Class 2 – AWS Certified Solutions Architect – Associate

Identity and Access Management

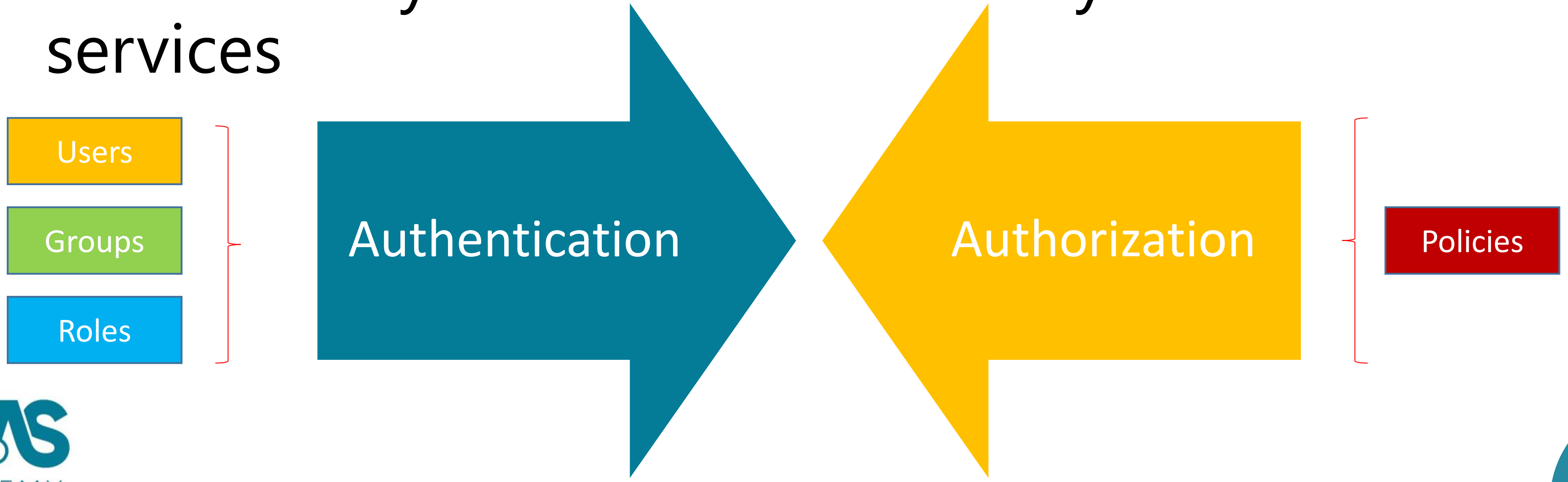
Amazon Web Services

What is IAM

You can use AWS IAM to securely control individual and group access to your AWS resources. You can create and manage user identities ("IAM users") and grant permissions for those IAM users to access your resources. You can also grant permissions for users outside of AWS (federated users).

What is IAM

You can use identity concepts such as users, groups, roles and access policies to control who and what can use your AWS account, what services they can use and how they can use those services



Users

Groups

Roles

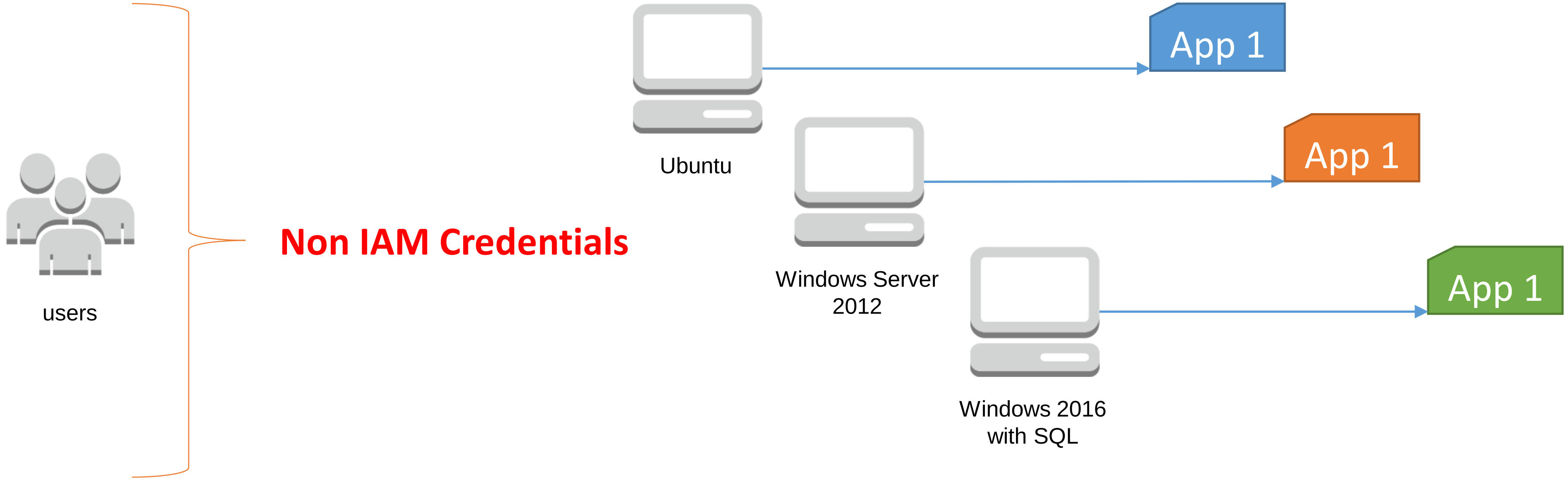
Authentication

Authorization

Policies

What IAM is NOT

IAM is not and identity management system for your applications or operating systems. IAM is used to assign permissions and grant access to services on the AWS Infrastructure.



Accessing IAM

- AWS Management Console – Browser based management system
- AWS Command Line – Fastest way to work with IAM
  - AWS Command Line Interface (AWS CLI)
  - AWS Tools for PowerShell
- AWS SDKs - create programmatic access to IAM and AWS
- IAM HTTPS API

IAM Features

- Shared access to your AWS account
  - Cross Account Access
    - Test/Dev Account, UAT and Production Accounts
    - Business Partners who need access to your AWS Account
- Granular permissions
- Identity federation (Microsoft Active Directory, Facebook, Google)
- Multifactor Authentication
- Secure access to AWS resources for applications that run on Amazon EC2
- AWS IAM Password Policies enables you to confirm password complexity rules
- Integration with other AWS Services
- Free to use

IAM Identities - Principals

Root User

- Don’t use the Root Account for day to day operations. Create an IAM Administrator instead if you need full administrative access.

IAM User

- Users enable you to enforce the concept of Least Privileges
- Note when you setup a IAM users, they need to login to a special AWS URL that is designated for your account. Format is https://accountID.signin.aws.amazon.com/console - This URL is customizable
- Always use the ‘Principal of Least Privileges’

IAM Identities - Groups

You can use groups to specify permissions for a collection of users, which can make those permissions easier to manage for those users.

- Groups are used to collect users that share common tasks.
- Groups allow permission inheritance
- Groups allow best practices to assign permissions

IAM Authentication

Three ways to authenticate a principal:

- User Name / Password – A principal such as a user can provide a username/password combination to verify their identity. You can also set password policies to enforce complexity
- Access Key – A combination of an access key ID and access secret key can be used to authenticate against AWS resources when making API calls
- Access Key/Session Token – when an access is requested under an assumed role such as an application assuming the role of an EC2 instance which has read/write access to an S3 bucket, the temporary token to authenticate includes a session token in addition to the access key

Access Keys

In addition to your user name and passwords (setup for root user or IAM users), you have Access Keys which is a key pair that comprises of an *access key id* and a *secret access key*.

You use Access Keys to programmatically access resources on the AWS platform. For example you can programmatically access an S3 bucket to read its content using APIs and here you would supply your Access Keys to authenticate yourself instead of a username and password.

Rotating Keys

There are security risks associated with using Access Keys, especially for programmatic access. Its always best practice to rotate these keys much the same way as you would change your password on a regular basis.

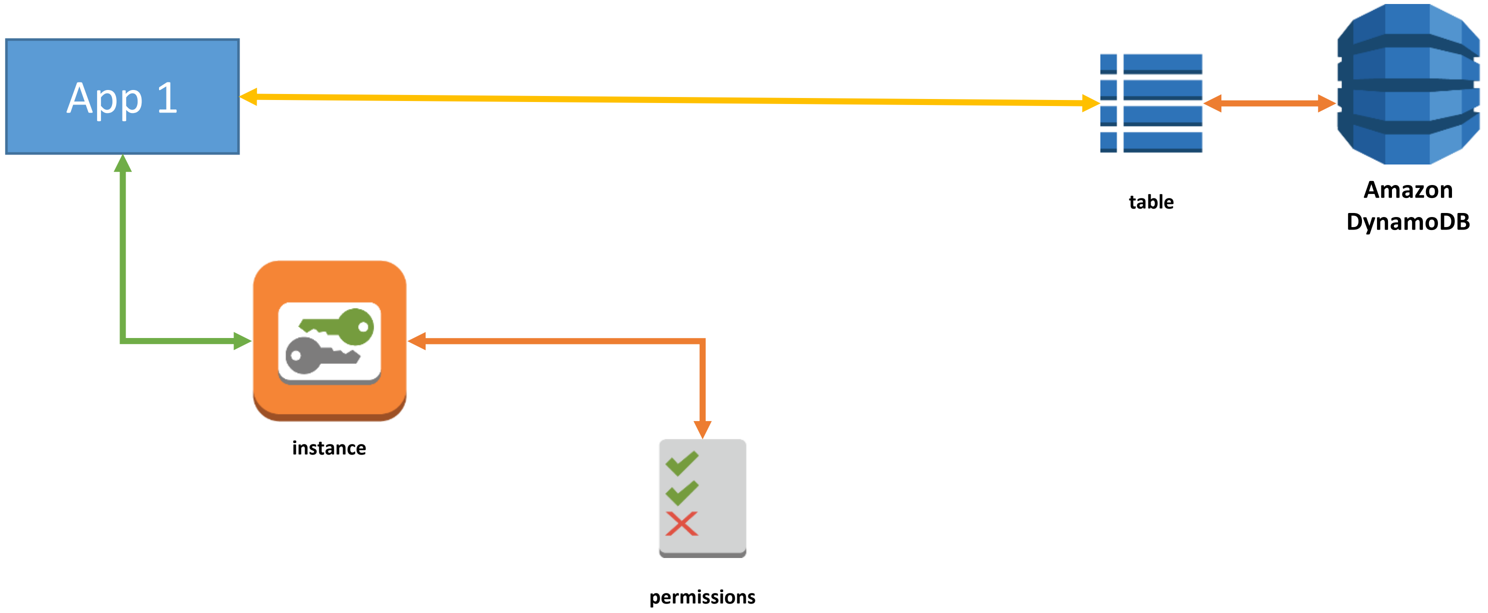
IAM Identities - Roles

Roles are used to grant specific privileges to AWS services and resources for a specified duration. Roles are assumed by authorized entities, such as IAM users, applications, or an AWS service such as EC2 to grant them the ability to make AWS Service Requests.

- E.G – An Application needs to update a DynamoDB Database Table with new customers who sign up for an account. Two ways to grant access:
  - Provide the application with an *access key ID* and *secret access key* by placing them in a Config file. Challenges Include
    - Security
    - Rotation of Keys when for example AWS deploys spot instances or when you use Auto Scaling Groups.
  - Assign an IAM Role (that has been granted necessary permissions) for the application to assume to enable it to access and update the database

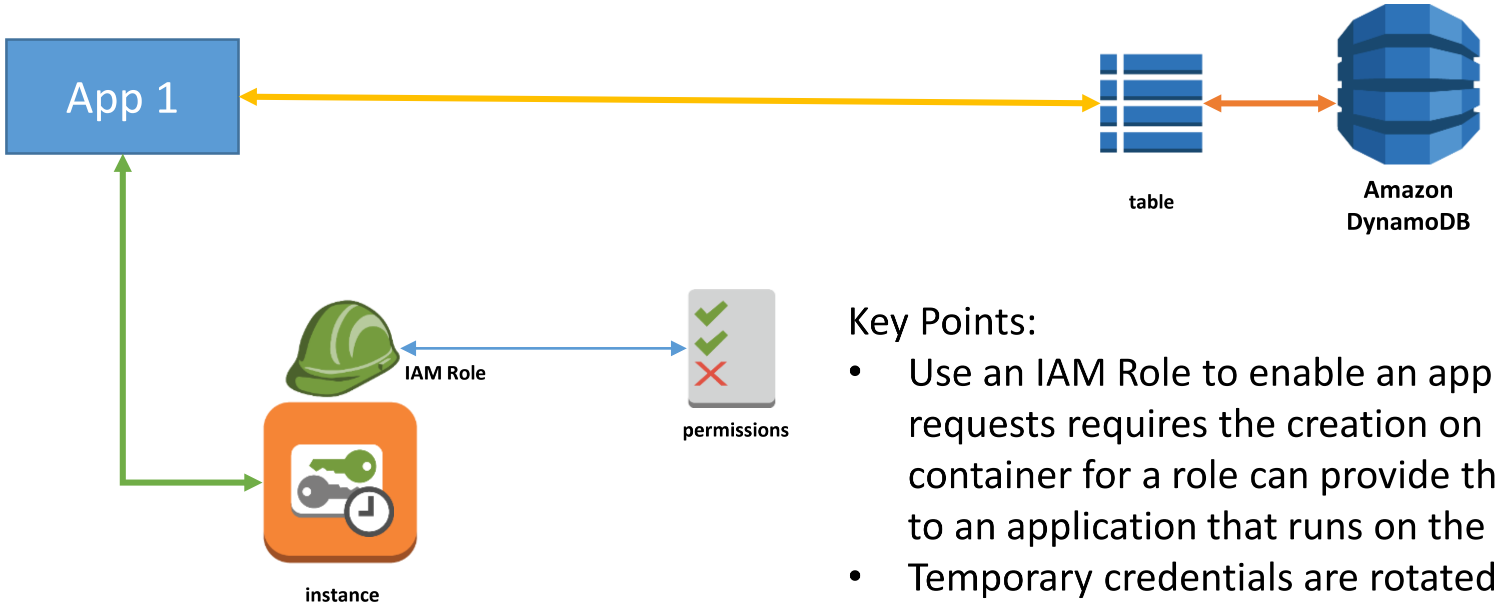
IAM Identities - Roles

Granting access via *access key ID* and *secret access key*



IAM Identities - Roles

Granting access IAM Roles



Key Points:

- Use an IAM Role to enable an application to make services requests requires the creation on an Instance Profile which is a container for a role can provide the role's temporary credentials to an application that runs on the instance.
- Temporary credentials are rotated automatically by AWS
- The application obtains temporary security credentials from Amazon EC2 instance metadata

IAM Roles

Use IAM Roles when you have the following scenarios:

- When you are creating an application that runs on an Amazon Elastic Compute Cloud (Amazon EC2) instance and that application makes requests to AWS.
- You need to grant cross-account access. Use an IAM role to establish trust between accounts, and then grant users in one account limited permissions to access the trusted account.
- When you are creating an app that runs on a mobile phone and that makes requests to AWS
- When you have users in a company are authenticated in your corporate network and they want to use their existing corporate Identity Services (e.g. Active Directory) to gain single sign on into the AWS account

IAM Roles

Key Points:

- IAM roles removes the need to store AWS Credentials in configuration files and is therefore more secure
- Temporary security credentials obtained through IAM roles and other features of the AWS Security Token Service expire after a short period of time. Use temporary security credentials to help reduce your risk in case credentials are accidentally exposed.
- The temporary security token has a lifetime of 15 mins to 36 hours
- You cannot attach multiple IAM roles to a single instance, but you can attach a single IAM role to multiple instances.
- You assign a role at the time of launching an EC2 Instance and now you can also attach/detach a role to a running instance

Multi Factor Authentication - MFA

Enables you to add an additional layer of security over and above your standard login credentials or authentication to APIs via Access Keys. You receive a 2nd challenge to your authentication requests once you have entered your primary login details. Using MFA, you must provide a One Time Password from a physical or virtual device.

MFA thus enables you to verify your identity with both something you know (your username and password) and with something you have (OTP PIN). For example when you log onto the console with an account configured for MFA, you are required to enter the code displayed on your MFA device to complete the 2 factor authentication process

Best Practice Tip – Ensure you configure MFA protection for the Root User Account

IAAS ACADEMY

Labs & Demos
IAM Initial Configuration

Amazon Web Services