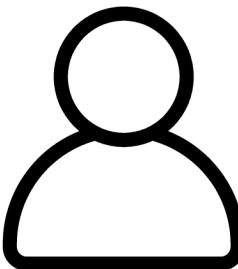
- **Version** : Indique la version de la politique. "2012-10-17" est la version standard actuelle.
- **Statement** : La politique est divisée en plusieurs blocs Statement pour spécifier différentes permissions.
 - Premier Statement :
 - **Effect** : "Allow" signifie que l'accès est autorisé.
 - **Action** : "s3
 - " permet à l'utilisateur de lister les objets dans le bucket.
 - **Resource** : "arn:aws:s3:::mon-bucket-exemple" se réfère spécifiquement au bucket S3 nommé mon-bucket-exemple.
 - Deuxième Statement :
 - **Effect** : "Allow" signifie que l'accès est autorisé.
 - **Action** : Inclut "s3:GetObject" (lecture d'objets), "s3:PutObject" (écriture d'objets), et "s3:DeleteObject" (suppression d'objets) dans le bucket.
 - **Resource** : "arn:aws:s3:::mon-bucket-exemple/*" se réfère à tous les objets dans ce bucket.
 - Troisième Statement :
 - **Effect** : "Allow" signifie que l'accès est autorisé.
 - **Action** : "s3
 - " permet à l'utilisateur de télécharger ou de mettre à jour des objets dans le bucket.
 - **Resource** : "arn:aws:s3:::mon-bucket-exemple/*" se réfère à tous les objets dans ce bucket.
 - **Condition** : Cette section impose des restrictions supplémentaires. Ici, l'action "s3:PutObject" n'est autorisée que si l'horodatage actuel (aws:CurrentTime) se situe entre le 1er août 2024 à 00:00 UTC et le 31 août 2024 à 23:59 UTC.

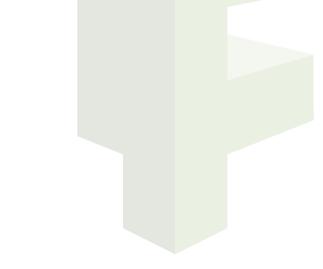
1.

Résumé

Les politiques **IAM** (Identity and Access Management) d'AWS sont des **règles** qui définissent les permissions des utilisateurs, rôles, et services pour accéder aux ressources AWS. Elles permettent de contrôler qui peut faire quoi et sur quelle ressource, en assurant une gestion sécurisée des accès. Il existe différents types de politiques : gérées par AWS pour un usage facile, gérées par le client pour un contrôle personnalisé, et en ligne pour des besoins très spécifiques. Lorsqu'une requête d'accès est faite, AWS vérifie l'authentification, l'autorisation, les actions permises, et l'accès à la ressource, avant de décider d'autoriser ou de refuser la requête. Ces politiques sont cruciales pour sécuriser les environnements cloud tout en assurant une flexibilité opérationnelle.

Recapitulatif

Entité	Description	cas d'usage
 utilisateurs	Chaque personne ayant besoin d'accéder à vos ressources AWS doit avoir un compte utilisateur. Les permissions sont ensuite définies via des politiques d'accès.	Accordez un accès spécifique à des individus pour certaines tâches.

Entité	Description	cas d'usage
 Groupes IAM	<p>Ensemble d'utilisateurs ayant des permissions communes. Les utilisateurs peuvent être ajoutés ou supprimés selon les besoins.</p> 	<p>Accordez des permissions communes à un groupe de comptes utilisateurs.</p>
 Roles IAM	<p>Les rôles sont utilisés pour accorder temporairement des permissions à des utilisateurs ou à des ressources AWS.</p> 	<p>Utilisez des rôles lorsque vous avez besoin d'accorder temporairement des permissions à des utilisateurs ou des ressources.</p>
 Policy IAM	<p>Document JSON définissant les permissions pour une ou plusieurs entités IAM</p>	<p>Utilisez des politiques d'accès pour définir des permissions granulaires pour chaque entité IAM dans votre compte AWS</p>

Exercices IAM

Exercice 1 : Créer un Utilisateur IAM avec des Permissions de Lecture Seule

1. Connectez-vous à la console AWS et accédez à la section IAM.
2. Créez un nouvel utilisateur IAM nommé "UtilisateurLectureSeule".
3. Attribuez-lui l'accès programmatique (clé d'accès et secret) et l'accès à la console de gestion AWS.
4. Attachez une politique gérée par AWS qui permet uniquement des actions en lecture seule, comme ReadOnlyAccess.
5. Connectez-vous avec les identifiants de cet utilisateur et vérifiez qu'il ne peut effectuer que des actions en lecture seule sur les services AWS.

Exercice 2 : Créer et Attacher une Politique IAM Personnalisée

1. Toujours dans IAM, créez une nouvelle politique personnalisée.
2. Dans cette politique, permettez à l'utilisateur de l'exercice 1 de lire des objets dans un bucket S3 spécifique, mais pas de les modifier ou de les supprimer.
3. Attachez cette politique à l'utilisateur "UtilisateurLectureSeule".
4. Testez la politique en essayant d'accéder au bucket S3 avec l'utilisateur, puis en tentant de modifier un fichier (ce qui devrait échouer).

Exercice 3 : Utiliser les Rôles IAM pour Accéder à un Service

1. Créez un rôle IAM qui permet d'accéder à un bucket S3 avec des permissions complètes (lecture, écriture, suppression).
2. Configurez une instance EC2 et attachez-lui ce rôle.
3. Connectez-vous à l'instance EC2 et utilisez l'AWS CLI pour vérifier que vous pouvez accéder au bucket S3 avec les permissions accordées par le rôle.
4. Vérifiez que ces permissions ne sont pas disponibles en dehors de l'instance EC2.



Exercice 4 : Configurer une Politique avec des Conditions de Temps

1. Créez une politique IAM qui permet à un utilisateur IAM d'écrire dans un bucket S3 uniquement entre 9h00 et 17h00 UTC.
2. Attachez cette politique à un nouvel utilisateur IAM nommé "UtilisateurRestreint".
3. Connectez-vous en tant qu'UtilisateurRestreint et testez l'accès en dehors de l'intervalle de temps autorisé (l'écriture devrait être refusée).
4. Essayez à nouveau pendant l'intervalle autorisé pour vérifier que l'accès est accordé.

Policy IAM

Exercice 1 : Créer une Politique pour Accorder l'Accès en Lecture à DynamoDB

1. Accédez à l'IAM Policy Generator dans la console AWS.
2. Créez une nouvelle politique qui permet à un utilisateur IAM d'accéder en lecture seule (List, Get, Query) à une table DynamoDB spécifique.
3. Utilisez le ARN (Amazon Resource Name) de la table pour restreindre la politique à cette table uniquement.
4. Générez la politique et attachez-la à un utilisateur IAM.
5. Testez la politique en essayant d'accéder à la table DynamoDB avec l'utilisateur.

Exercice 2 : Créer une Politique pour Restreindre l'Accès à S3 par Adresse IP

1. Utilisez l'IAM Policy Generator pour créer une politique qui permet à un utilisateur IAM d'accéder à un bucket S3, mais uniquement depuis une adresse IP spécifique (par exemple, 203.0.113.0/24).
2. Ajoutez une condition `IpAddress` dans la politique pour restreindre l'accès.
3. Générez la politique et attachez-la à un utilisateur IAM.
4. Testez l'accès au bucket S3 depuis l'adresse IP autorisée et une autre adresse IP pour vérifier que la restriction fonctionne correctement.

Exercice 3 : Créer une Politique pour Autoriser l'Accès à la Console de Facturation AWS

1. Dans l'IAM Policy Generator, créez une politique qui permet à un utilisateur IAM d'accéder uniquement à la console de facturation AWS.
2. Sélectionnez les actions liées à la facturation, comme aws-portal:ViewBilling et aws-portal:ViewPaymentMethods.
3. Générez la politique et attachez-la à un utilisateur IAM.
4. Testez la politique en vous connectant à la console avec l'utilisateur pour vérifier qu'il peut accéder uniquement aux informations de facturation.

Exercice 4 : Créer une Politique pour Autoriser l'Accès Complet à un Seul Service AWS

1. Utilisez l'IAM Policy Generator pour créer une politique qui permet à un utilisateur IAM d'avoir un accès complet (FullAccess) à Amazon S3.
2. Sélectionnez toutes les actions S3 (s3:*) et spécifiez * pour la ressource, ou limitez l'accès à un seul bucket si vous souhaitez restreindre la portée.
3. Générez la politique et attachez-la à un utilisateur IAM.
4. Testez l'accès en utilisant l'utilisateur pour vérifier qu'il a bien un accès complet à S3, mais pas à d'autres services.

Exercice 5 : Créer une Politique pour Autoriser l'Accès aux Logs CloudWatch

1. Utilisez l'IAM Policy Generator pour créer une politique qui permet à un utilisateur IAM d'accéder aux journaux CloudWatch Logs.
2. Ajoutez des actions comme logs>CreateLogGroup, logs>CreateLogStream, logs:PutLogEvents, et logs:DescribeLogGroups.
3. Spécifiez les ressources ARN des groupes de journaux spécifiques si nécessaire.
4. Générez la politique et attachez-la à un utilisateur IAM.
5. Testez la politique en essayant de créer et de visualiser des journaux CloudWatch Logs avec l'utilisateur.



Haruna Rashid Yakubu