aws professional services

Identity & Access Management Overview

Agenda

- The AWS APIs
- AWS Identity and Access Management (IAM)

Goals

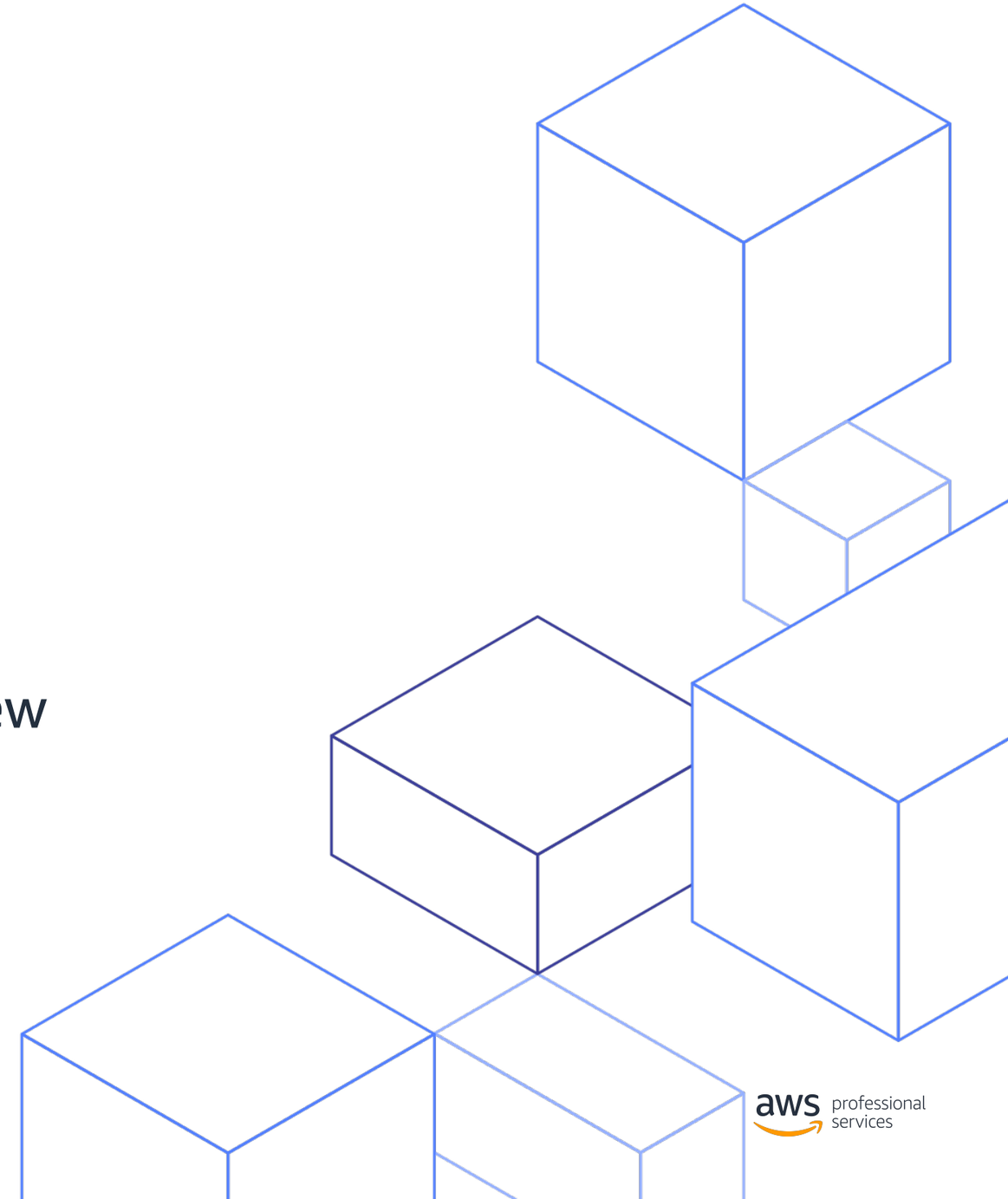
- Learn the foundations of AWS Identity & Access Management
- Understand when and where to use AWS IAM

Outcomes

- Decision on the AWS access model (IAM users vs federated access)
- Decision on a root credentials management procedure to be implemented

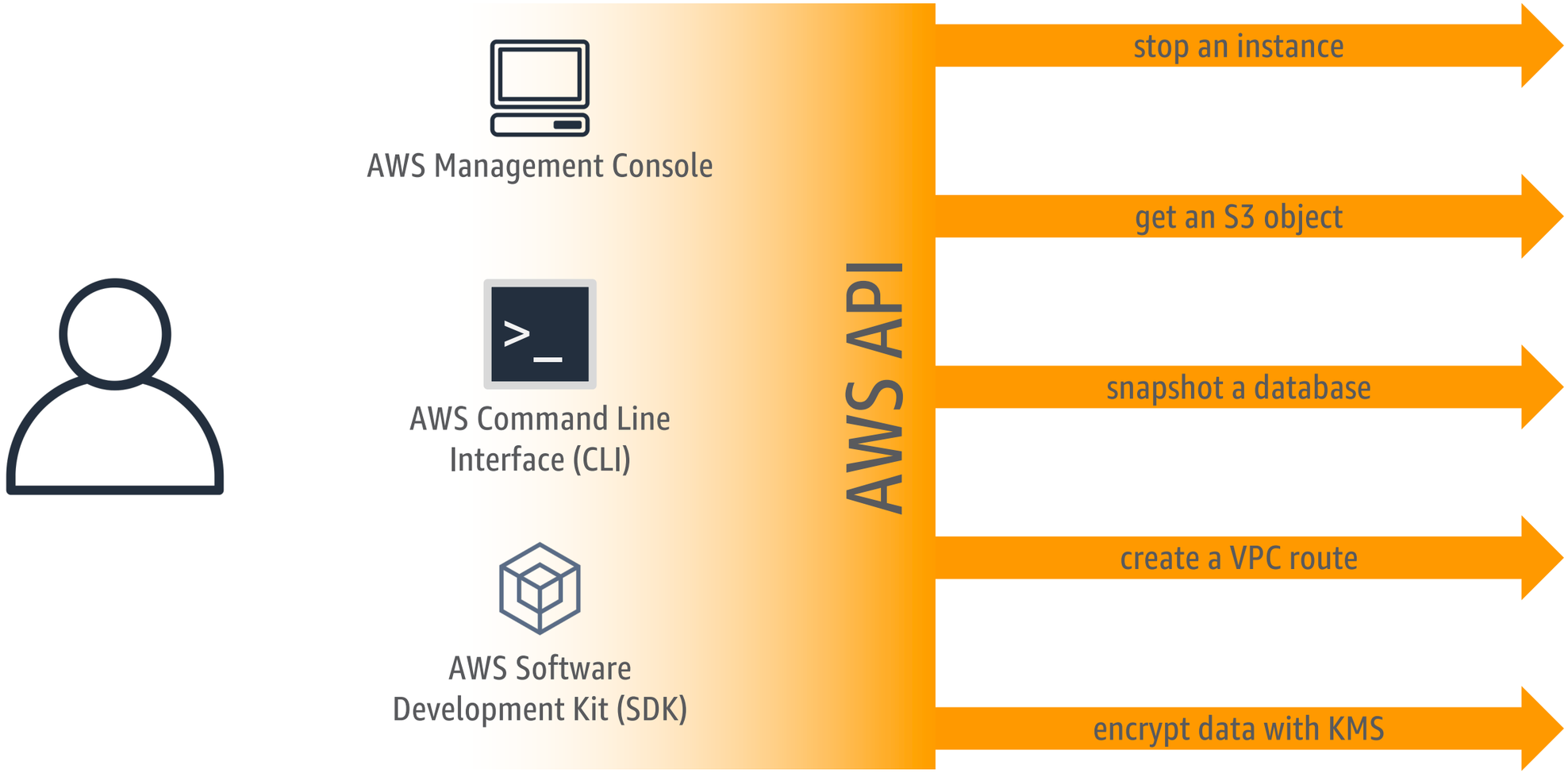
AWS API Calls

AWS Identity & Access Management Overview

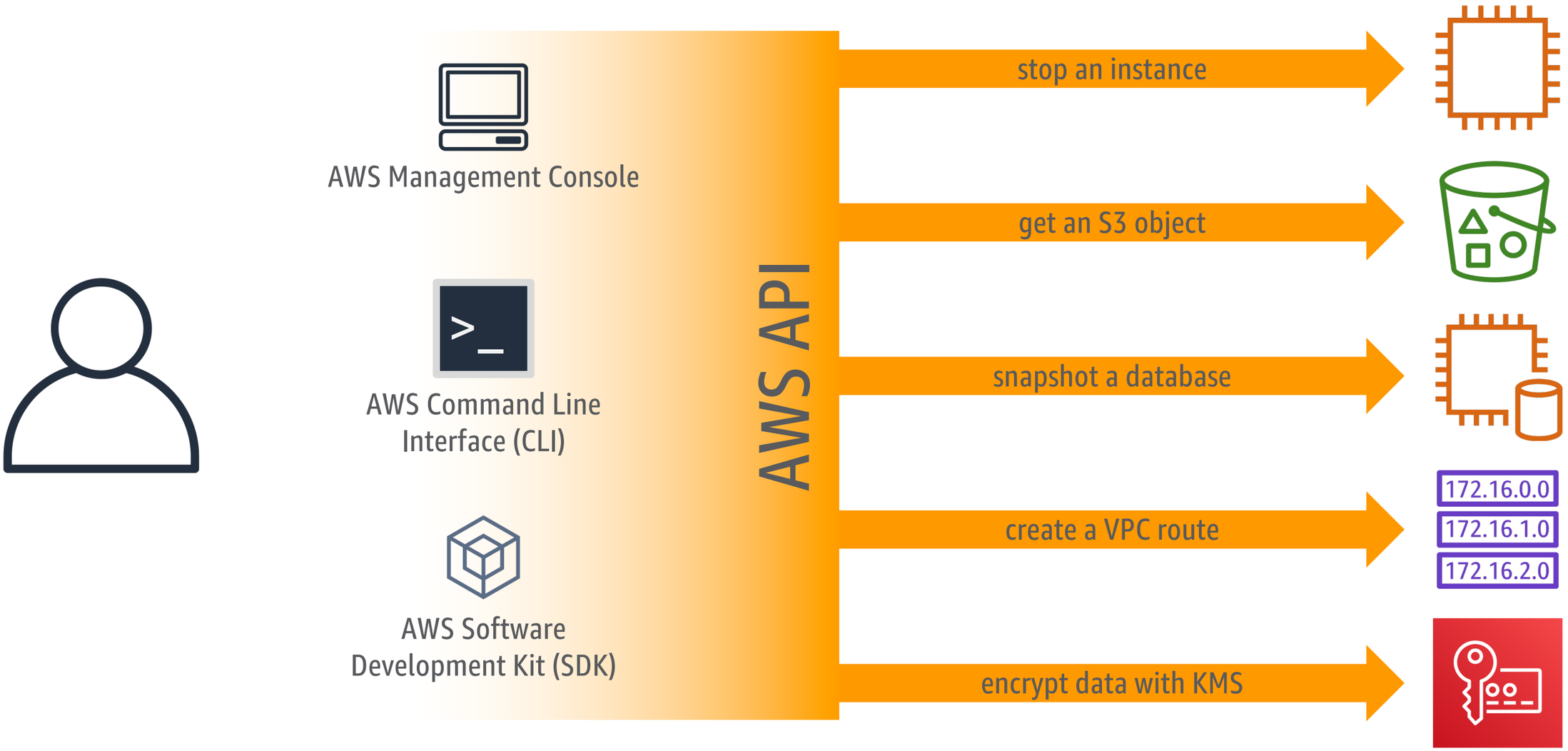


aws professional services

Making API Calls

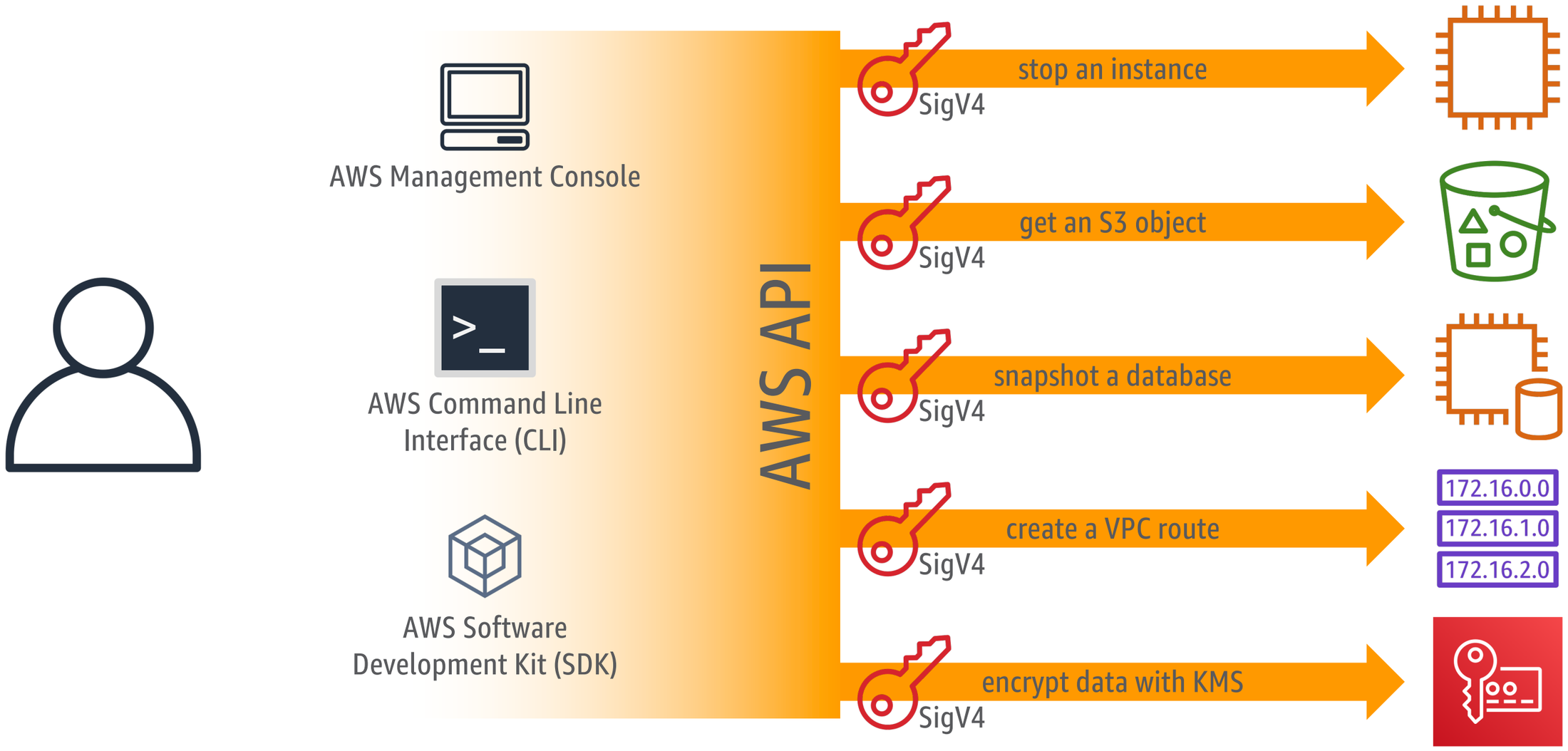


Making API Calls



aws professional services

Making API Calls



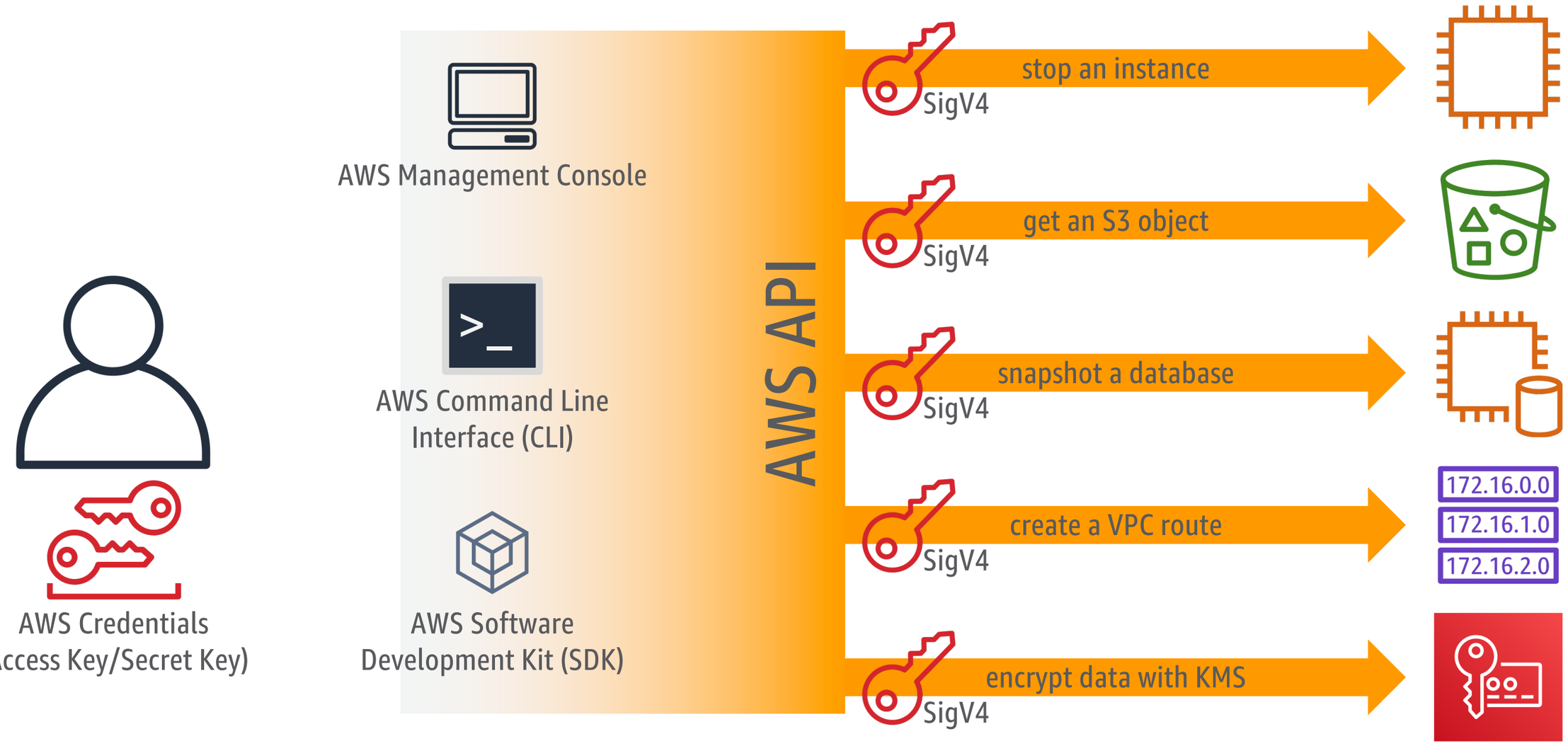
aws professional services

AWS Signature Version 4

- AWS Signature Version 4 is the process to add authentication information to AWS requests.
    - The AWS SDKs or CLI tools will construct, sign, and send requests for you, with the **access keys** you provide.
    - If you are constructing AWS API requests yourself, you will have to include code to sign the requests.

More information can be found here:
http://docs.aws.amazon.com/general/latest/gr/signature-version-4.html

Making API Calls



aws professional services

AWS Identity & Access Management

AWS Identity & Access Management Overview

aws professional services

AWS IAM Concepts

AWS Accounts

- Strong separation of duties
- Consolidate billing into a single account
- Plan your account strategy in advance (e.g. per function, per criticality, etc.)

AWS IAM Concepts

AWS Resources

Defined uniquely by an **Amazon Resource Name** (ARN)

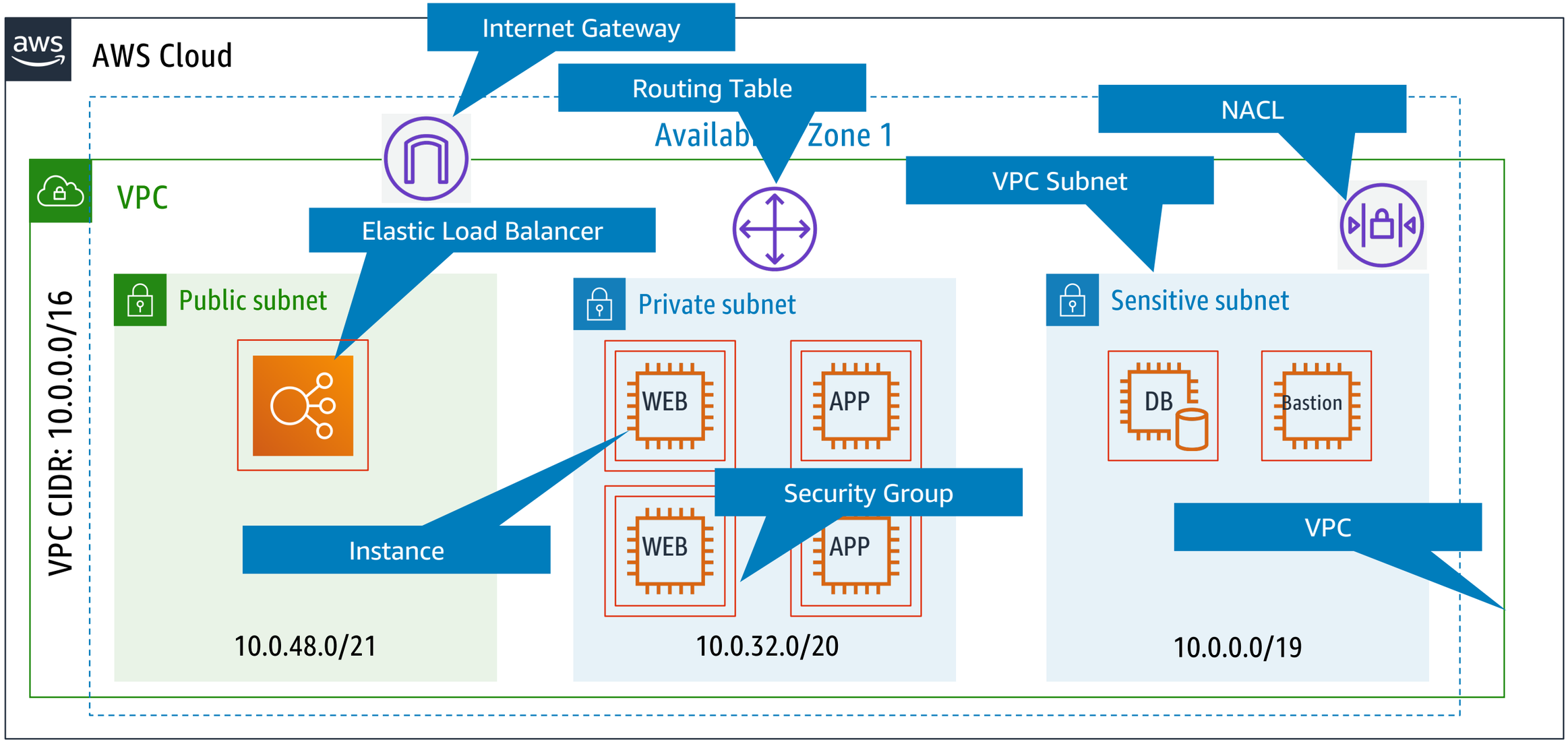
Ex: EC2 instance, DynamoDB table, IAM user, etc.

Not: OS installed on EC2, data inside an EBS volume, etc.

```
arn:aws:service:region:account:resource
```

```
<!- Amazon EC2 instance -->
arn:aws:ec2:us-east-1:123456789012:instance/i-1a2b3c4d

<!-- Amazon RDS tag -->
arn:aws:rds:eu-west-1:123456789012:db:mysql-db

<!-- Amazon S3 all objects in a bucket -->
arn:aws:s3:::my_corporate_bucket/*
```

AWS IAM Concepts



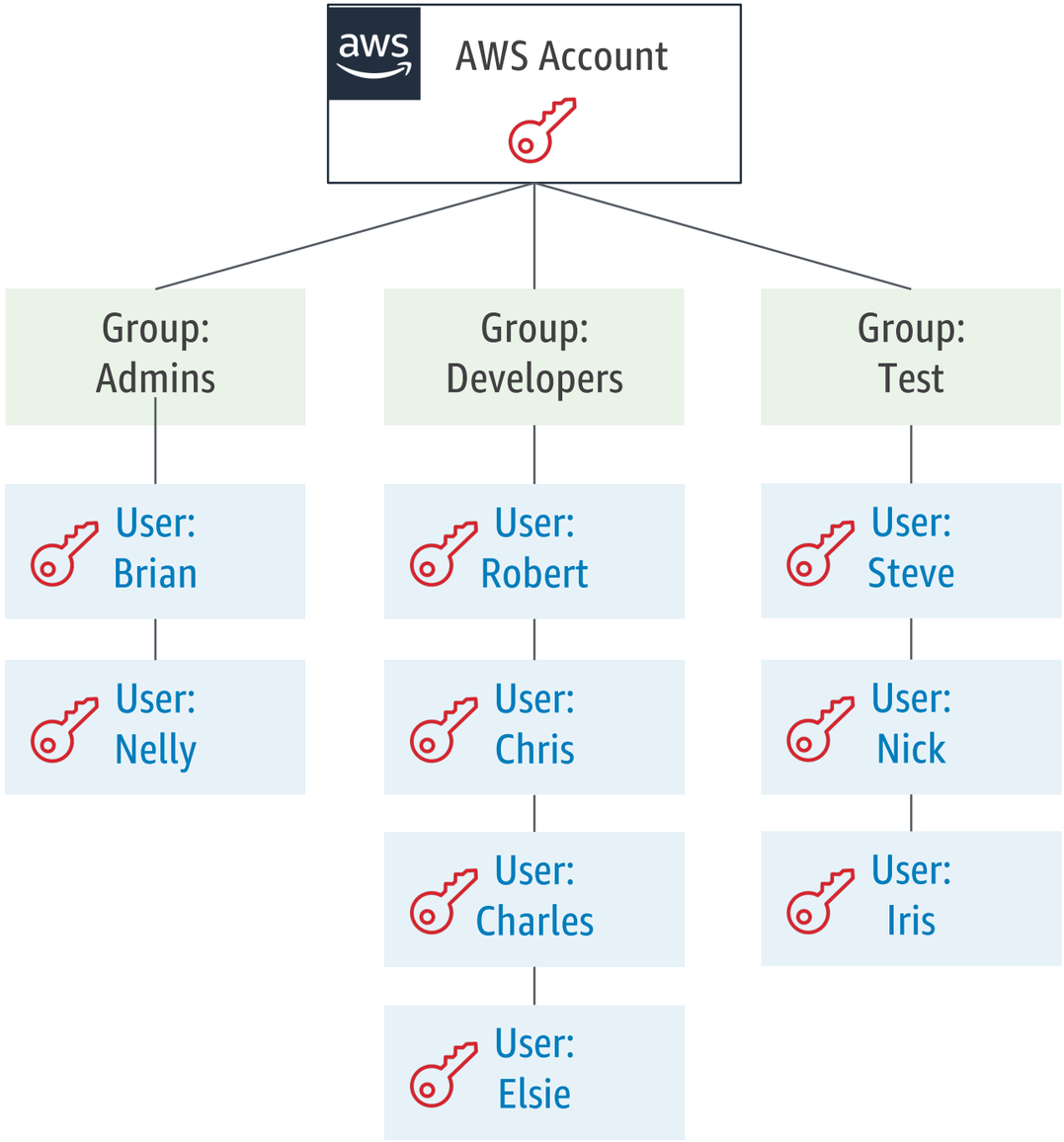
aws professional services

AWS IAM Concepts - Resources

```
<-- S3 Bucket -->
"Resource":"arn:aws:s3:::my_corporate_bucket/*"

<-- SQS queue-->
"Resource":"arn:aws:sqs:us-west-2:123456789012:queue1"

<-- Multiple DynamoDB tables -->
"Resource":["arn:aws:dynamodb:us-west-2:123456789012:table/books_table",
          "arn:aws:dynamodb:us-west-2:123456789012:table/magazines_table"]

<-- All EC2 instances for an account in a region -->
"Resource": "arn:aws:ec2:us-east-1:123456789012:instance/*"
```

AWS IAM Concepts

- A username for each user
- Groups to manage multiple users
- Centralised access control
- Optional provisions:
    - Password for console access
    - Policies to control access
    - Use Access Key to sign API calls
    - Multifactor Authentication



AWS IAM Concepts - Roles

- Set of permissions granted to a trusted entity
- Assumed by IAM users, applications or AWS services like EC2
    - Use case:
    - Cross-services
    - Temporary access
    - Cross-account
    - Federation
- Benefits
    - Security: no sharing of secrets
    - Control: revoke access anytime

AWS IAM Concepts - Roles: Breakdown of IAM role use cases

Grant Access to AWS Services

Federate Identities into AWS

Enable Cross Account Access

AWS IAM Concepts - Roles: Grant Access to AWS Services

Roles for EC2

Best Practice!

- Enable applications running on EC2 to make AWS API calls.
- Manage security credentials for you.
- Rotates credentials automatically.
- Easy to attach and detach an IAM role to a new or existing instance.
- Add/update permissions without logging into the instance.

Service Roles

- Grant AWS services access to perform actions on your behalf.
- Control permissions that service can run.
- Track actions AWS services perform on your behalf using CloudTrail.
- Examples: AWS Config, AWS OpsWorks, and AWS Directory Service.

Service-linked Roles

- Grant AWS services access to perform actions on your behalf.
- *Pre-defined* permissions that the linked service requires.
- Protect you from inadvertently deleting a role.
- Track actions AWS services perform on your behalf using CloudTrail.
- Examples: Amazon Lex

AWS IAM Concepts - Roles: Federate Identities into AWS

If your organization has its own identity system, create an IAM identity provider entity to establish trust between your AWS account and the IdP. Use IAM roles to grant permissions to your users.

SAML IDP

- Enable users to log into the AWS Management Console
- Call the AWS APIs without you having to create an IAM user for everyone in your organization.
- Enable federated single sign-on (SSO) to your AWS accounts.

OpenID Connect Provider

- Enable your application users to sign in using an identity provider (IdP), such as Login with Amazon, Facebook, Google, or any other OpenID Connect (OIDC) compatible IdPs.
- Helps you keep your AWS account secure, because you don't have to embed and distribute long-term security credentials in your application

AWS IAM Concepts - Roles: Enable Cross Account Access

Assume Role

Best Practice!

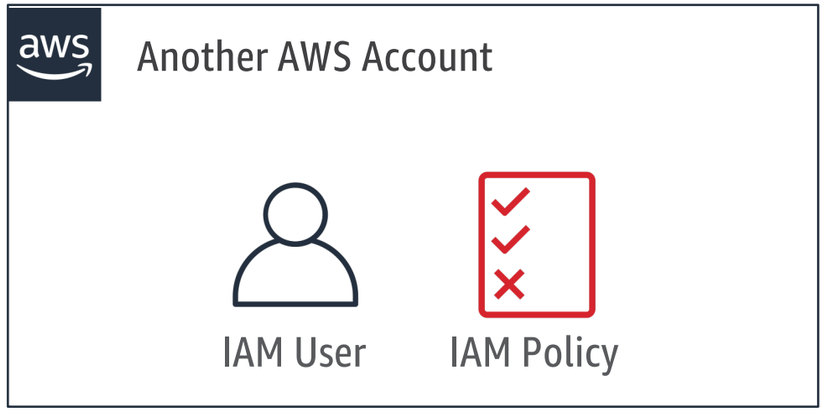
- Configure the AWS Command Line Interface to use a role by creating a profile.
- Configure your application or script to assume a role for temporary credentials use to call AWS APIs.

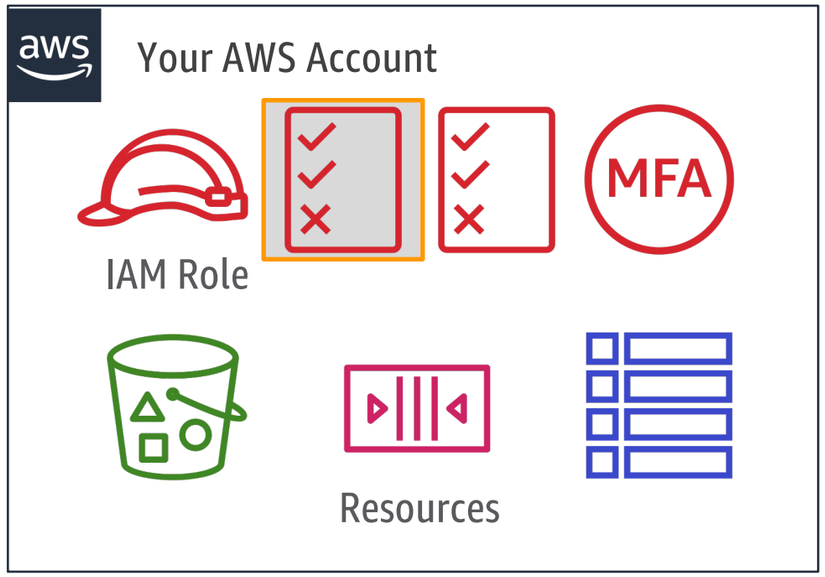
Switch Role in the Console

- Sign in to the console as an IAM user or via federated Single Sign-On and then switch the console to manage another account without having to enter (or remember) another user name and password.

aws professional services

AWS IAM Concepts - Roles





Create an **IAM Role**

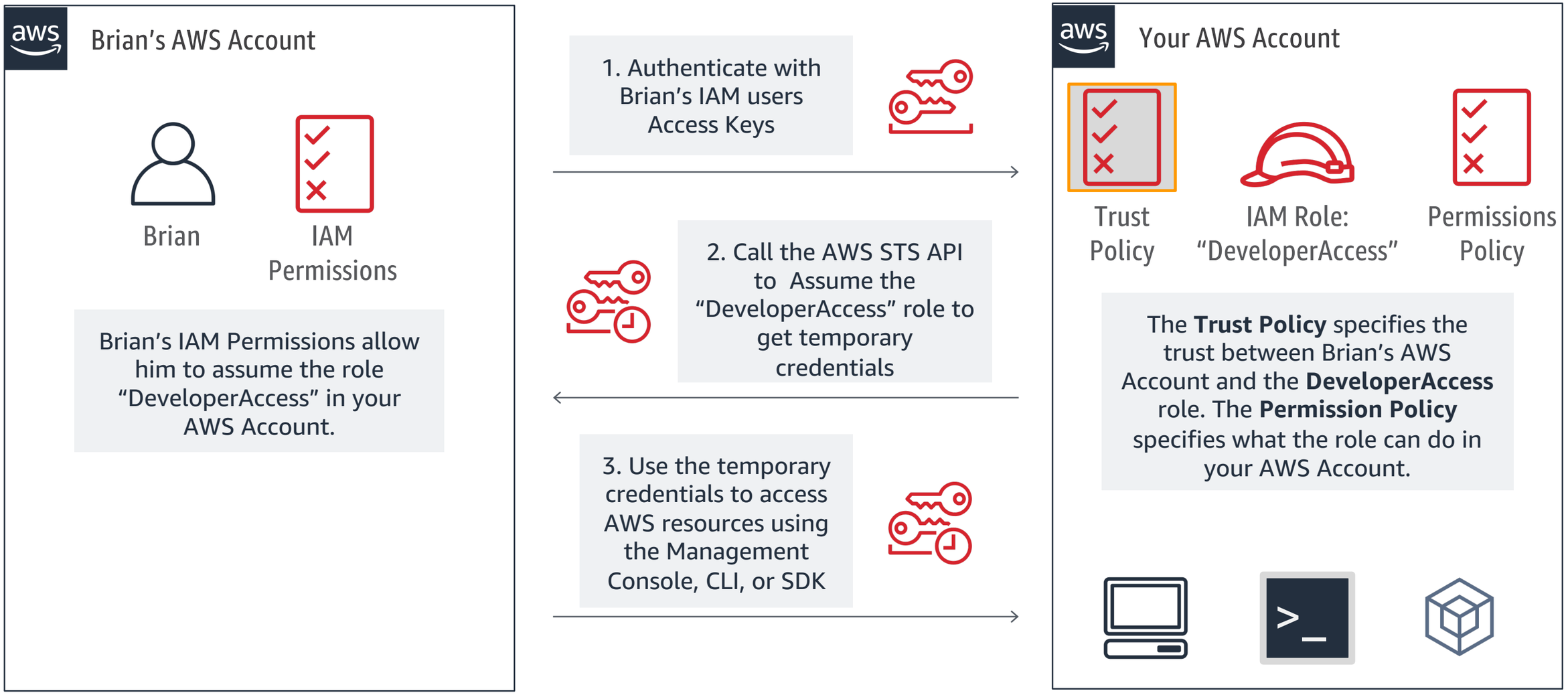
Trust Policy: Trust another AWS Account

Permission Policy: Grant Permissions

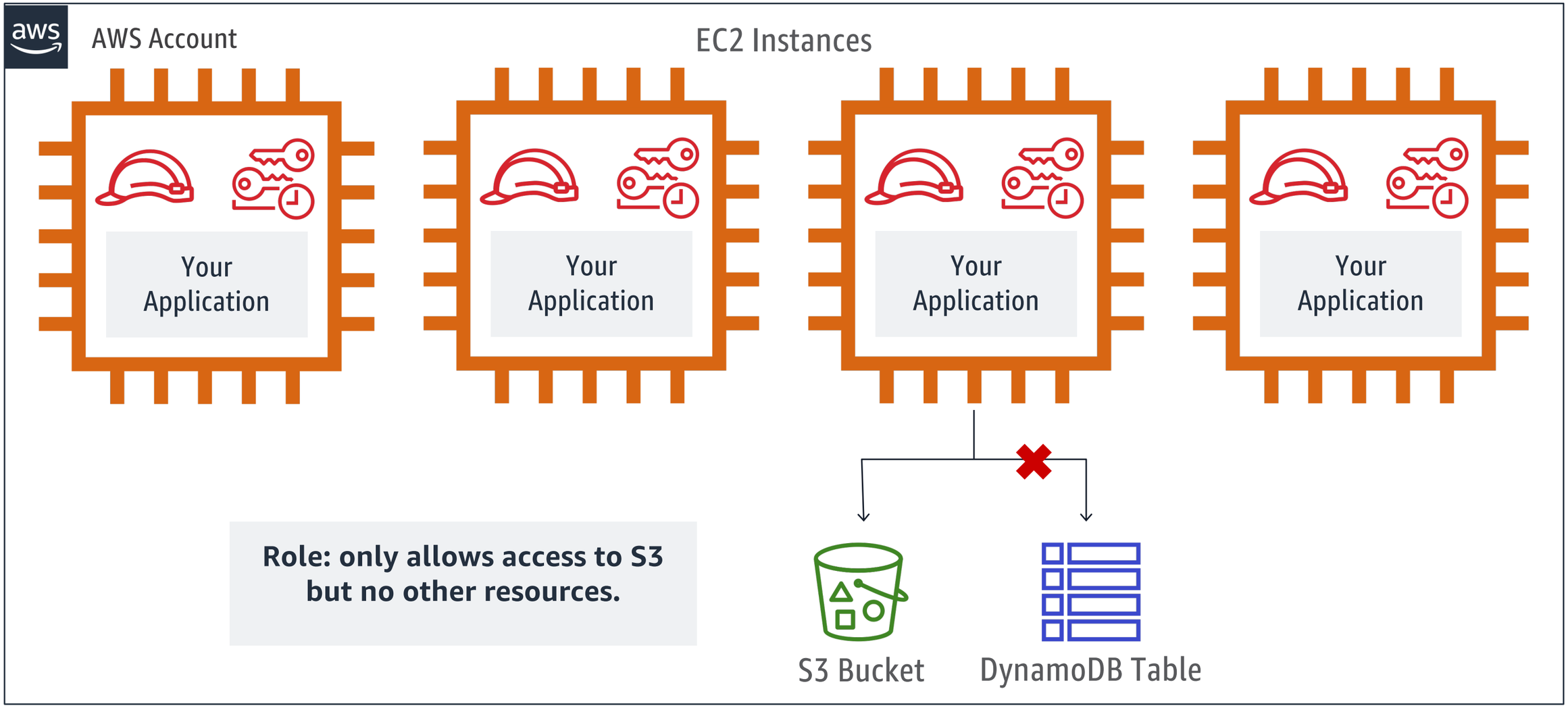
Another account's **IAM user** can assume the role if his **Permission Policy** allows him to

Use **MFA** to protect role assumption for privileged access

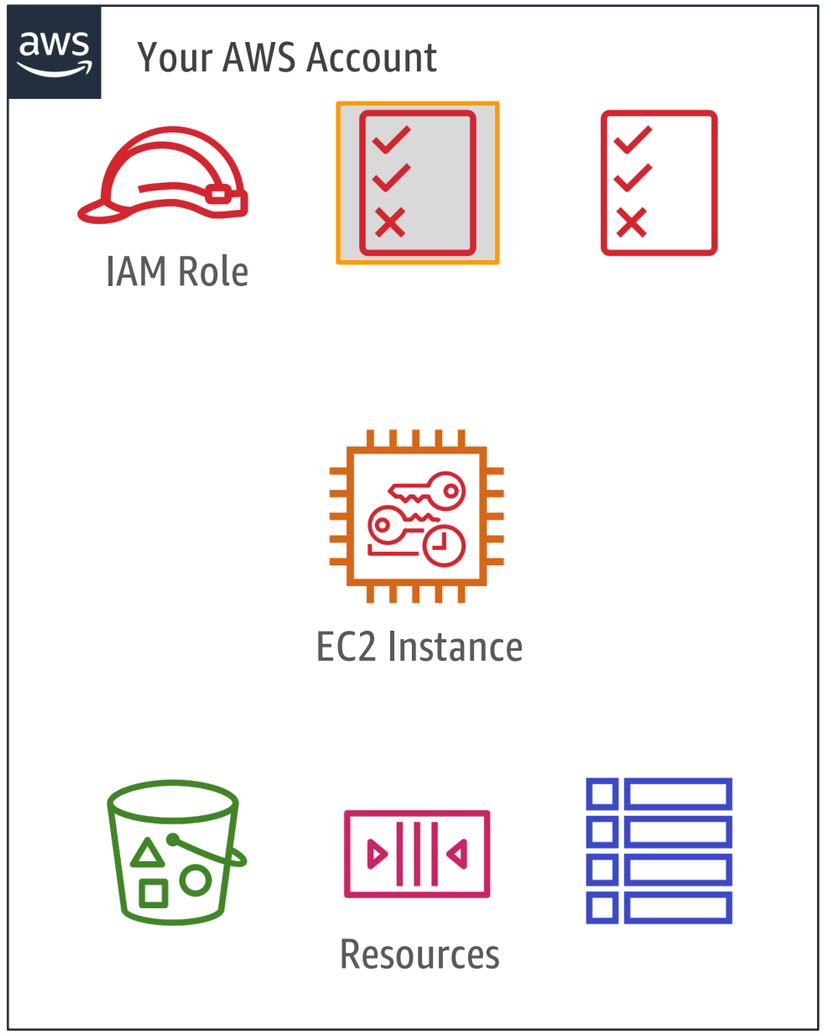
AWS IAM Concepts - Roles



AWS IAM Concepts - Roles



AWS IAM Concepts - Roles



Create an **IAM Role**

Trust Policy: Allow EC2 instances to assume this role

Permission Policy: Grant Permissions to resources

Launch an **EC2 Instance** with the **IAM Role** or attach the **IAM Role** to an existing **EC2 Instance**

Temporary Credentials are available on the EC2 Instance through the metadata URL.

aws professional services

AWS IAM Concepts - Roles: Best practices

Trust: Validate your trust policies to ensure only appropriate entities can assume the role.

Permissions: Set granular permissions on your IAM roles an use scope down policies to further control access.

Tracking and Monitoring: Track cross account access using AWS CloudTrail and IAM Access Analyzer

Analogy

Account Owner ID (Root Account)

- Access to all subscribed services.
- Access to billing.
- Credentials can’t be disabled.
- Access to console and APIs.

DO NOT USE after initial set-up

Door Key
Keys to the Kingdom

IAM Users

- Access to specific services.
- Access to console and/or APIs.
- Credentials can be revoked and invalidated.

Employee ID Badge

Temporary Security Credentials / IAM Roles

- Access to specific services.
- Access to console and/or APIs.

Hotel Key

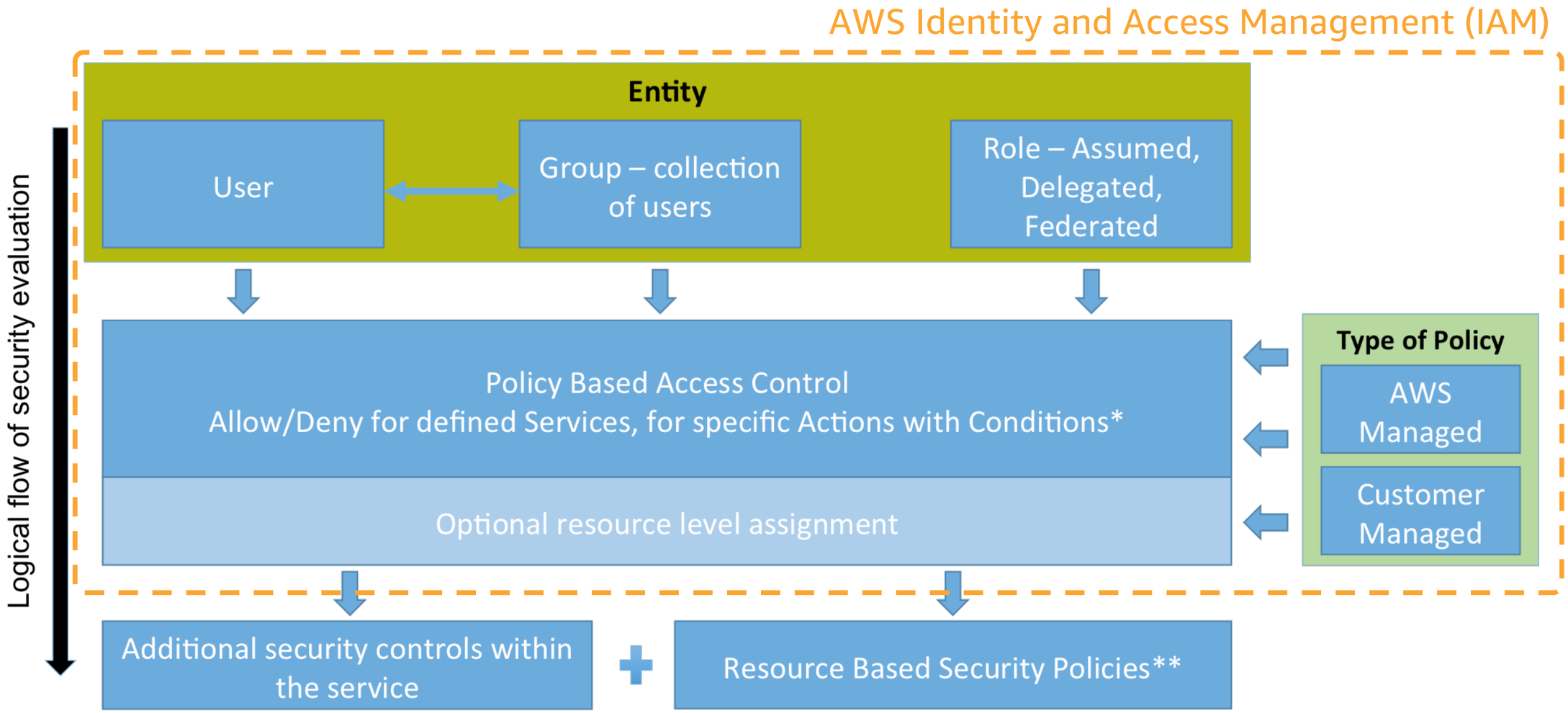
aws professional services

AWS IAM Concepts

- Permissions
  - Authorize (or not) to perform an action
  - Use Policies to grant permission
- Policy
  - Set of instructions which define permission
  - Can be simple or very granular

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

Secure Access with IAM



* Service specific conditions also available, e.g. EC2, RDS, KMS, Elastic Beanstalk, etc.
** Available for S3, SQS, SNS, KMS, VPC Endpoint

aws professional services

Questions