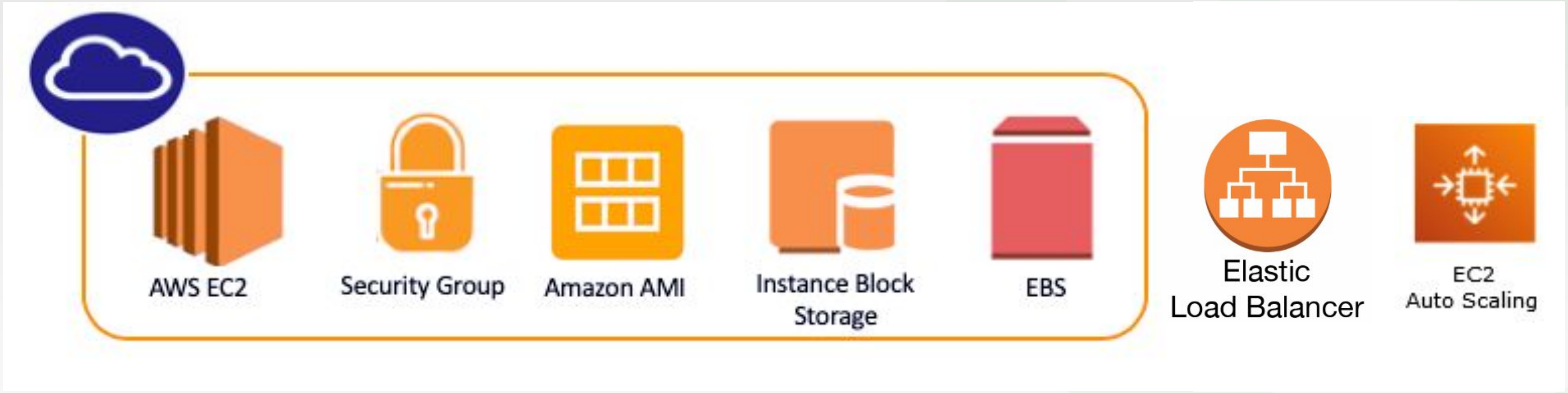
BATCH	:	146 - 149
LESSON	:	**AWS**
DATE	:	02.08.2023
SUBJECT	:	**AWS-IAM**

ZOOM GİRİŞLERİNİZİ LÜTFEN **LMS** SİSTEMİ ÜZERİNDEN YAPINIZ

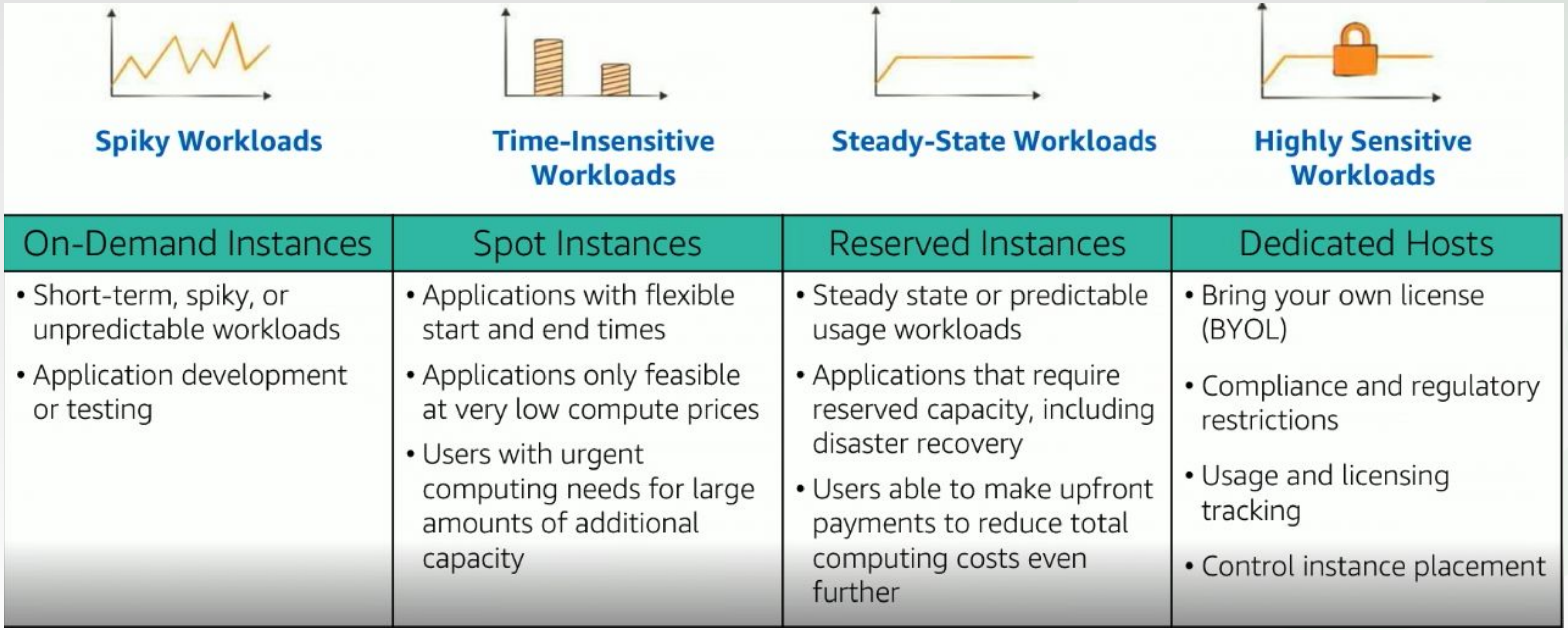
AWS EC2

EC2

EC2 Basic Components:



EC2



Spiky Workloads	Time-Insensitive Workloads	Steady-State Workloads	Highly Sensitive Workloads
On-Demand Instances	Spot Instances	Reserved Instances	Dedicated Hosts
• Short-term, spiky, or unpredictable workloads • Application development or testing	• Applications with flexible start and end times • Applications only feasible at very low compute prices • Users with urgent computing needs for large amounts of additional capacity	• Steady state or predictable usage workloads • Applications that require reserved capacity, including disaster recovery • Users able to make upfront payments to reduce total computing costs even further	• Bring your own license (BYOL) • Compliance and regulatory restrictions • Usage and licensing tracking • Control instance placement

EC2

Instance Families

General Purpose

- Provide a balance of compute, memory and networking resources, and can be used for a variety of diverse workloads.

Compute Optimized

- Have a higher ratio of virtual CPUs to memory than the other families and the lowest cost per virtual CPU of all the EC2 instance types.

Memory Optimized

- Designed for memory-intensive applications, these instances have the lowest cost per GiB of RAM of all EC2 instance types.

Storage Optimized

- Storage optimized instances are designed for workloads that require high, sequential read and write access to very large data sets on local storage. They are optimized to deliver tens of thousands of low-latency, random I/O operations per second (IOPS) to applications.

Accelerated Computing

- Provide access to hardware-based compute accelerators such as graphics processing units (GPUs)

AWS IAM

IAM

What is IAM ?

- AWS IAM stands for **Identity & Access Management** and is the **primary service that handles authentication and authorization processes** within AWS environments.

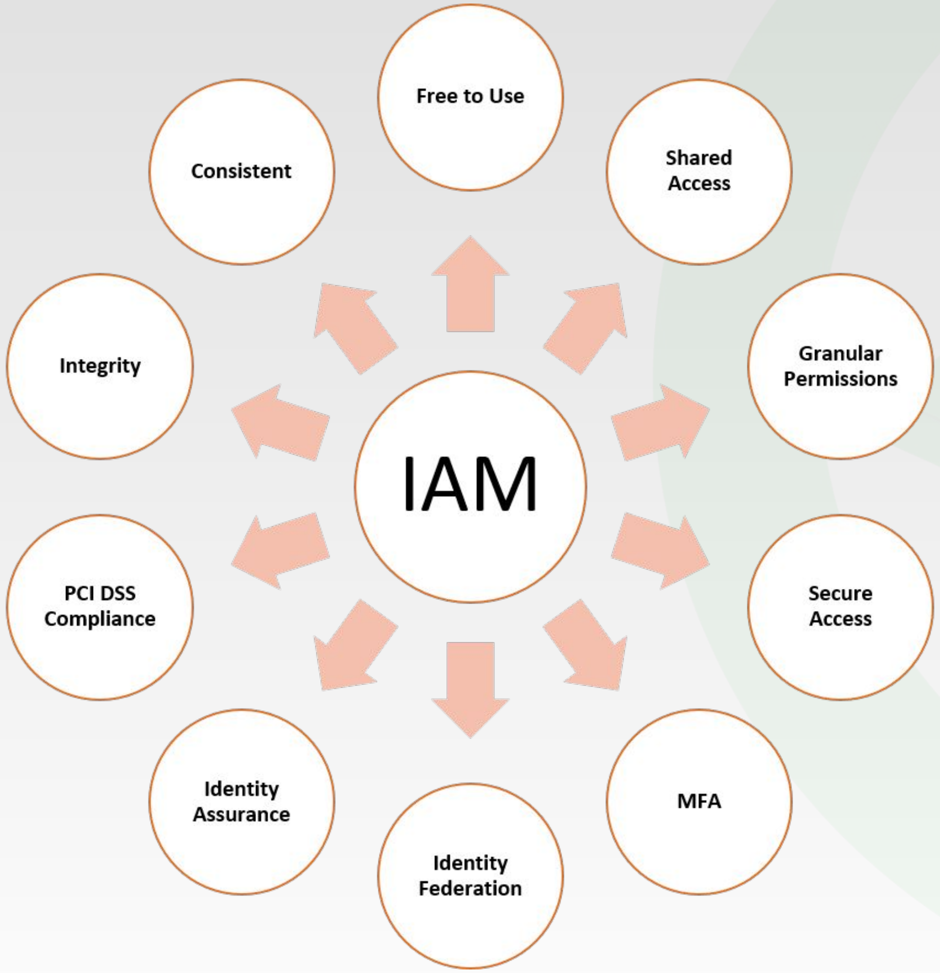
IAM

What is IAM?:

- By using AWS IAM, you can **manage users and their access level**.
- All **account settings** are made through this service.
- It allows us to **create and manage objects such as User, Group, Role, and Policy**.
- Account owner can identify and allow the user to use specified services.
- All kinds of **user password restrictions, access keys and multifactor authentication settings** are also made through IAM.

IAM

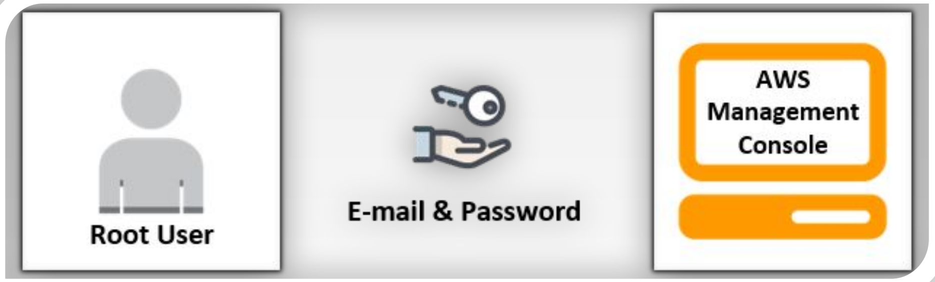
IAM Features:

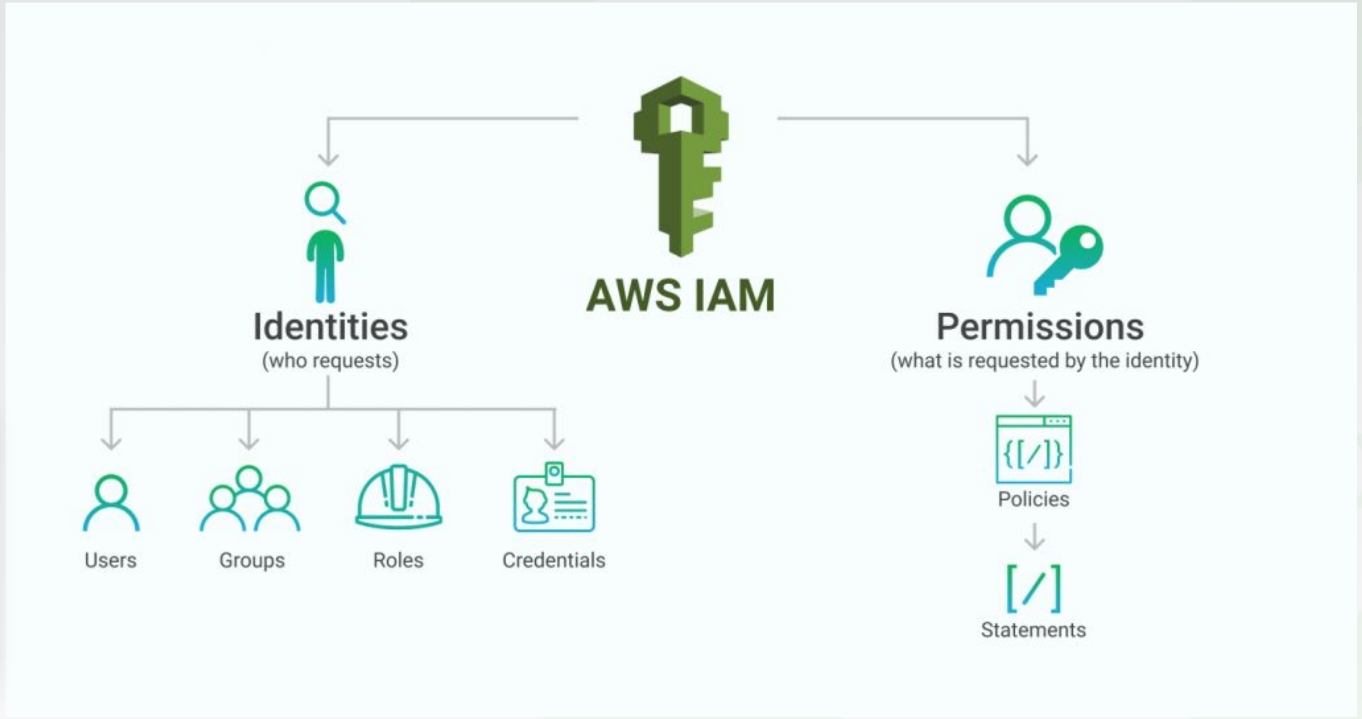


IAM

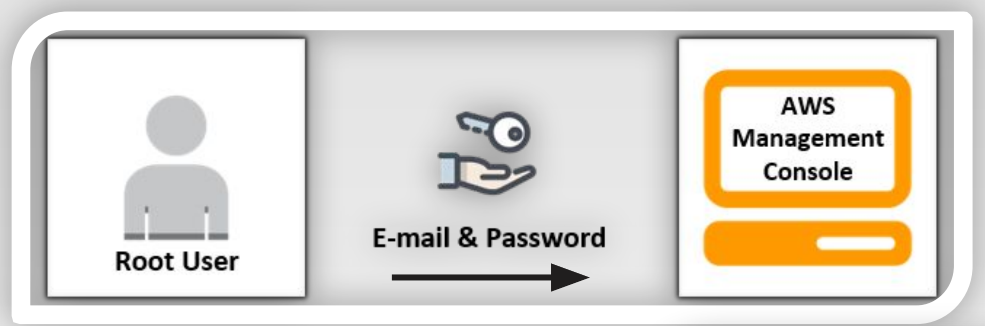
Categorizing IAM Components

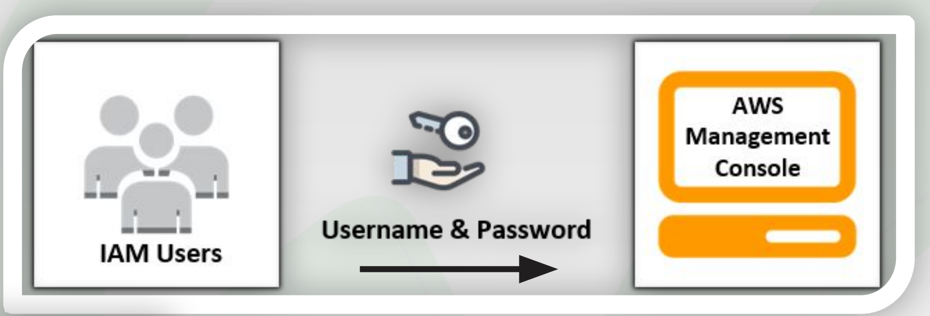
- IAM components can be mainly categorized under two terms; **identities and permissions**.

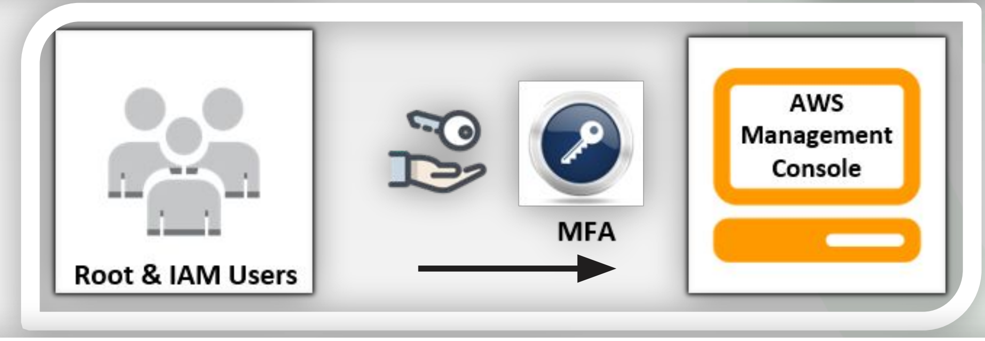


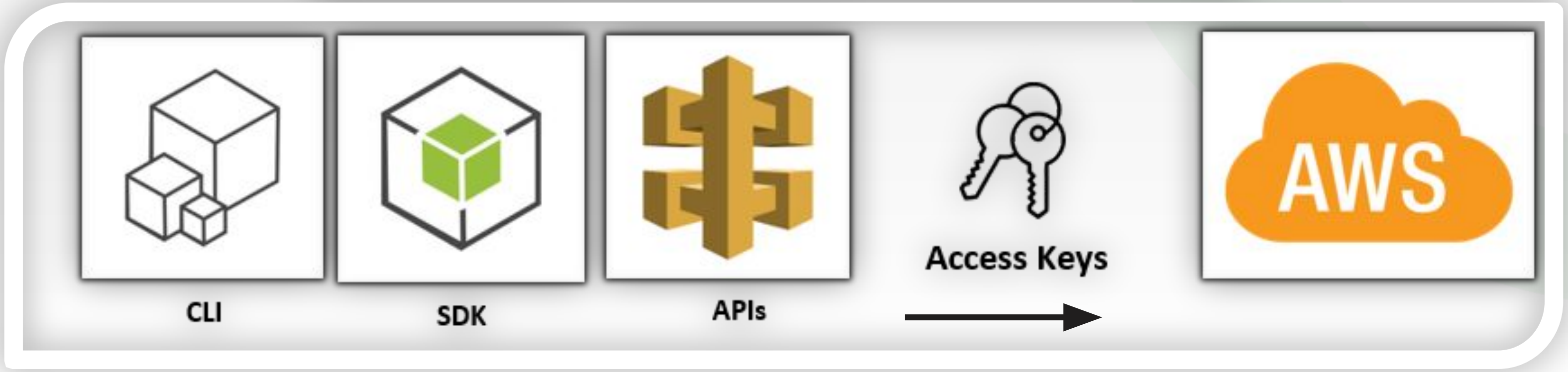


IAM


Root User
E-mail & Password
AWS Management Console


IAM Users
Username & Password
AWS Management Console


Root & IAM Users
MFA
AWS Management Console


CLI
SDK
APIs
Access Keys
AWS

IAM

What is an IAM User?

An **IAM user is an entity that you create** in AWS.

- The IAM user represents the **person or service who uses** AWS services.
- A primary use for IAM users is to give people the ability to **sign in to the AWS Management Console** for interactive tasks and to **make programmatic requests to AWS services** using the API or CLI.
- A user in AWS consists of **a name, a password** to sign in to the AWS Management Console, and up to **two access keys** that can be used with the API or CLI.
- When you create **an IAM user, you grant it permissions by making it a member of a group** that has appropriate permission policies attached (recommended), or by **directly attaching policies** to the user.
- You can also **clone the permissions of an existing IAM user**, which automatically makes the new user a member of the same groups and attaches all the same policies.

IAM

IAM User Types

- Real Person
- Web Application
- Service Accounts
- Software

IAM

IAM – Users – Account Root User

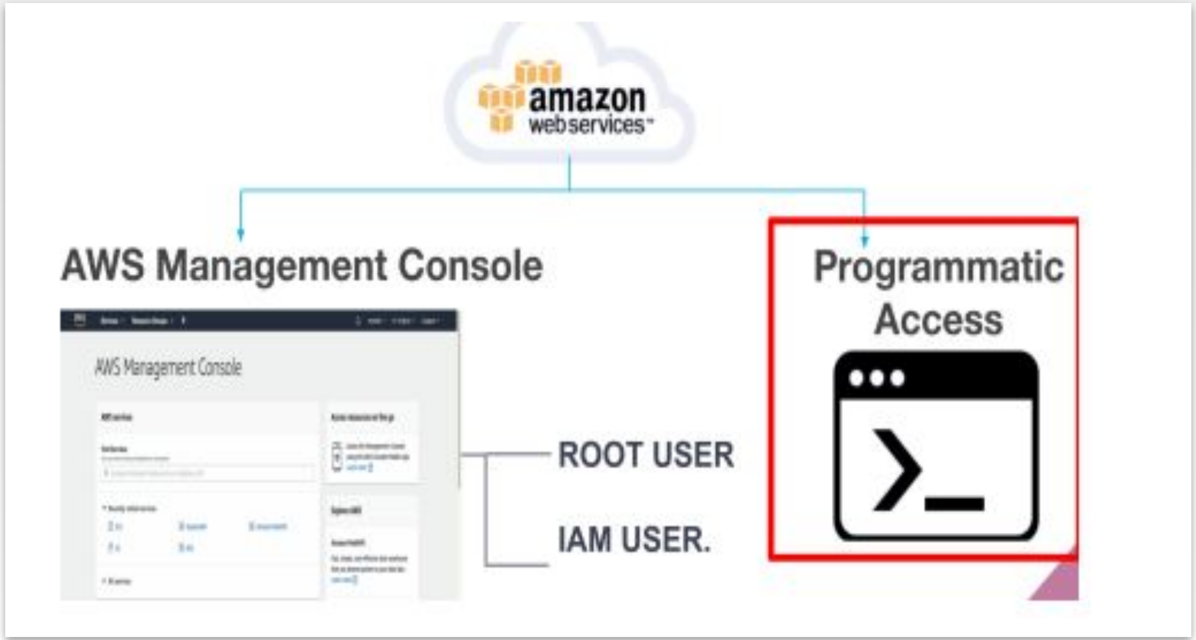
- By first creating an AWS account, you create a root user identity account that is used to log in to the AWS. This identity is called the AWS Account **Root User**.

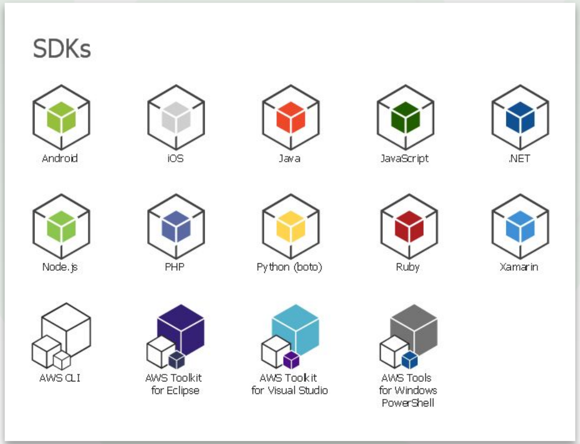
- A root user can create new **IAM users** and give them authorization for using AWS services within the account. The limit of creating new **IAM users is restricted to 5000 users per account.**

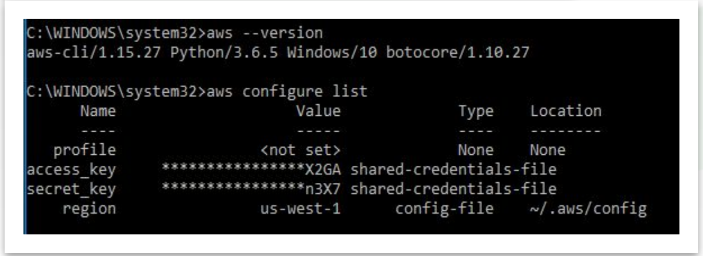
IAM

What is an IAM user & Credentials

- An **IAM user represents a person or service that interacts with AWS**. You define the user within your AWS account.
- An IAM user consists of a **name and a set of credentials**. When creating a user, you can choose to provide the user.







```
C:\WINDOWS\system32>aws --version
aws-cli/1.15.27 Python/3.6.5 Windows/10 botocore/1.10.27

C:\WINDOWS\system32>aws configure list
      Name                    Value             Type    Location
      ----                    -----             ----    --------
   profile                <not set>             None    None
access_key     ****************X2GA shared-credentials-file
secret_key     ****************n3X7 shared-credentials-file
    region                us-west-1      config-file    ~/.aws/config
```

IAM

What is an IAM Policy?

To **manage access and provide permissions** to AWS services and resources, you **create IAM policies** and **attach them to IAM users, groups, and roles**.

Most policies are stored in AWS as **JSON documents** with several policy elements.

IAM

In this policy, there are four major JSON elements: Version, Effect, Action, and Resource.

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

The **Version** element defines the version of the policy language.

The Effect element specifies whether the statement will allow or deny access. In this policy, the Effect is "Allow", which means you're providing access to a particular resource.

The Action element describes the type of action that should be allowed or denied. In the sample policy, the action is "*". This is called a wildcard, and it is used to symbolize every action inside your AWS account.

The **Resource** element specifies the object or objects that the policy statement covers. In the policy example above, the resource is also the wildcard `"*"`. This represents all resources inside your AWS console.

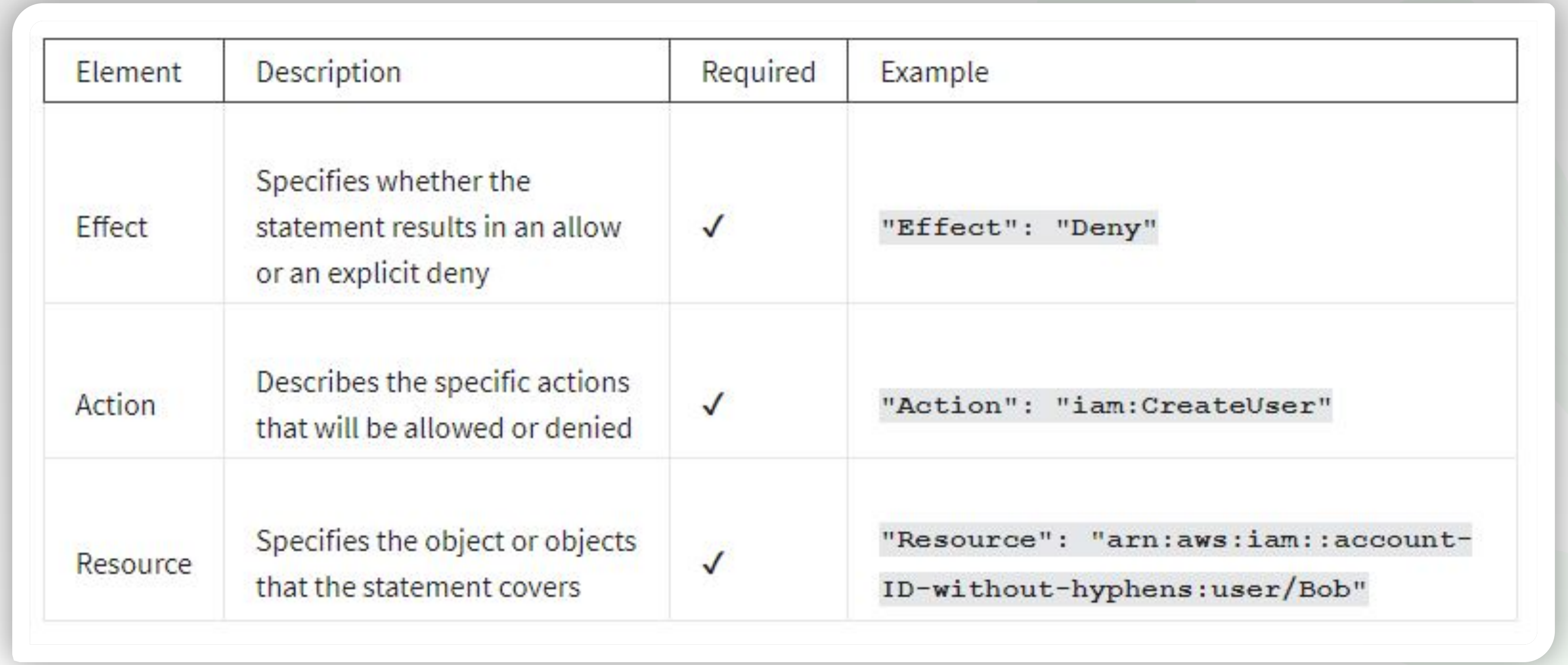
IAM

- In this policy, there are four major JSON elements: Version, Effect, Action, and Resource.

```
{ "Version": "2012-10-17",
    "Statement": [{
       "Effect": "Allow",
       "Action": ["iam: ChangePassword", "iam: GetUser"]
       "Resource":"arn:aws:iam::123456789012:user/${aws:username}"
    }]
```

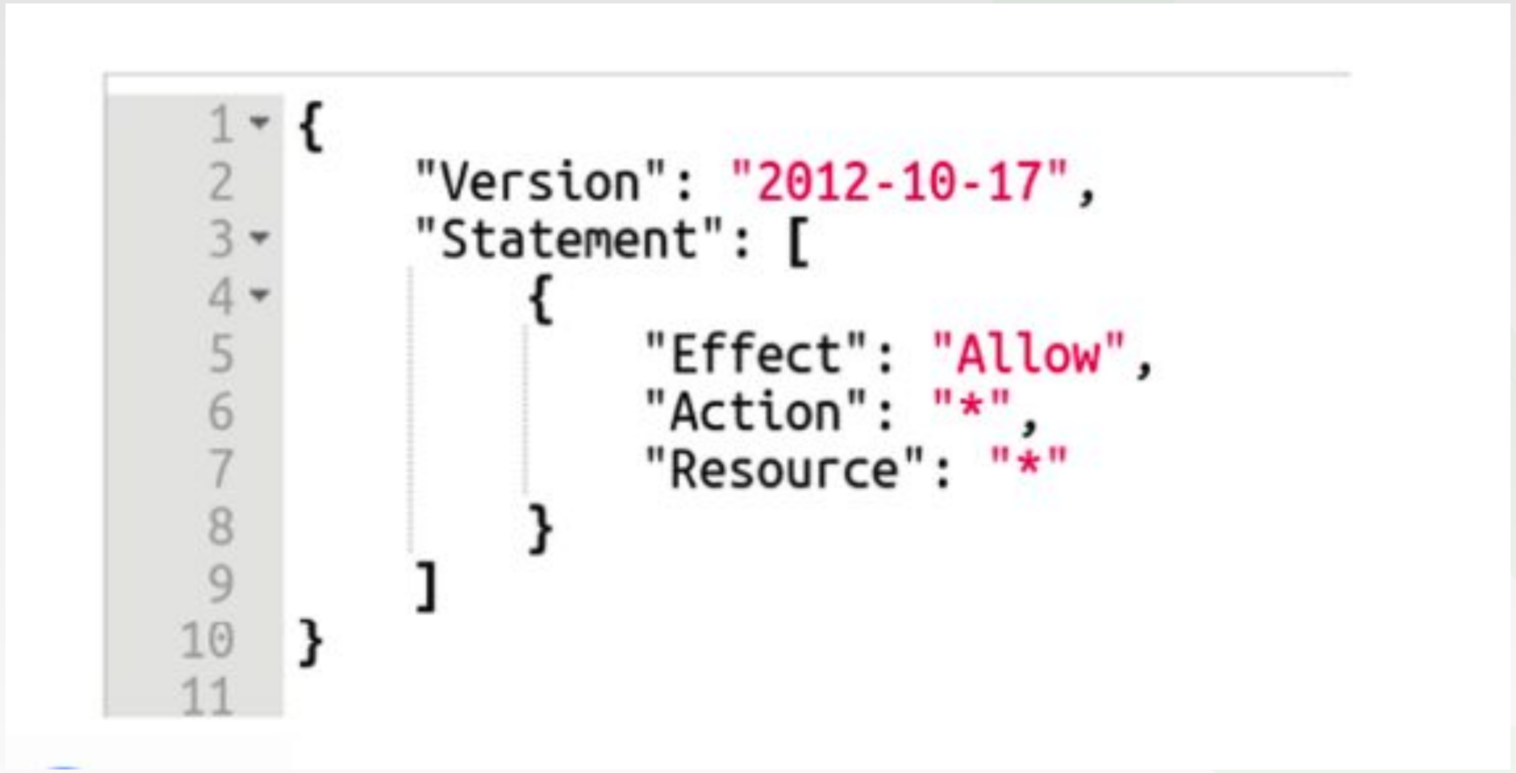
IAM

- When creating a policy, it is required to have each of the following elements inside a policy statement.



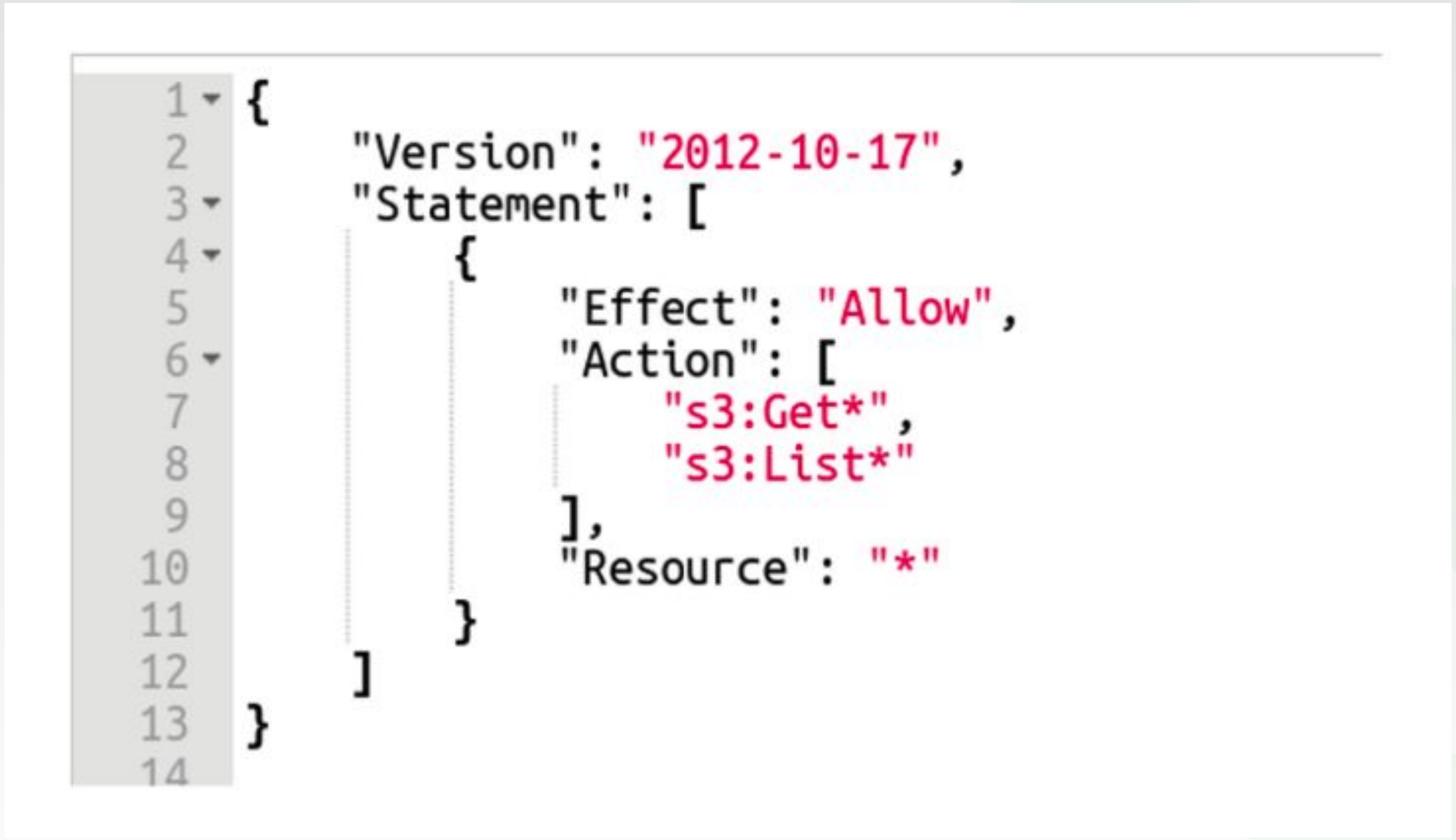
Element	Description	Required	Example
Effect	Specifies whether the statement results in an allow or an explicit deny	✓	`"Effect": "Deny"`
Action	Describes the specific actions that will be allowed or denied	✓	`"Action": "iam:CreateUser"`
Resource	Specifies the object or objects that the statement covers	✓	`"Resource": "arn:aws:iam::account-ID-without-hyphens:user/Bob"`

IAM



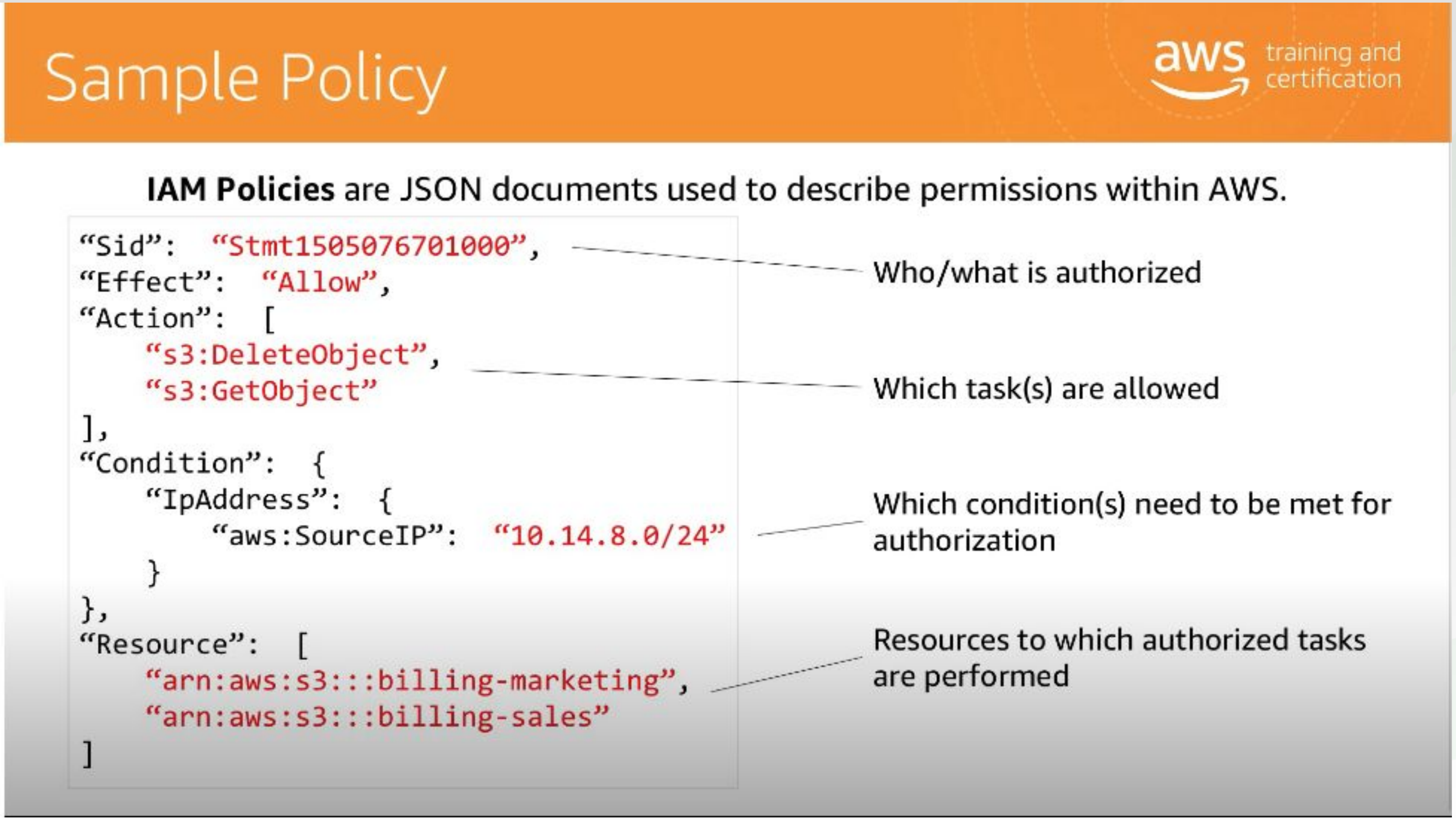
```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

IAM



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

IAM



Sample Policy

aws training and certification

IAM Policies are JSON documents used to describe permissions within AWS.

```
“Sid”:  “Stmt1505076701000”,
“Effect”:  “Allow”,
“Action”:  [
    “s3:DeleteObject”,
    “s3:GetObject”
],
“Condition”:  {
    “IpAddress”:  {
        “aws:SourceIP”:  “10.14.8.0/24”
    }
},
“Resource”:  [
    “arn:aws:s3:::billing-marketing”,
    “arn:aws:s3:::billing-sales”
]
```

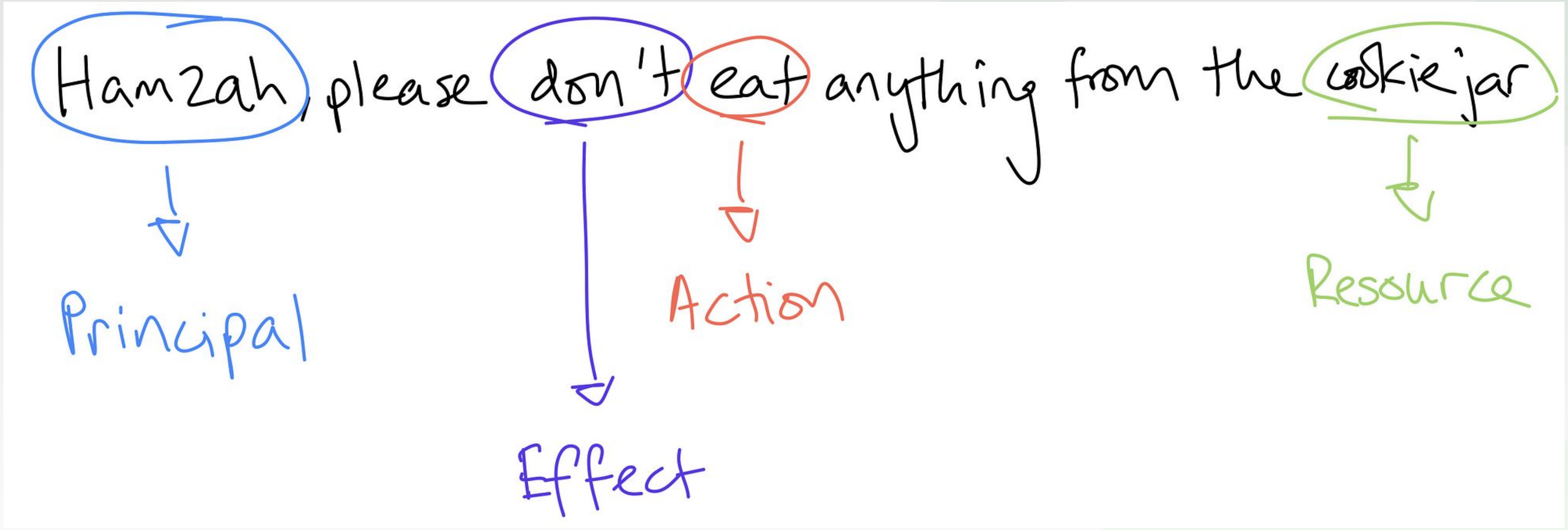
Who/what is authorized

Which task(s) are allowed

Which condition(s) need to be met for authorization

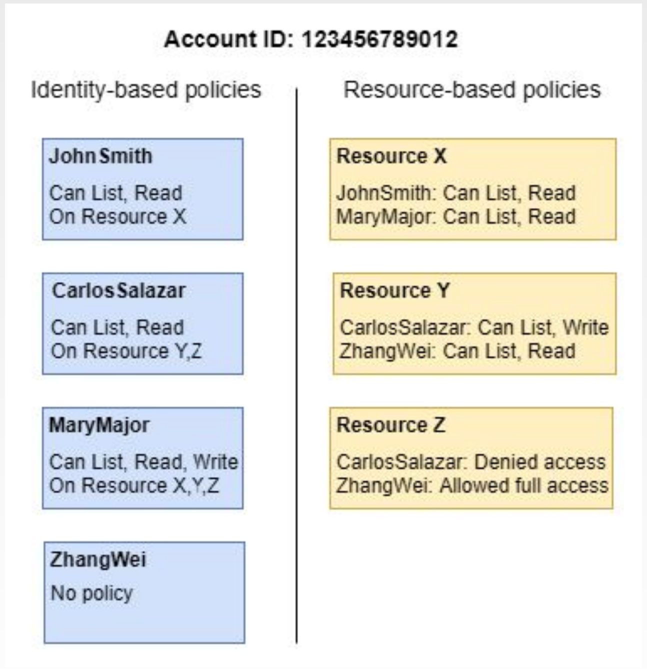
Resources to which authorized tasks are performed

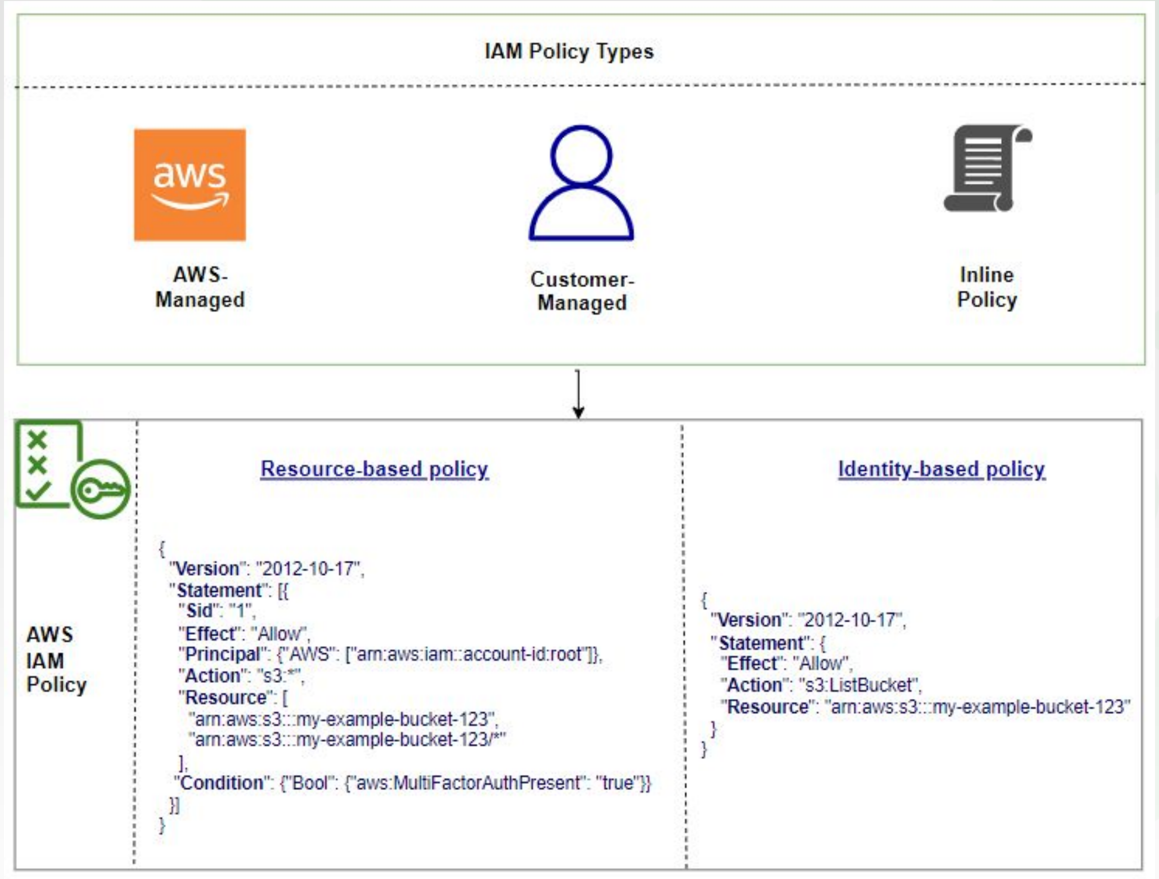
IAM


Hamzah, please don't eat anything from the cookie jar
Principal
Effect
Action
Resource

IAM

- Identity Based Policy: Attached to **users, groups or roles**
- Resource Based Policy: Attached to a **resource**, defines **permissions for a principal accessing the resource**

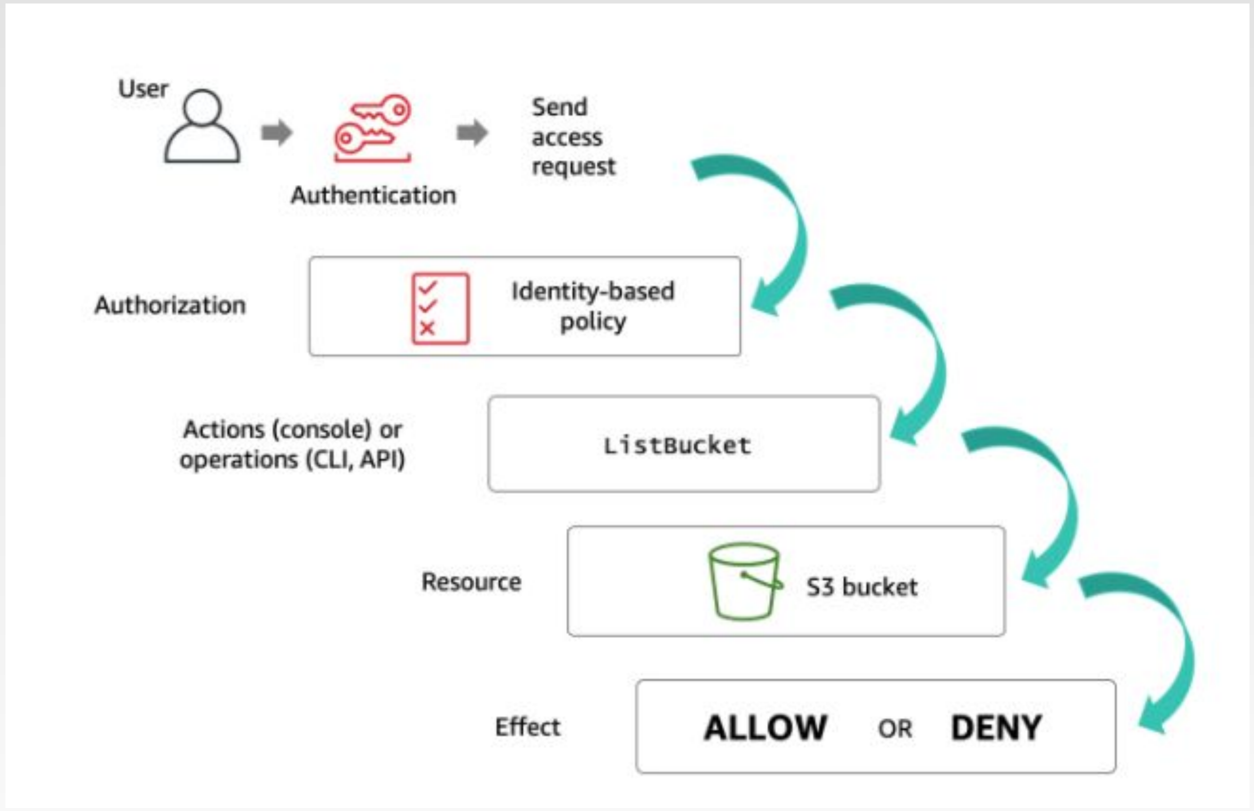


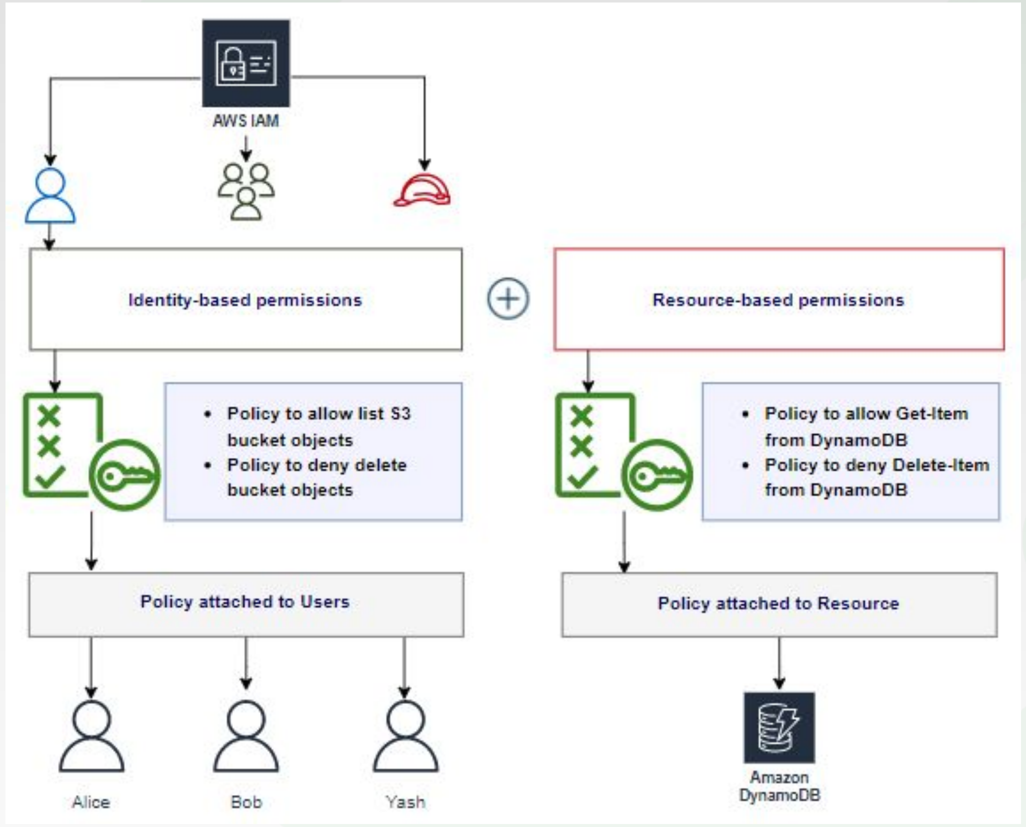


```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Sid": "1",
  "Effect": "Allow",
  "Principal": {"AWS": ["arn:aws:iam::account-id:root"]},
  "Action": "s3:*",
  "Resource": [
   "arn:aws:s3:::my-example-bucket-123",
   "arn:aws:s3:::my-example-bucket-123/*"
  ],
 "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
 }]
}
```

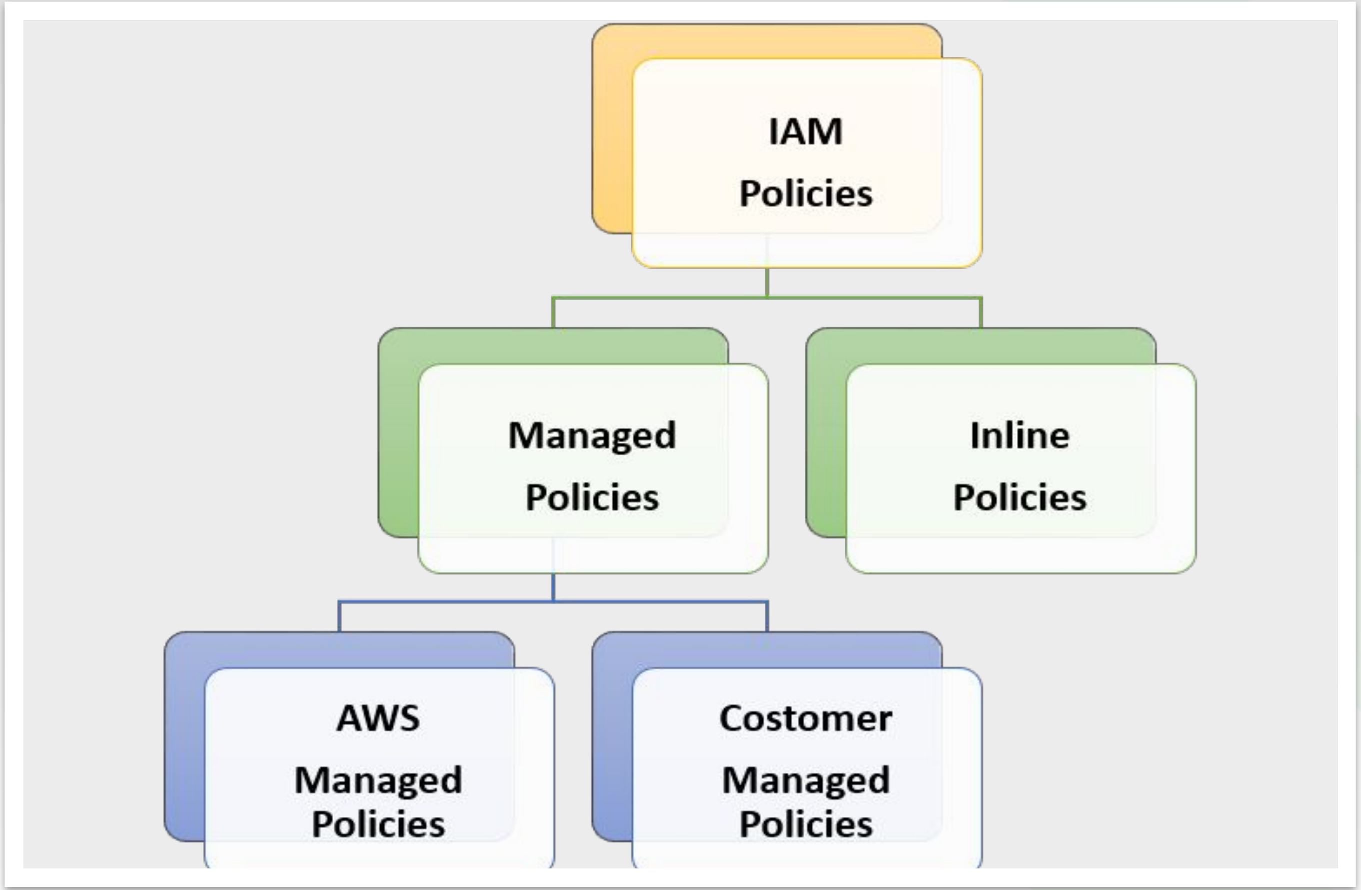
```
{
 "Version": "2012-10-17",
 "Statement": {
  "Effect": "Allow",
  "Action": "s3:ListBucket",
  "Resource": "arn:aws:s3:::my-example-bucket-123"
 }
}
```

IAM


User
Authentication
Send access request
Authorization
Identity-based policy
Actions (console) or operations (CLI, API)
ListBucket
Resource
S3 bucket
Effect
ALLOW OR DENY

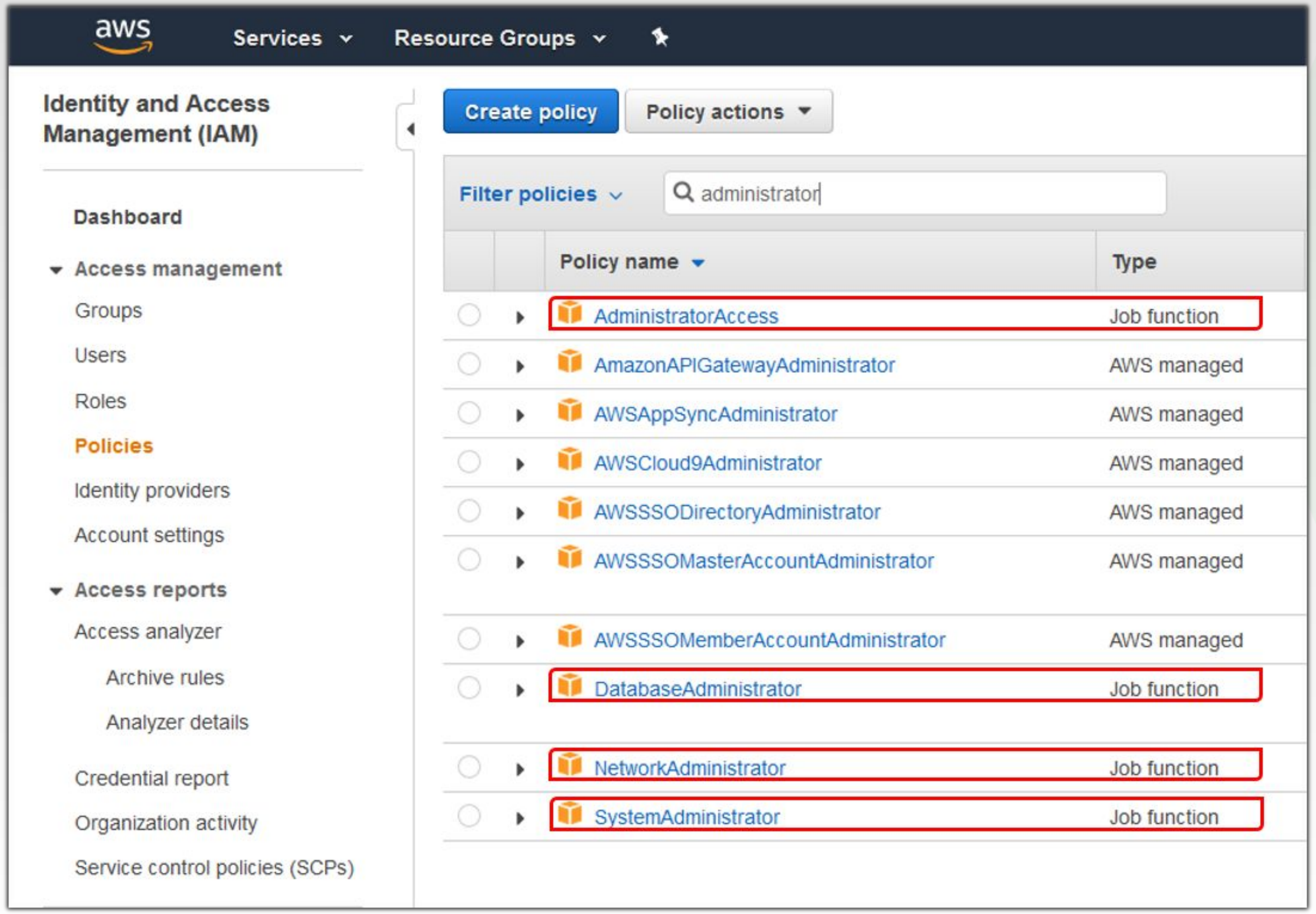

AWS IAM
Identity-based permissions
Resource-based permissions
Policy to allow list S3 bucket objects
Policy to deny delete bucket objects
Policy to allow Get-Item from DynamoDB
Policy to deny Delete-Item from DynamoDB
Policy attached to Users
Policy attached to Resource
Alice
Bob
Yash
Amazon DynamoDB

IAM Policy Types



Job Function Policies

Job Function Policies

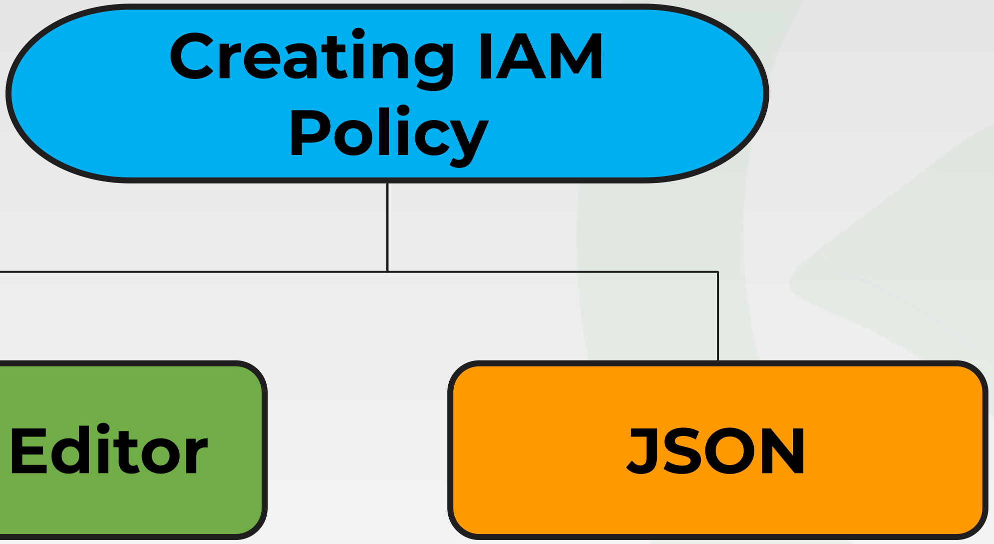


Policy name	Type
AdministratorAccess	Job function
AmazonAPIGatewayAdministrator	AWS managed
AWSAppSyncAdministrator	AWS managed
AWSCloud9Administrator	AWS managed
AWSSSODirectoryAdministrator	AWS managed
AWSSSOMasterAccountAdministrator	AWS managed
AWSSSOMemberAccountAdministrator	AWS managed
DatabaseAdministrator	Job function
NetworkAdministrator	Job function
SystemAdministrator	Job function

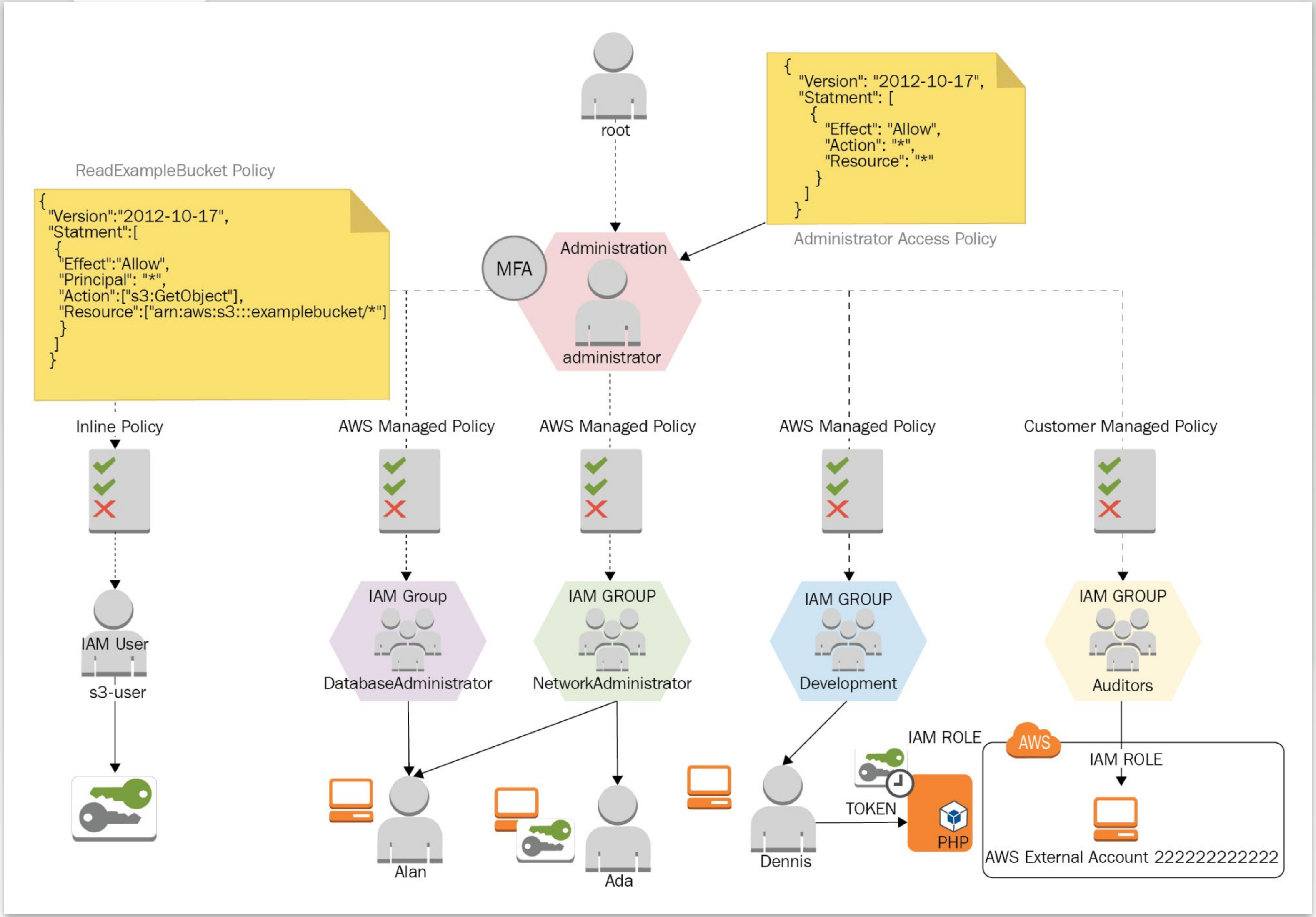
Managed policies in job function status are listed below:

- Administrator
- Billing
- Database Administrator
- Data Scientist
- Developer Power User
- Network Administrator
- Security Auditor
- Support User
- System Administrator
- View-Only User

Creating IAM Policies



Designing IAM Groups



ReadExampleBucket Policy

```
{
 "Version":"2012-10-17",
 "Statment":[
  {
   "Effect":"Allow",
   "Principal": "*",
   "Action":["s3:GetObject"],
   "Resource":["arn:aws:s3:::examplebucket/*"]
  }
 ]
}
```

```
{
  "Version": "2012-10-17",
  "Statment": [
    {
      "Effect": "Allow",
      "Action": "*",
      "Resource": "*"
    }
  ]
}
```

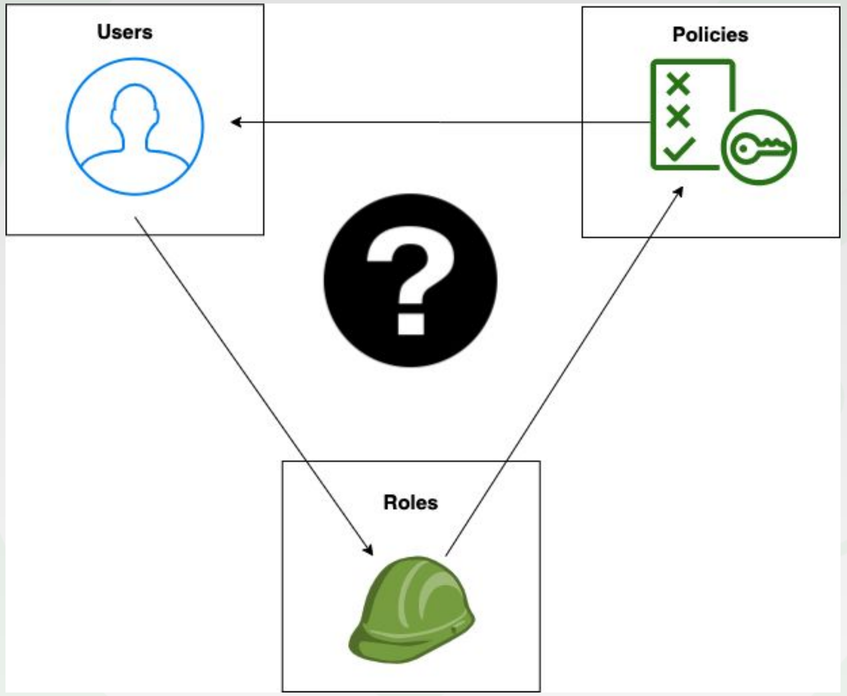
Administrator Access Policy

1. • Create IAM Groups as many as you need (max=300).
2. • Attach policies to the groups. (One or more managed/inline policies)
3. • If not, create IAM users for groups.
4. • Assign users to the groups.

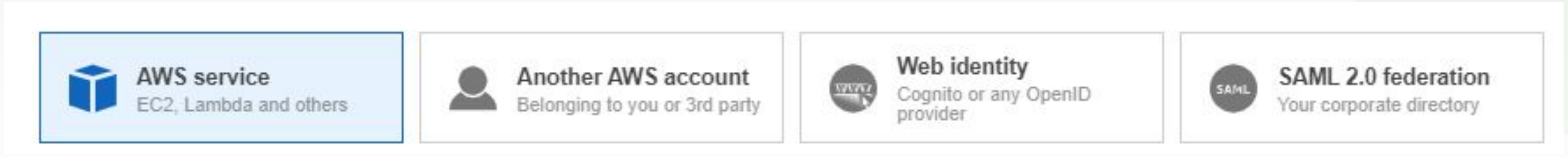
IAM Roles

Role

- An **IAM role, similar to an IAM user, is an IAM identity that has specific permissions** that you can create in your account.
- It **tells which identity can access which AWS resources**.



- **Who can assume an IAM Role ?**



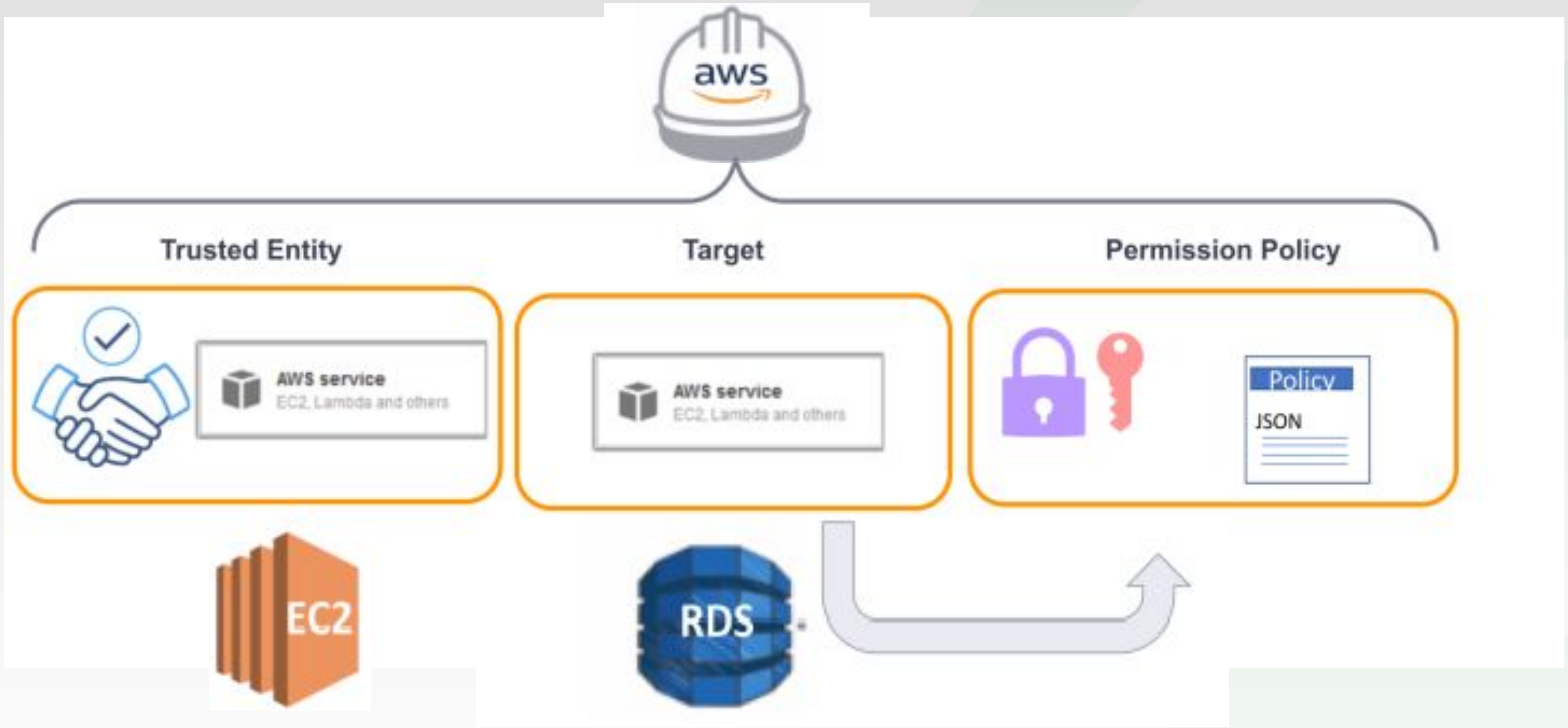
AWS service
EC2, Lambda and others

Another AWS account
Belonging to you or 3rd party

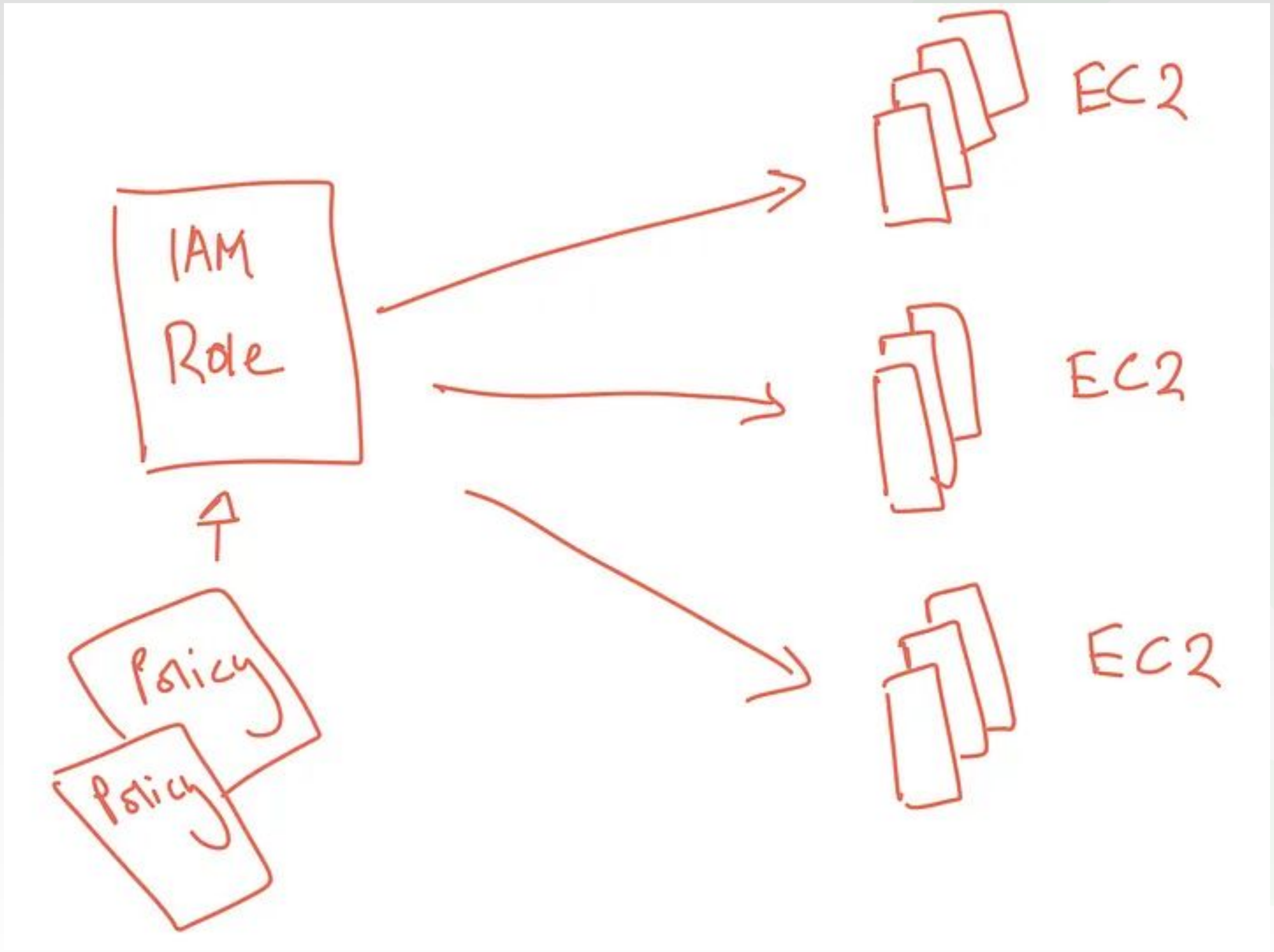
Web identity
Cognito or any OpenID provider

SAML 2.0 federation
Your corporate directory

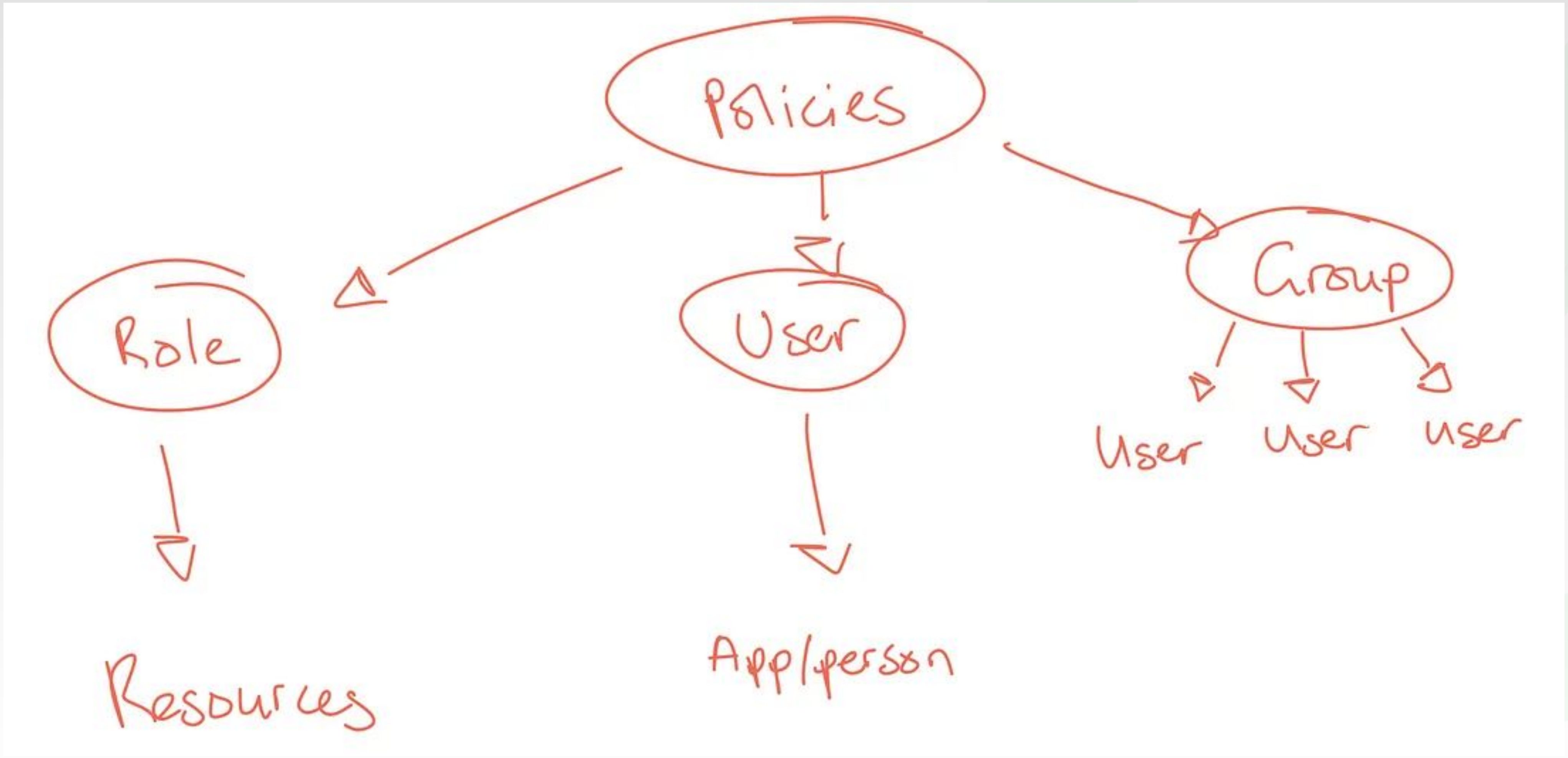
Anatomy of Role



Anatomy of Role



Anatomy of Role



IAM Best Practices

- Unless **Allowed**, every action is **Denied** by default.
- **Lock away root user** account
- Create **individual IAM user accounts**
- Use **Groups to assign permissions to IAM users**
- Grant **least privilege**
- Configure **a strong password policy for users**
- Use **MFA**
- Use **roles for applications that run on Amazon EC2 instances**
- **Do not share access keys**
- **Rotate credentials regularly**
- **Remove unnecessary users or credentials**
- Use **policy conditions for extra security**

IAM

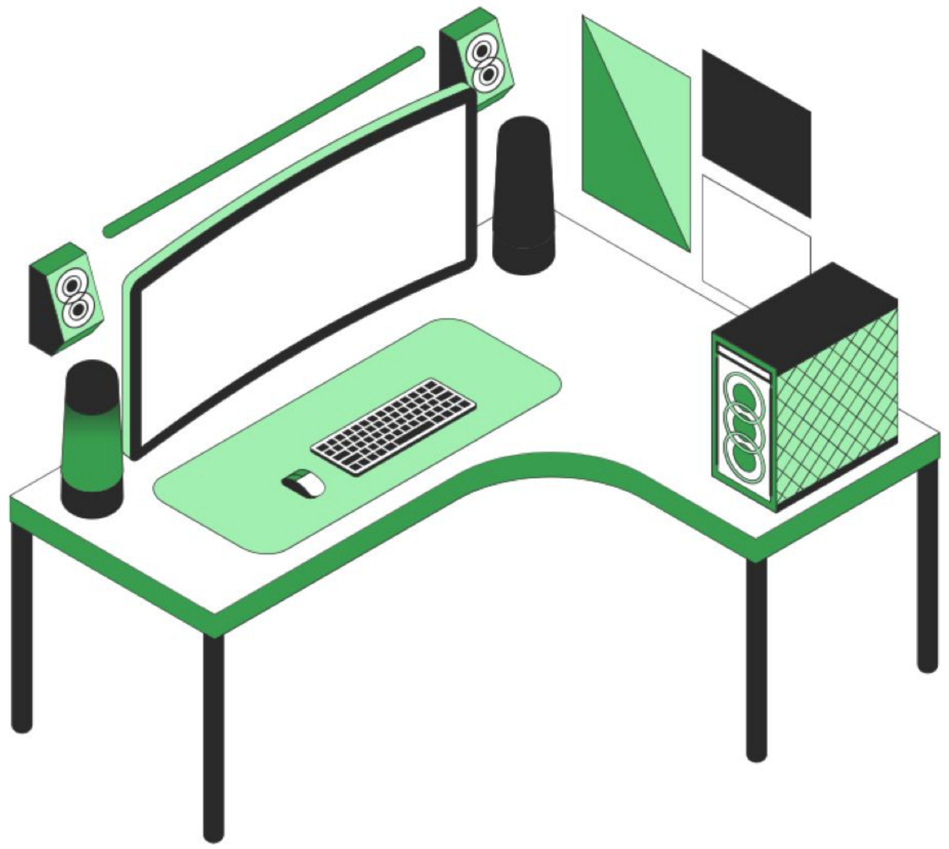
Certification Test Cases

1 A Developer is using an EC2 instance to work with AWS service DynamoDB.
How should you manage permissions for him/her the best way?
Attach Role to EC2 instance

2 A company just met with AWS by creating the first user account. They need to assign permissions to users based on job function. How can you manage permissions the best way?
Create groups based on jobs. Assign users to groups.

3 A developer needs to make API calls from AWS CLI.
Tell/show him/her what to do.
Use Access Keys

4 A solutions architect needs to restrict access to an AWS service based on source IP.
What should he/she do?
Add Conditions in Policy



Do you have any questions?

Send it to us! We hope you learned something new.