Einführung in AWS Identity and Access Management (IAM)

Einleitung

- **Wer** kann **was** mit **welchen** Ressourcen in **AWS** tun?
- IAM ermöglicht die **Verwaltung von Identitäten und Zugriffsberechtigungen** in AWS.
- **Zentrale Steuerung** der Zugriffsrechte für Benutzer und Dienste.

- **Warum braucht man AWS IAM?**
  - Sicherheitskontrolle und Zugriffsverwaltung in AWS.
  - Verhinderung unautorisierten Zugriffs auf Ressourcen.
  - Kontrolle über Berechtigungen für Benutzer und Dienste.

IAM im Vergleich mit Security Groups

- **IAM versus Security Groups:**
  - IAM:
    - Identitäts- und Zugriffsverwaltung für Benutzer und Dienste.
    - Steuerung des Zugriffs auf AWS-Ressourcen.
    - Z.B. darf Benutzer A auf S3-Bucket B zugreifen.
  - Security Groups:
    - Firewall-Regeln zur Steuerung des Netzwerkverkehrs.
    - **Es geht immer um die Netzwerkkommunikation**.
    - Z.B. darf EC2-Instanz A auf Port 80 von EC2-Instanz B zugreifen.

User (Benutzer)

- Es gibt verschiedene Arten von Benutzern in AWS IAM:
  - **Das Root-Benutzerkonto**
  - **IAM-Benutzer**

- **Das Root-Benutzerkonto:**
  - Standard-Benutzerkonto, das bei der Einrichtung erstellt wird.
  - Hat uneingeschränkten Zugriff auf alle AWS-Ressourcen.
  - Höchste Berechtigungsstufe, sollte mit Vorsicht verwendet werden.

- **Erstellen und Verwalten von IAM-Benutzern:**
  - Benutzerkonten für einzelne Personen oder Dienste.
  - Jeder Benutzer hat eindeutige Anmeldeinformationen.
  - Möglichkeit, benutzerspezifische Berechtigungen zuzuweisen.
  - Es gibt auch die Möglichkeit programmatisch zuzugreifen (Access Key und Secret Access Key).

Policy (Richtlinien)

- **Was sind Policies in IAM?**
  - JSON-Dokumente, die Berechtigungen definieren.
  - Legen fest, welche Aktionen auf welche Ressourcen durchgeführt werden dürfen.
  - Können an Benutzer, Gruppen oder Rollen angehängt werden.
  - **Werden im JSON-Format geschrieben**.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": "*"
        }
    ]
}
```

- **Wie sind Policies aufgebaut?**
  - Bestehen aus einer Liste von Erlaubnissen (Allow) und Verweigerungen (Deny).
  - Statement-Objekte definieren die Berechtigungen.
  - Jedes Statement enthält **Effect**, **Aktionen**, **Ressourcen** und **Bedingungen**.
  - **Effect** (Effekt) definiert, ob die Aktion **erlaubt -> Allow** oder **verweigert -> Deny** wird.
  - **Action** definiert die Aktionen, die erlaubt oder verweigert werden.
  - **Resource** definiert die Ressourcen, auf die die Aktionen angewendet werden.
  - **Condition** definiert optionale Bedingungen, unter denen die Berechtigung gilt.

- **Warum benötigt man Policies?**
  - Gewährleistung der Sicherheit und Einhaltung von Compliance-Vorschriften.
  - Flexibilität bei der Definition von Zugriffssteuerungen.
  - Feingranulare Kontrolle über Ressourcenzugriffe.

- **Beispiel für eine Policy:**
  - Erlaubt einem Benutzer das Lesen von Objekten in einem bestimmten S3-Bucket.
  - Verhindert einem Benutzer das Löschen von EC2-Instanzen.

Gruppen in AWS IAM

- **Was sind Gruppen?**
  - Sammlung von Benutzern mit ähnlichen Berechtigungen.
  - Ermöglicht die zentrale Verwaltung von Zugriffsrechten.
  - Vereinfacht die Berechtigungsverwaltung in großen Umgebungen.
  - Beispiel: Entwicklergruppe, Administratorengruppe.

- **Erstellen und Verwalten von Gruppen:**
  - Neue Gruppen können erstellt werden, um Benutzer zu organisieren.
  - Benutzer können Mitgliedern von mehreren Gruppen sein.
  - Gruppen können Policies enthalten, die an alle Mitglieder angewendet werden.

Rolle in AWS IAM

- **Was sind Rollen?**
  - Temporäre Anmeldeinformationen für Benutzer oder Dienste.
  - Gewähren Zugriff auf Ressourcen ohne dauerhafte Anmeldeinformationen.
  - Häufig für Anwendungen oder Dienste verwendet, die auf AWS-Ressourcen zugreifen müssen.
  - Beispiel: EC2-Instanz, die auf S3-Bucket zugreifen muss.

- **Wie verwendet man Rollen?**
  - Erstellen von Rollen mit spezifischen Berechtigungen.
  - Zuweisen von Rollen für:
    - Benutzer in anderen AWS-Konten.
    - Anwendungen oder Dienste, die auf AWS-Ressourcen zugreifen müssen.
    - AWS-Services wie z.B. EC2-Instanzen.
  - Rollen können auch an EC2-Instanzen angehängt werden.

Best Practices

- Nutzung des Root-Benutzerkontos vermeiden.
- **Niemals mehrere Benutzer mit denselben Anmeldeinformationen**.
- **Prinzip des geringsten Privilegs** anwenden.
- **Regelmäßige Überprüfung und Aktualisierung von Berechtigungen**.
- Wenn es möglich ist Gruppen verwenden.