🔐 Securing AWS Resources with IAM: A Hands-On Project

📜 Overview

In today's cloud-native landscape, ensuring secure, controlled access to cloud resources is more critical than ever. This project-based, 2-hour hands-on course dives deep into the foundational aspects of **Identity and Access Management (IAM)** on **Amazon Web Services (AWS)**. IAM is the backbone of cloud security—enabling organizations to define *who* can access *what* resources and *how* they can interact with them.

Through this guided experience, I explored essential IAM functionalities by simulating a real-world scenario in which a **financial institution** provides **secure cross-account access** to a **third-party auditing firm**. The goal was to ensure secure and auditable access to financial reports stored in AWS S3.

🎯 Learning Objectives and Key Tasks

✅ 1. Project Setup and Overview

- Gained familiarity with the project structure, prerequisites, and the overall security objective.
- Reviewed the cloud architecture of a typical enterprise security workflow.

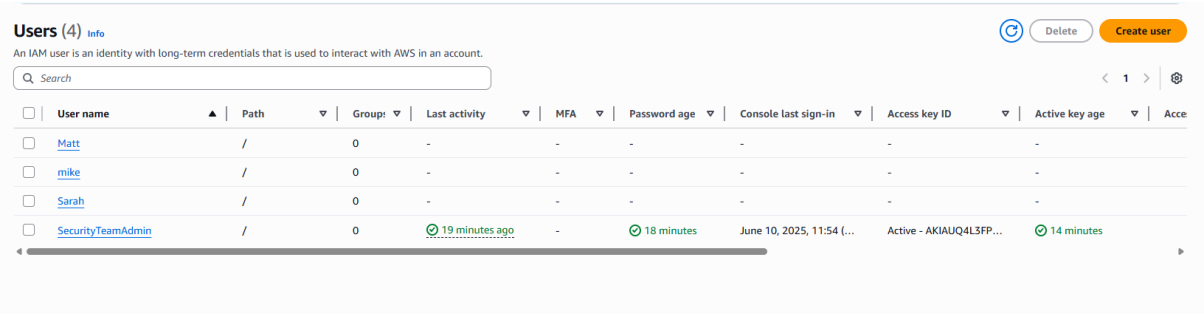
✅ 2. Creating IAM Users (Console & CLI)

- **Console**: Created a user with programmatic and console access, attached policies, and generated secure credentials.
- **AWS CLI**: Used aws iam create-user and aws iam create-login-profile to automate IAM user creation.

💡 *Commands Used:*

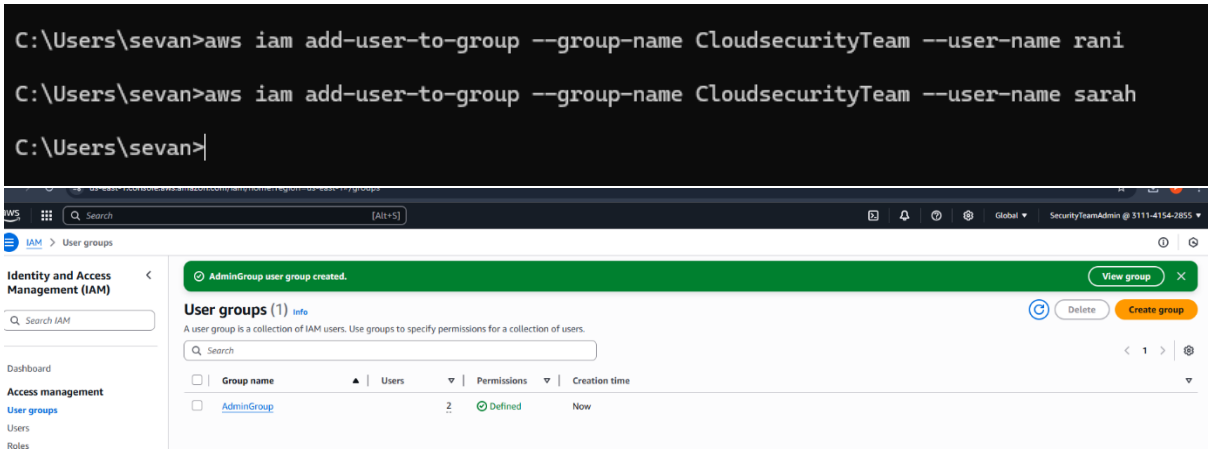
aws iam create-user --user-name AuditUser
aws iam create-login-profile --user-name AuditUser --password 'SecureP@ssw0rd'

```
C:\Users\sevan>aws iam create-user --user-name Matt
USER    arn:aws:iam::311141542855:user/Matt     2025-06-10T06:40:26+00:00       /       AIDAUQ4L3FPDZT3NWCHNQ   Matt


C:\Users\sevan>aws iam create-user --user-name Sarah
USER    arn:aws:iam::311141542855:user/Sarah    2025-06-10T06:40:39+00:00       /       AIDAUQ4L3FPDW46WDYFMT   Sarah


C:\Users\sevan>aws iam create-user --user-name mike
USER    arn:aws:iam::311141542855:user/mike     2025-06-10T06:41:19+00:00       /       AIDAUQ4L3FPDXEDMMGK3R   mike


C:\Users\sevan>
```



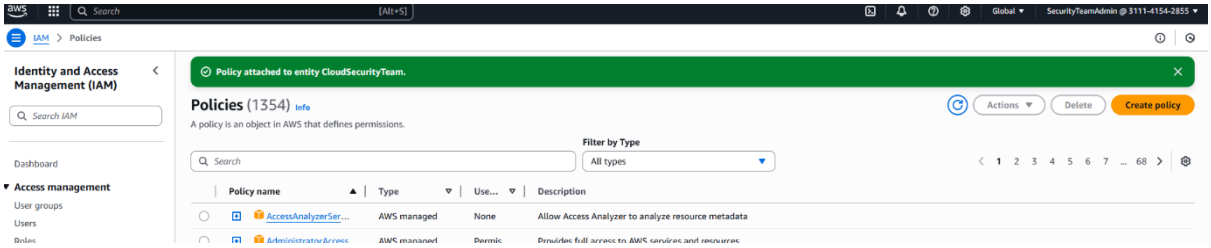
✅ 3. Creating IAM Groups and Adding Users

- Created an IAM group (e.g., AuditorsGroup) with limited permissions.
- Attached policies to the group and added users to ensure permission consistency and scalability.



✅ 4. Implementing IAM Policies

- Created custom IAM policies using **JSON syntax** to grant granular access.
- Assigned policies to users and groups to restrict or allow access to specific S3 buckets.



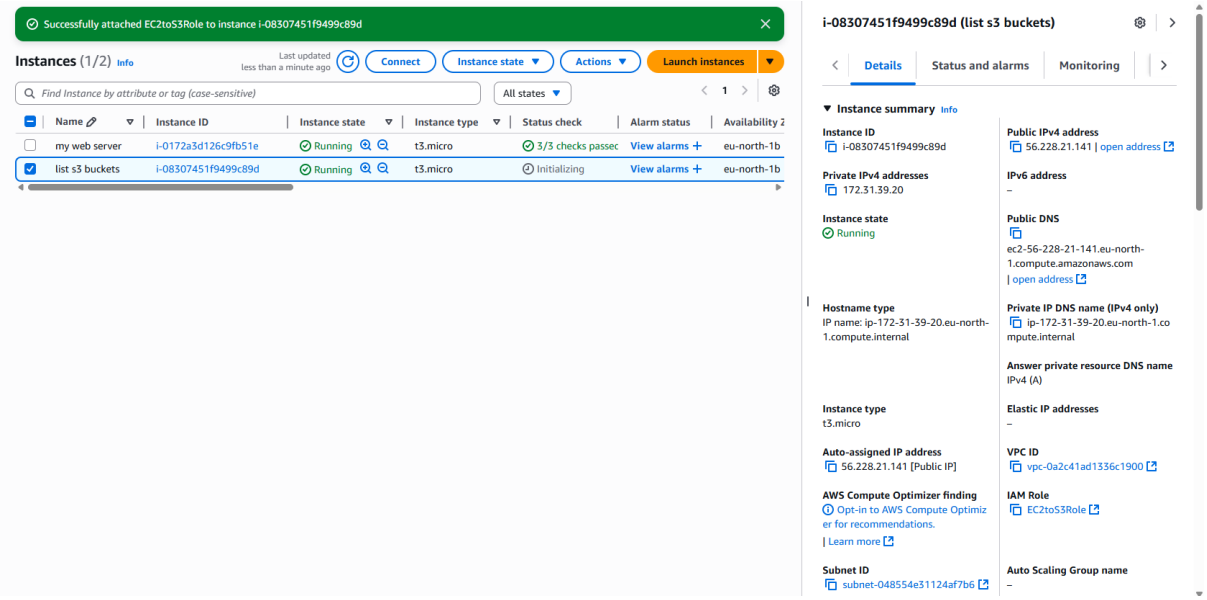
✅ 5. Creating and Uploading to S3 Buckets

- Created a new **S3 bucket** to simulate secure financial report storage.
- Uploaded documents using the AWS CLI and ensured proper encryption and access policies.

✅ 6. Creating IAM Roles for AWS Services

- Created service roles (e.g., for EC2 and Lambda) with necessary trust relationships and attached policies.

🧠 *Example:* EC2 role that allows read/write to an S3 bucket for backup purposes.



✅ 7. Cross-Account Access Using IAM Roles

- Set up a **role in the main account** (Account A) that allows **auditing account** (Account B) to assume it.
- Attached a policy that allows read-only access to financial reports in S3.

json

CopyEdit

```
{
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::ACCOUNT-B-ID:root"
  },
  "Action": "sts:AssumeRole",
  "Condition": {
    "StringEquals": {
      "sts:ExternalId": "AuditorExternalId123"
    }
  }
}
```

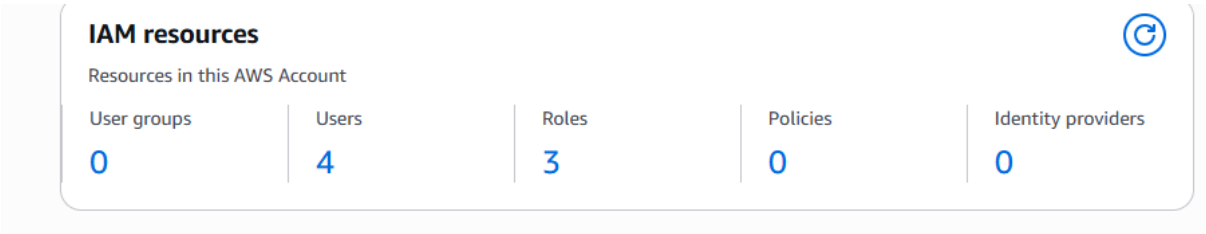
```
C:\Users\sevan>aws iam attach-group-policy --group-name CloudSecurityTeam --policy-arn "arn:aws:iam::aws:policy/AmazonS3FullAccess"

C:\Users\sevan>
```

✅ 8. Using External IDs for Secure Role Assumption

- Implemented **external ID** mechanisms to prevent confused deputy attacks.
- Shared role ARN and external ID with the auditing firm for secure, traceable access.

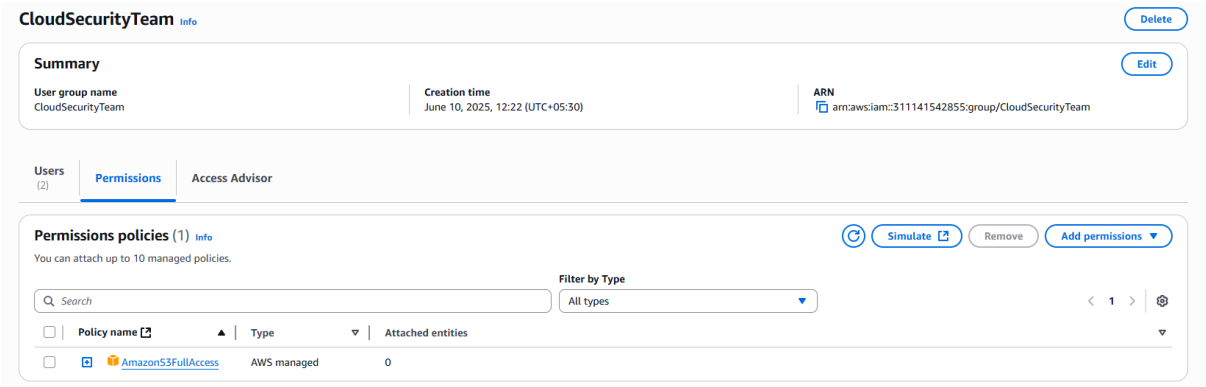
✅ 9. Revoking IAM Roles

- Demonstrated how to **revoke role access** by detaching policies or deleting the trust relationship.



✅ 10. Setting Permissions Boundaries

- Implemented **permissions boundaries** to ensure users can't exceed granted permissions, even if a policy allows more.



✅ 11. Testing Policies with IAM Policy Simulator

- Used the **IAM Policy Simulator** to verify and debug IAM policies.
- Confirmed expected behavior before deployment—reducing misconfigurations and improving security posture.

🧠 Key Takeaways

- IAM is the **foundation of cloud security**—understanding it is crucial for any AWS-based application.
- Using **least privilege principles** and **role assumption with external IDs** ensures secure and auditable third-party access.
- Combining **console operations**, **CLI scripting**, and **policy simulation** provides robust control and testing capabilities.
- Practical, scenario-based learning fosters a deep understanding of real-world challenges in cloud security.

🔗 **Technologies Used**

- **AWS IAM**
- **Amazon S3**
- **AWS CLI**
- **IAM Policy Simulator**
- **JSON (for policy definitions)**

🏁 **Final Outcome**

By the end of the project, I was able to simulate a **real-world IAM security scenario** where a third-party auditor could securely access specific S3 resources in a cross-account setup—achieved through carefully crafted IAM roles, policies, and permissions boundaries.

This project reinforced my understanding of **identity and access management**, policy design, and AWS security best practices—an essential foundation for any role in **cloud engineering** or **DevSecOps**.