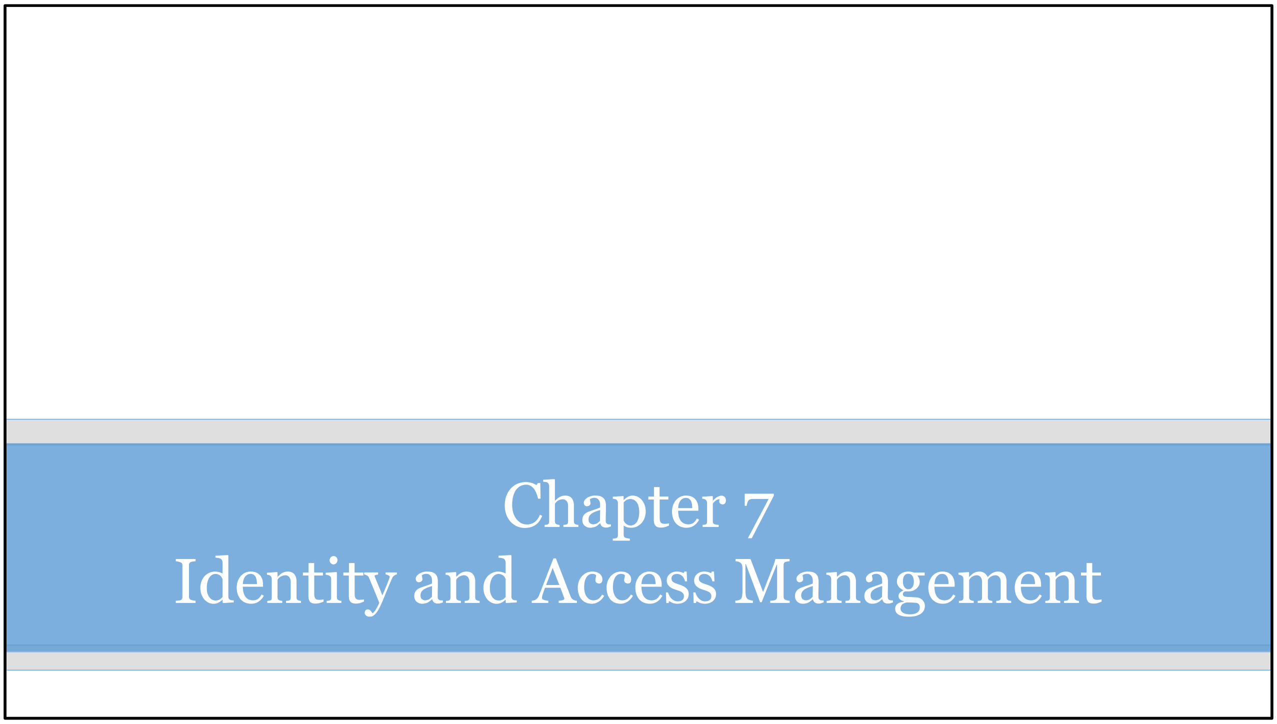
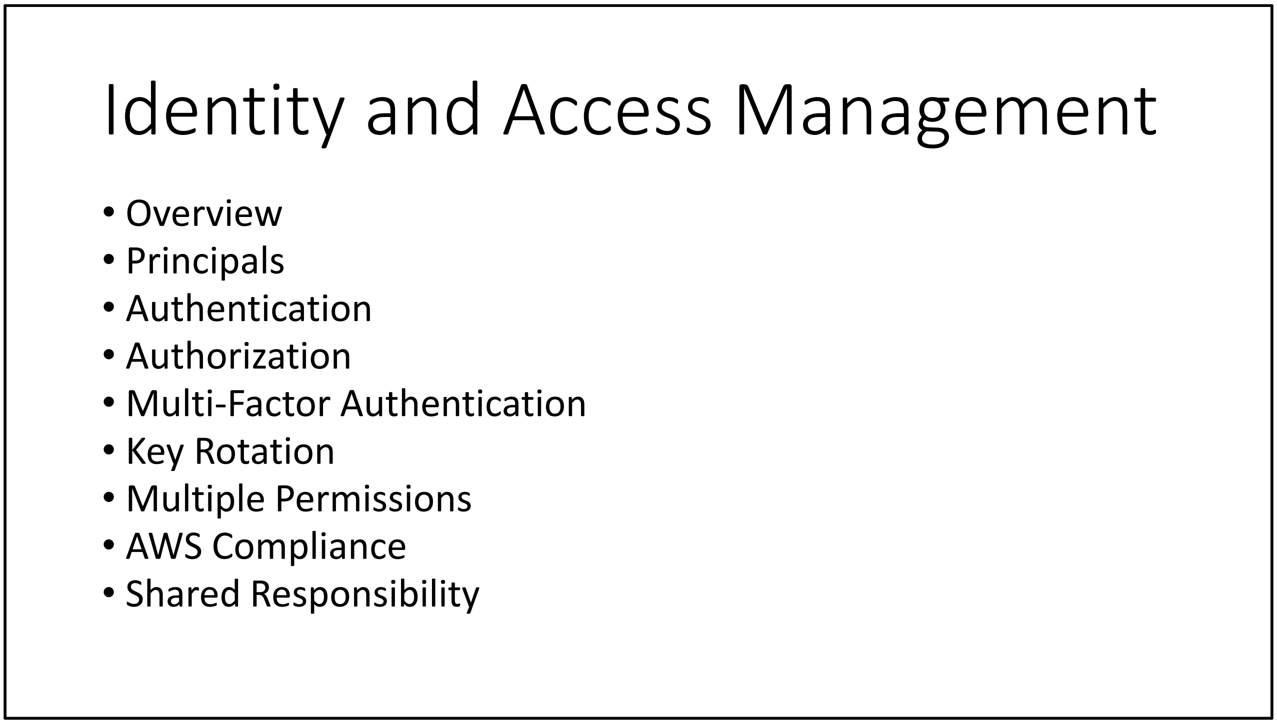


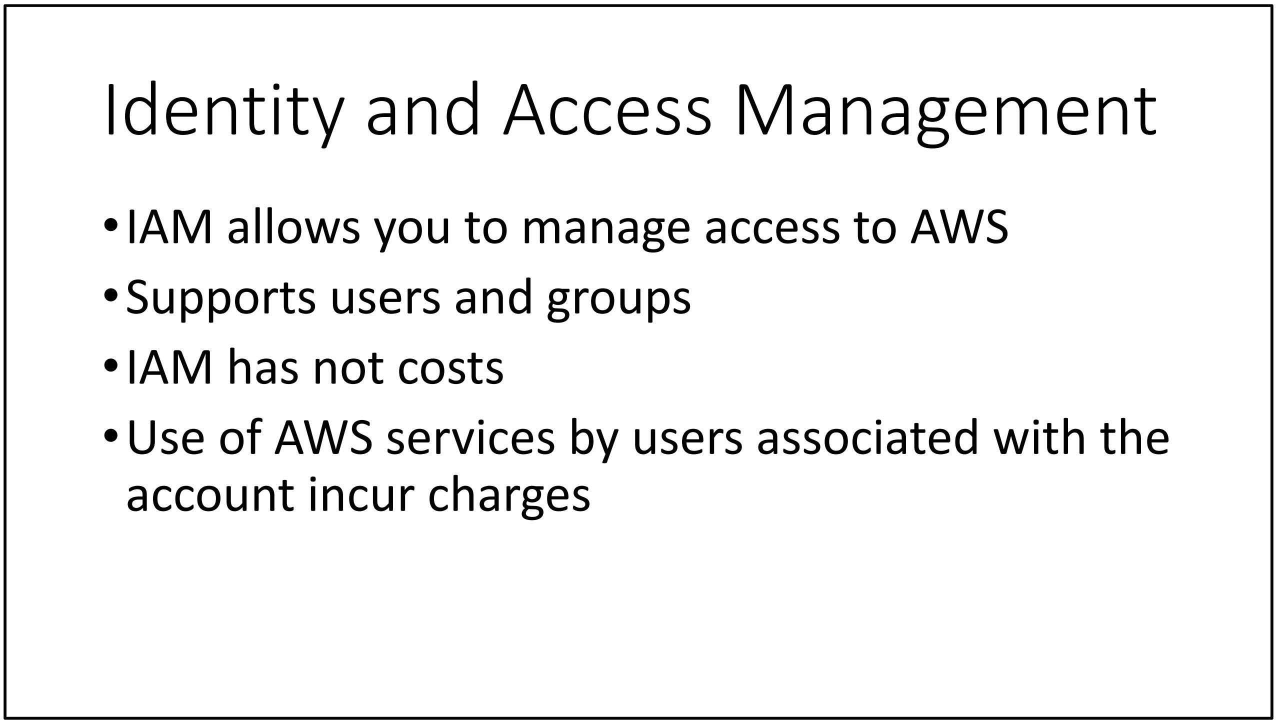
Chapter 7
Identity and Access Management

Episode 7.01
Identity and Access Management (IAM) Overview



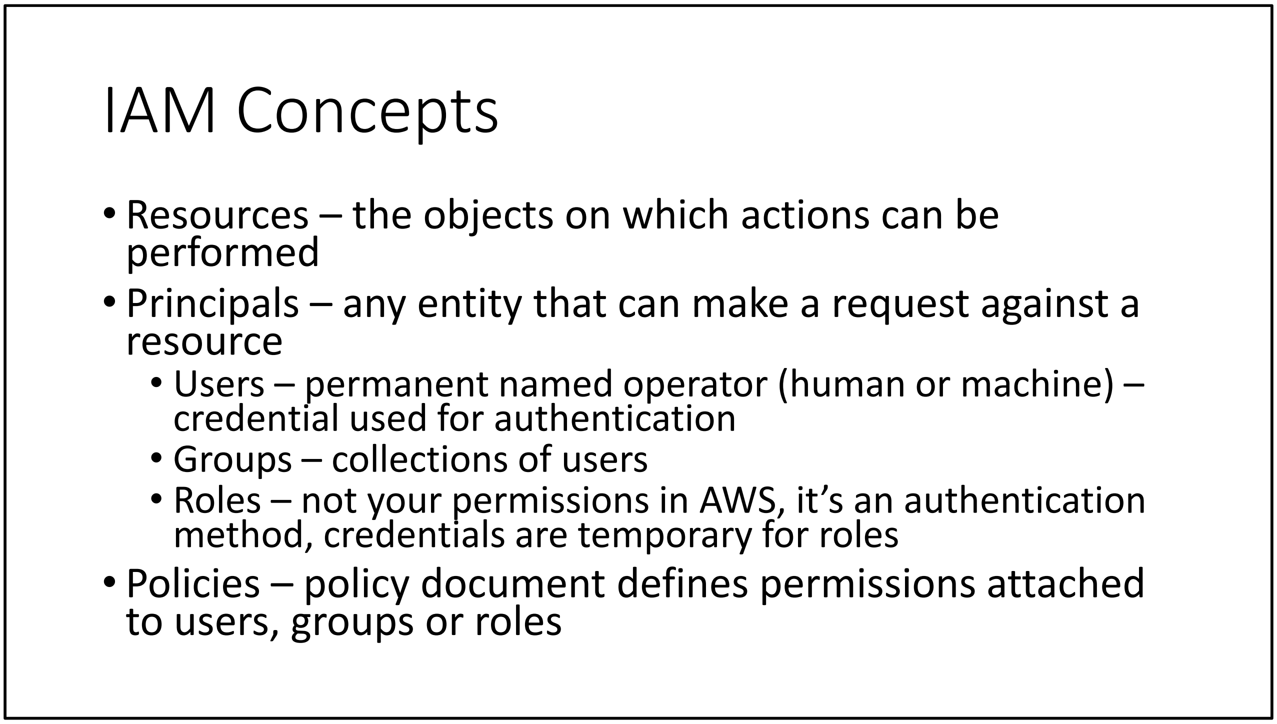
Identity and Access Management

- Overview
- Principals
- Authentication
- Authorization
- Multi-Factor Authentication
- Key Rotation
- Multiple Permissions
- AWS Compliance
- Shared Responsibility



Identity and Access Management

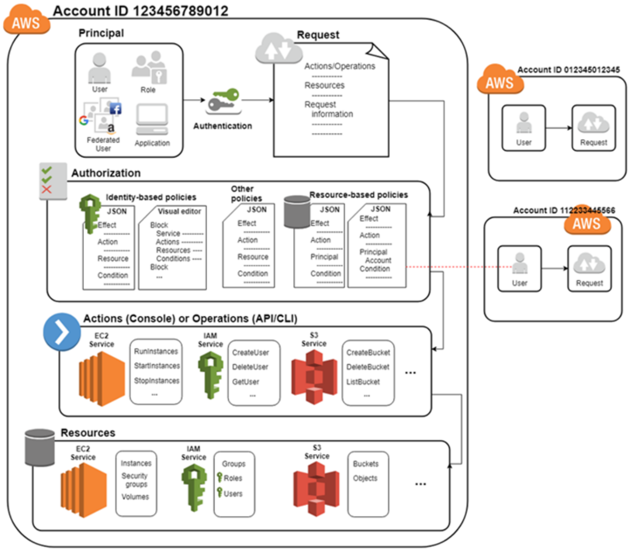
- IAM allows you to manage access to AWS
- Supports users and groups
- IAM has not costs
- Use of AWS services by users associated with the account incur charges

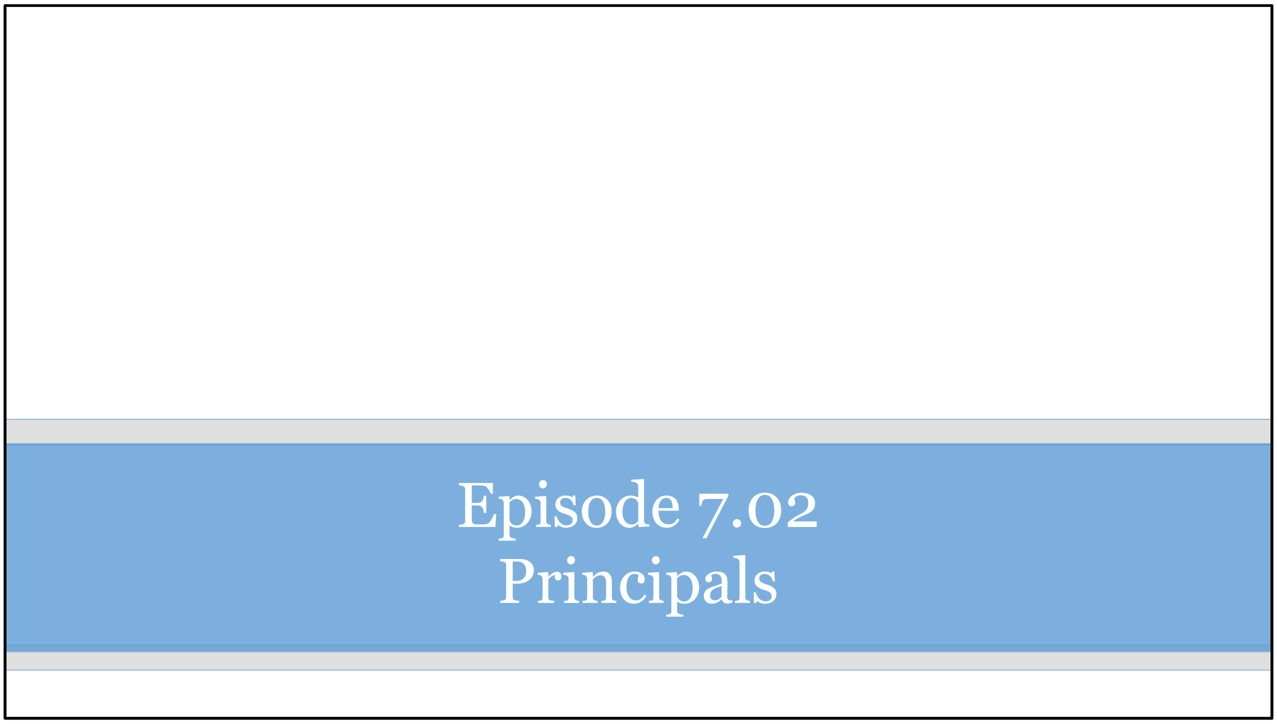


IAM Concepts

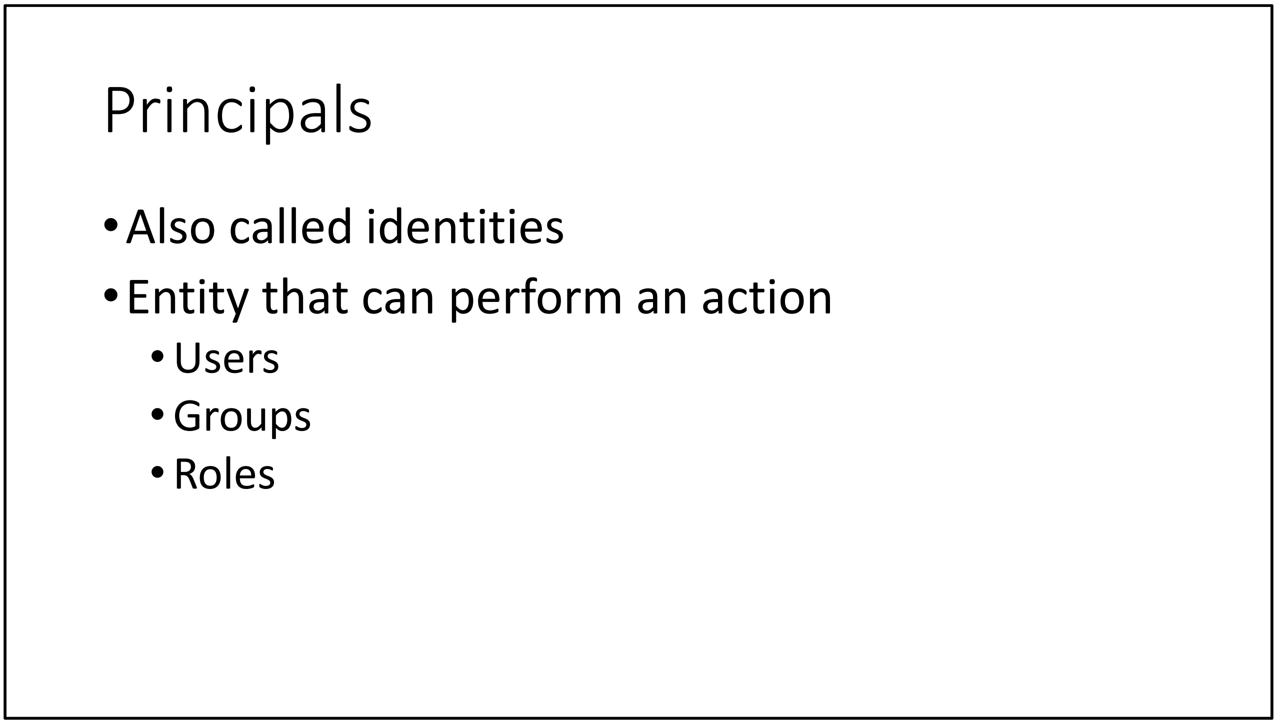
- Resources – the objects on which actions can be performed
- Principals – any entity that can make a request against a resource
  - Users – permanent named operator (human or machine) – credential used for authentication
  - Groups – collections of users
  - Roles – not your permissions in AWS, it’s an authentication method, credentials are temporary for roles
- Policies – policy document defines permissions attached to users, groups or roles

IAM Model


AWS
Account ID 123456789012
Principal
User
Role
Federated User
Application
Authentication
Request
Actions/Operations
Resources
Request information
Authorization
Identity-based policies
JSON
Effect
Action
Resource
Condition
Visual editor
Block
Service
Actions
Resources
Conditions
Block
...
Other policies
JSON
Effect
Action
Resource
Condition
Resource-based policies
JSON
Effect
Action
Principal
Condition
JSON
Effect
Action
Principal
Account
Condition
Actions (Console) or Operations (API/CLI)
EC2 Service
RunInstances
StartInstances
StopInstances
...
IAM Service
CreateUser
DeleteUser
GetUser
...
S3 Service
CreateBucket
DeleteBucket
ListBucket
...
...
Resources
EC2 Service
Instances
Security groups
Volumes
IAM Service
Groups
Roles
Users
S3 Service
Buckets
Objects
...
AWS
Account ID 012345012345
User
Request
Account ID 112233445566
AWS
User
Request

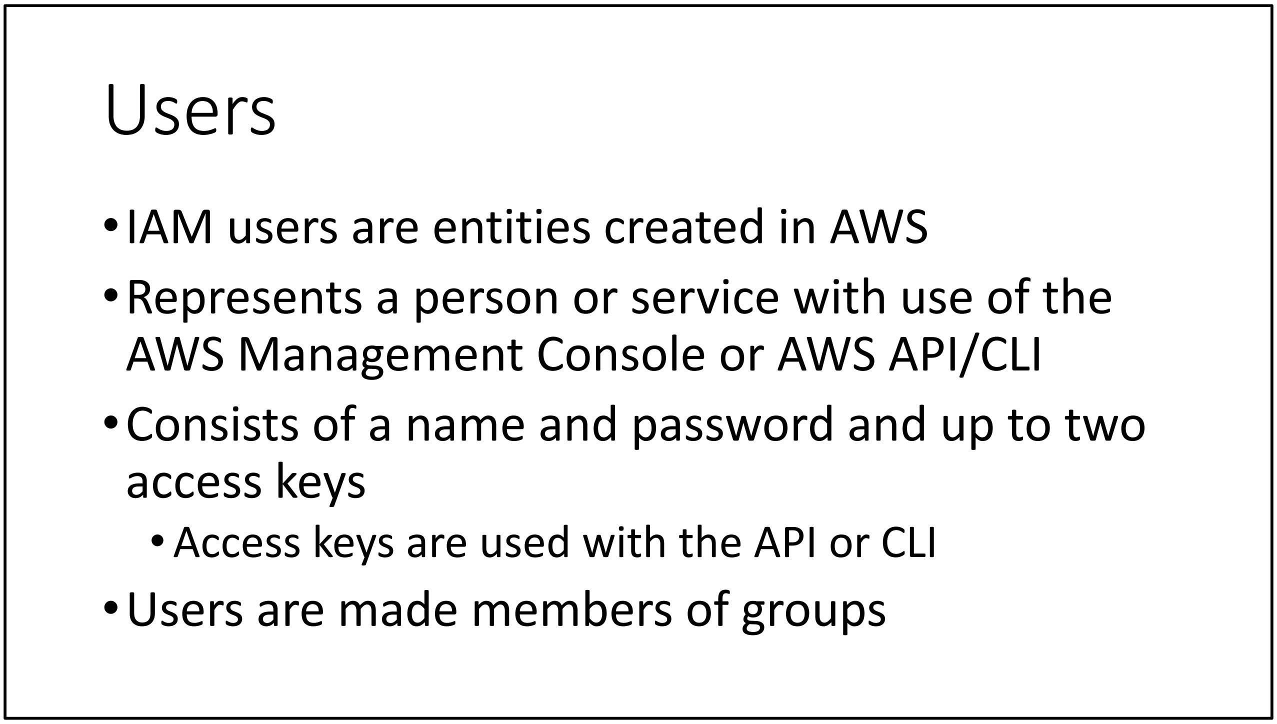


Episode 7.02
Principals



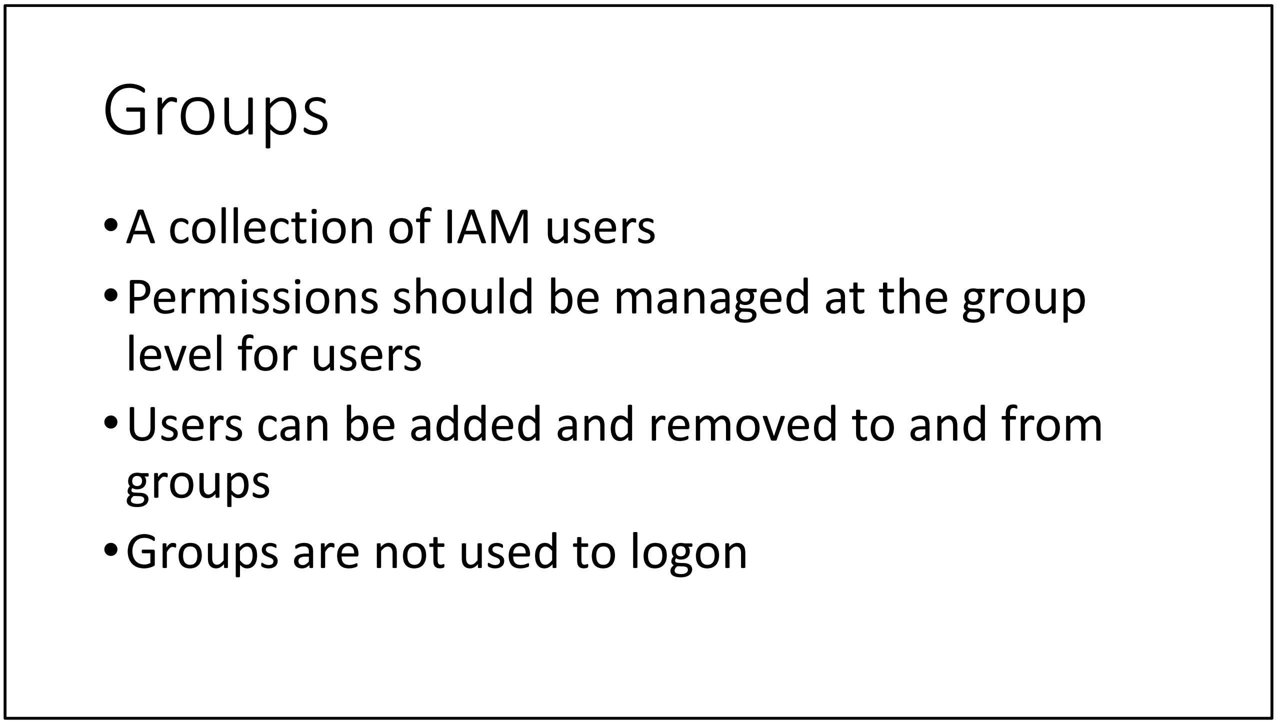
Principals

- Also called identities
- Entity that can perform an action
  - Users
  - Groups
  - Roles



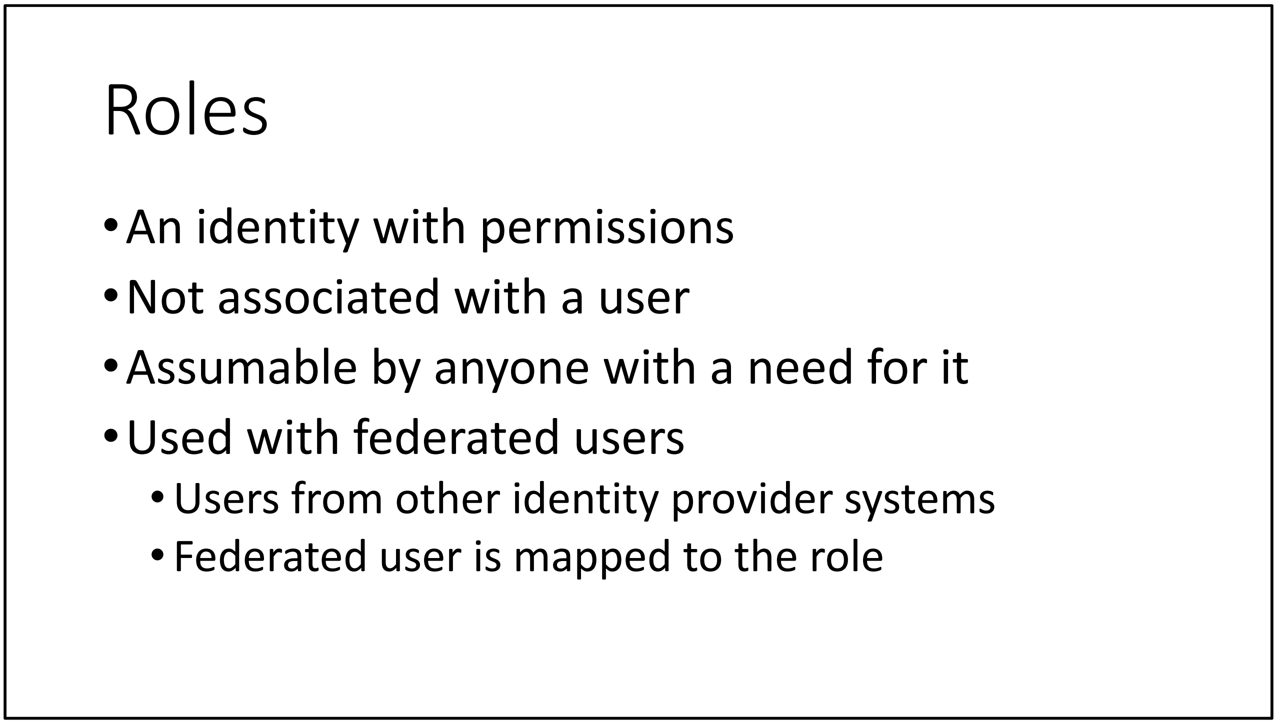
Users

- IAM users are entities created in AWS
- Represents a person or service with use of the AWS Management Console or AWS API/CLI
- Consists of a name and password and up to two access keys
  - Access keys are used with the API or CLI
- Users are made members of groups



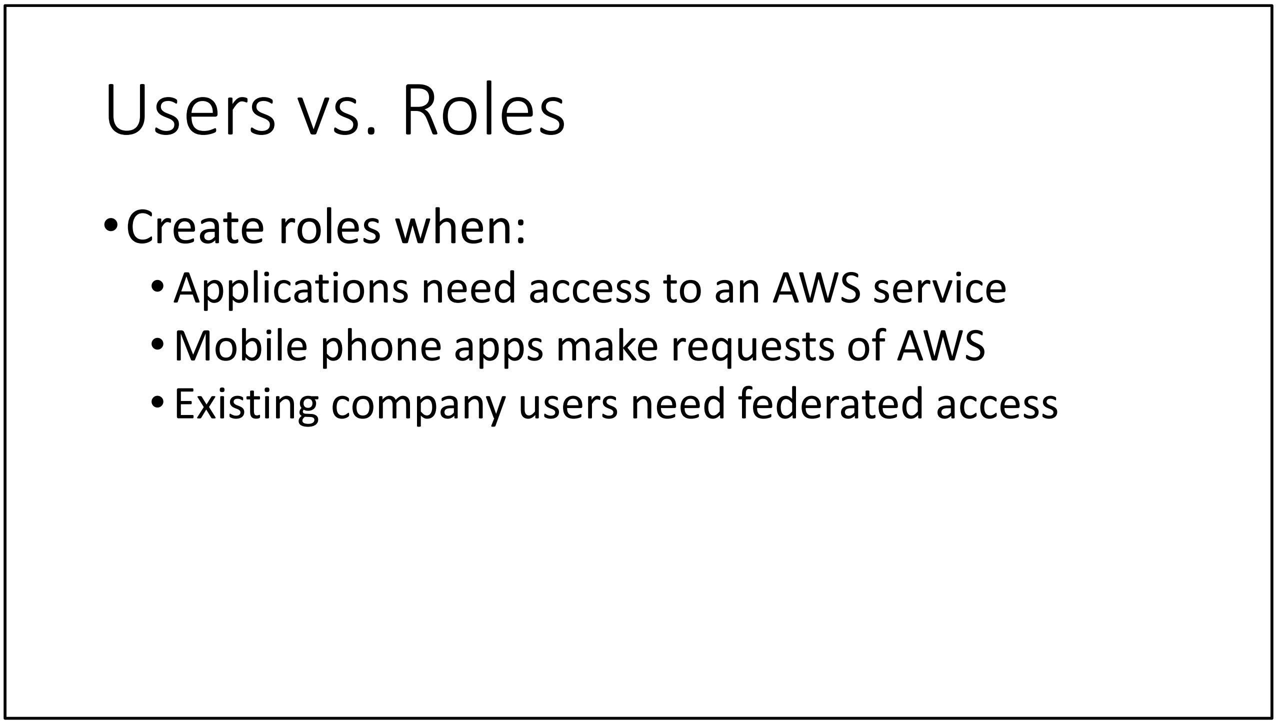
Groups

- A collection of IAM users
- Permissions should be managed at the group level for users
- Users can be added and removed to and from groups
- Groups are not used to logon



Roles

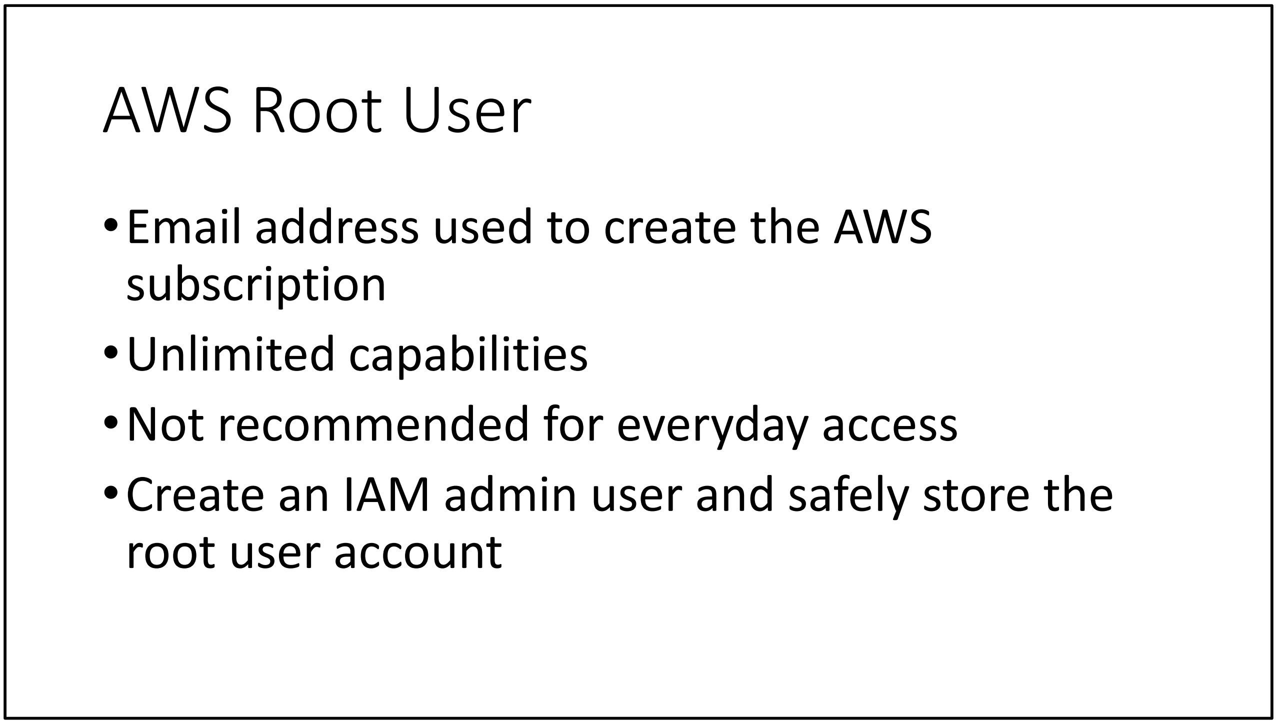
- An identity with permissions
- Not associated with a user
- Assumable by anyone with a need for it
- Used with federated users
  - Users from other identity provider systems
  - Federated user is mapped to the role



Users vs. Roles

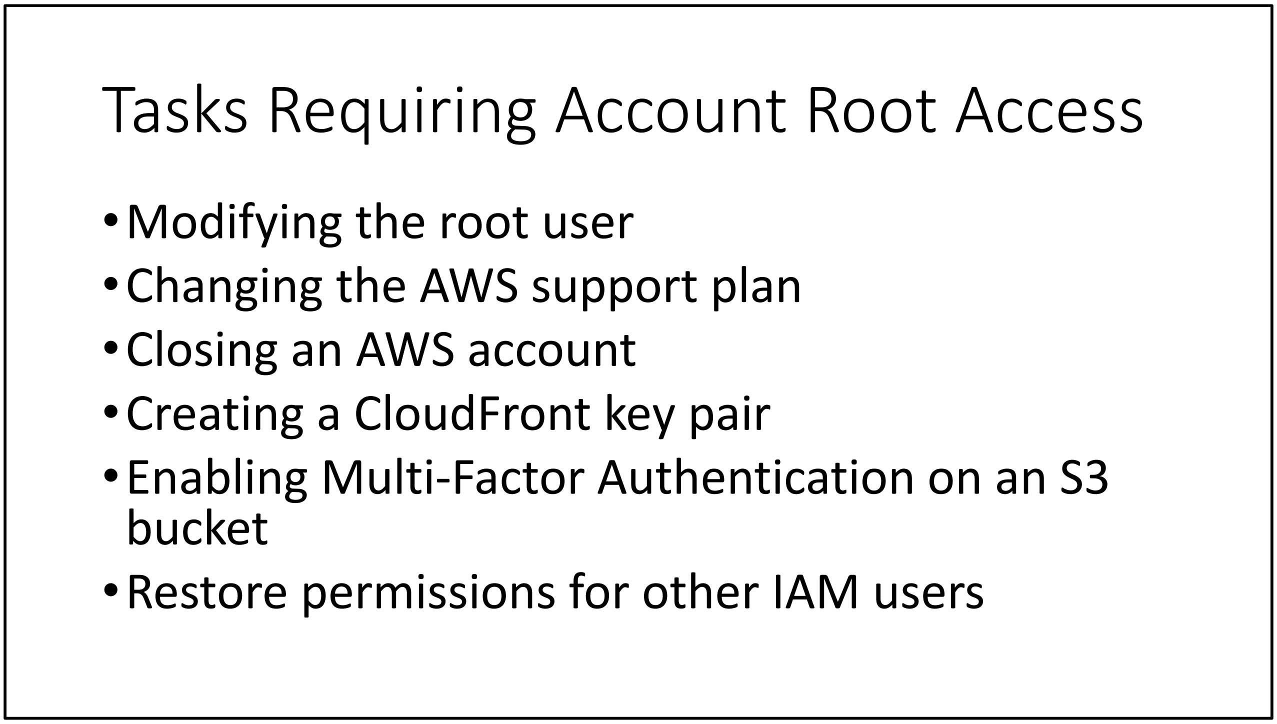
- Create roles when:
  - Applications need access to an AWS service
  - Mobile phone apps make requests of AWS
  - Existing company users need federated access

Episode 7.03
Root User



AWS Root User

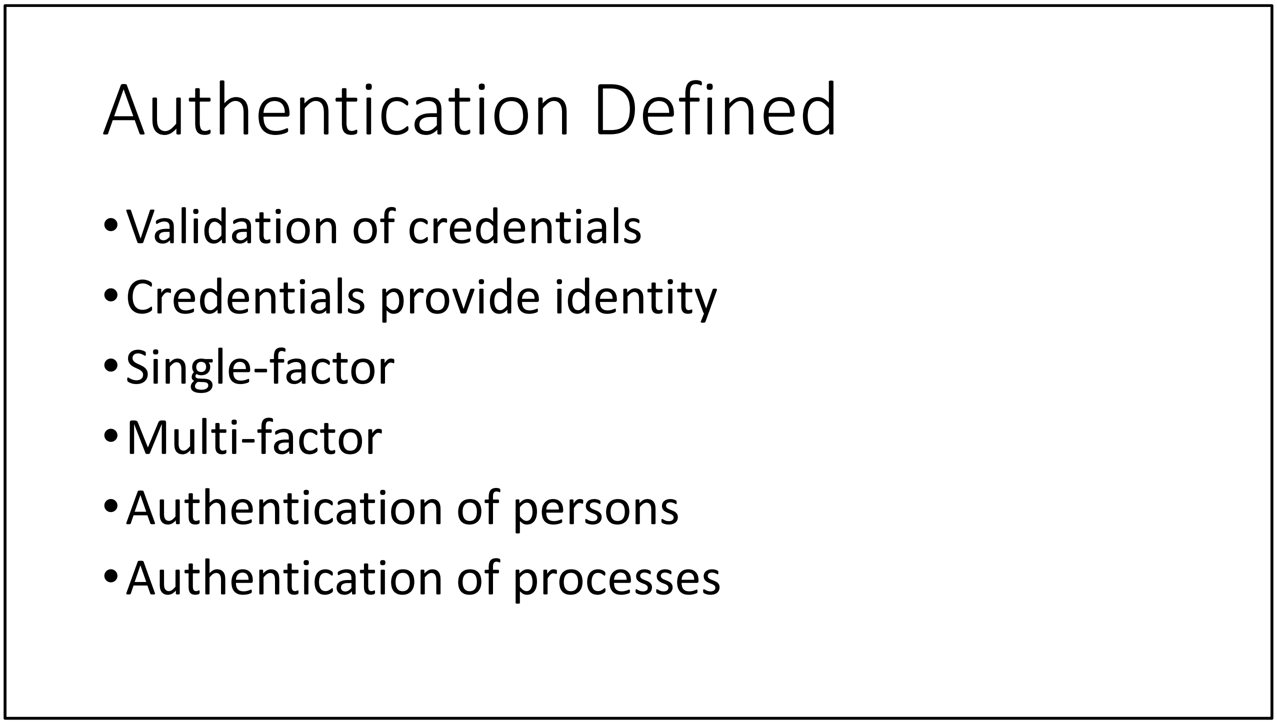
- Email address used to create the AWS subscription
- Unlimited capabilities
- Not recommended for everyday access
- Create an IAM admin user and safely store the root user account



Tasks Requiring Account Root Access

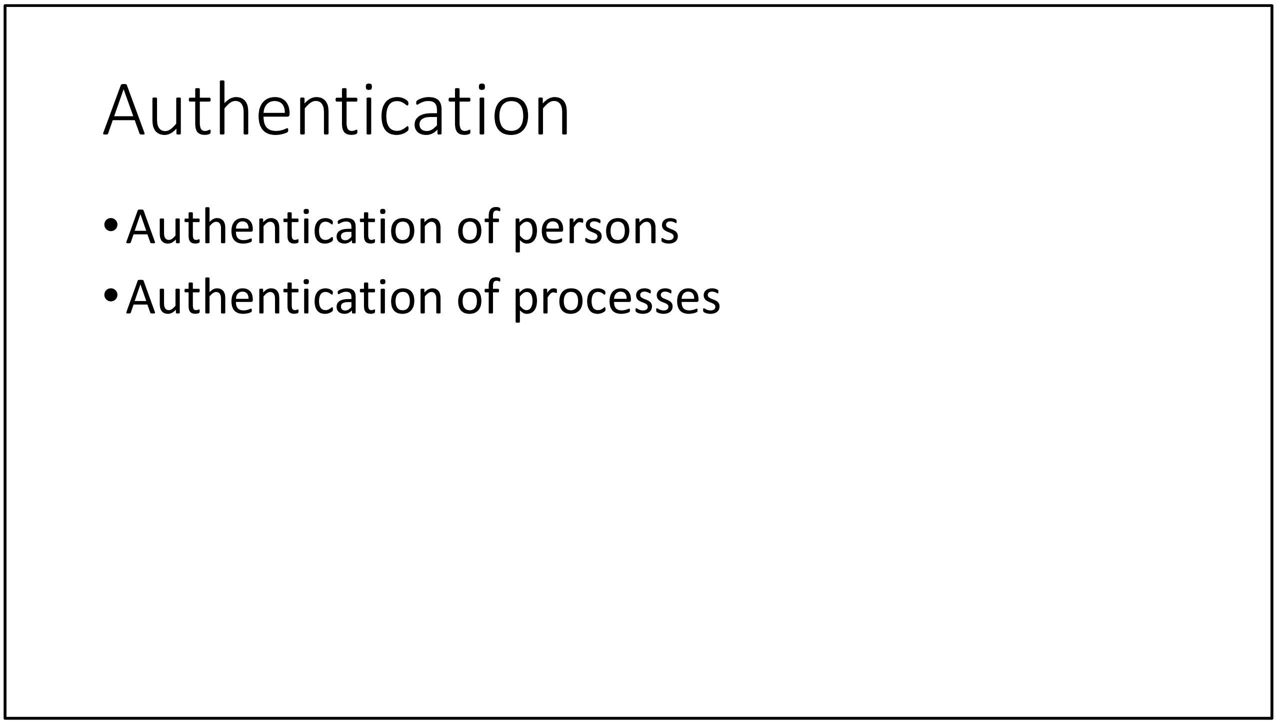
- Modifying the root user
- Changing the AWS support plan
- Closing an AWS account
- Creating a CloudFront key pair
- Enabling Multi-Factor Authentication on an S3 bucket
- Restore permissions for other IAM users

Episode 7.04
Authentication



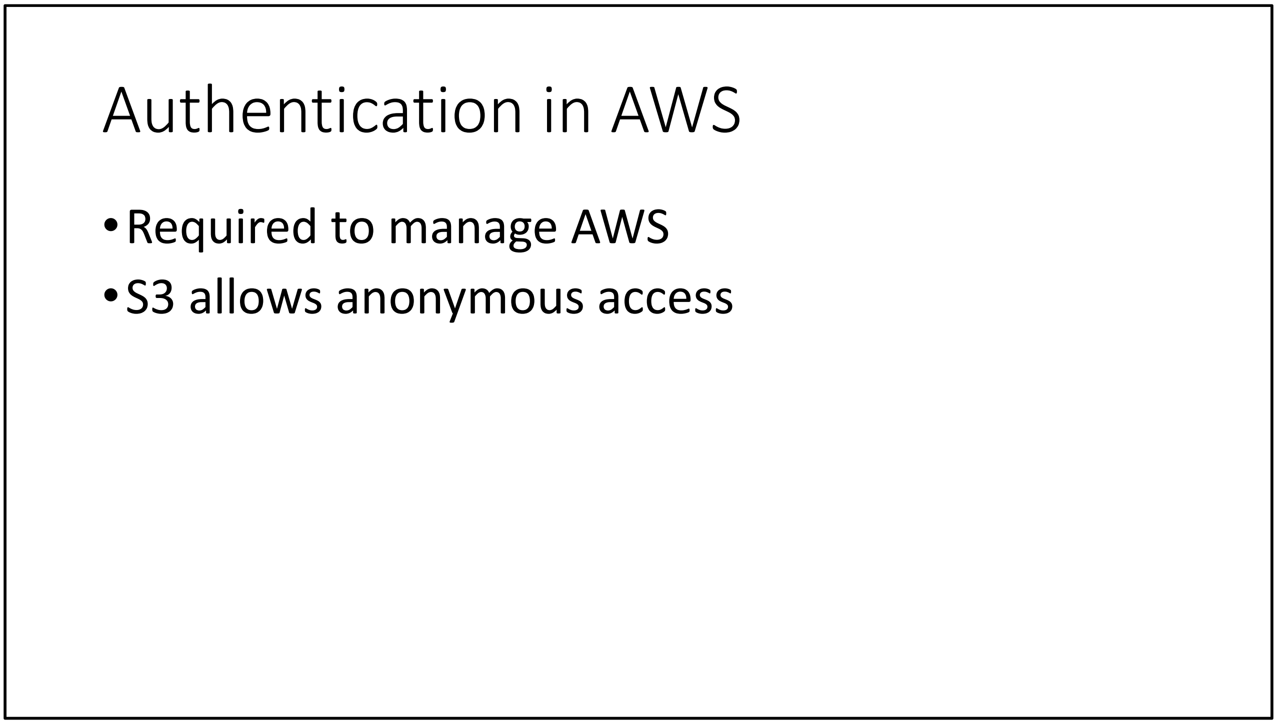
Authentication Defined

- Validation of credentials
- Credentials provide identity
- Single-factor
- Multi-factor
- Authentication of persons
- Authentication of processes



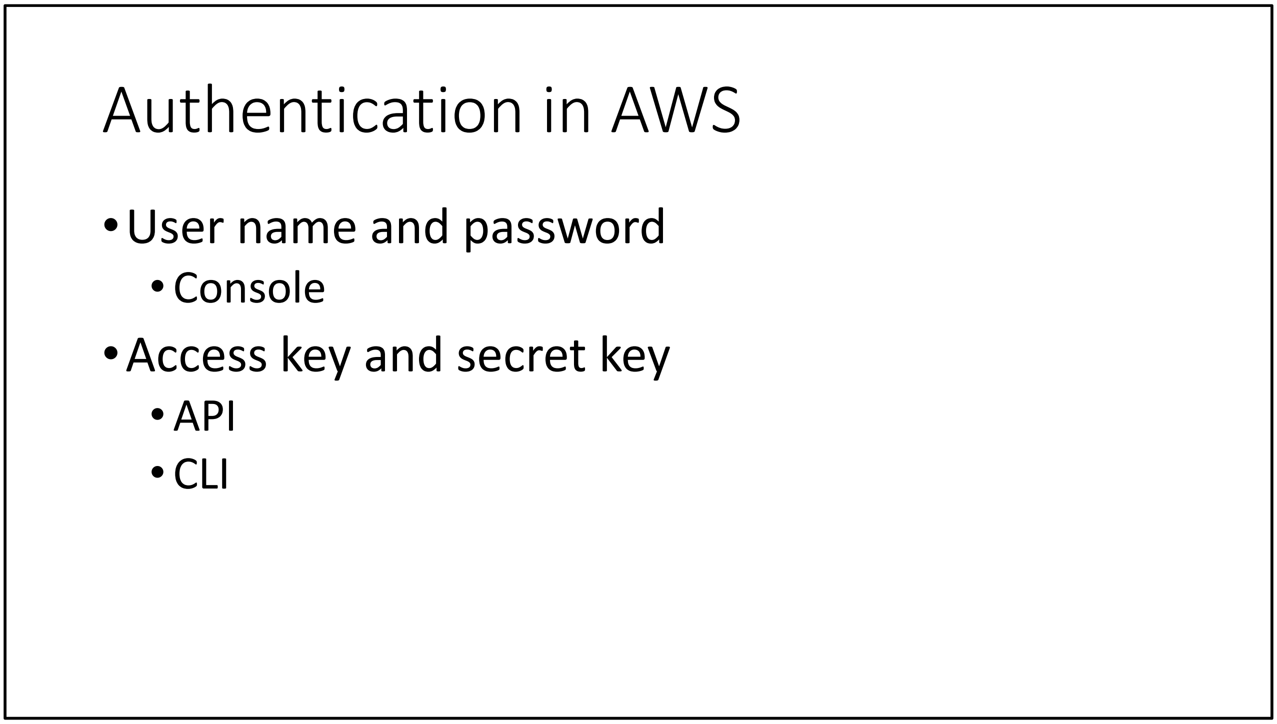
Authentication

- Authentication of persons
- Authentication of processes



Authentication in AWS

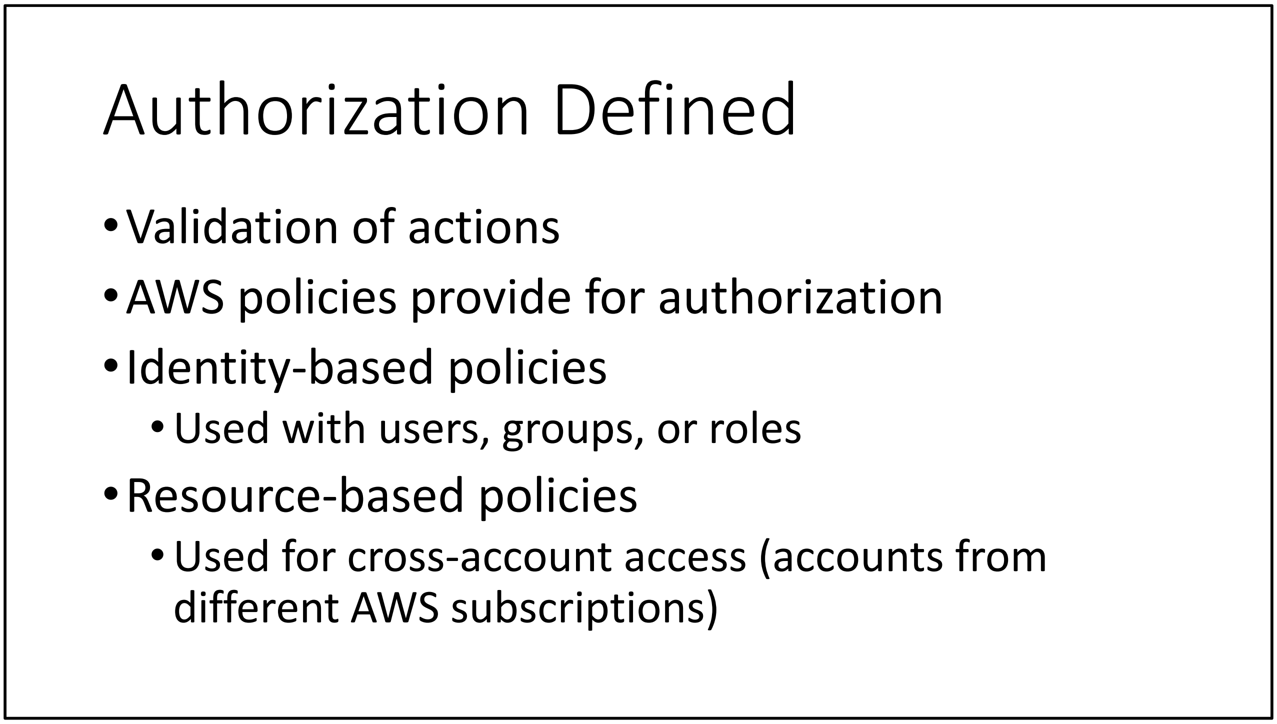
- Required to manage AWS
- S3 allows anonymous access



Authentication in AWS

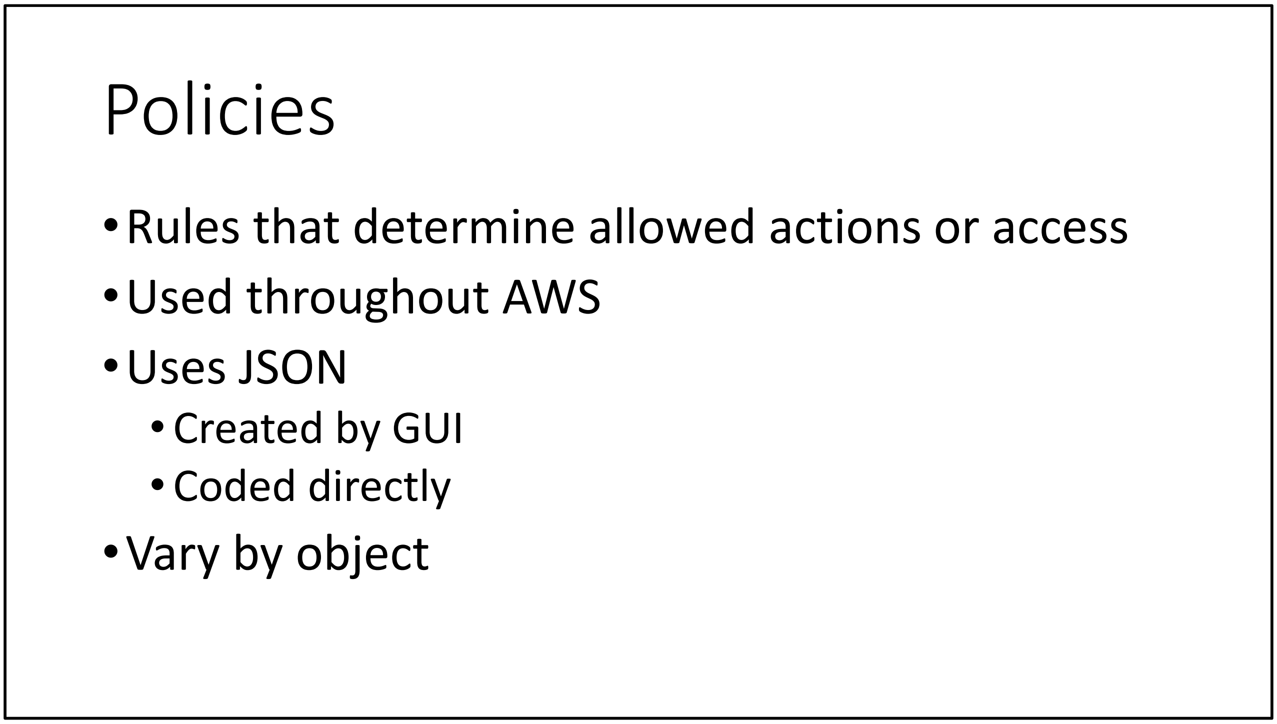
- User name and password
  - Console
- Access key and secret key
  - API
  - CLI

Episode 7.05
Authorization Policies



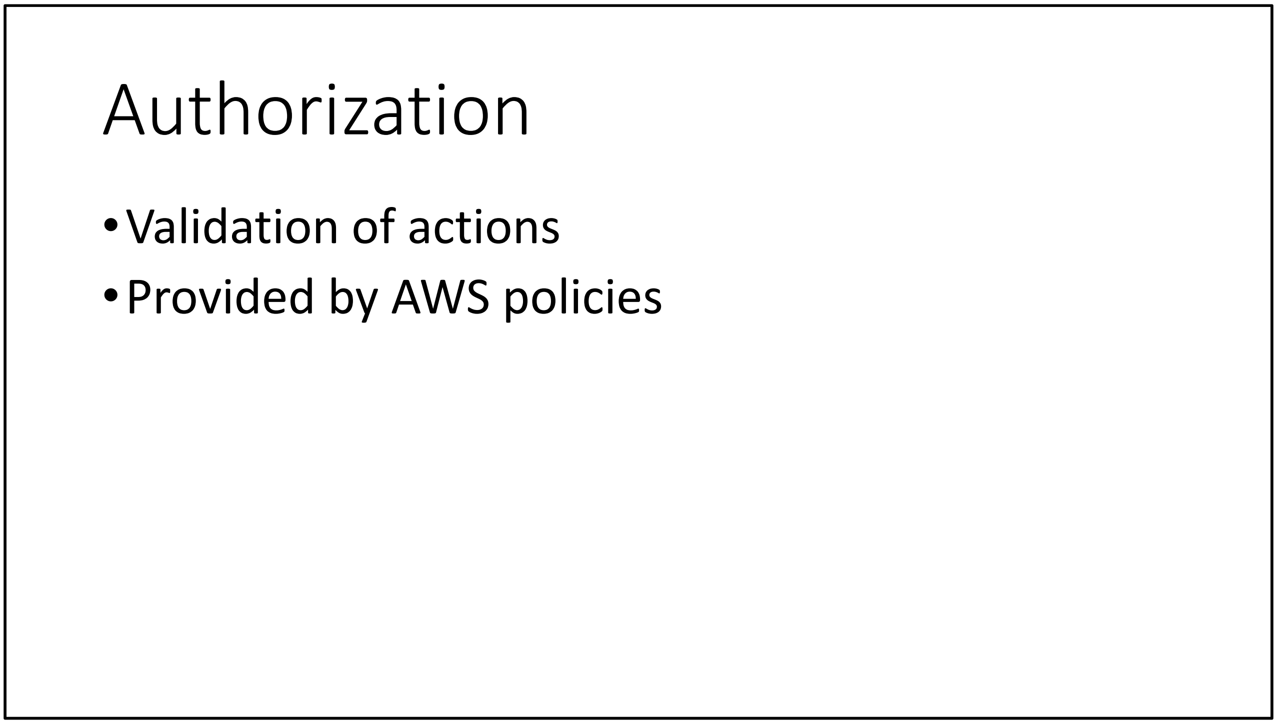
Authorization Defined

- Validation of actions
- AWS policies provide for authorization
- Identity-based policies
  - Used with users, groups, or roles
- Resource-based policies
  - Used for cross-account access (accounts from different AWS subscriptions)



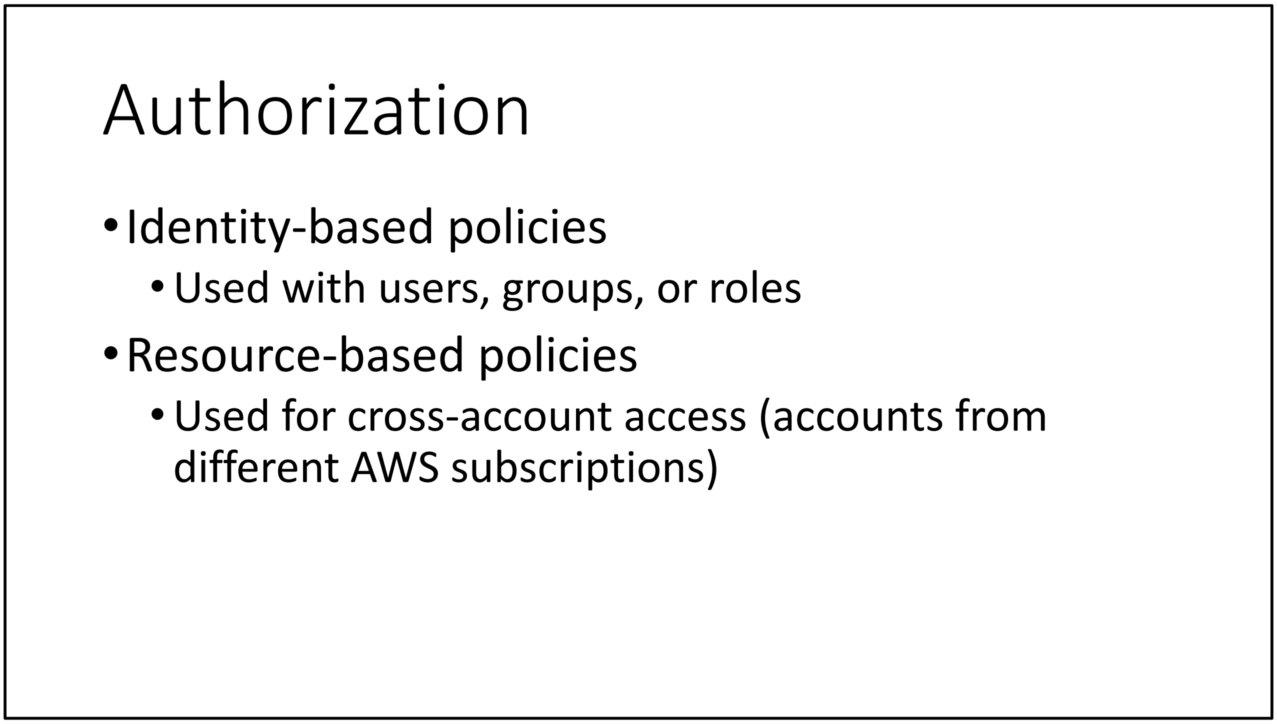
Policies

- Rules that determine allowed actions or access
- Used throughout AWS
- Uses JSON
  - Created by GUI
  - Coded directly
- Vary by object



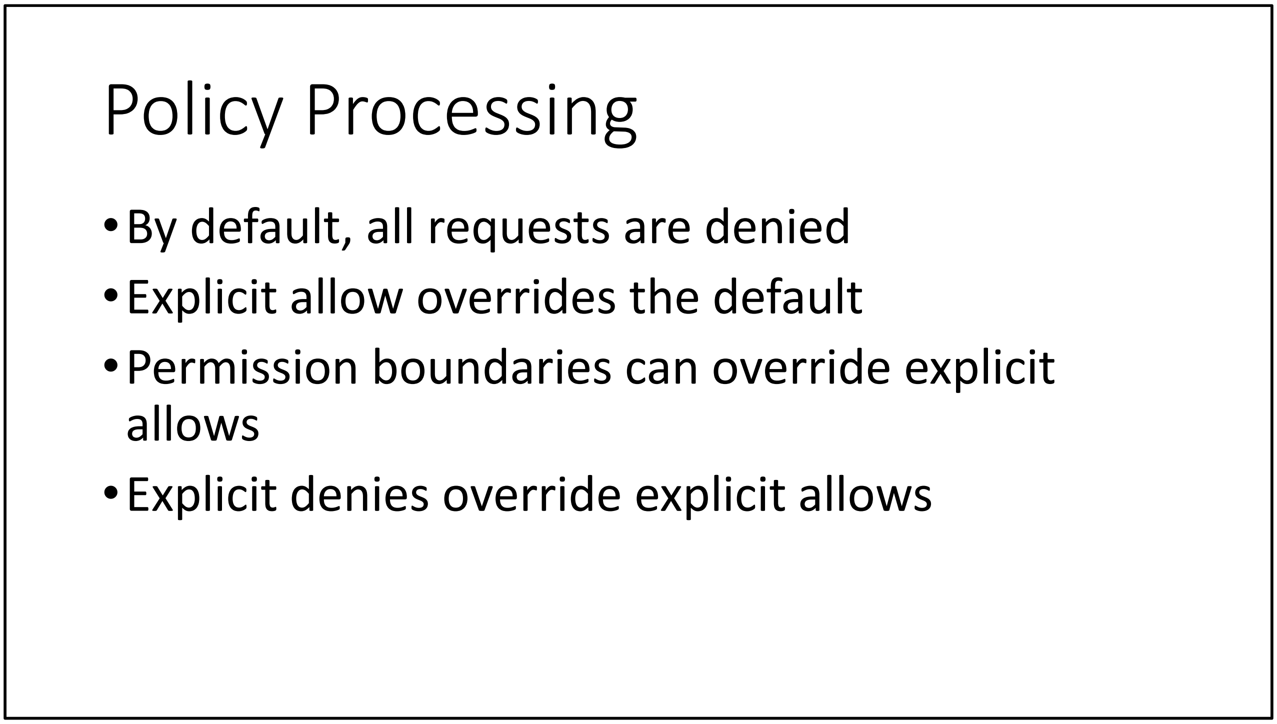
Authorization

- Validation of actions
- Provided by AWS policies



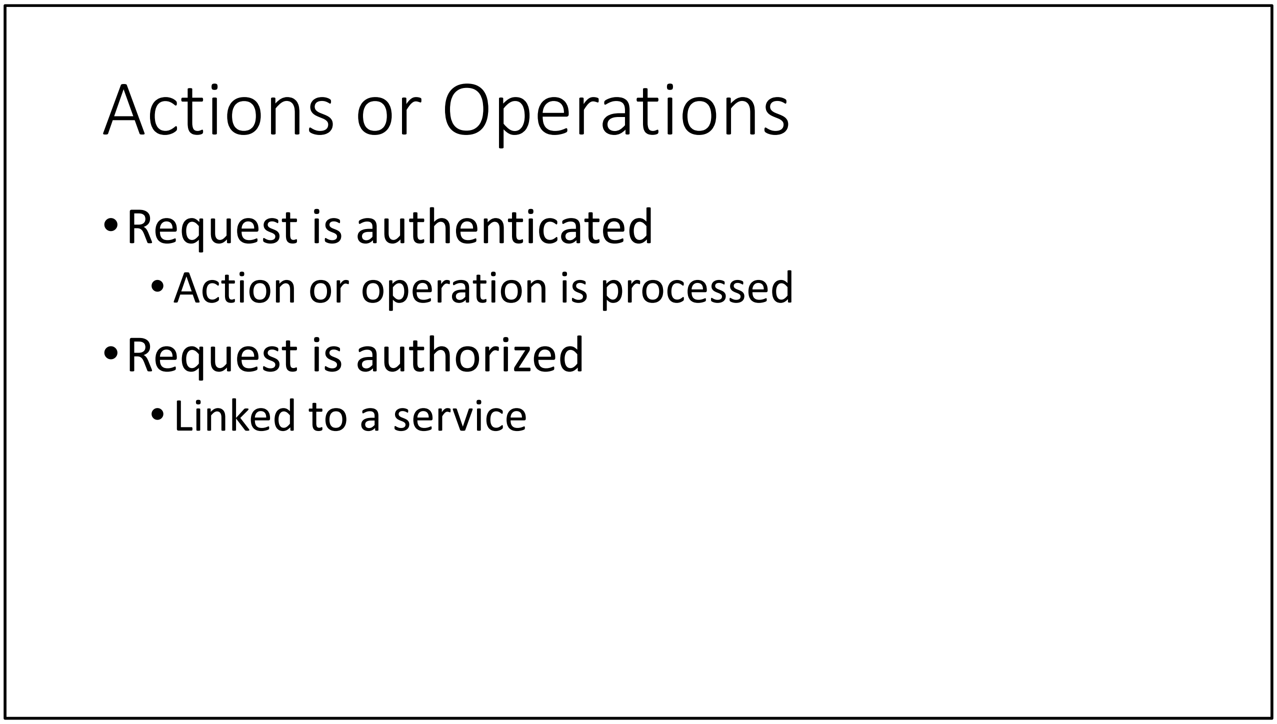
Authorization

- Identity-based policies
  - Used with users, groups, or roles
- Resource-based policies
  - Used for cross-account access (accounts from different AWS subscriptions)



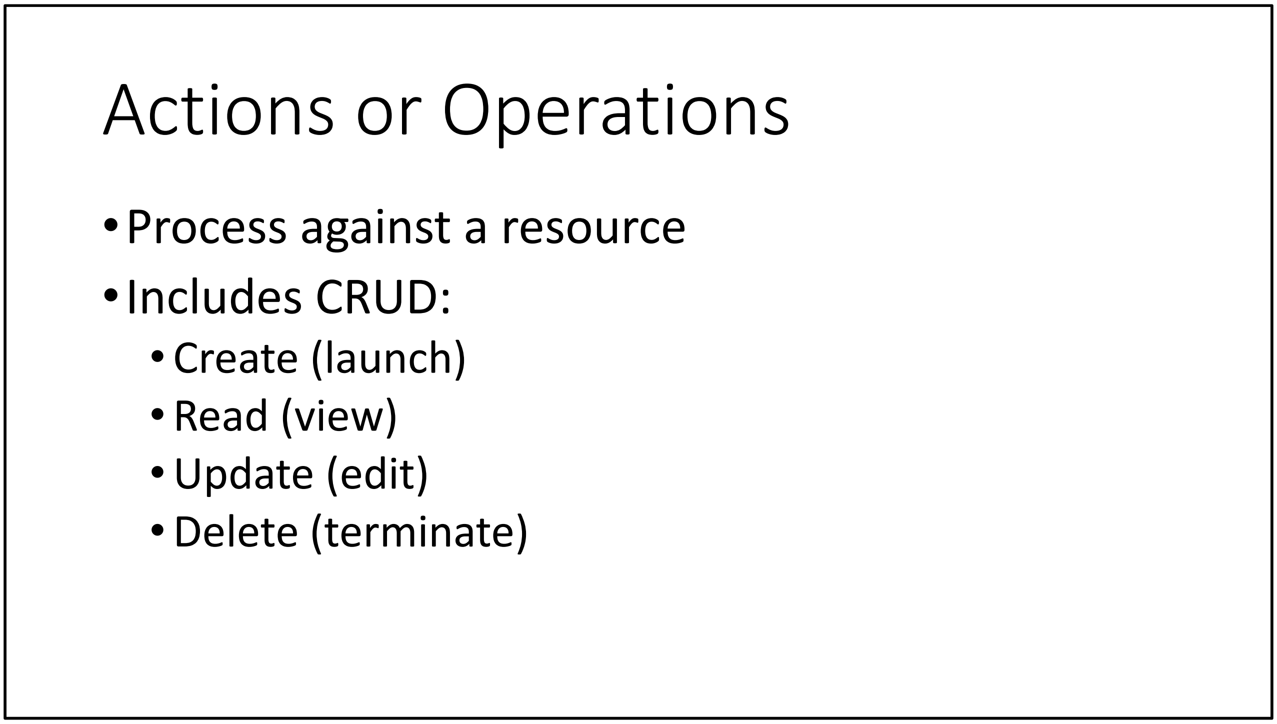
Policy Processing

- By default, all requests are denied
- Explicit allow overrides the default
- Permission boundaries can override explicit allows
- Explicit denies override explicit allows



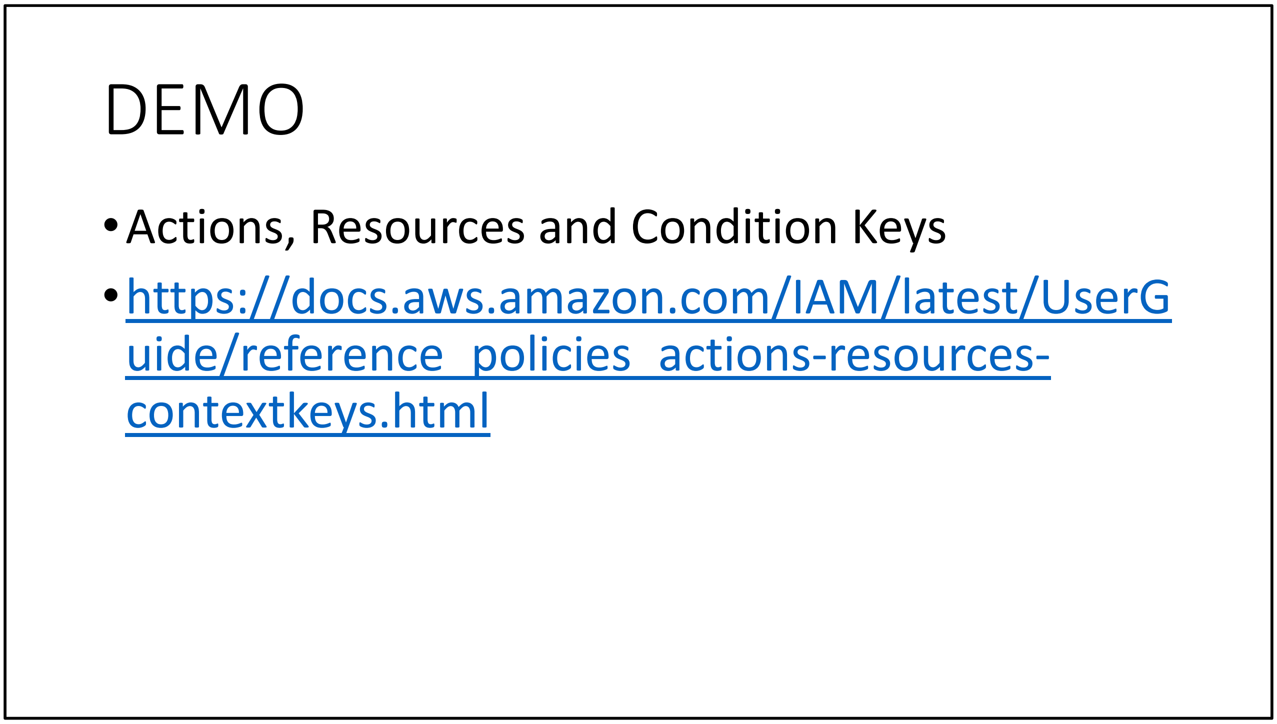
Actions or Operations

- Request is authenticated
  - Action or operation is processed
- Request is authorized
  - Linked to a service



Actions or Operations

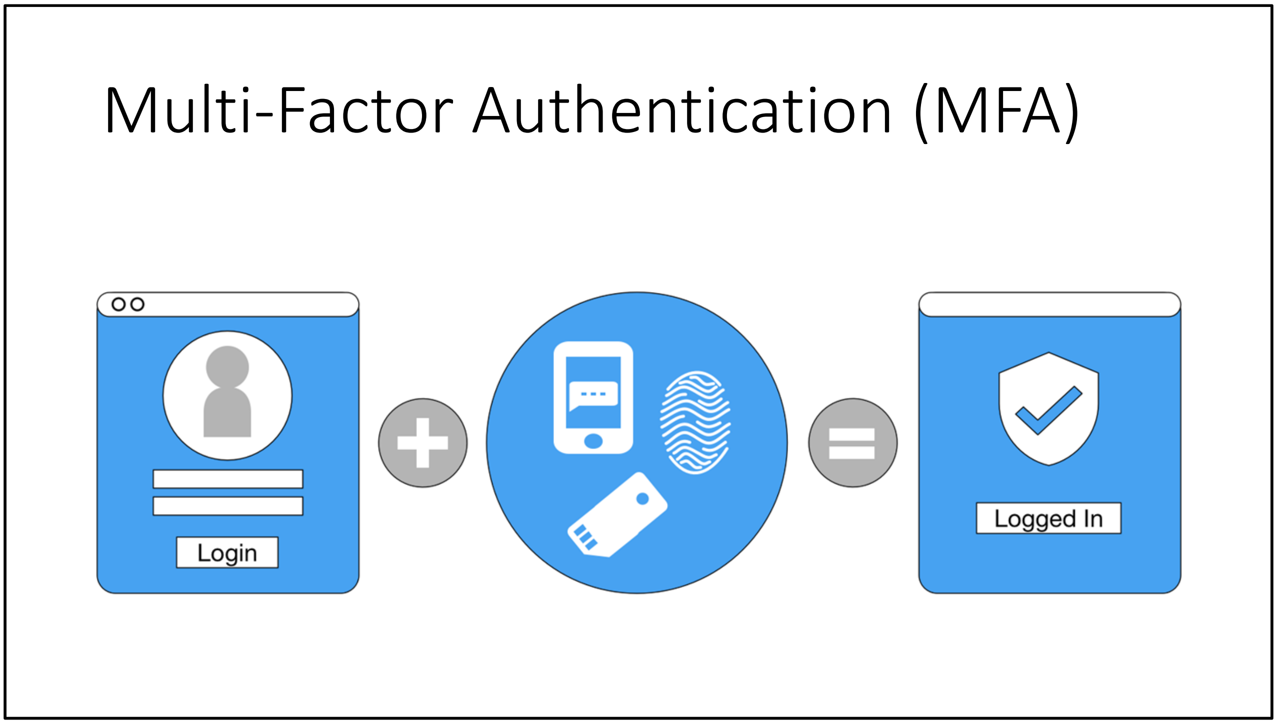
- Process against a resource
- Includes CRUD:
  - Create (launch)
  - Read (view)
  - Update (edit)
  - Delete (terminate)



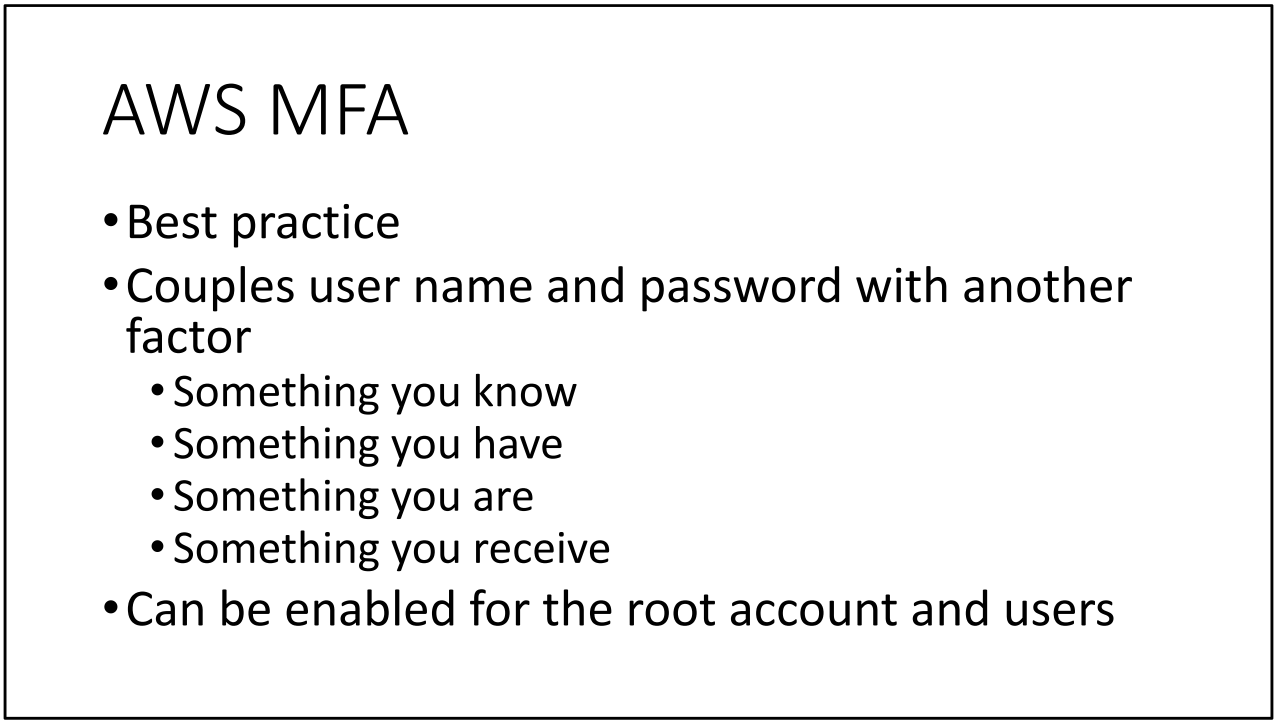
DEMO

- Actions, Resources and Condition Keys
- https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_actions-resources-contextkeys.html

Episode 7.06
Multi-Factor Authentication



Multi-Factor Authentication (MFA)



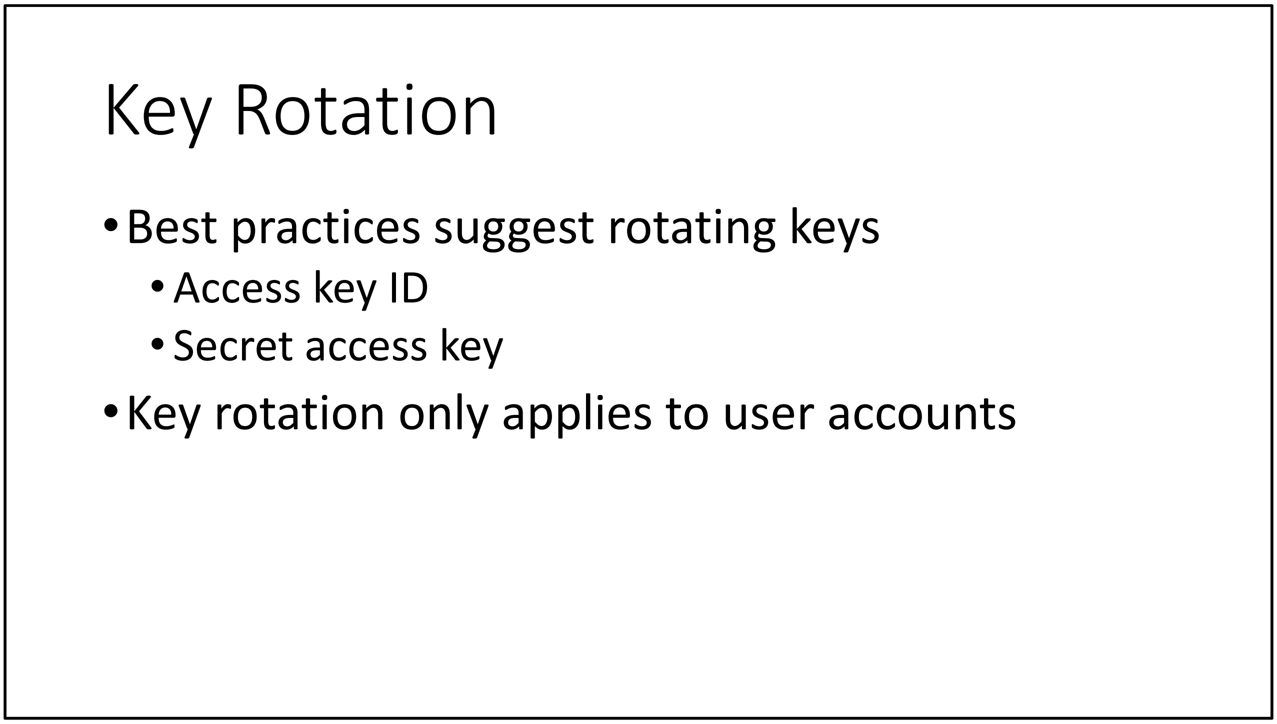
AWS MFA

- Best practice
- Couples user name and password with another factor
  - Something you know
  - Something you have
  - Something you are
  - Something you receive
- Can be enabled for the root account and users

DEMO

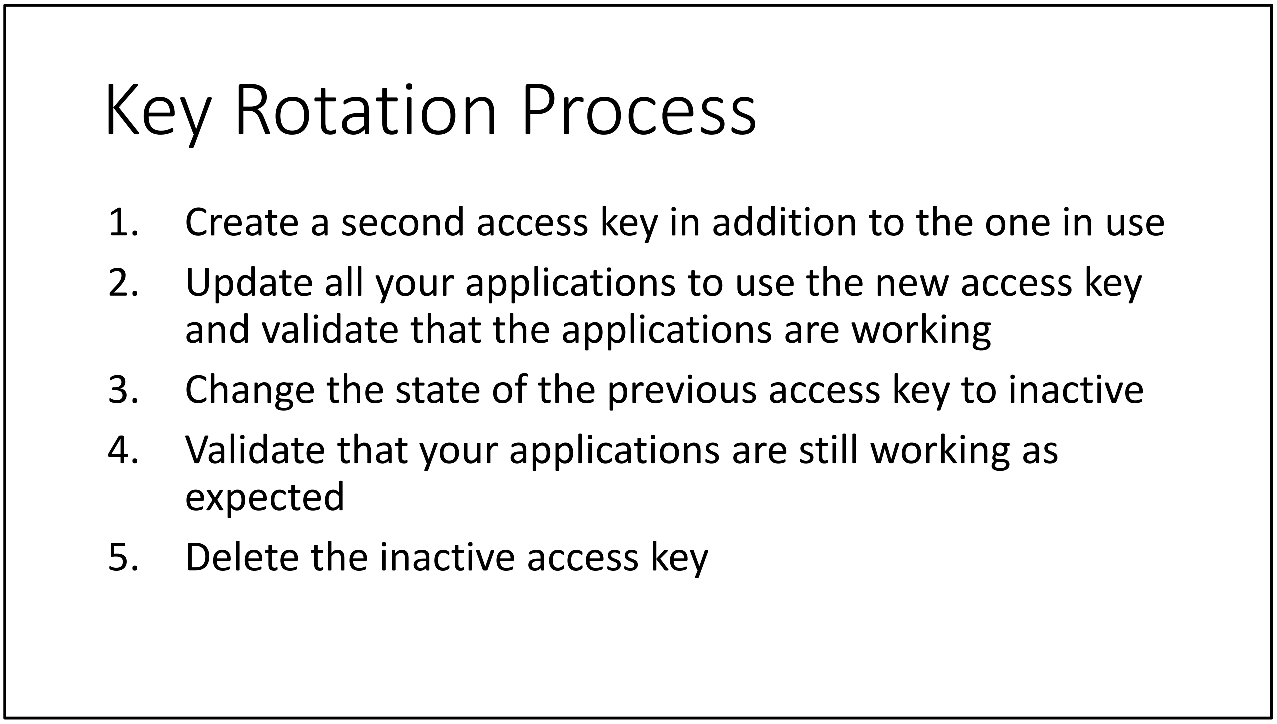
- MFA Form Factors
- https://aws.amazon.com/iam/details/mfa/

Episode 7.07
Key Rotation



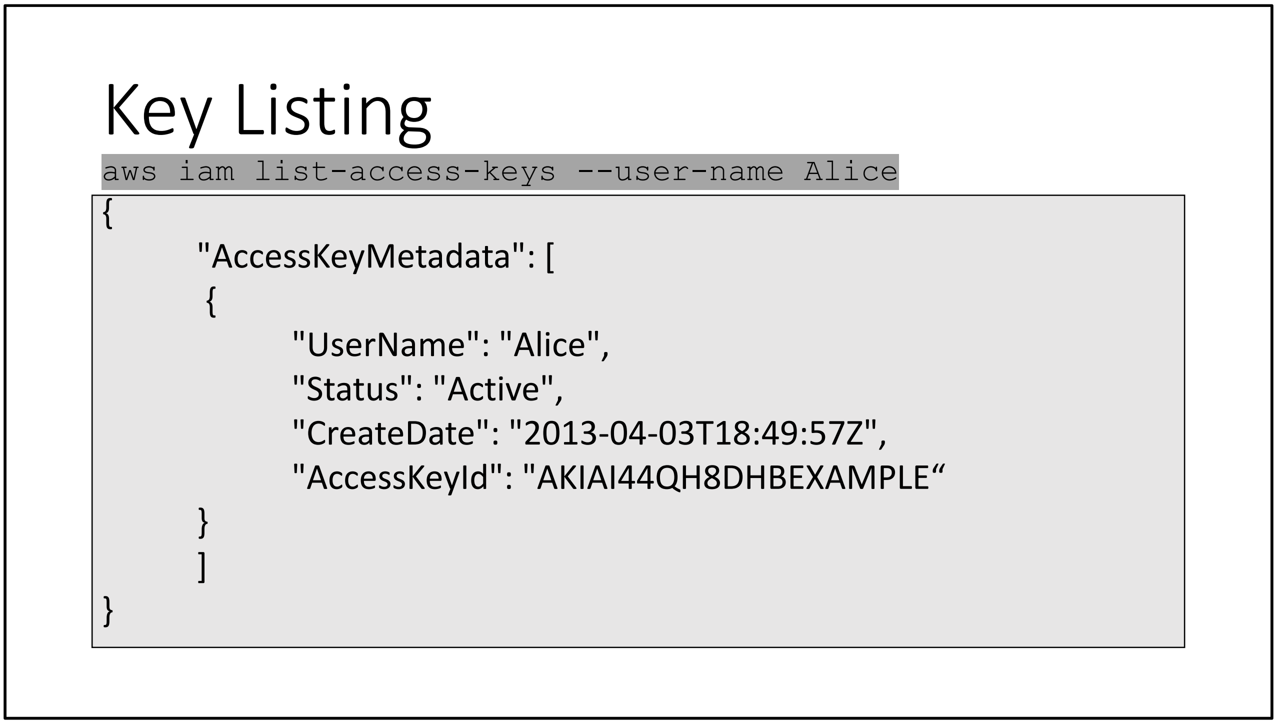
Key Rotation

- Best practices suggest rotating keys
  - Access key ID
  - Secret access key
- Key rotation only applies to user accounts



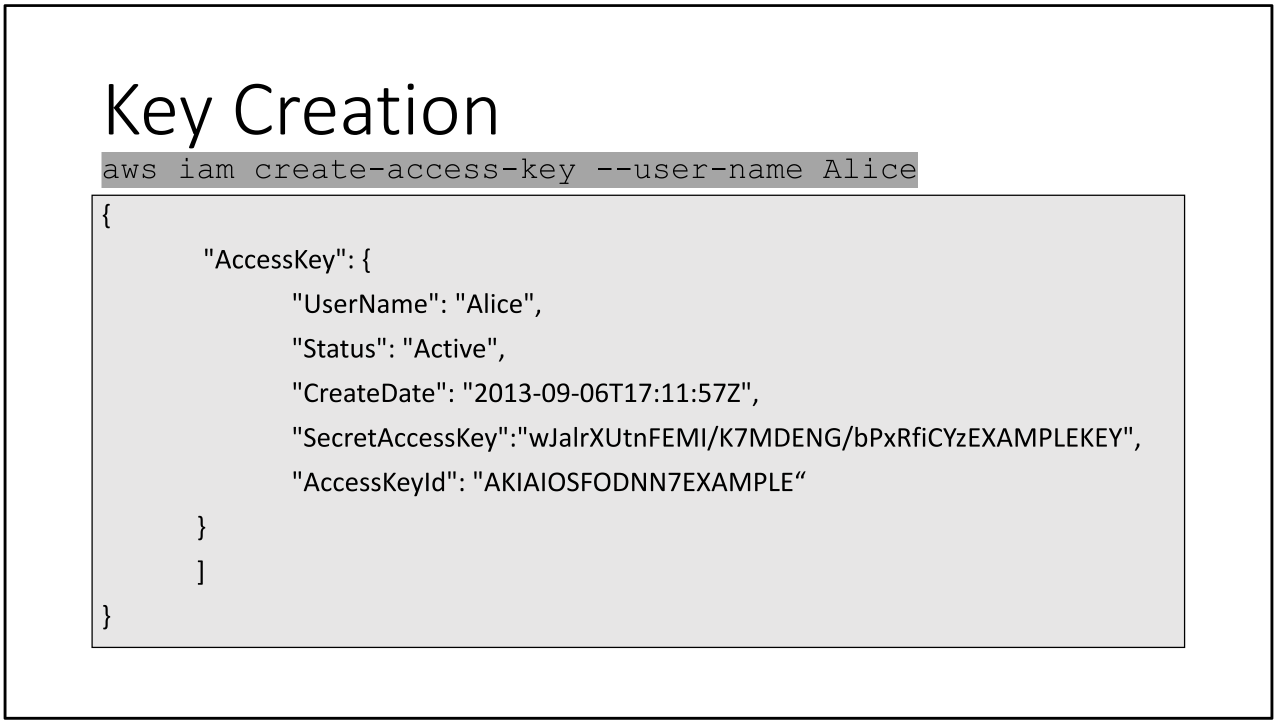
Key Rotation Process

1. Create a second access key in addition to the one in use
2. Update all your applications to use the new access key and validate that the applications are working
3. Change the state of the previous access key to inactive
4. Validate that your applications are still working as expected
5. Delete the inactive access key



Key Listing

```
aws iam list-access-keys --user-name Alice
{
        "AccessKeyMetadata": [
         {
                "UserName": "Alice",
                "Status": "Active",
                "CreateDate": "2013-04-03T18:49:57Z",
                "AccessKeyId": "AKIAI44QH8DHBEXAMPLE“
        }
        ]
}
```

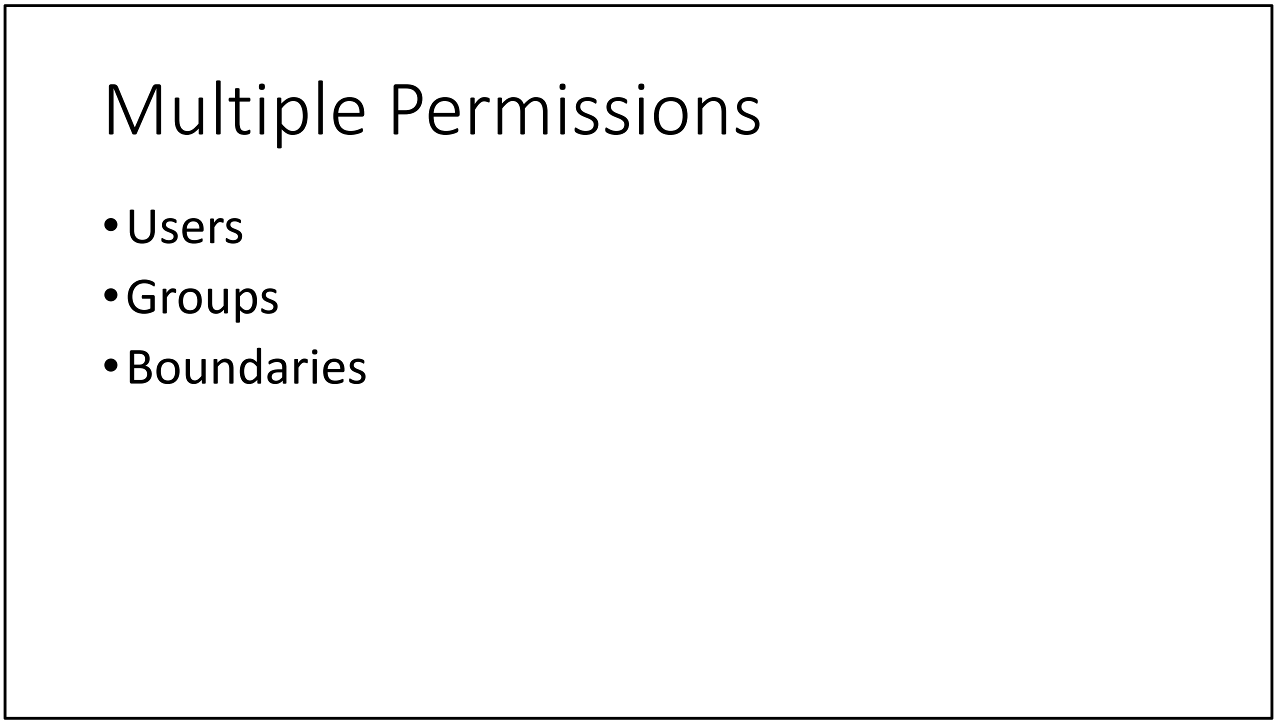


Key Creation

```
aws iam create-access-key --user-name Alice
```

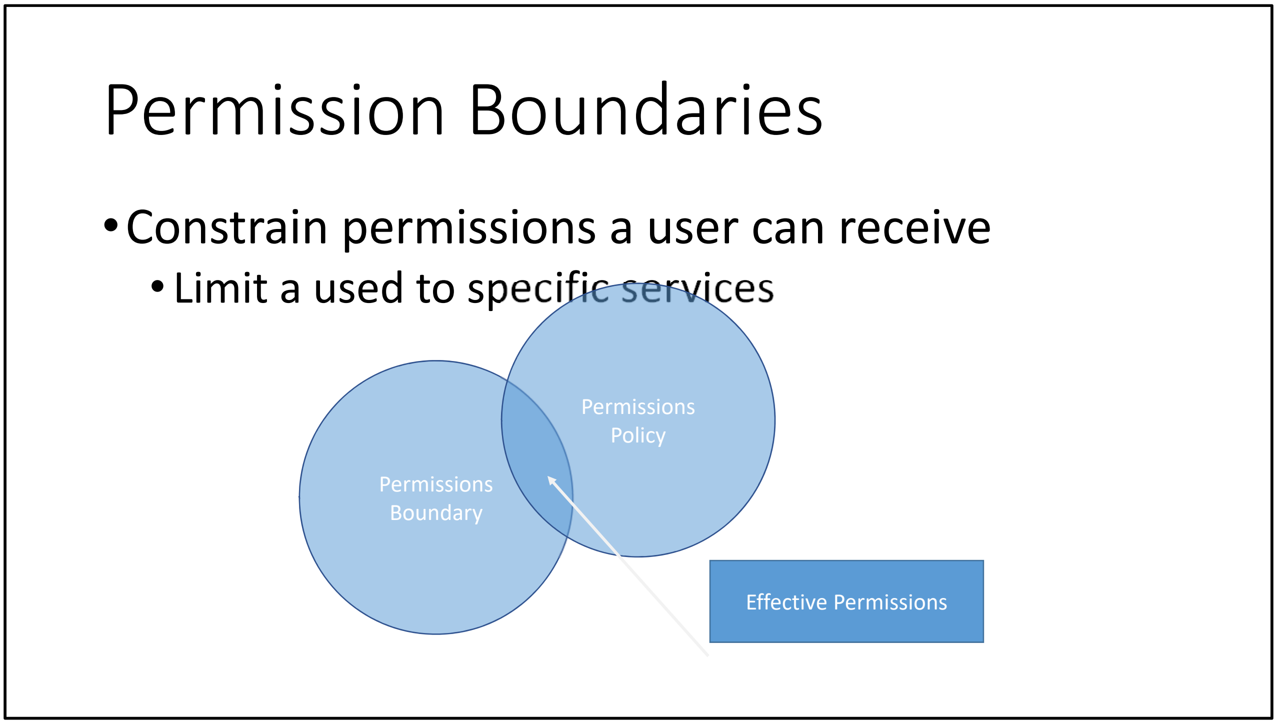
```
{
        "AccessKey": {
                "UserName": "Alice",
                "Status": "Active",
                "CreateDate": "2013-09-06T17:11:57Z",
                "SecretAccessKey":"wJalrXUtnFEMI/K7MDENG/bPxRfiCYzEXAMPLEKEY",
                "AccessKeyId": "AKIAIOSFODNN7EXAMPLE“
        }
        ]
}
```

Episode 7.08
Multiple Permissions



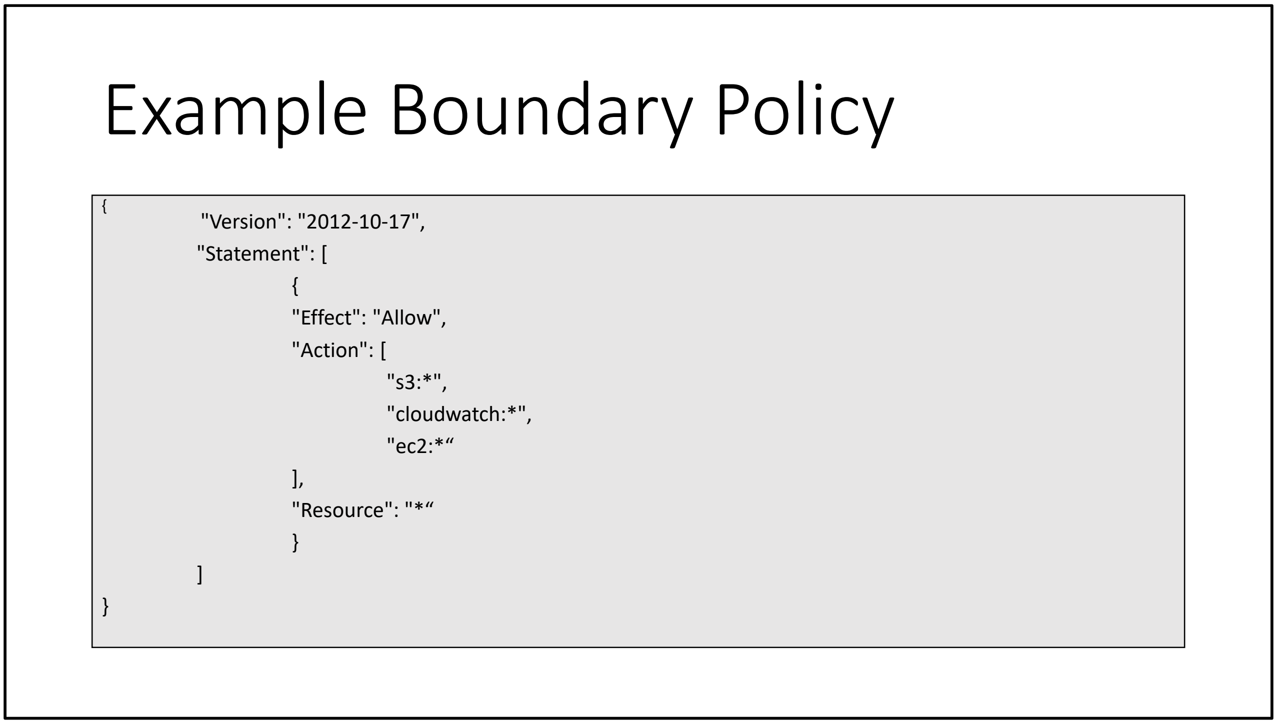
Multiple Permissions

- Users
- Groups
- Boundaries



Permission Boundaries

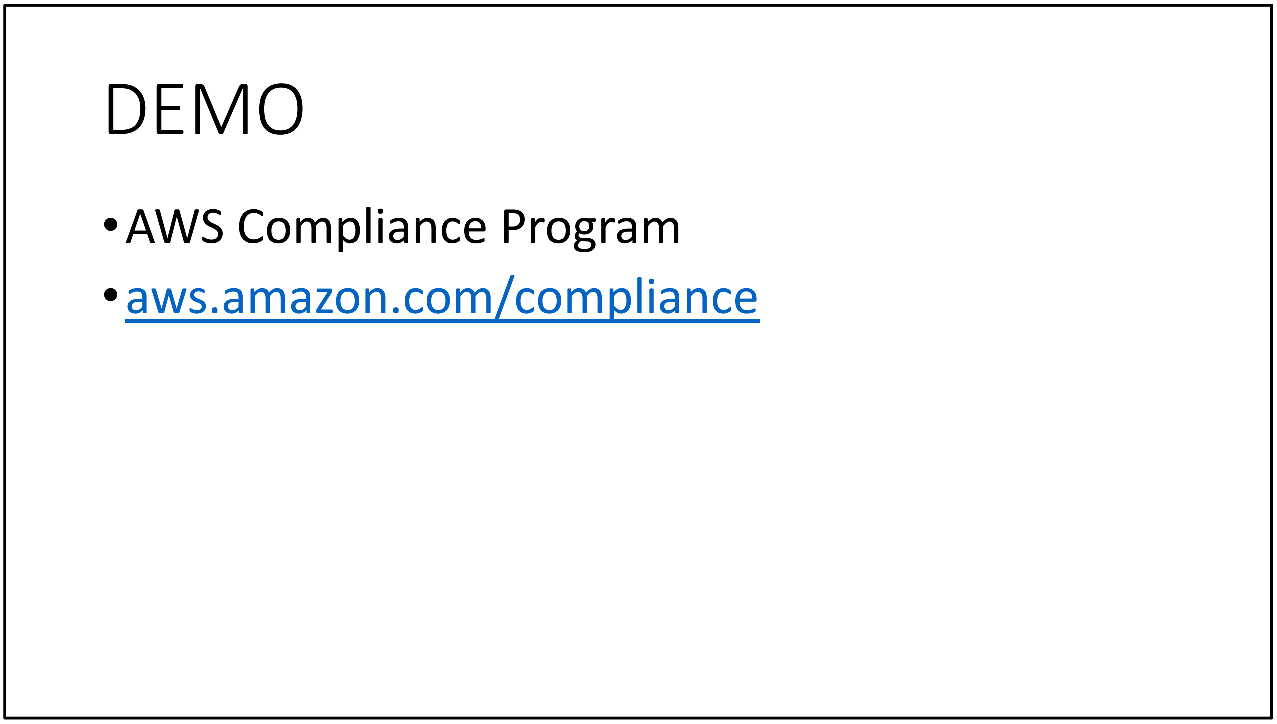
- Constrain permissions a user can receive
  - Limit a used to specific services



Example Boundary Policy

```
{
        "Version": "2012-10-17",
       "Statement": [
               {
               "Effect": "Allow",
               "Action": [
                      "s3:*",
                      "cloudwatch:*",
                      "ec2:*“
               ],
               "Resource": "*“
               }
       ]
}
```

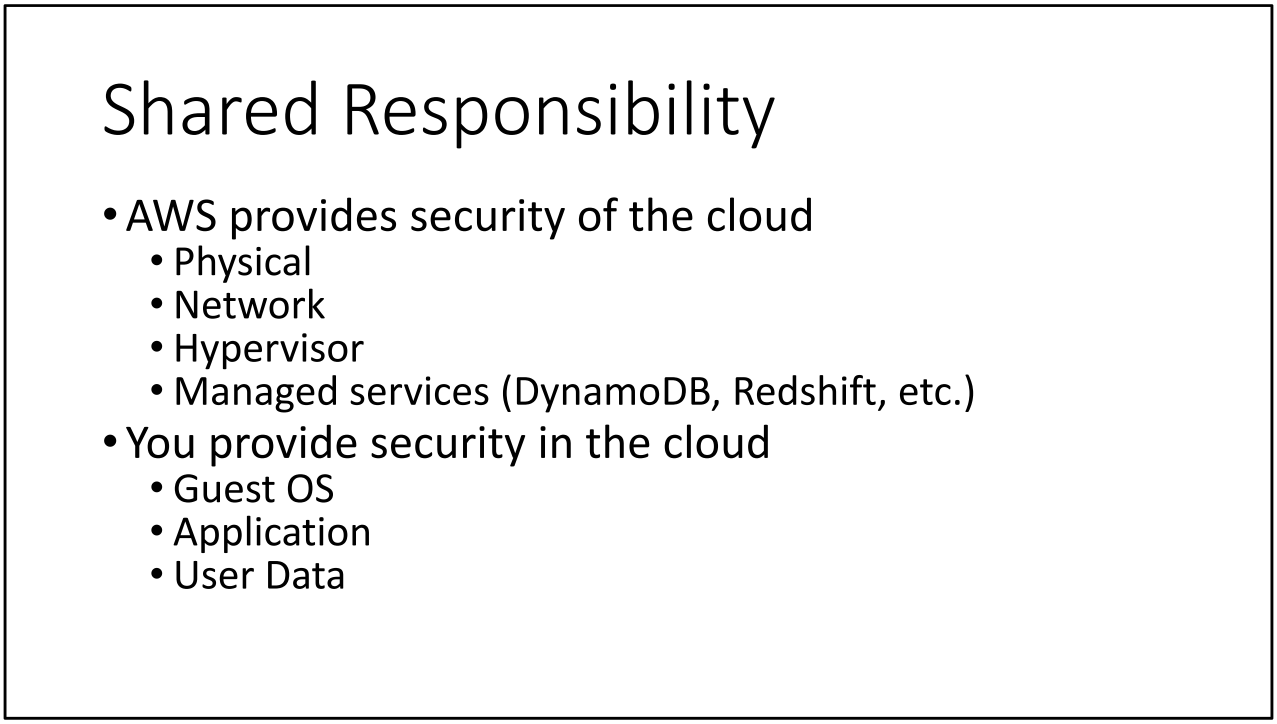
Episode 7.09
AWS Compliance Program



DEMO

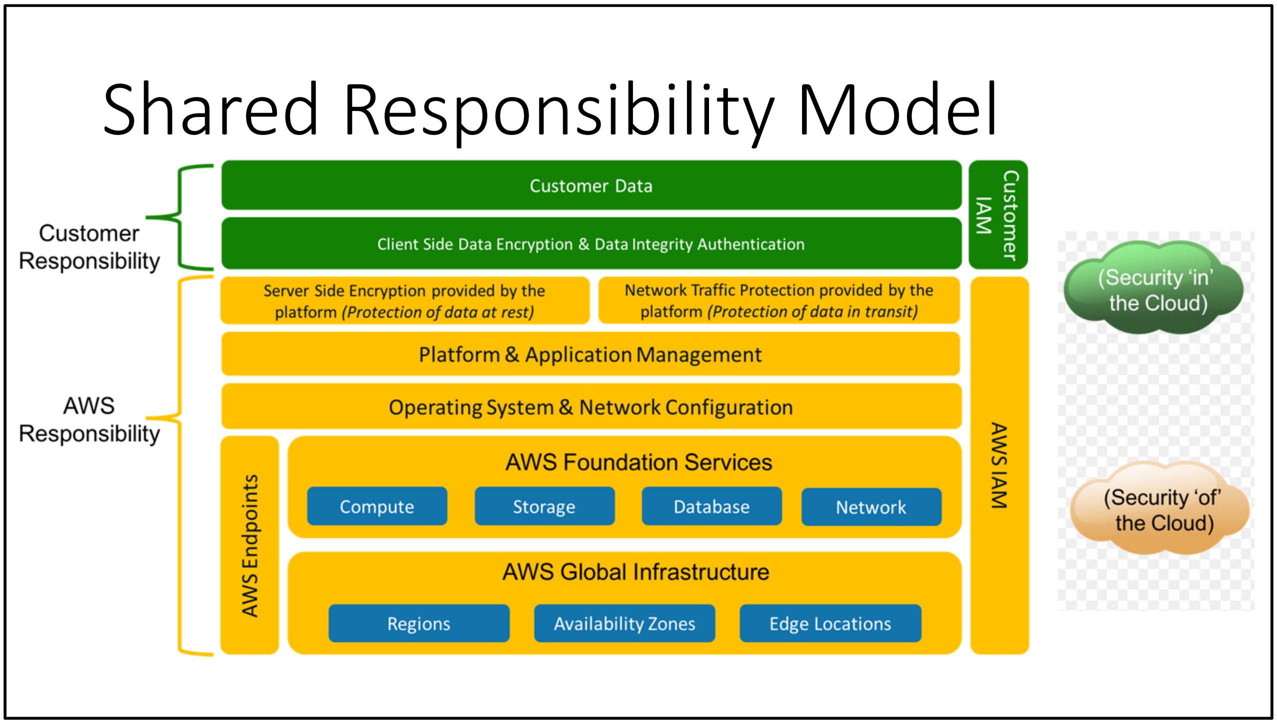
- AWS Compliance Program
- aws.amazon.com/compliance

Episode 7.10
Shared Responsibility Model



Shared Responsibility

- AWS provides security of the cloud
  - Physical
  - Network
  - Hypervisor
  - Managed services (DynamoDB, Redshift, etc.)
- You provide security in the cloud
  - Guest OS
  - Application
  - User Data



Shared Responsibility Model

Customer Responsibility

Customer Data

Client Side Data Encryption & Data Integrity Authentication

Customer IAM

AWS Responsibility

Server Side Encryption provided by the platform *(Protection of data at rest)*

Network Traffic Protection provided by the platform *(Protection of data in transit)*

Platform & Application Management

Operating System & Network Configuration

AWS Endpoints

AWS Foundation Services

Compute | Storage | Database | Network

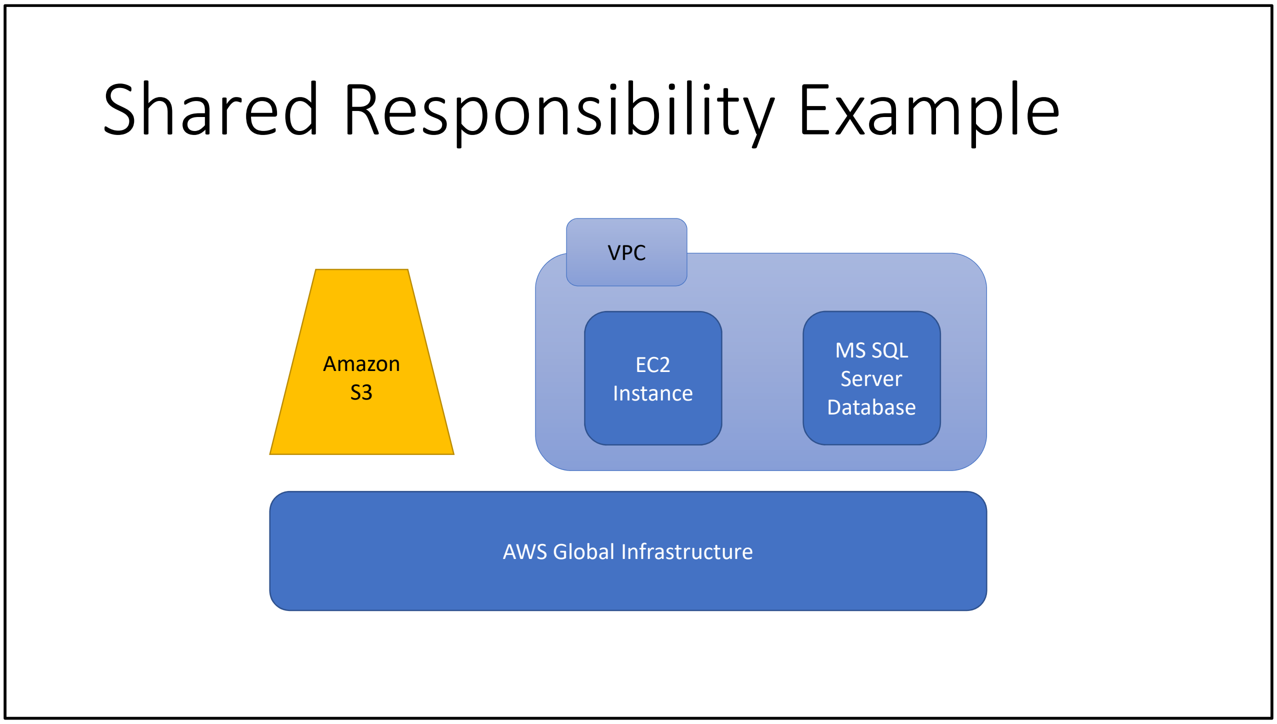
AWS Global Infrastructure

Regions | Availability Zones | Edge Locations

AWS IAM

(Security 'in' the Cloud)

(Security 'of' the Cloud)



Shared Responsibility Example