DAY-02 | IAM: Identity Access & Management | AWS Cloud Practitioner Certification CLF-C02

Created on 2024-07-16 18:00
Published on 2024-07-17 03:55

IAM: Identity Access & Management

➤ What Is IAM?

➥ IAM: Users & Groups

➥ IAM: Permissions

➥ IAM Policies Inheritance

➥ IAM Policies Structure

➥ IAM – Password Policy

➥ IAM Roles for Services

➥ IAM Security Tools

➥ IAM Guidelines & Best Practices

➥ Shared Responsibility Model for IAM

➤ Multi Factor Authentication - MFA

➤ MFA devices options in AWS

➤ How can users access AWS ?

➤ What’s the AWS CLI?

➤ What’s the AWS SDK?

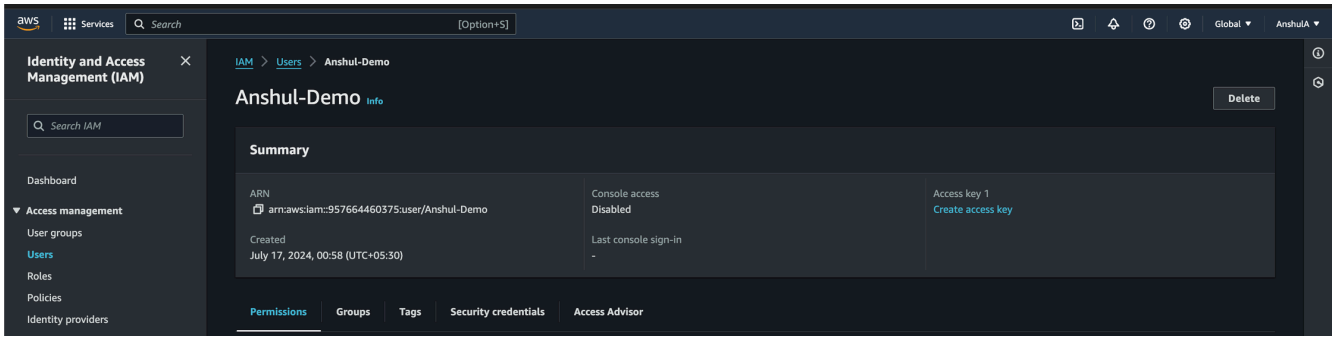
➤ IAM Section – Summary

➔ What Is IAM?

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. IAM is used to control who is authenticated (signed in) and authorized (has permissions) to use resources.

☞ IAM: Users & Groups

- **IAM** = Identity and Access Management, Global service
- **Root account** created by default, shouldn’t be used or shared
- **Users** are people within your organization, and can be grouped
- **Groups** only contain users, not other groups
- Users don’t have to belong to a group, and user can belong to multiple groups



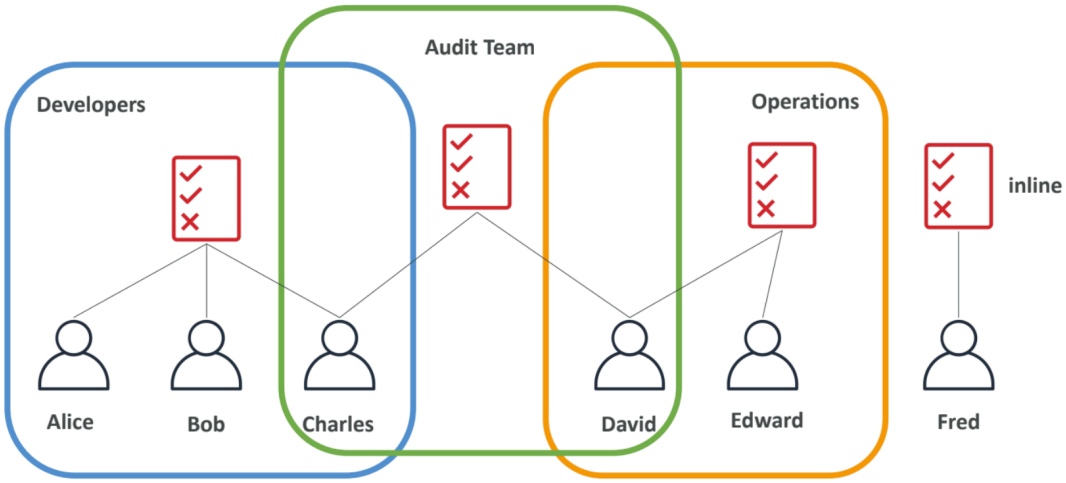
AWS Identity and Access Management (IAM)

☞ IAM: Permissions

Permissions in IAM are granted through policies. A policy is a document that specifies permissions as JSON. AWS evaluates these policies to determine whether to allow or deny a request. In AWS you apply the least privilege principle, don’t give more permissions than a user needs.

☞ IAM Policies Inheritance

Policies can be attached to users, groups, or roles. When a user is part of a group, the user inherits the permissions assigned to the group. Similarly, if a role is assigned to an AWS resource, the resource inherits the permissions of that role.



IAM Policies Inheritance

☞ IAM Policies Structure

IAM policies consist of:

- **Version**: Specifies the version of the policy language.
- **Id**: an identifier for the policy (optional)

- **Statement**: one or more individual statements (required)

➙ **Statement**: *Statements consists of*

↦ **Sid**: an identifier for the statement (optional)

↦ **Effect**: whether the statement allows or denies access (Allow, Deny)

↦ **Principal**: account/user/role to which this policy applied to

↦ **Action**: list of actions this policy allows or denies

↦ **Resource**: list of resources to which the actions applied to

↦ **Condition**: conditions for when this policy is in effect (optional)

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "EC2",
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "ec2:Get*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ELB",
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Sid": "Cloudwatch",
```

IAM Policies Structure

☞ **IAM – Password Policy**

Password policies in IAM are used to define the complexity requirements for user passwords. This can include:

↦ Minimum length

↦ Requirement for uppercase letters, lowercase letters, numbers, and special characters

↦ Password expiration and reuse restrictions

☞ **IAM Roles for Services**

Some AWS service will need to perform actions on your behalf. To do so, we will assign permissions to AWS services with IAM Roles.

Common roles :

↦ EC2 Instance Roles

↦ Lambda Function Roles

↦ Roles for CloudFormation

☞ **IAM Security Tools**

AWS provides several tools to enhance IAM security:

↦ **IAM Access Analyzer**: Helps identify resources shared with an external entity.

↦ **Credential Reports**: Provides a report of all your account's users and the status of their credentials.

↦ **IAM Policy Simulator**: Tests and troubleshoots policies to see the effect of policy changes.

☞ **IAM Guidelines & Best Practices**

- Don’t use the root account except for AWS account setup
- One physical user = One AWS user
- Follow the principle of least privilege: Grant only the permissions required to perform a task.
- Use groups to manage permissions.
- Regularly rotate security credentials.
- Create a **strong password policy**
- Enable MFA for privileged users.
- Monitor and audit IAM activities with AWS CloudTrail.
- Audit permissions of your account with the IAM Credentials Report
- **Never share IAM users & Access Keys**

IAM Guidelines & Best Practices

☞ Shared Responsibility Model for IAM

AWS operates under a shared responsibility model:

- **AWS**: Responsible for protecting the infrastructure that runs all the services offered in the AWS Cloud.

- **Customer**: Responsible for managing their data, identity, and access management, including IAM policies and credentials.

AWS	YOU
Infrastructure (global network security)	Users, Groups, Roles, Policies management and monitoring
Configuration and vulnerability analysis	Enable MFA on all accounts
Compliance validation	Rotate all your keys often, Use IAM tools to apply appropriate permissions, Analyze access patterns & review permissions

Shared Responsibility Model for IAM

➔ Multi Factor Authentication - MFA

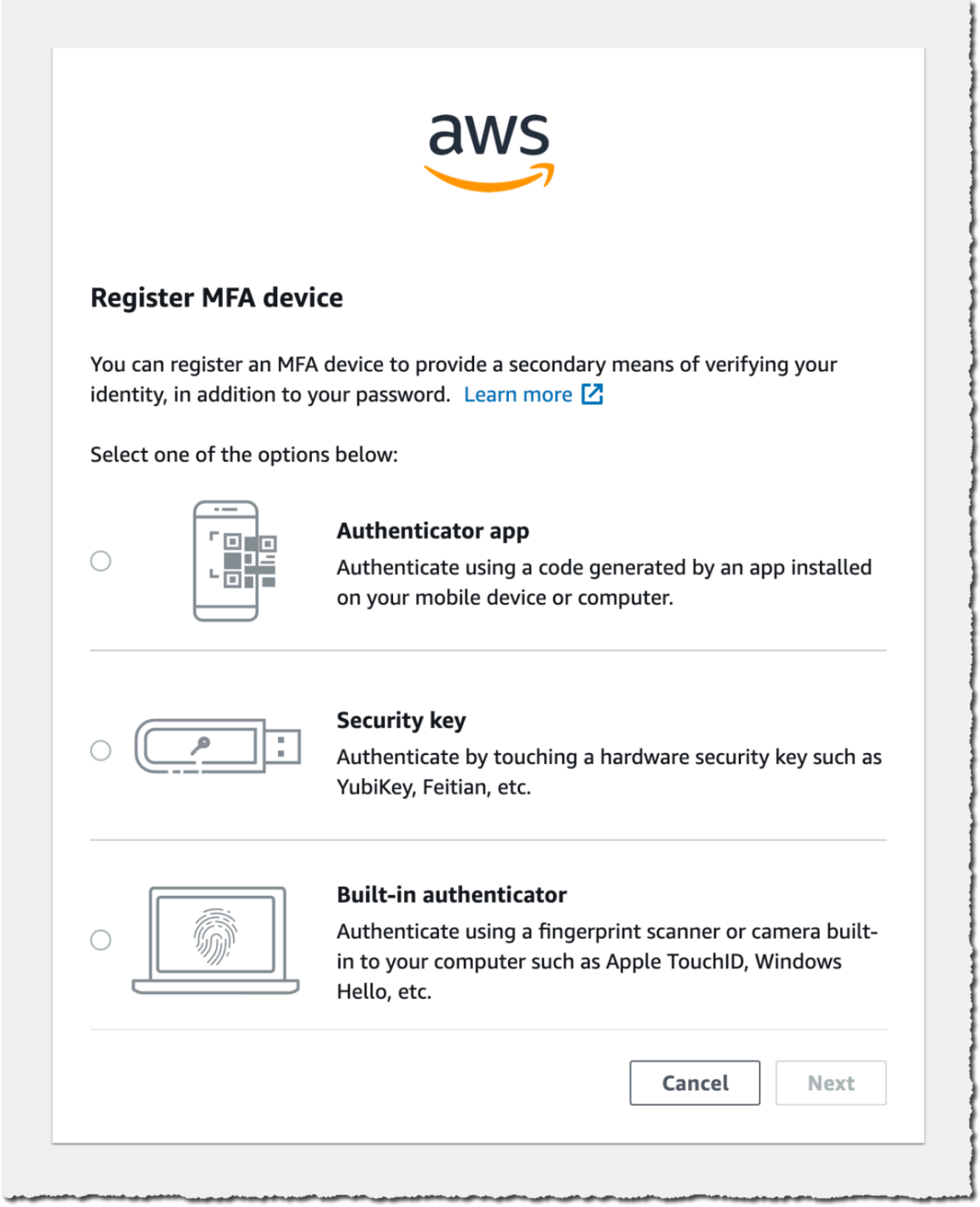
MFA enhances security by requiring users to provide multiple forms of verification (e.g., a password and a one-time code).

MFA Devices Options in AWS

↦ **Virtual MFA devices**: Software-based apps that generate a one-time code.

↦ **Hardware MFA devices**: Physical devices that generate a one-time code.

↦ **U2F security keys**: Physical USB keys for authentication.



MFA Devices Options in AWS

➔ MFA devices options in AWS

- Virtual MFA device (Support for multiple tokens on a single device.)Google Authenticator (phone only)Authy (multi-device)

- Universal 2nd Factor (U2F) Security Key (Support for multiple root and IAM users using a single security key)YubiKey by Yubico (3rd party)
- Hardware Key Fob MFA Device
- Hardware Key Fob MFA Device for AWS GovCloud (US)

➔ How can users access AWS ?

Users can access AWS via:

- **AWS Management Console**: A web-based user interface.
- **AWS Command Line Interface (CLI)**: A tool to control AWS services using commands in your command-line shell.
- **AWS SDKs**: Software Development Kits (SDKs) that provide APIs to interact with AWS services in various programming languages.

⇾ Access Keys are generated through the AWS Console

⇾ Users manage their own access keys

⇾ Access Keys are secret, just like a password. Don’t share them

⇾ Access Key ID ~= username

⇾ Secret Access Key ~= password

➔ What’s the AWS CLI?

The AWS Command Line Interface (CLI) is a unified tool to manage your AWS services. It’s open-source https://github.com/aws/aws-cli. With just one tool, you can

control multiple AWS services from the command line and automate them through scripts.

➔ What's the AWS SDK?

The AWS Software Development Kit (SDK) provides a set of APIs in various programming languages (like Python, Java, .NET, and others) to interact with AWS services. SDKs simplify using AWS services by providing high-level abstractions and handling many low-level details like authentication, retries, and data marshaling.

➔ IAM Section – Summary

- **Users:** mapped to a physical user, has a password for AWS Console
- **Groups:** contains users only
- **Policies:** JSON document that outlines permissions for users or groups
- **Roles:** for EC2 instances or AWS services
- **Security:** MFA + Password Policy
- **AWS CLI:** manage your AWS services using the command-line
- **AWS SDK:** manage your AWS services using a programming language
- **Access Keys:** access AWS using the CLI or SDK
- **Audit:** IAM Credential Reports & IAM Access Advisor

Happy Learning !