IAM Section

Module 3

What is IAM?

- IAM stands for Identity Access Management.
- IAM allows you to manage users and their level of access to the aws console.
- It is used to set users, permissions and roles. It allows you to grant access to the different parts of the aws platform.
- AWS Identity and Access Management is a web service that enables Amazon Web Services (AWS) customers to manage users and user permissions in AWS.
- With IAM, Organizations can centrally manage users, security credentials such as access keys, and permissions that control which AWS resources users can access.
- Without IAM, Organizations with multiple users must either create multiple user accounts, each with its own billing and subscriptions to AWS products or share an account with a single security credential. Without IAM, you also don't have control about the tasks that the users can do.
- IAM enables the organization to create multiple users, each with its own security credentials, controlled and billed to a single aws account. IAM allows the user to do only what they need to do as a part of the user's job.

IAM: Users & Groups

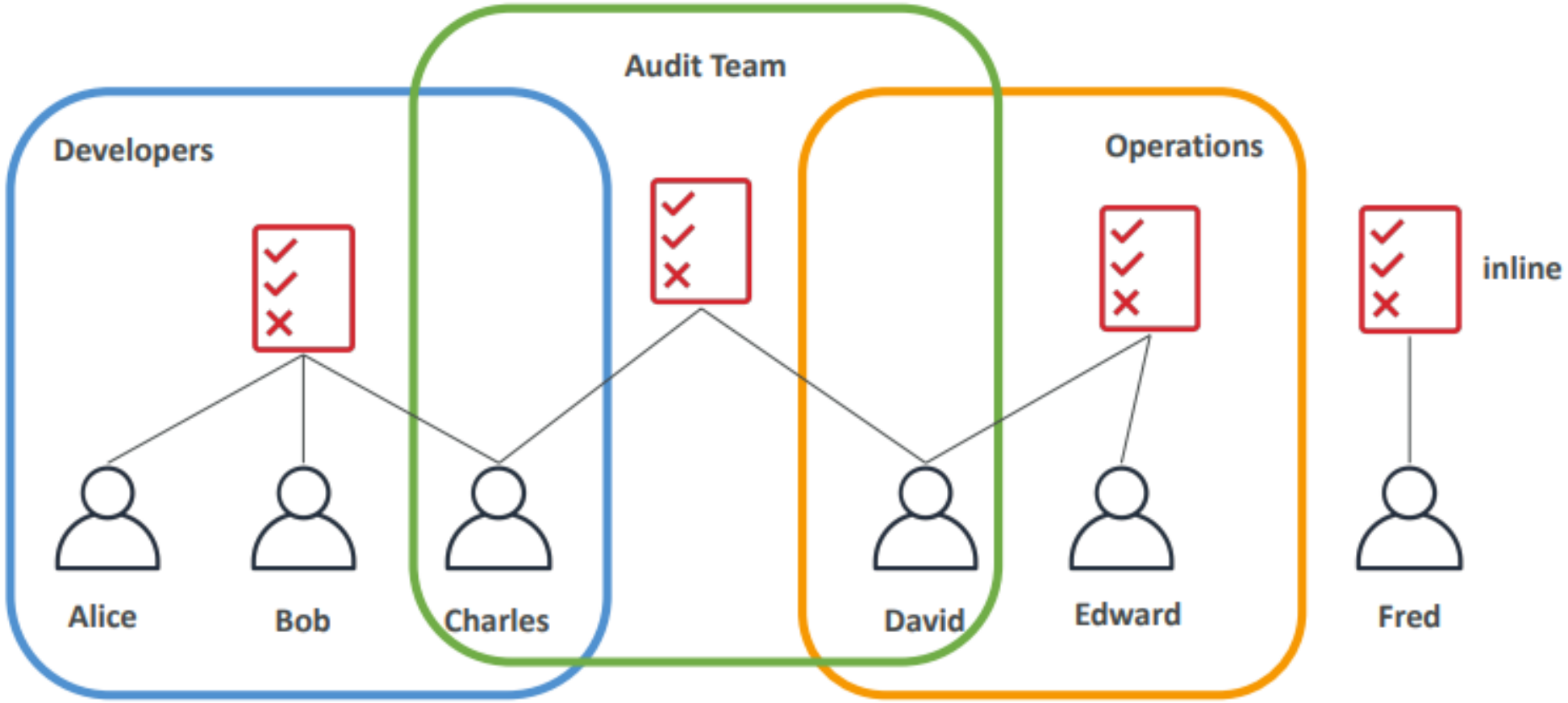
- IAM = Identity and Access Management, Global service
- Root account created by default, shouldn’t be used or shared
- Users are people within your organization, and can be grouped
- Groups only contain users, not other groups
- Users don’t have to belong to a group, and user can belong to multiple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

IAM: Permissions

- Users or Groups can be assigned JSON documents called policies
- These policies define the permissions of the users
- In AWS you apply the least privilege principle: don’t give more permissions than a user needs

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "elasticloadbalancing:Describe*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "cloudwatch:ListMetrics",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:Describe*"
            ],
            "Resource": "*"
        }
    ]
}
```

IAM Policies inheritance



Features of IAM

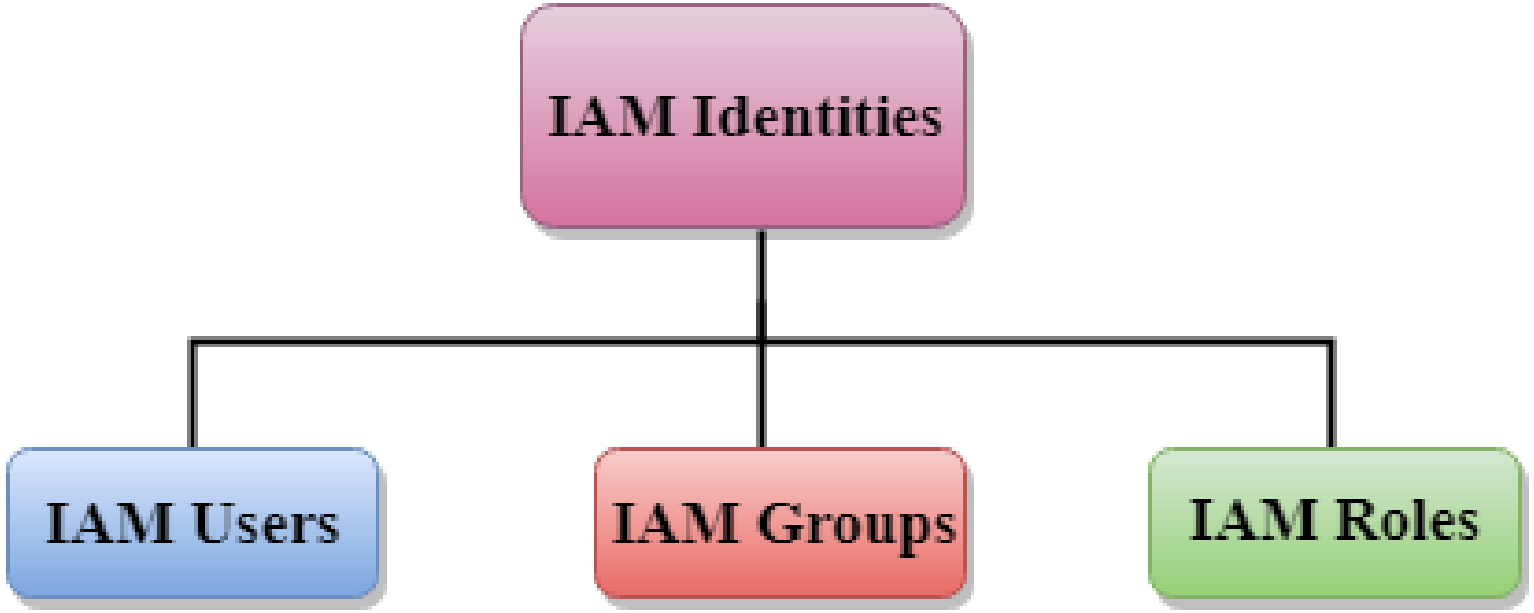
- **Centralised control of your AWS account:** You can control creation, rotation, and cancellation of each user's security credentials. You can also control what data in the aws system users can access and how they can access.
- **Shared Access to your AWS account:** Users can share the resources for the collaborative projects.
- **Granular permissions:** It is used to set a permission that user can use a particular service but not other services.
- **Identity Federation:** An Identity Federation means that we can use Facebook, Active Directory, LinkedIn, etc with IAM. Users can log in to the AWS Console with same username and password as we log in with the Active Directory, Facebook, etc.
- **Multifactor Authentication:** An AWS provides multifactor authentication as we need to enter the username, password, and security check code to log in to the AWS Management Console.
- **Permissions based on Organizational groups:** Users can be restricted to the AWS access based on their job duties, for example, admin, developer, etc.

Features of IAM

- **Networking controls:** IAM also ensures that the users can access the AWS resources within the organization's corporate network.
- **Provide temporary access for users/devices and services where necessary:** If you are using a mobile app and storing the data in AWS account, you can do this only when you are using temporary access.
- **Integrates with many different aws services:** IAM is integrated with many different aws services.
- **Supports PCI DSS Compliance:** PCI DSS (Payment Card Industry Data Security Standard) is a compliance framework. If you are taking credit card information, then you need to pay for compliance with the framework.
- **Eventually Consistent:** IAM service is eventually consistent as it achieves high availability by replicating the data across multiple servers within the Amazon's data center around the world.
- **Free to use:** AWS IAM is a feature of AWS account which is offered at no additional charge. You will be charged only when you access other AWS services by using IAM user.

IAM Identities

- IAM identities are created to provide authentication for people and processes in your aws account.



- IAM Users
- IAM Groups
- IAM Roles
- AWS Account Root User
- When you first create an AWS account, you create an account as a root user identity which is used to sign in to AWS.
- You can sign to the AWS Management Console by entering your email address and password. The combination of email address and password is known as **root user credentials**.
- When you sign in to AWS account as a root user, you have unrestricted access to all the resources in AWS account.
- The Root user can also access the billing information as well as can change the password also.

What is a Role?

- A role is a set of permissions that grant access to actions and resources in AWS. These permissions are attached to the role, not to an IAM User or a group.
- An IAM User can use a role in the same AWS account or a different account.
- An IAM User is similar to an IAM User; role is also an AWS identity with permission policies that determine what the identity can and cannot do in AWS.
- A role is not uniquely associated with a single person; it can be used by anyone who needs it.
- A role does not have long term security credential, i.e., password or security key. Instead, if the user uses a role, temporarily security credentials are created and provided to the user.
- You can use the roles to delegate access to users, applications or services that generally do not have access to your AWS resources.

Situations in which "IAM Roles" can be used:

- Sometimes you want to grant the users to access the AWS resources in your AWS account.
- Sometimes you want to grant the users to access the AWS resources in another AWS account.
- It also allows the mobile app to access the AWS resources, but not want to store the keys in the app.
- It can be used to grant access to the AWS resources which have identities outside of AWS.
- It can also be used to grant access to the AWS resources to the third party so that they can perform an audit on AWS resources.

Following are the important terms associated with the "IAM Roles":

- **Delegation:** Delegation is a process of granting the permissions to the user to allow the access to the AWS resources that you control. Delegation sets up the trust between a trusted account (an account that owns the resource) and a trusting account (an account that contains the users that need to access the resources).
  The trusting and trusted account can be of three types:
  - Same account
  - Two different accounts under the same organization control
  - Two different accounts owned by different organizations.

- To delegate permission to access the resources, an IAM role is to be created in the trusting account that has the two policies attached.
- **Permission Policy:** It grants the user with a role the needed permissions to carry out the intended tasks.
- **Trust Policy:** It specifies which trusted account members can use the role.
- **Federation:** Federation is a process of creating the trust relationship between the external service provider and AWS. For example, Facebook allows the user to login to different websites by using their facebook accounts.
- **Trust policy:** A document was written in JSON format to define who is allowed to use the role. This document is written based on the rules of the IAM Policy Language.
- **Permissions policy:** A document written in JSON format to define the actions and resources that the role can use. This document is based on the rules of the IAM Policy Language.
- **Permissions boundary:** It is an advanced feature of AWS in which you can limit the maximum permissions that the role can have. The permission boundaries can be applied to IAM User or IAM role but cannot be applied to the service-linked role.
- **Principal:** A principal can be AWS root account user, an IAM User, or a role. The permissions that can be granted in one of the two ways:
  - Attach a permission policy to a role.
  - The services that support resource-based policies, you can identify the principal in the principal element of policy attached to the resource.
- **Cross-account access: Roles vs Resource-Based Policies:** It allows you to grant access to the resources in one account to the trusted principal in another account is known as cross-account access. Some services allow you to attach the policy directly, known as Resource-Based policy. The services that support Resource-Based Policy are Amazon S3 buckets, Amazon SNS, Amazon SQS Queues.

IAM Roles Use Cases

- There are two ways to use the roles:
- **IAM Console:** When IAM Users working in the IAM Console and want to use the role, then they access the permissions of the role temporarily. An IAM Users give up their original permissions and take the permissions of the role. When IAM User exits the role, their original permissions are restored.
- **Programmatic Access:** An AWS service such as Amazon EC2 instance can use role by requesting temporary security credentials using the programmatic requests to AWS.

An IAM Role can be used in the following ways:

- **IAM User:** IAM Roles are used to grant the permissions to your IAM Users to access AWS resources within your own or different account. An IAM User can use the permissions attached to the role using the IAM Console. A Role also prevents the accidental access to the sensitive AWS resources.
- **Applications and Services:** You can grant the access of permissions attached with a role to applications and services by calling the AssumeRole API function. The AssumeRole function returns a temporary security credentials associated with a role. An application and services can only take those actions which are permitted by the role. An application cannot exit the role in the way the IAM User in Console does, rather it stops using with the temporary credentials and resumes its original credentials.
- **Federated Users:** Federated Users can sign in using the temporary credentials provided by an identity provider. AWS provides an IDP (identity provider) and temporary credentials associated with the role to the user. The credentials grant the access of permissions to the user.

IAM Policies Structure

- Consists of
  - **Version:** policy language version, always include "2012-10-17"
  - **Id:** an identifier for the policy (optional)
  - **Statement:** one or more individual statements (required)
- 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",
    "Id": "S3-Account-Permissions",
    "Statement": [
        {
            "Sid": "1",
            "Effect": "Allow",
            "Principal": {
                "AWS": ["arn:aws:iam::123456789012:root"]
            },
            "Action": [
                "s3:GetObject",
                "s3:PutObject"
            ],
            "Resource": ["arn:aws:s3:::mybucket/*"]
        }
    ]
}
```

IAM – Password Policy

- Strong passwords = higher security for your account
- In AWS, you can setup a password policy:
  - Set a minimum password length
  - Require specific character types:
    - including uppercase letters
    - lowercase letters
    - numbers
    - non-alphanumeric characters
  - Allow all IAM users to change their own passwords
  - Require users to change their password after some time (password expiration)
  - Prevent password re-use

Multi Factor Authentication - MFA

- Users have access to your account and can possibly change configurations or delete resources in your AWS account
- **You want to protect your Root Accounts and IAM users**
- MFA = password *you know* + security device *you own*

- **Main benefit of MFA:**
  if a password is stolen or hacked, the account is not compromised

MFA devices options in AWS

Virtual MFA device

Google Authenticator
(phone only)

Authy
(multi-device)

Support for multiple tokens on a single device.

Universal 2nd Factor (U2F) Security Key

YubiKey by Yubico (3rd party)

Support for multiple root and IAM users using a single security key

MFA devices options in AWS

Hardware Key Fob MFA Device

Provided by Gemalto (3rd party)

Hardware Key Fob MFA Device for AWS GovCloud (US)

Provided by SurePassID (3rd party)

How can users access AWS ?

- To access AWS, you have three options:
- AWS Management Console (protected by password + MFA)
- AWS Command Line Interface (CLI): protected by access keys
- AWS Software Developer Kit (SDK) - for code: protected by access keys
- 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?

- A tool that enables you to interact with AWS services using commands in your command-line shell
- Direct access to the public APIs of AWS services
- You can develop scripts to manage your resources
- It’s open-source https://github.com/aws/aws-cli
- Alternative to using AWS Management Console

What’s the AWS SDK?

- AWS Software Development Kit (AWS SDK)
- Language-specific APIs (set of libraries)
- Enables you to access and manage AWS services programmatically
- Embedded within your application
- Supports
  - SDKs (JavaScript, Python, PHP, .NET, Ruby, Java, Go, Node.js, C++)
  - Mobile SDKs (Android, iOS, …)
  - IoT Device SDKs (Embedded C, Arduino, …)
- Example: AWS CLI is built on AWS SDK for Python

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

- IAM Credentials Report (account-level)
- a report that lists all your account's users and the status of their various credentials

- IAM Access Advisor (user-level)
- Access advisor shows the service permissions granted to a user and when those services were last accessed.
- You can use this information to revise your policies.

IAM Guidelines & Best Practices

- Don’t use the root account except for AWS account setup
- One physical user = One AWS user
- Assign users to groups and assign permissions to groups
- Create a strong password policy
- Use and enforce the use of Multi Factor Authentication (MFA)
- Create and use Roles for giving permissions to AWS services
- Use Access Keys for Programmatic Access (CLI / SDK)
- Audit permissions of your account using IAM Credentials Report & IAM Access Advisor
- Never share IAM users & Access Keys

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

- Thank you…