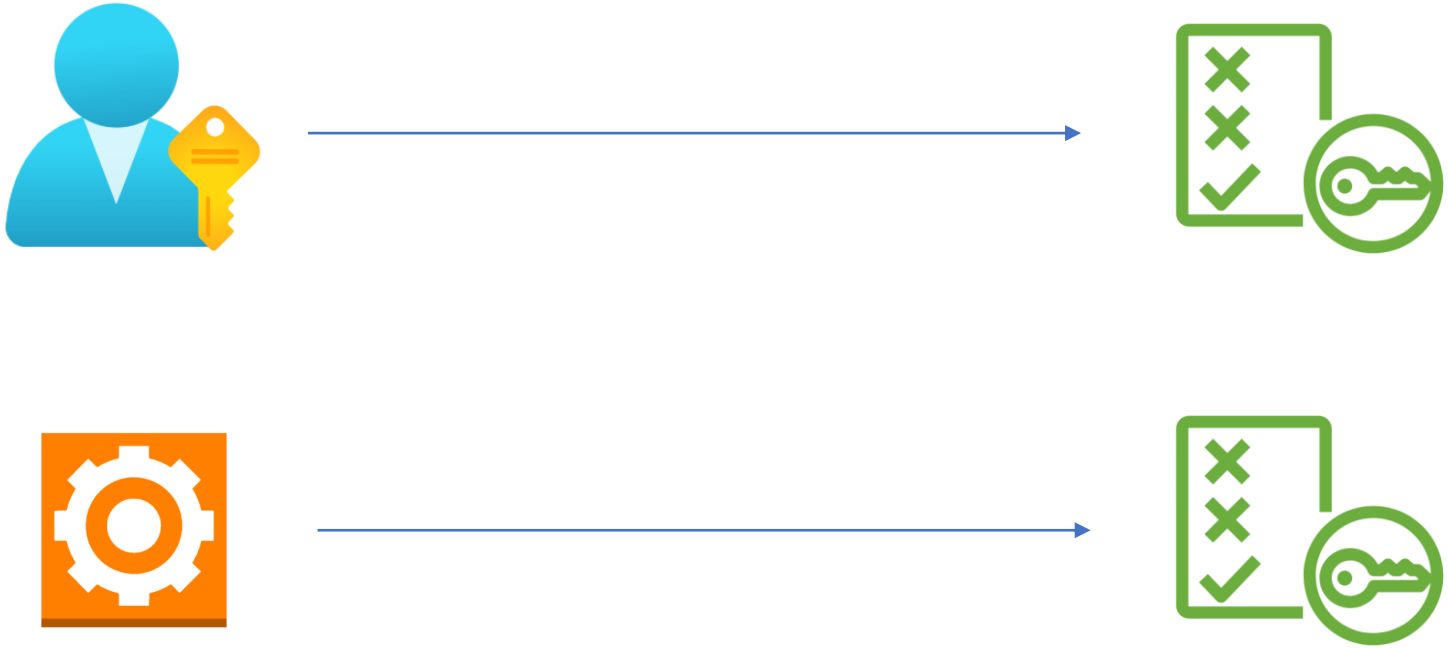
Identity and Access Management (IAM)

Contents

Overview of AWS IAM service

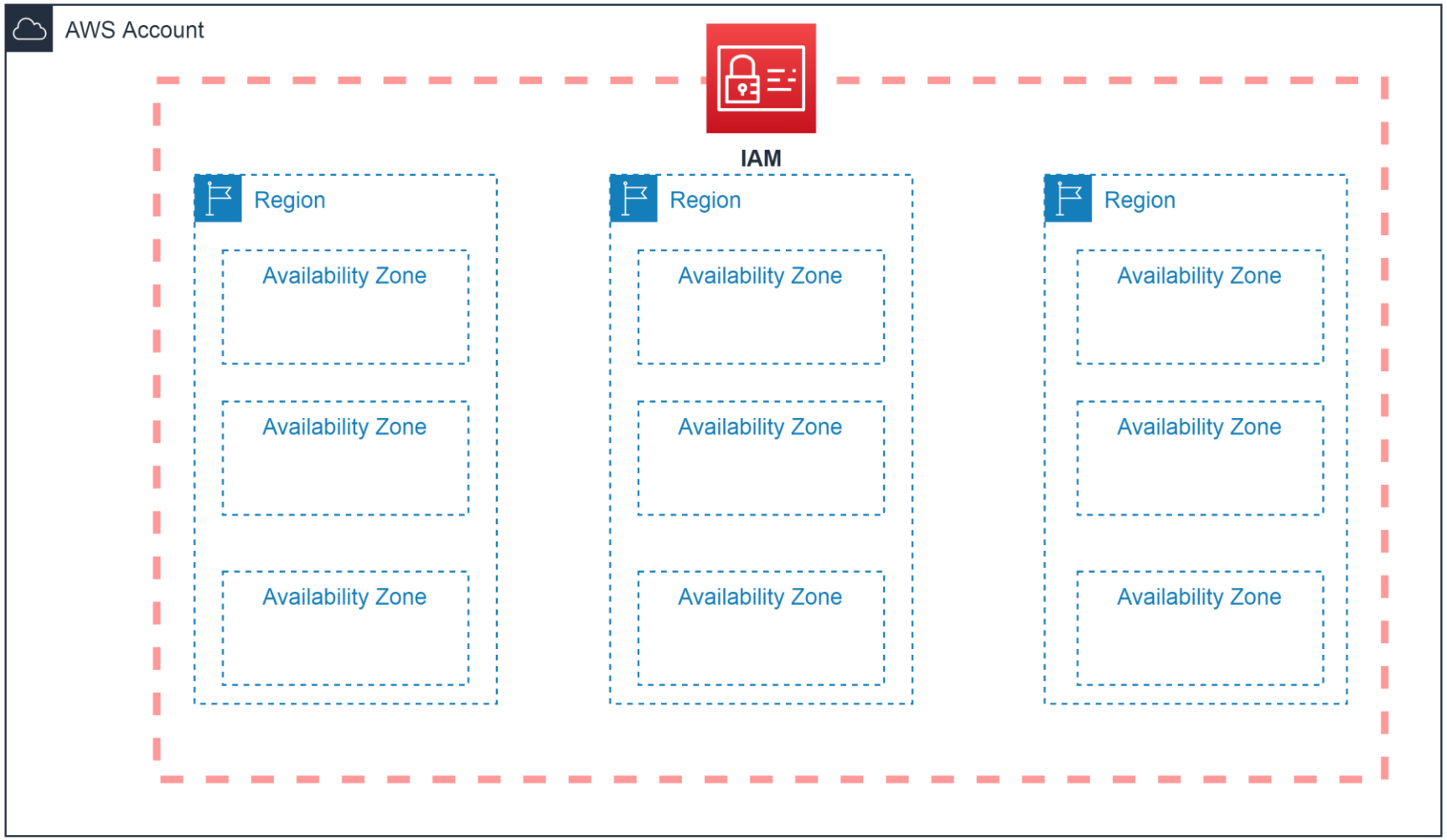


WHO CAN DO WHAT ?

Overview of AWS IAM service

- IAM stands for **Identity and Access Management**
- AWS IAM is a web service that helps you securely control access to AWS resources
- You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resource
- IAM is a Global service
- When you first create an AWS account, **Root** user is created by default, shouldn't be used or shared
- It is recommended that you do not use the root user for your everyday tasks, even the administrative ones

Global service vs Regional service



Understanding how IAM works

- IAM provides the infrastructure necessary to control **authentication** and **authorization** for your account.
- The IAM includes the following elements:

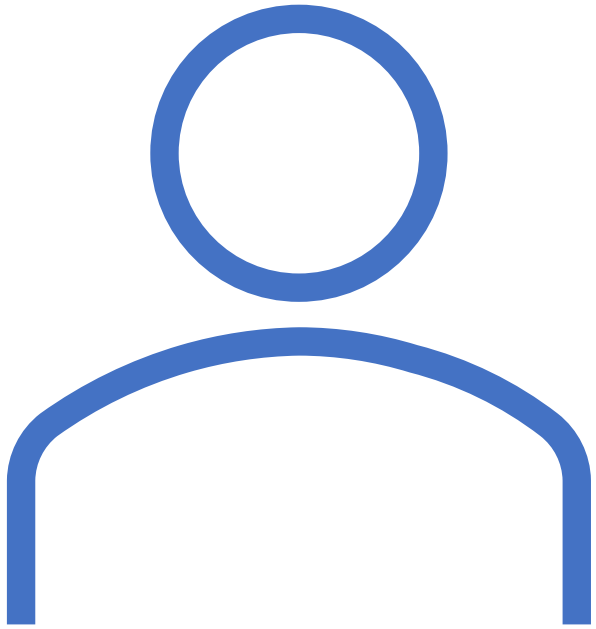
1. **Terms** – IAM Resources (User, Group, Role, Policy), IAM Identities (User, Group, and Role)
2. **Principal** – A principal is a person or application that can make a request for an action or operation on an AWS resource. [include federated users and assumed roles]
3. **Request –** When a principal tries to use the AWS Management Console, the AWS API, or the AWS CLI, that principal sends a request to AWS
4. **Authentication –** A principal must be authenticated (signed into AWS) using their credentials to send a request to AWS. [using your Username and Password]

Understanding how IAM works

5. **Authorization** – During authorization, AWS uses values from the request context to check for policies (JSON documents) that apply to the request. It then uses the policies to determine whether to allow or deny the request
6. **Actions or operations** – After your request has been authenticated and authorized, AWS approves the actions or operations in your request

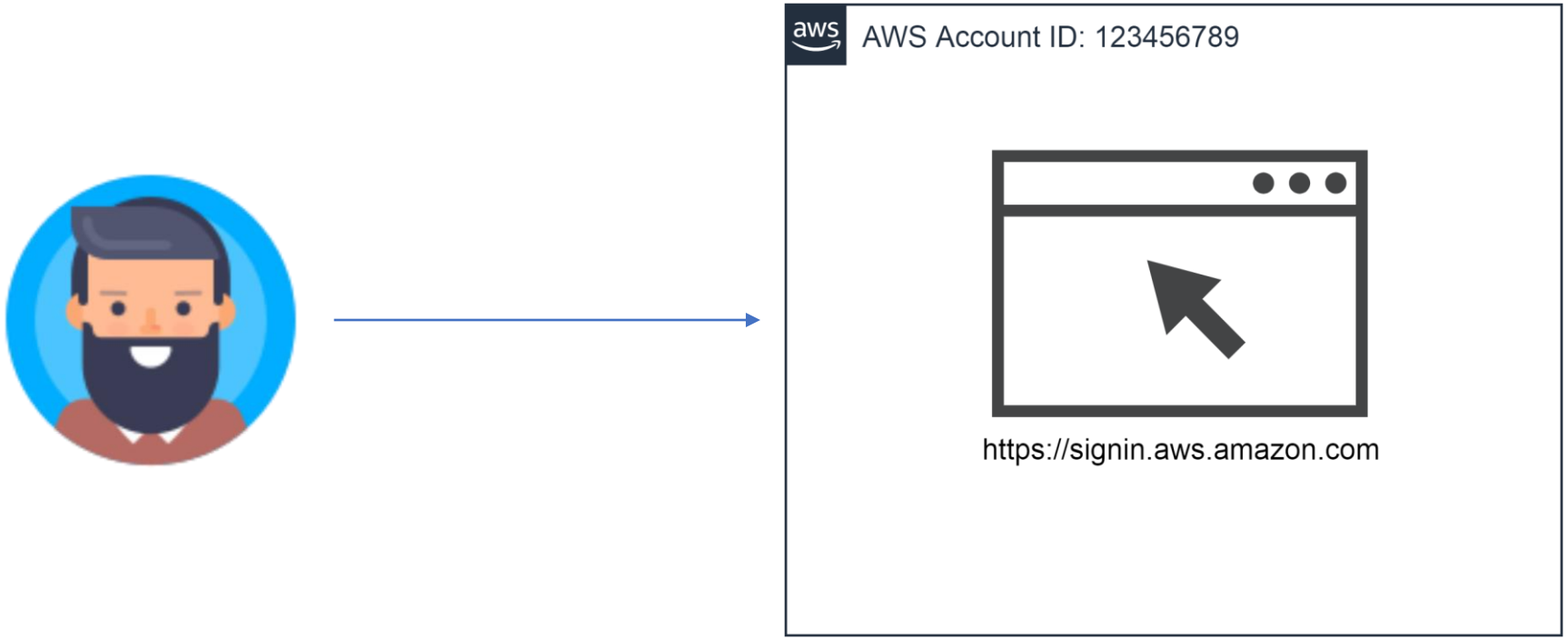
   Operations are defined by a service, and include things that you can do to a resource, such as viewing, creating, editing, and deleting that resource

   Example: CreateUser, DeleteUser, GetUser
7. **Resources –** A resource is an object that exists within a service. Examples include an Amazon EC2 instance, an IAM user, and an Amazon S3 bucket.



AWS IAM User

AWS IAM User



AWS IAM Users

- An AWS Identity and Access Management (IAM) user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS
- A user in AWS consists of a name and credentials
- An IAM user with administrator permissions is not the same thing as the AWS account root user.

How IAM Users are identified in AWS?

When you create a user, IAM creates these ways to identify that user:

1. A "friendly name" for the user, which is the name that you specified when you created the user, e.g. Dave. These are the names you see in the AWS Management Console
2. An Amazon Resource Name (ARN) for the user

   arn:aws:iam::account-ID-without-hyphens:user/<user_name>
3. A unique identifier for the user. This ID is returned only when you use the API, Tools for Windows PowerShell, or AWS CLI to create the user; you do not see this ID in the console

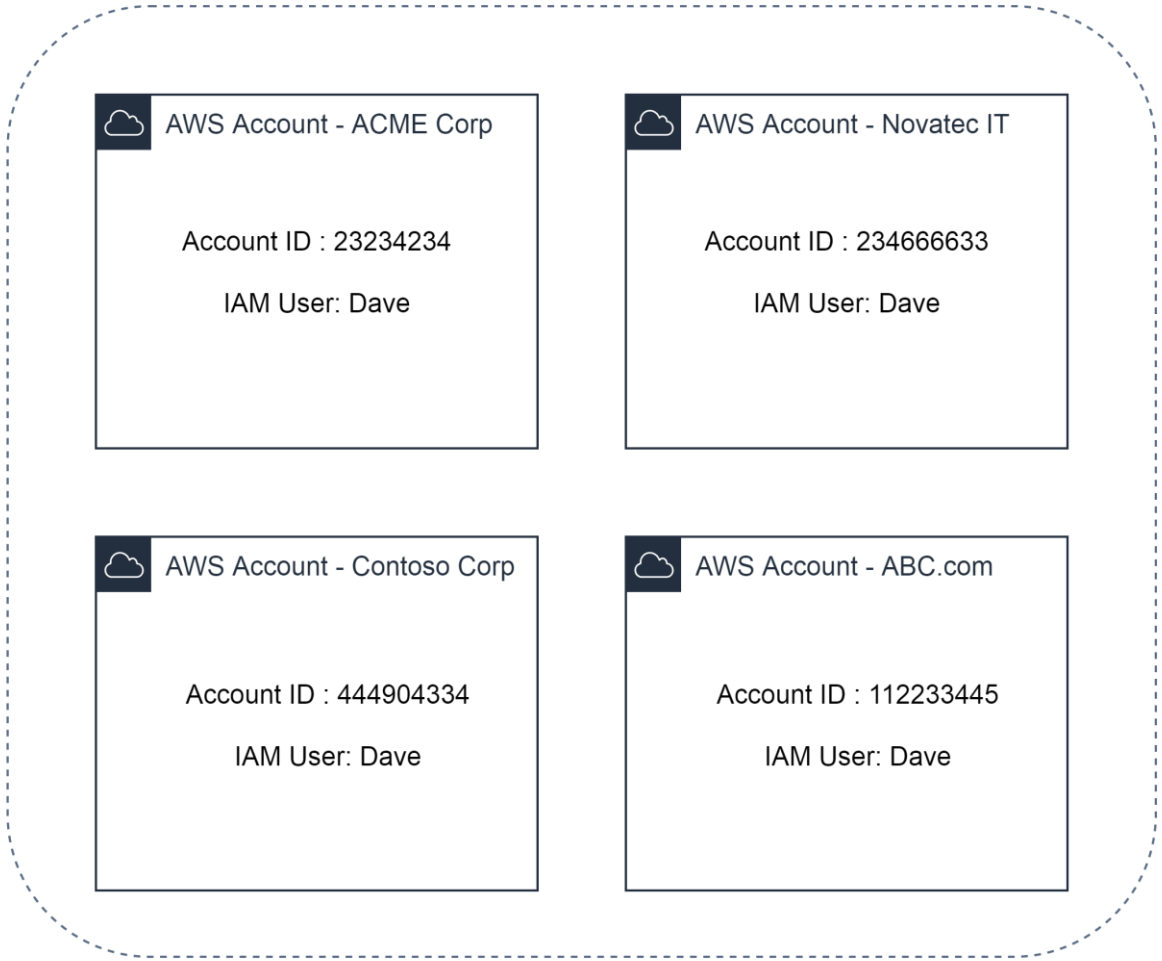
Different ways of accessing AWS

1. **AWS Console**: IAM user can use her **Username** and **Password** to sign into interactive sessions such as the AWS Management Console
2. **Access Keys:** A combination of an **access key ID** and a **secret access key**. You can assign two to a user at a time. These can be used to make programmatic calls to AWS. Example. AWS CLI, AWS API, Tools for PowerShell
3. **Server Certificates:** SSL/TLS certificates that you can use to authenticate with some AWS services. It is recommended that you use AWS Certificate Manager (ACM) service to provision, manage, and deploy your server certificates

Root user vs IAM User

Root User
Email : xyz@email.com
Password : **********

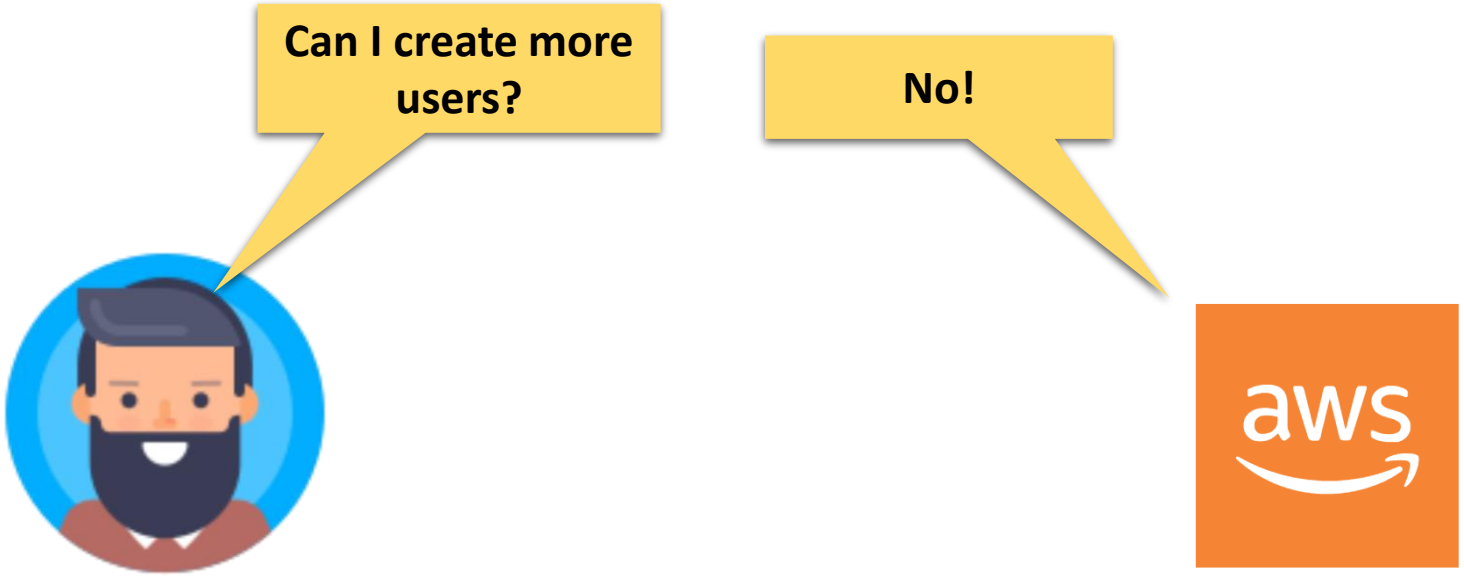
IAM User
Username : Dave
Password : **********
Account No. : 23234234



IAM User Limits

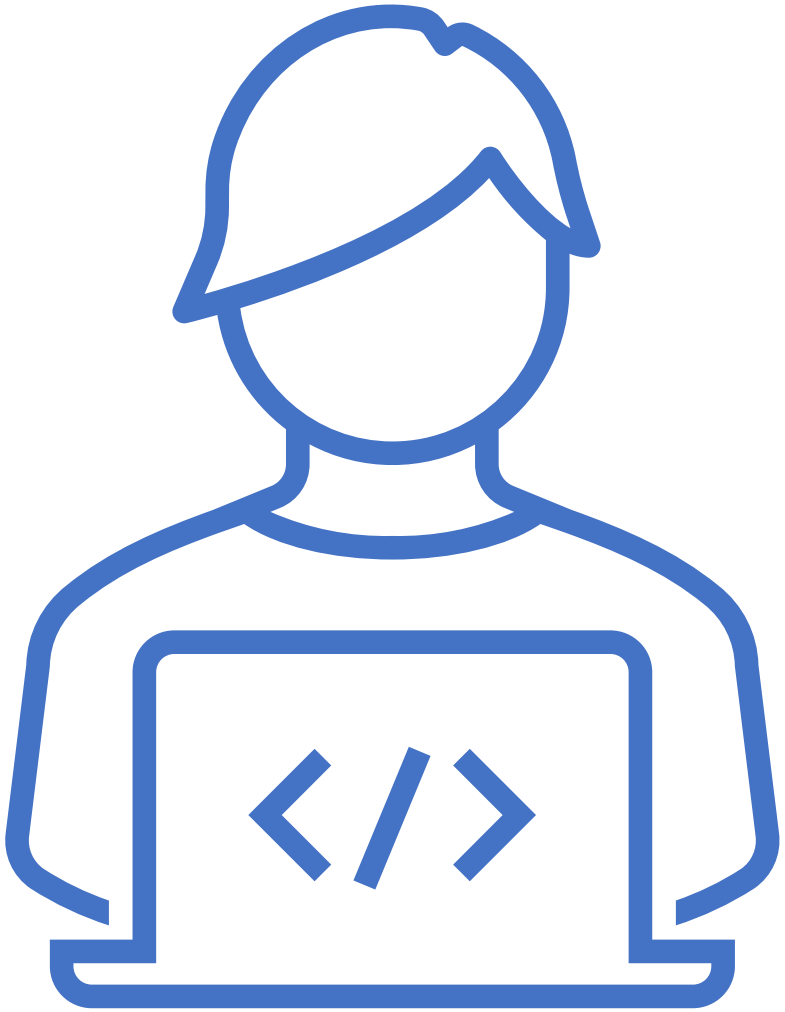
IAM Users Max: 5000 Users per Account

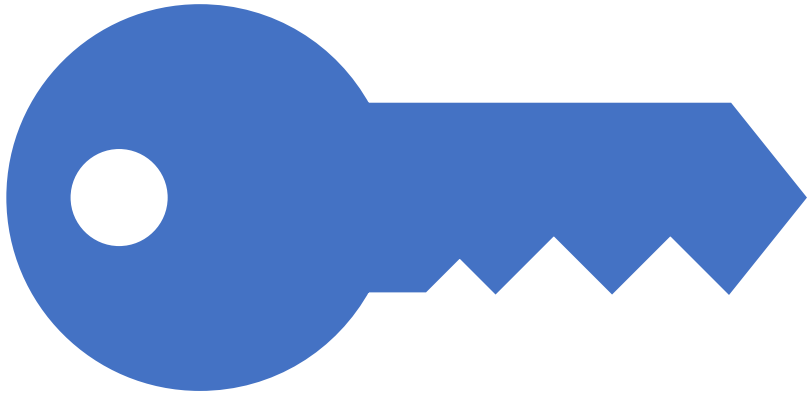
Root User Max: 1 per Account



Ref: https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_iam-quotas.html

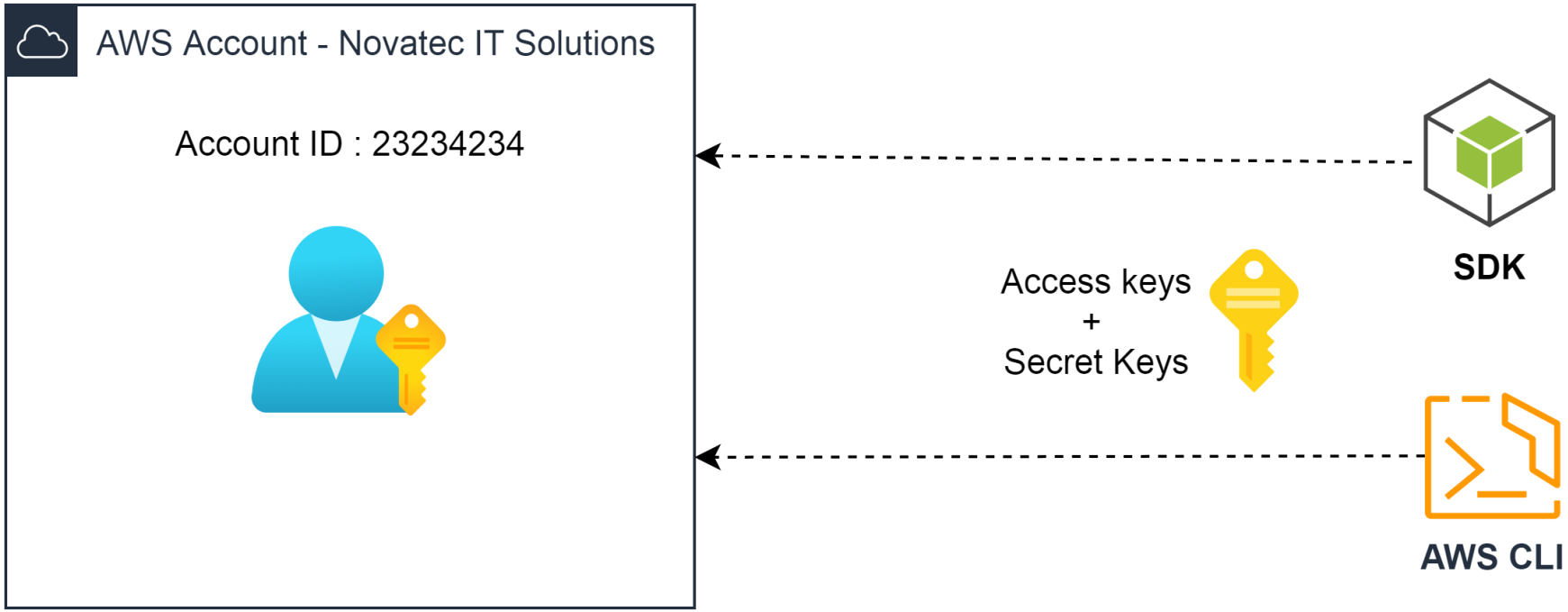
Hands-on Labs



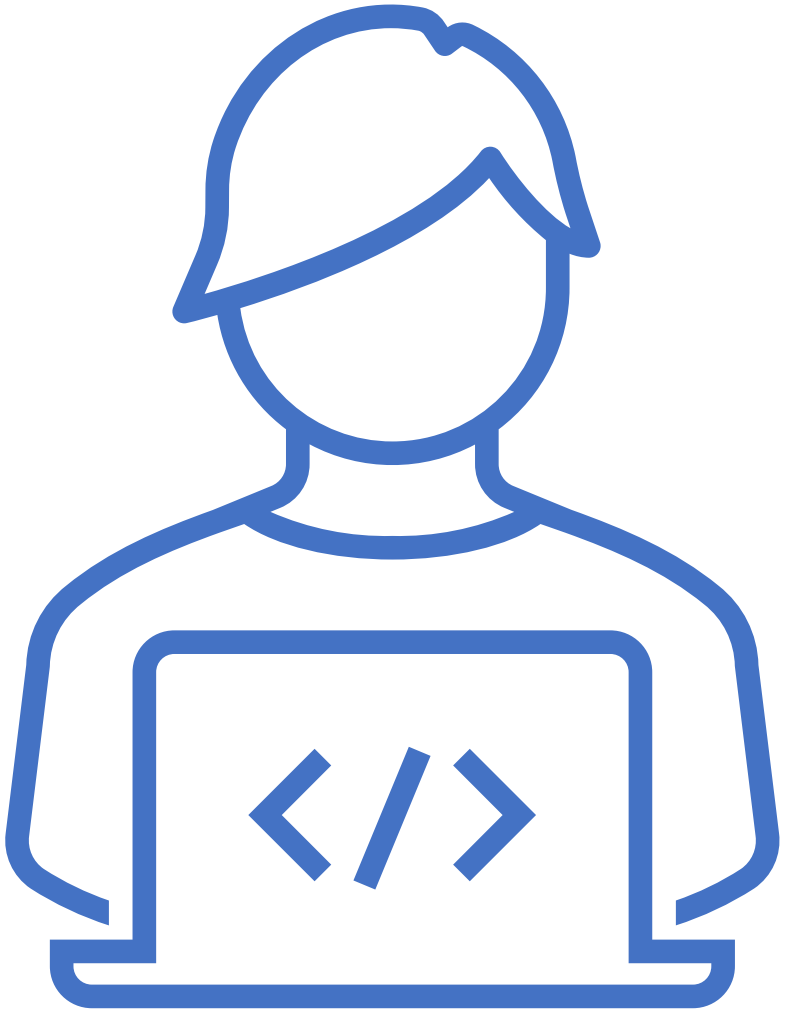


IAM Access Keys

IAM Access Keys



Hands-on Labs



Lab: Creating an AWS IAM User from AWS console

Lab: Creating an AWS IAM User using AWS CLI

Command: aws iam create-user - -user-name <username>

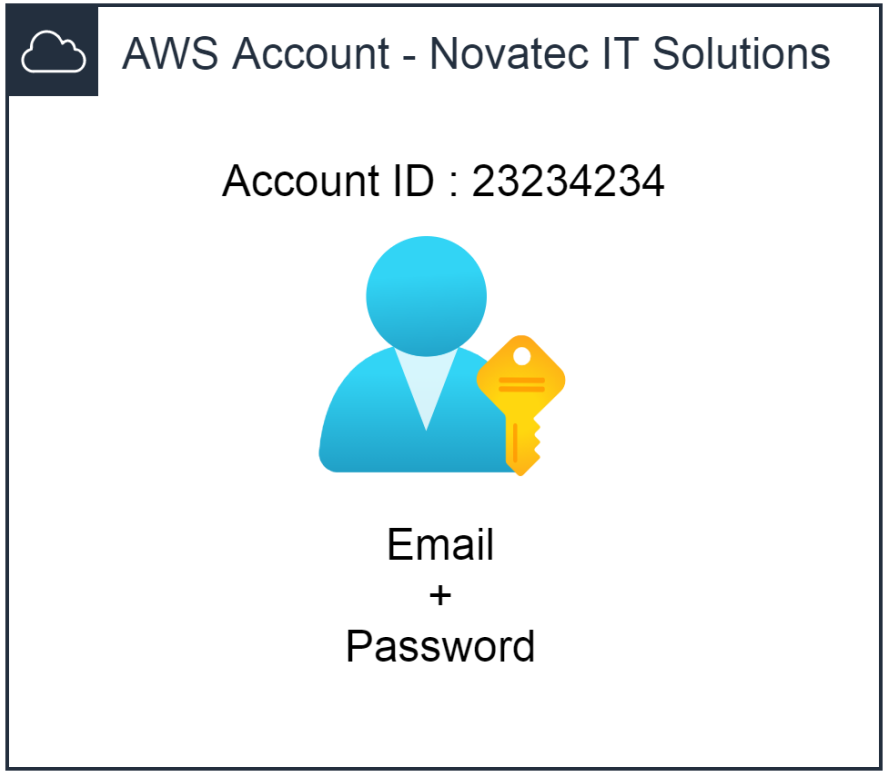
Ref: https://docs.aws.amazon.com/cli/latest/reference/iam/create-user.html for other available attributes.

Real World User Types

Real World Users

- Root User
- IAM User (Team Members)
- Federated Users
- Application
- Cross-Account User

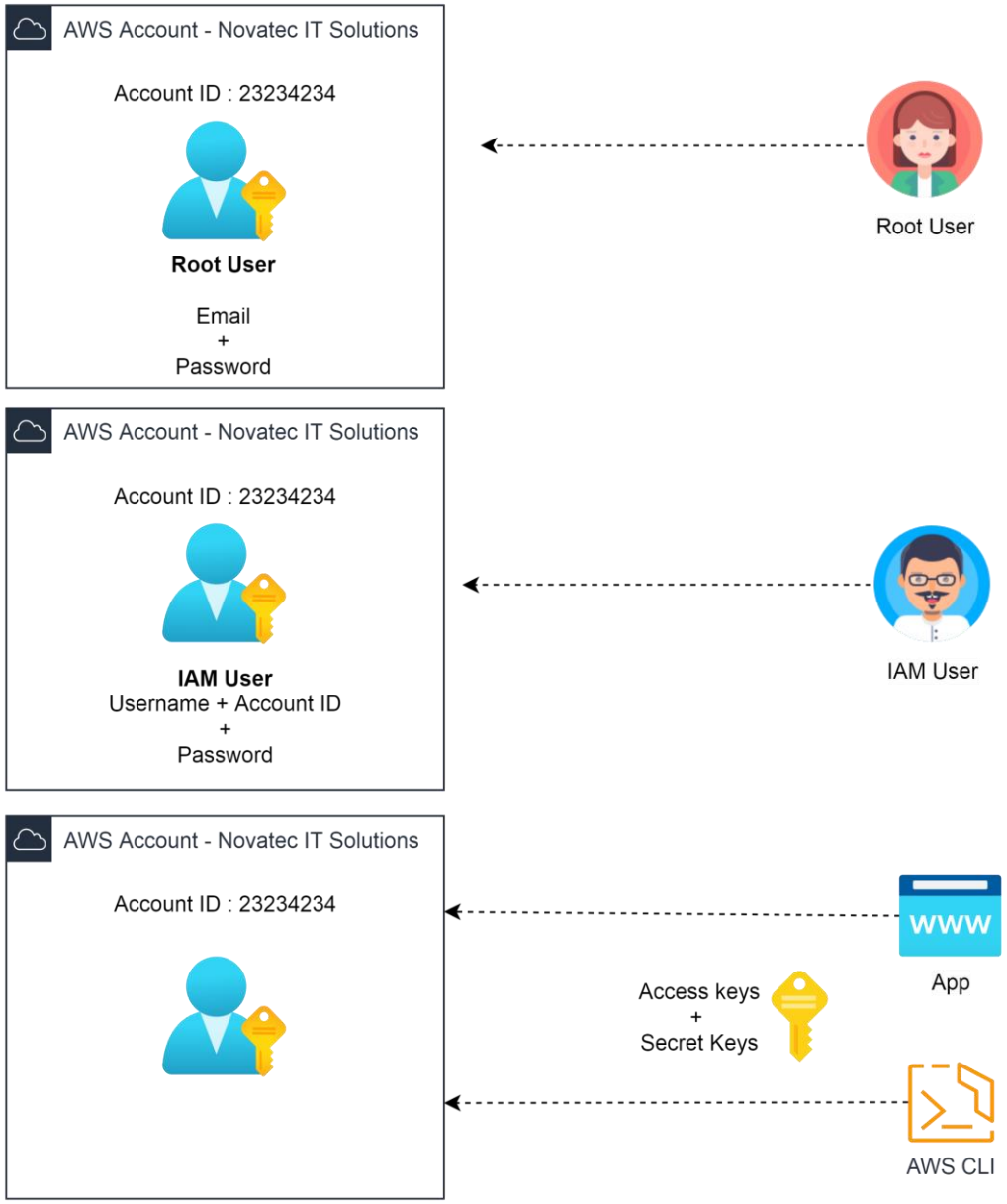
Root User



- Gets automatically created
- 1 Root user per Account
- "Owner" of the Account
- Able to delete the Account
- Has all the permissions
- AWS Organization can limit Root user
- Best Practice – Do not use the Root user credentials (Access Keys/Passwords)

Credentials

- Credential could be one or more information for proving your identity
- To prove you are, what you say you are

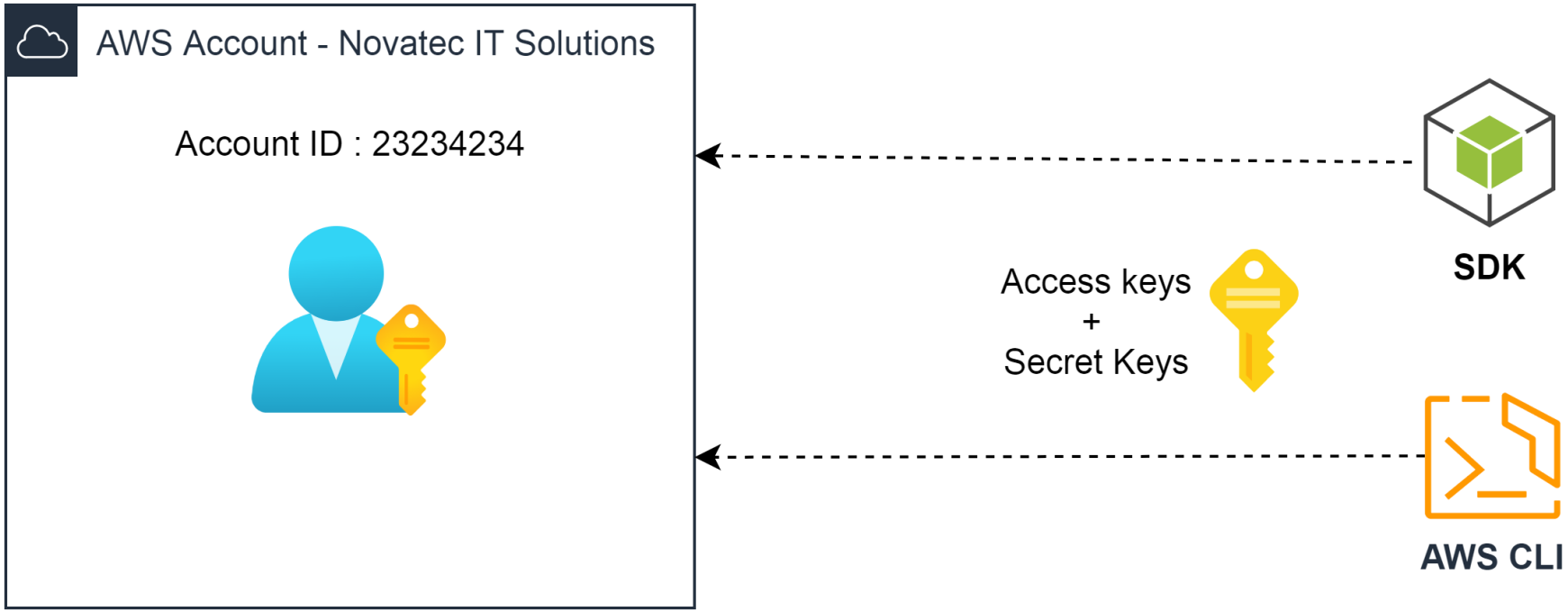


Federated Users

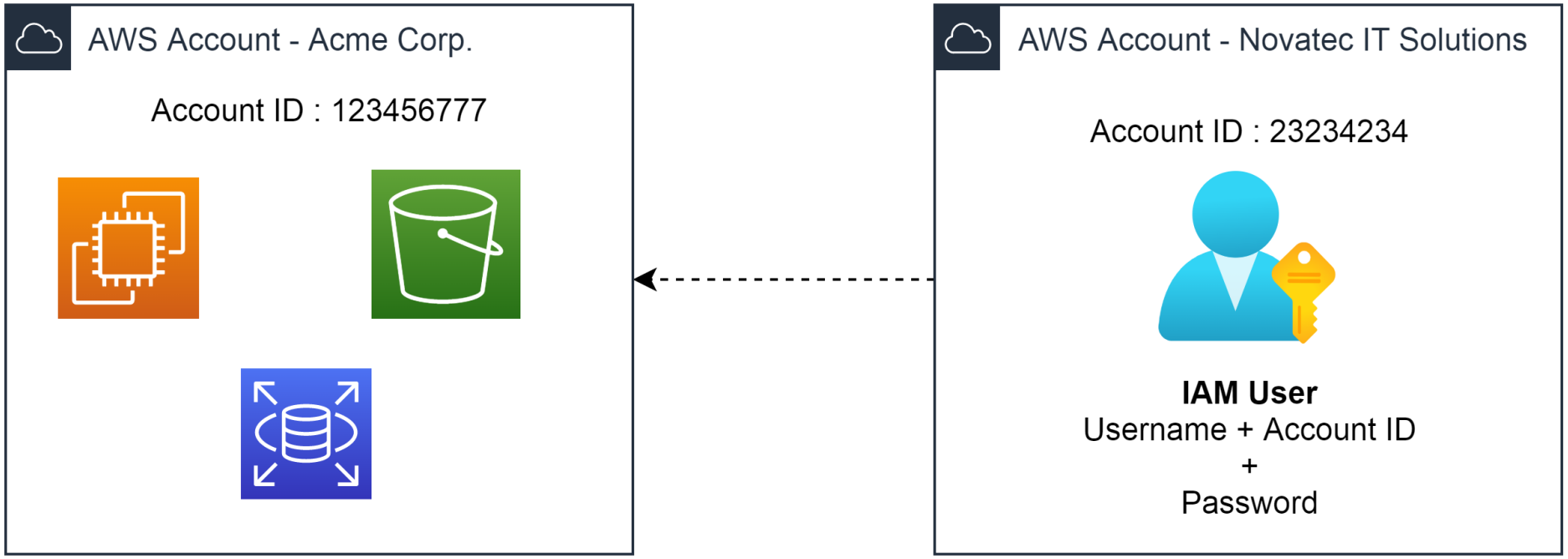
- Federated user is a common use-case with 100s or 1000s of users when access the resources in AWS Account
- Typical use-case is a company who has an Identity Management Solution like Active Directory
- The employees typically use their corporate username and password to login to their workstation. They can use the same credentials to access the resources in the AWS Account
- This is also known as Single-Sign-On

Application

Applications also need credentials to access the AWS resources



Cross-Account User



AWS IAM Group

IAM User Groups

- An IAM user group is a collection of IAM users
- User groups let you specify permissions for multiple users, which can make it easier to manage the permissions for those users
- For example, you could have a user group called *Admins* and give that user group the types of permissions that administrators typically need
- Any user in that user group automatically has the permissions that are assigned to the user group

Use-case: If a new user joins your organization and needs administrator privileges, you can assign the appropriate permissions by adding the user to that user group

Characteristics of IAM User Groups

1) A user group can contain many users, and a user can belong to multiple user groups

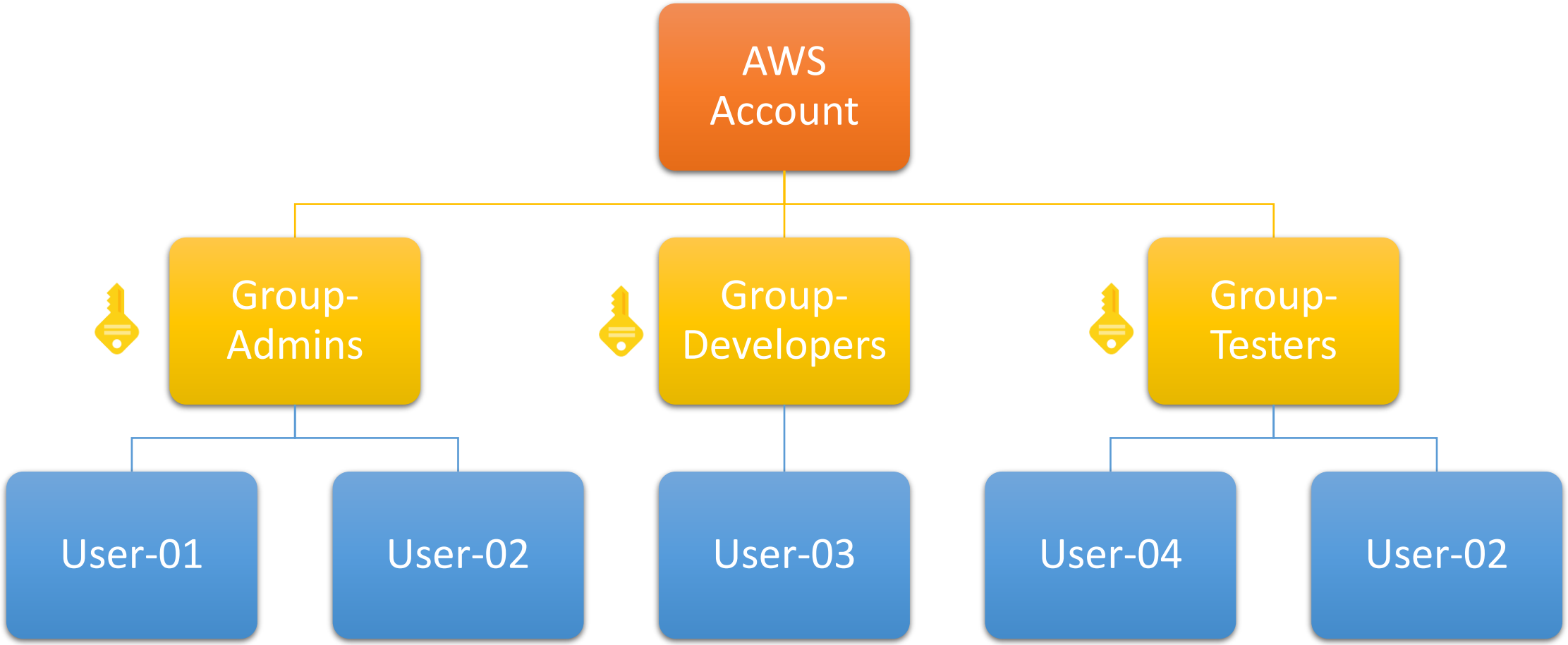
2) User groups can't be nested; they can contain only users, not other user groups

3) There is no default user group that automatically includes all users in the AWS account

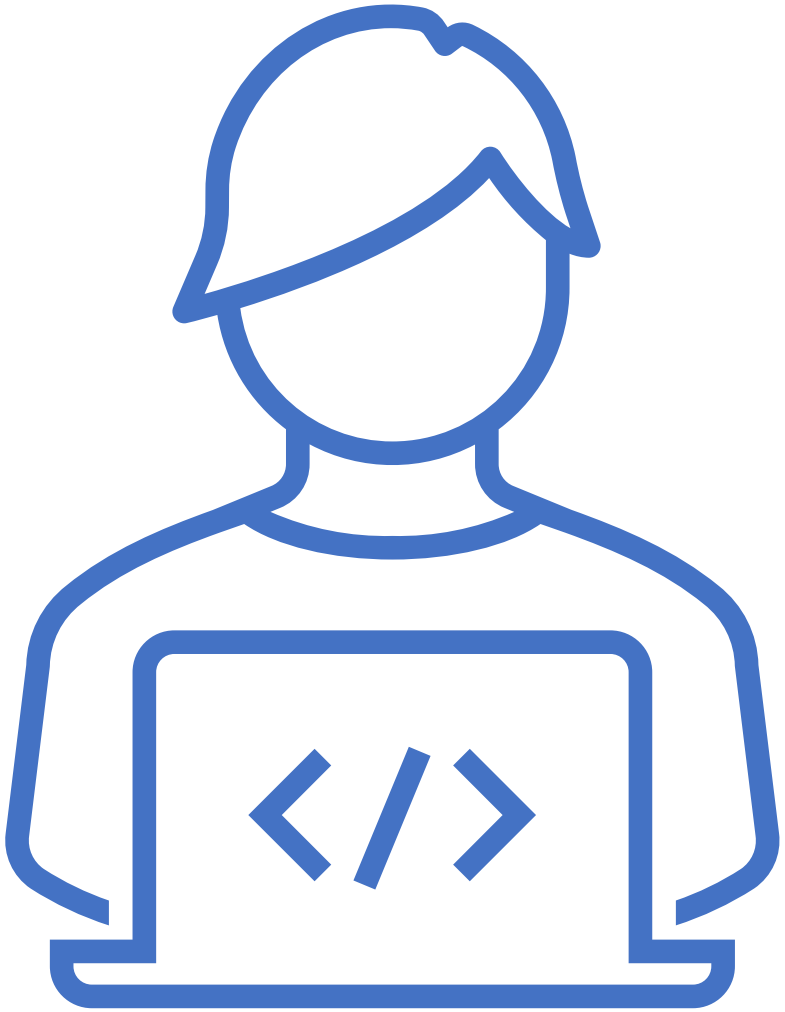
4) The number and size of IAM resources in an AWS account are limited

Ref: https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_iam-quotas.html for all the IAM quota limits

Sample IAM User Groups setup



Hands-on Labs



Lab: Creating an AWS IAM Group from AWS console

Lab: Creating an AWS IAM Group using AWS CLI

Command: aws iam create-group - -group-name <groupname>

Ref: https://docs.aws.amazon.com/cli/latest/reference/iam/create-group.html for other available attributes.

AWS IAM Policy

AWS IAM Policy

1) Users or Groups can be assigned JSON documents called **Policies**
2) These Policies define the permissions of the Users, Groups and Roles when associated with it
3) In AWS you must follow the ***least privilege principle***: don't give more permissions than a user needs
4) You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles)

AWS IAM Policy – Examples

1) Allow Read S3 Objects

2) Allow Delete EC2 Instances

AWS IAM Policies Types

- Customer Managed Policy (Custom)
- AWS Managed Policies (Read Only)

AWS IAM Policy – Types

IAM Policies are basically categorized into following three types:

1) Identity-based policies (User, Group, Role)
2) Resource-based policies (S3, SQS)
3) Organizations SCPs (Service Control Policies)

AWS IAM Policy – Identity Based Policy

- Identity-based policies are JSON permissions policy documents that control what actions an identity (users, groups of users, and roles) can perform, on which resources, and under what conditions
- Identity-based policies can be further categorized:
  1. **Managed policies**: Standalone identity-based policies that you can attach to multiple users, groups, and roles in your AWS account
  2. **Inline policies**: Policies that you add directly to a single user, group, or role. Inline policies maintain a strict one-to-one relationship between a policy and an identity. They are deleted when you delete the identity

AWS IAM Policy – Resource Based Policy

- Resource-based policies are JSON policy documents that you attach to a resource such as an Amazon S3 bucket
- These policies grant the specified principal permission to perform specific actions on that resource and defines under what conditions this applies
- Resource-based policies are inline policies
- There are no managed resource-based policies

IAM Policy and Structure

```
{
    "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": "*"
        }
    ]
}
```

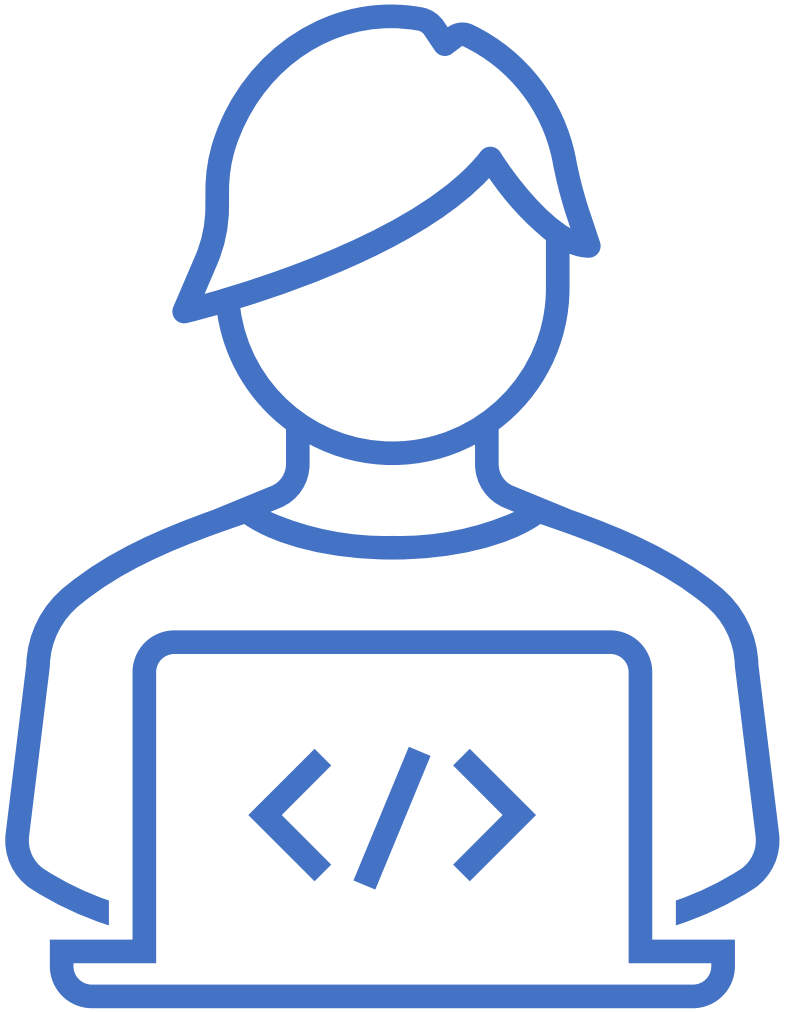
IAM Policy Structure – Statement Elements

1. **Sid:** Name of the Statement (Identifier and must be unique)
2. **Effect:** Allowed values are “Allow” or “Deny”
3. **Principal:** WHO? Example: arn:aws:iam:::3542342333:user/Dave. Used with Resource based policy and attached to the AWS Resources and not to the IAM Users/Groups
4. **Action:** Required “Verb”. Action related to AWS Resources. Eg. Service:Service_Action
5. **Resource:** Defines the list of resources on which Effect and Action is applied
6. **Condition:** Any specific condition in which you would like policy to be applied.

The difference between explicit and implicit denies

- A request results in an explicit deny if an applicable policy includes a Deny statement
- If policies that apply to a request include an Allow statement and a Deny statement, the Deny statement trumps the Allow statement. The request is explicitly denied
- An **implicit denial** occurs when there is no applicable Deny statement but also no applicable Allow statement
- IAM user, role, or federated user is denied access by default, they must be explicitly allowed to perform an action. Otherwise, they are implicitly denied access

Hands-on Labs



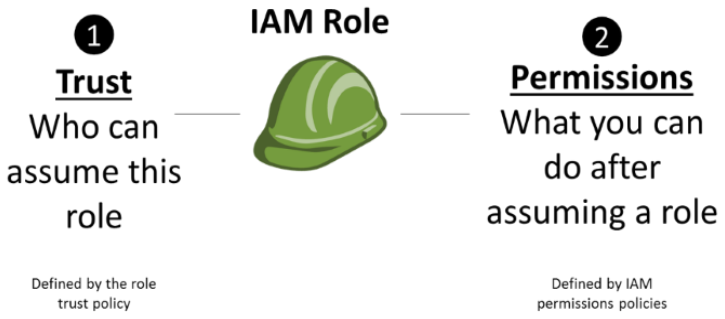
IAM Reference Links

https://docs.aws.amazon.com/service-authorization/latest/reference/reference_policies_actions-resources-contextkeys.html

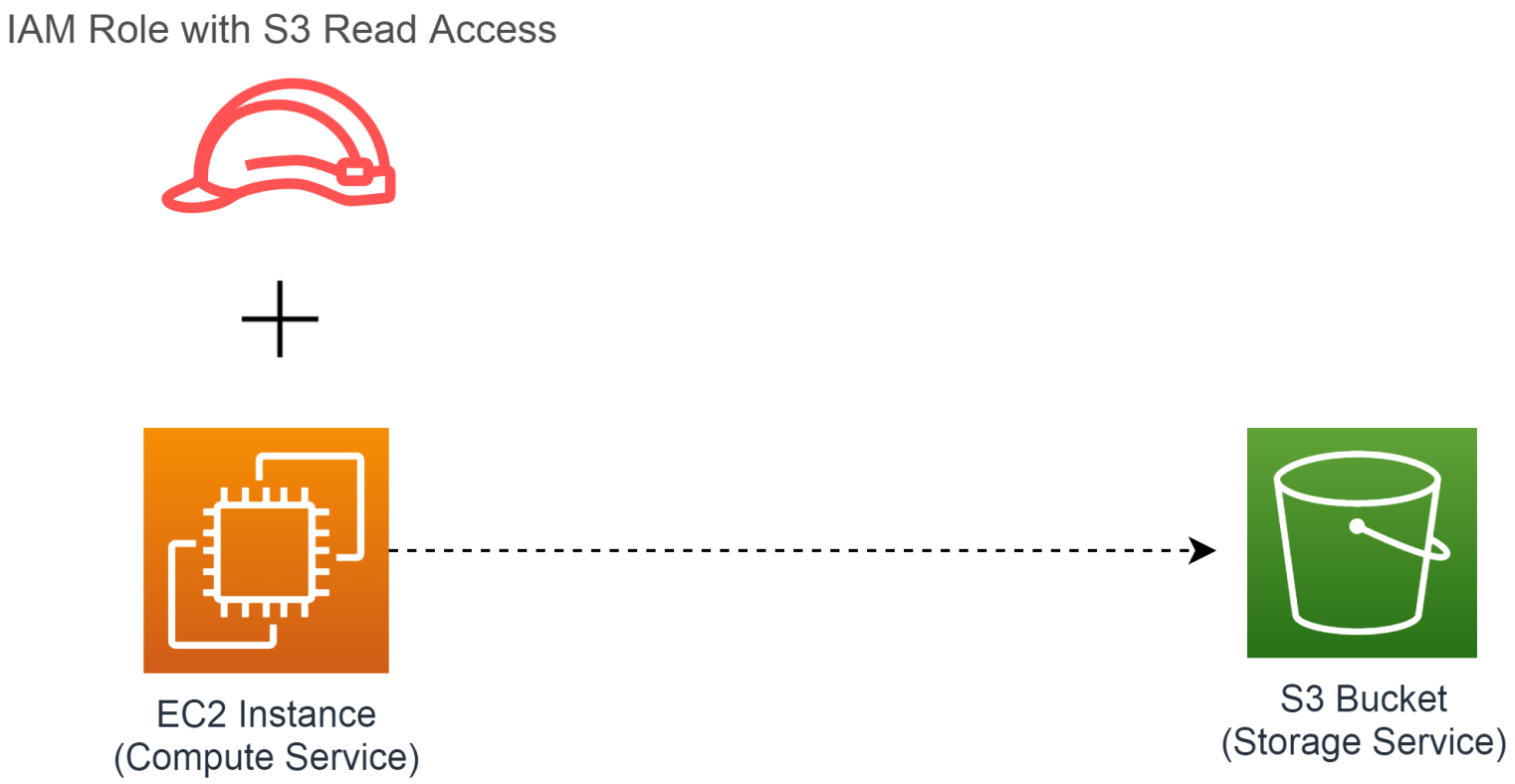
AWS IAM Role

AWS IAM Role

- An ***IAM Role*** is an IAM identity that you can create in your account that has specific permissions
- IAM Role allows AWS services to perform actions on your behalf
- A Role is intended to be assumable by anyone who needs it
- A Role does not have standard long-term credentials such as a password or access keys associated with it. Instead, when you assume a role, it provides you with temporary security credentials for your role session



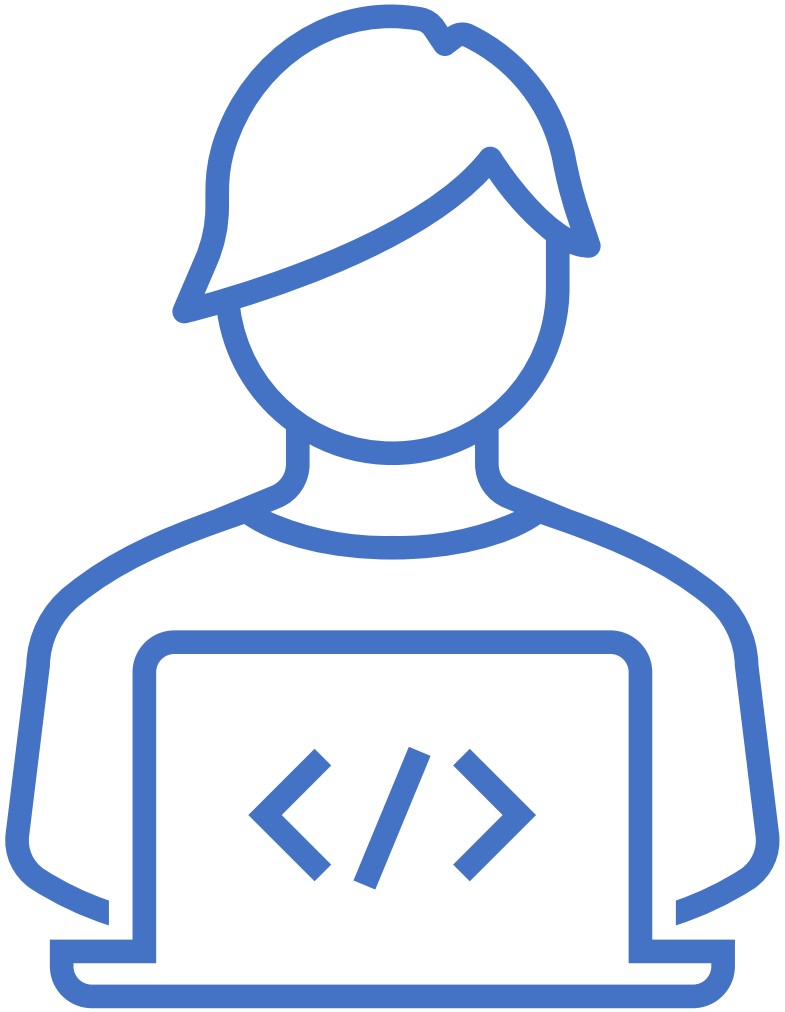
AWS IAM Role – Scenario



Creating IAM Role

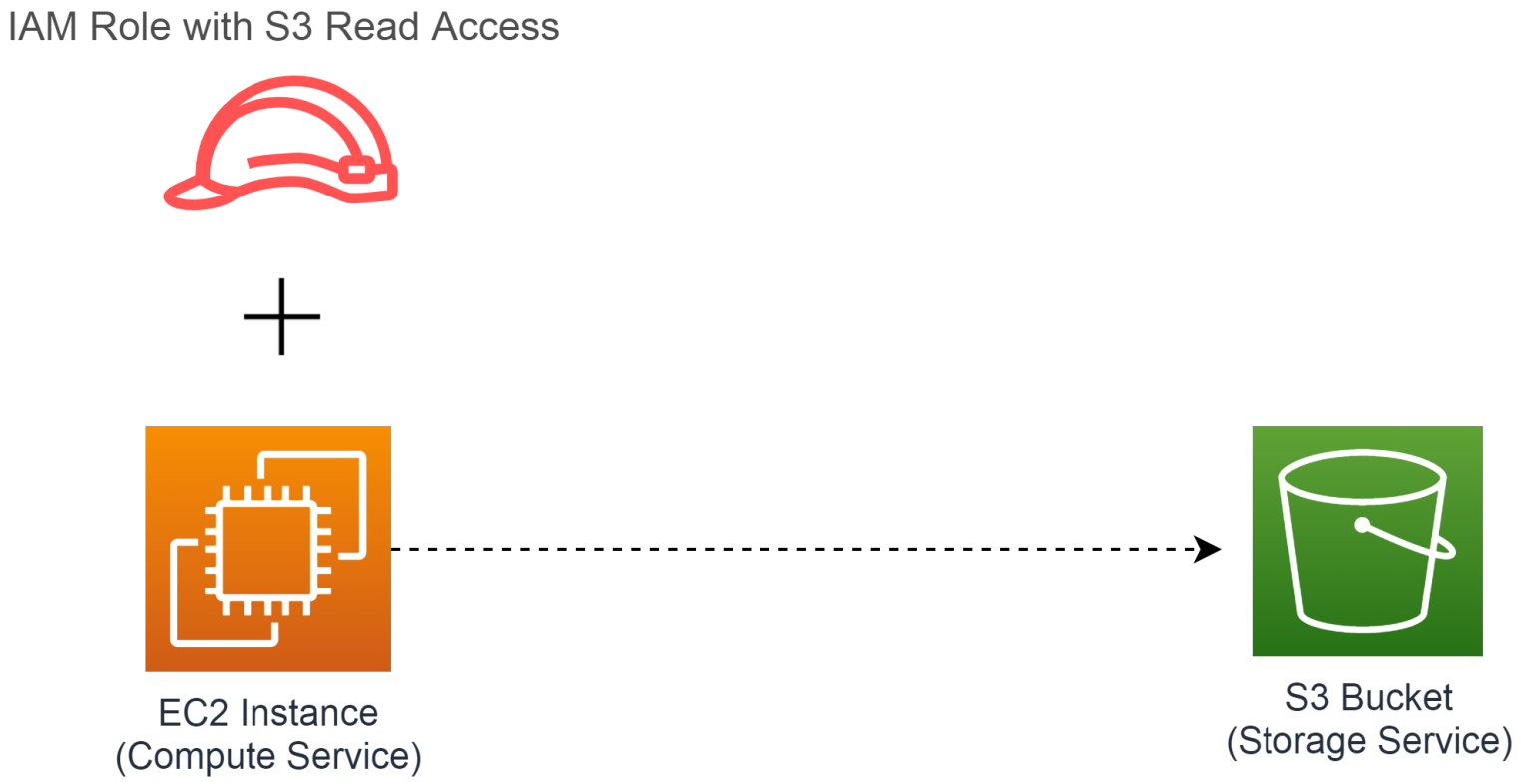
- If you use the AWS Management Console, a wizard guides you through the steps for creating a role
- The wizard has slightly different steps depending on whether you're creating a role for an **AWS service**, for an **AWS account**, or for a **Federated user**

Hands-on Labs



Lab: Creating an IAM Role to delegate permission to an AWS service

Lab: Creating a role to delegate permissions to an AWS service



IAM Multi-factor Authentication (MFA)

IAM MFA

- Users have access to your AWS Account and can possibly change the configurations or delete the resources
- If you want to add an additional layer of security to your AWS Account credentials (Username and Password)
- Main benefit of MFA is that if your password is stolen or hacked, the account is not compromised

Username and Password

+

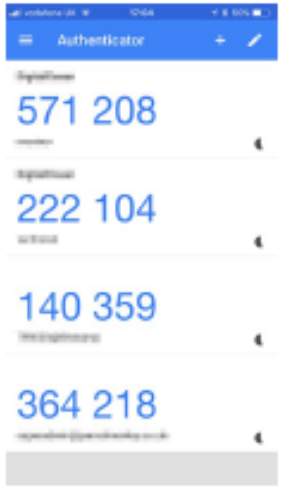
MFA

→

Successful Login

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

How can user access AWS?

There are basically three ways to access AWS:

1) AWS Management Console – Protected by password and MFA
2) AWS Command Line Interface (CLI) – Protected by Access keys
3) AWS Software Development Kit (SDK) – Protected by Access keys

Access keys are generated from the AWS Management console

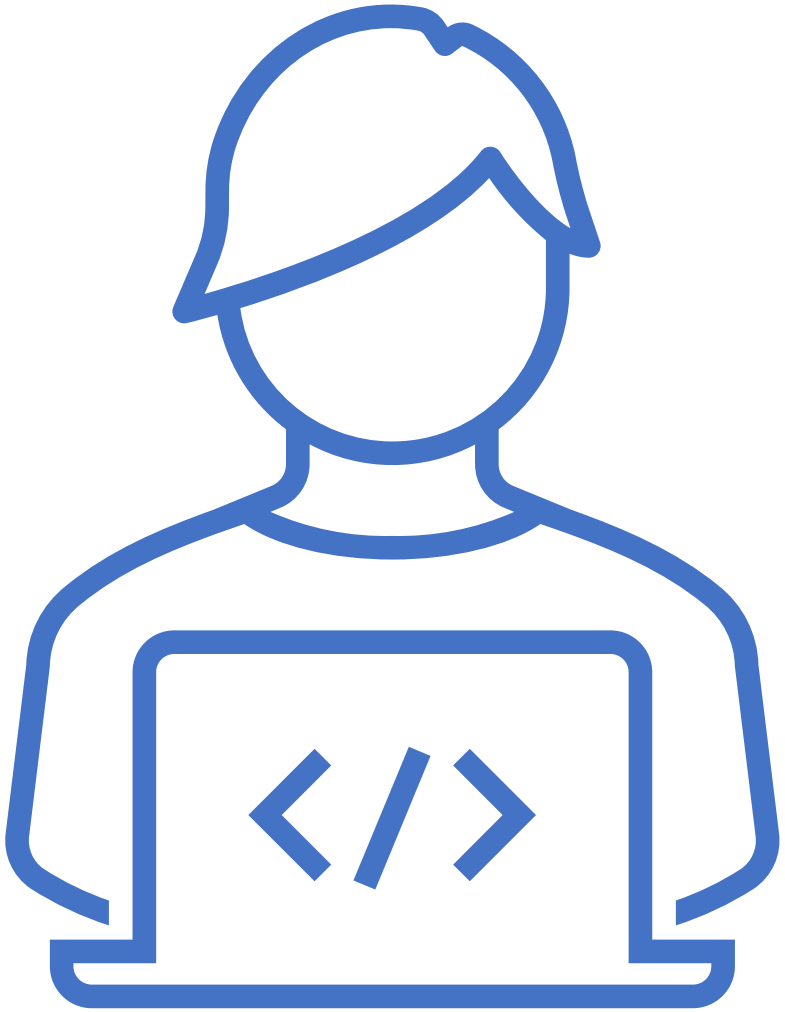
Note: Access keys are secret, just like password. Do not share them with anyone

Virtual MFA Applications

- Applications for your smartphone can be installed from the application store that is specific to your phone type
- The following table lists some applications for different smartphone types:

Android	Authy, Duo Mobile, LastPass Authenticator, Microsoft Authenticator, Google Authenticator
iPhone	Authy, Duo Mobile, LastPass Authenticator, Microsoft Authenticator, Google Authenticator

Hands-on Labs

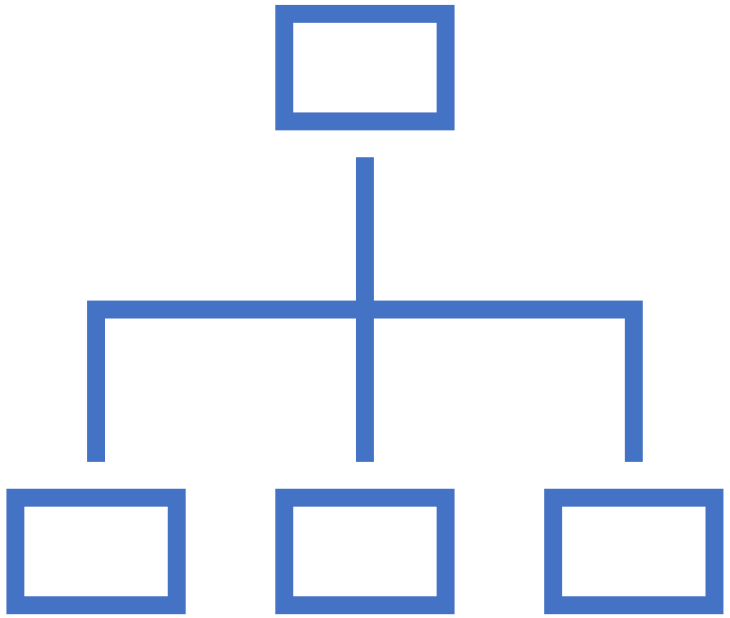


Lab: Enabling MFA of IAM user from AWS Console

Username and Password

+

Virtual MFA

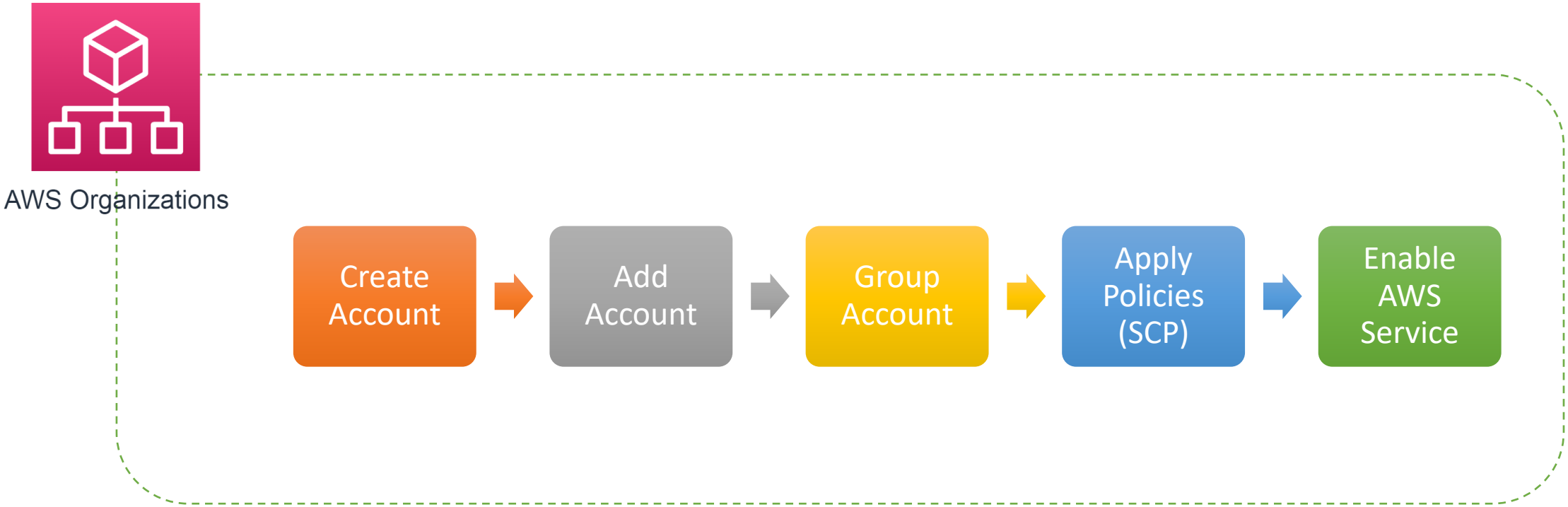


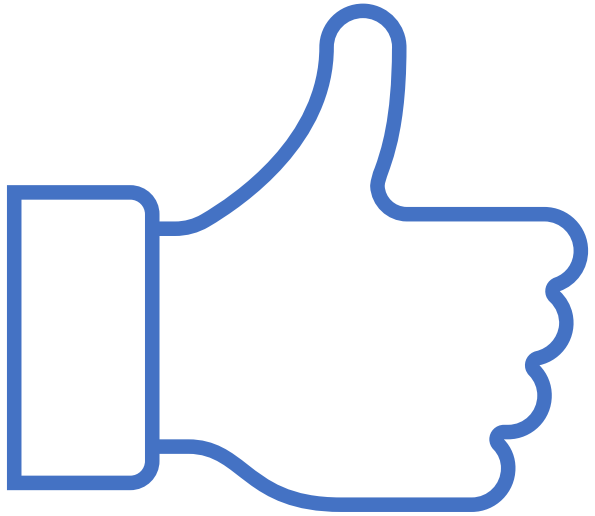
AWS Organizations

Overview of AWS Organizations

- ✓ AWS Organizations helps you centrally manage and govern your environment as you grow and scale your AWS resources
- ✓ You can programmatically create new AWS accounts and allocate resources, group accounts to organize your workflows, apply policies to accounts or groups for governance
- ✓ Simplify billing by using a single payment method for all of your accounts

Overview of AWS Organizations





IAM Best Practices

IAM Best Practices

- ✓ Do not use the root account except for AWS account setup
- ✓ One physical user EQUALS One AWS IAM 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)
- ✓ Never share IAM users Access Keys with multiple users

IAM – 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 AWS services

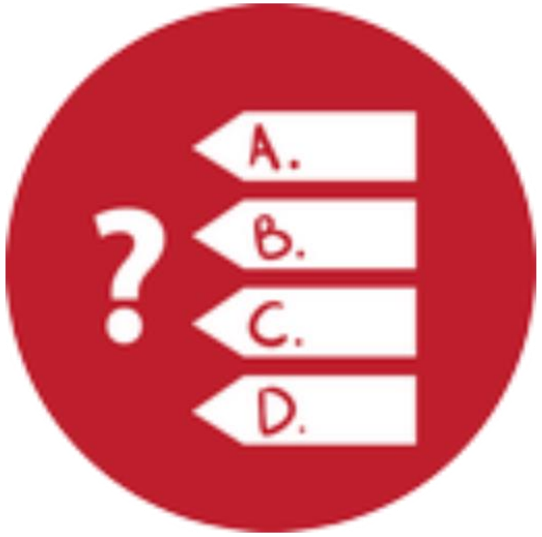
Security: MFA + Password Policy

Access Keys: access AWS using the CLI or SDK

Audit: IAM Credential Reports & IAM Access Advisor

Quiz and Exercises

Exercise



Quiz