

Cloud Computing Course Content

Farid Afzali, Ph.D., P.Eng.

Table of Contents

- ❑ What is Cloud Computing?
- ❑ AWS Identity & Access Management
- ❑ Amazon EC2
- ❑ Amazon EC2 Instance Storage
- ❑ Elastic Load Balancing & Auto Scaling Group
- ❑ Amazon S3
- ❑ Databases & Analytics
- ❑ Other Compute Services
- ❑ Deploying & Managing Infrastructure at Scale
- ❑ Global Infrastructure
- ❑ Cloud Integration

Table of Contents

- ❑ Cloud Monitoring
- ❑ Amazon VPC
- ❑ Security & Compliance
- ❑ Machine Learning
- ❑ Account Management, Billing, & Support
- ❑ Advanced Identity
- ❑ Other AWS Services
- ❑ AWS Architecting & EcoSystem

AWS Certifications

- **Foundational:** This is the starting point, requiring six months of fundamental AWS Cloud and industry knowledge. The certification at this level is "[AWS Certified Cloud Practitioner](#)."
- **Associate:** This level is for those who have one year of experience solving problems and implementing solutions using the AWS Cloud. It offers three certifications:
 - AWS Certified Solutions Architect – Associate
 - AWS Certified Developer – Associate
 - AWS Certified SysOps Administrator – Associate
- **Professional:** This is a more advanced level, requiring two years of experience in designing, operating, and troubleshooting solutions using the AWS Cloud. It includes two certifications:
 - AWS Certified Solutions Architect – Professional
 - AWS Certified DevOps Engineer – Professional
- **Specialty:** These certifications are for those who have technical AWS Cloud experience in specialized domains. These are more focused areas of expertise and require specific technical knowledge as specified in the exam guide. Specialty certifications include:
 - AWS Certified Advanced Networking – Specialty
 - [AWS Certified Data Analytics – Specialty](#)
 - AWS Certified Database – Specialty
 - [AWS Certified Machine Learning – Specialty](#)
 - AWS Certified Security – Specialty
 - AWS Certified SAP on AWS – Specialty

Create an AWS account

- **Step 1: Access the AWS Free Tier Page**
 1. Go to the AWS Free Tier page by visiting this URL: [AWS Free Tier](#)
- **Step 2: Click on "Create an AWS Account"**
 2. On the AWS Free Tier page, click on the "Create an AWS Account" button.
- **Step 3: Enter Your Account Information**
 - 3.1. Fill in your email address, a password for your AWS account, and a unique AWS account name.
 - 3.2. Click "Continue" to proceed.
- **Step 4: Contact Information**
 - 4.1. Provide your full name, contact number, and address details.
 - 4.2. Click "Next" to continue.

Create an AWS account

- **Step 5: Payment Information**

- 5.1. Enter your payment information. Don't worry, you won't be charged during the free tier period.
- 5.2. Click "Verify and Add" to validate your payment method.

- **Step 6: Identity Verification**

- 6.1. AWS will send you a verification code to your provided phone number.
- 6.2. Enter the code you received, and click "Verify code and continue."

- **Step 7: Choose a Support Plan**

- 7.1. Select the "Basic" plan, which is free.
- 7.2. Click "Continue."

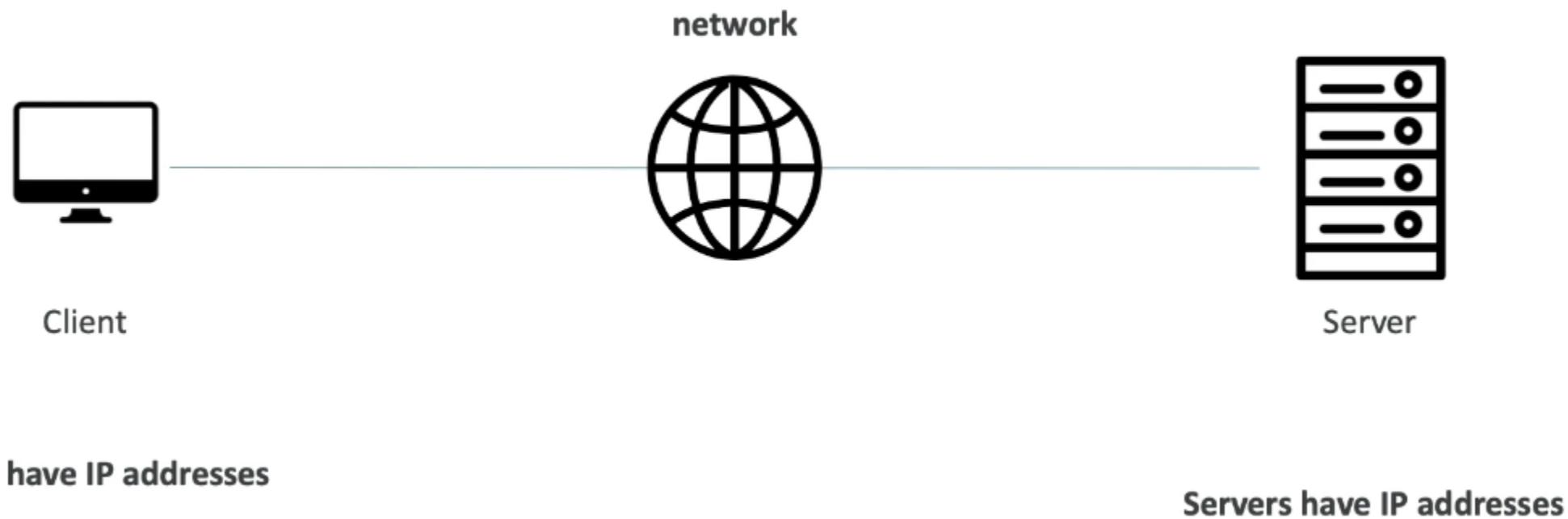
- **Step 8: Complete Sign-up**

- 8.1. Review the details you've entered and make sure they are accurate.
- 8.2. Click "Create Account and Continue" to finalize the process.

- **Step 9: Welcome to AWS**

- 9.1. Congratulations! You now have a free AWS account.
- 9.2. You can start exploring AWS services and resources available in the AWS Management Console.

How websites work



How websites work

- **Client:** Typically a user's device (like a computer or smartphone) that requests resources (web pages, images, data) from a server over the internet. The client has a unique IP (Internet Protocol) address that identifies it on the network.
- **Network:** The internet, represented by the globe icon, is the vast network of interconnected devices across the world. It facilitates the transmission of data between clients and servers.
- **Server:** A server is a computer that provides data to other computers. It hosts websites and serves their content to clients upon request. Like clients, servers also have unique IP addresses so they can be located on the network.
- When a user (client) wants to visit a website, the browser sends a request over the network to the server that hosts the website. The request contains the IP address of the client, so the server knows where to send the data. The server then responds with the website's data, which travels back across the network to the client's IP address. The browser receives this data and renders the website for the user to view.

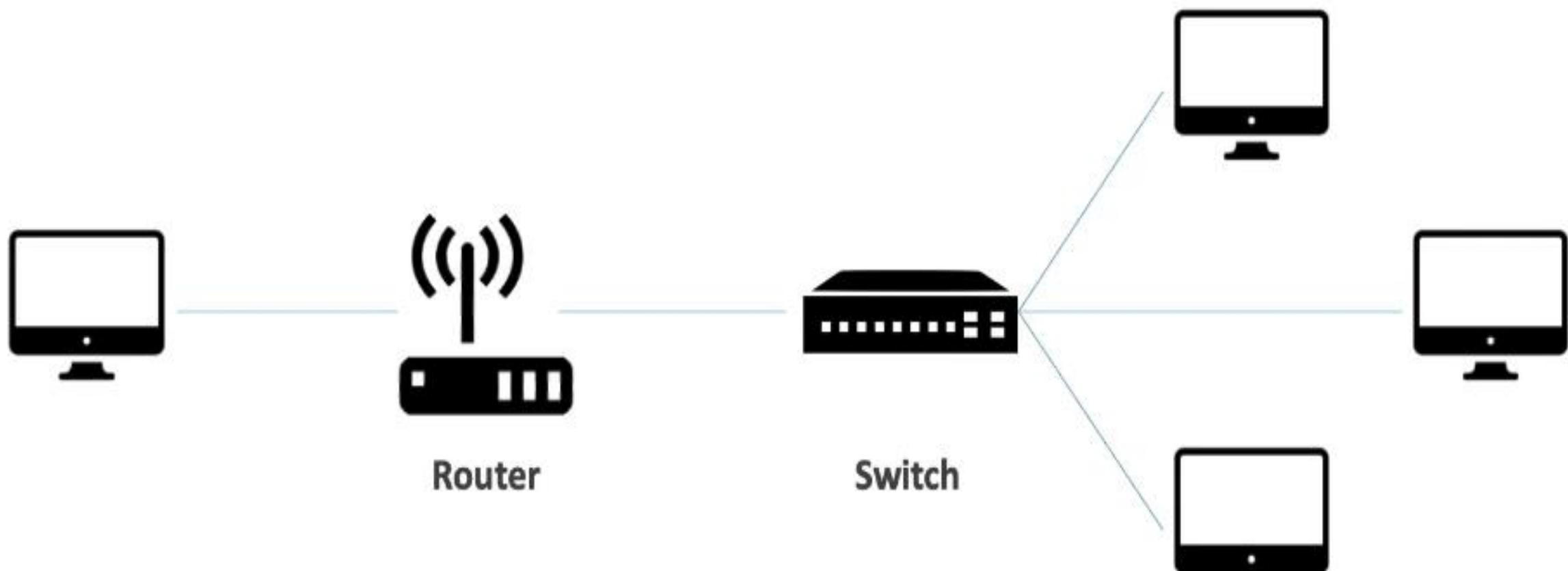
How websites work-Example

- Imagine a website as a library and the network as the postal service:
- **Client**: You are at home, wanting to read a book. Your home address is analogous to an IP address. It's unique and specifies where you are located.
- **Network**: You send a request via mail (postal service), asking for a specific book. The postal service represents the network, responsible for delivering messages (data packets) to and from various addresses (IP addresses).
- **Server**: The library is the server where books (web content) are stored. It has its own unique address (server IP address) so the postal service knows where to collect the book from.
- You write a letter (data request) asking for a book by title (URL), and send it to the library's address (server IP). The library receives your request, finds the book (processes the request for web content), and sends it back to you (sends the web data) via the postal service (network), using your home address (client IP) to deliver it directly to you.

What is server?

- **Compute (CPU):** The Central Processing Unit (CPU) is the brain of the server where most calculations take place. It executes commands from the server's operating system and applications.
- **Memory (RAM):** Random Access Memory (RAM) is the server's short-term memory, which stores data that is actively being used or processed by the CPU.
- **Storage:** This refers to where data is permanently stored, usually on hard drives or solid-state drives. It retains data even when the server is powered down.
- **Database:** A database is a structured set of data held in a computer, typically accessed by a server through database management software.
- **Network:** This includes devices like routers, which direct traffic across the internet; switches, which connect devices within a network; and DNS servers, which translate domain names into IP addresses that computers can understand.

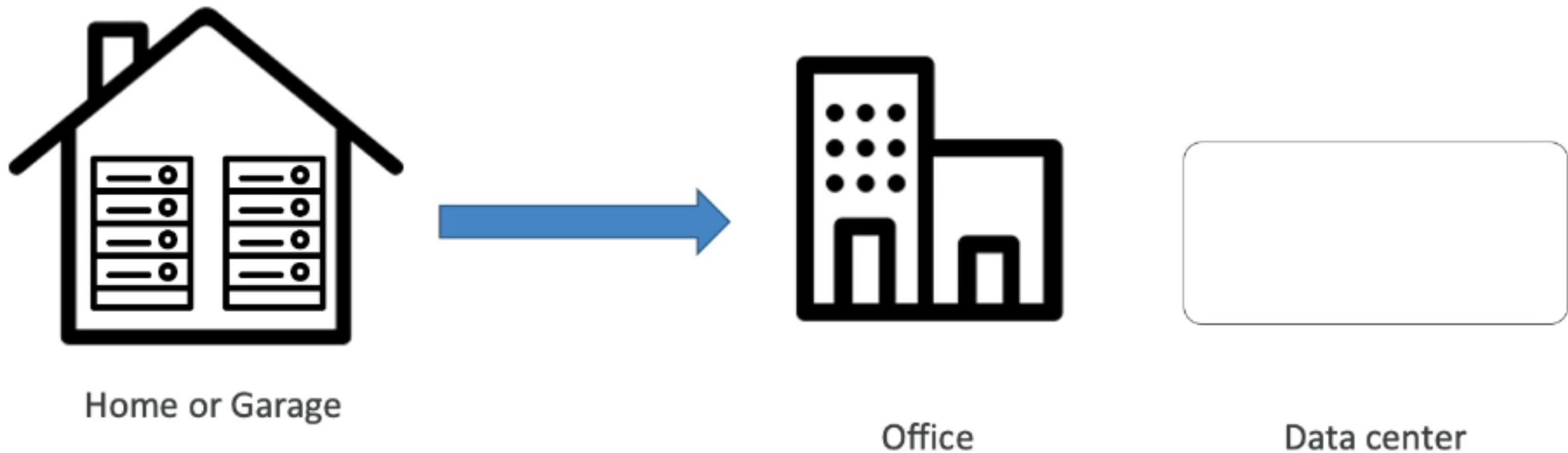
What is server?



What is server?

- 1.Router:** The router serves as the gateway between your local network and the internet. It assigns local IP addresses to each device on the network and directs incoming and outgoing traffic efficiently. In the diagram, it is shown connecting to one of the computers, symbolizing the connection to the internet.
- 2.Switch:** The switch is depicted as a central hub in the local network that receives data packets and forwards them to their destination within the local network. For example, if one computer wants to send a file to another on the same network, the switch will receive the data packet and direct it to the correct computer based on its MAC address. This is more efficient than broadcasting the packet to all connected devices.
- 3.Computers:** The computers on the left and right are clients in the network. They can communicate with each other through the switch or with external networks (like the internet) through the router.

Evolution of computing infrastructure



Home or Garage

Office

Data center

Evolution of computing infrastructure

- 1. Home or Garage:** Many technology companies begin with rudimentary setups, often in a home or garage. This symbolizes the early stages of a business or startup, where a few servers or computers are used to develop and host the initial product or service.
- 2. Office:** As the business grows, the need for more professional and scalable infrastructure becomes evident. Operations move to an office setting, which likely includes a dedicated space for servers or expanded IT infrastructure.
- 3. Data Center:** The blank space labeled 'Data Center' implies the next step in the infrastructure evolution. A data center is a large facility used to house computer systems and associated components. It is equipped with backup power supplies, data communications connections, environmental controls (e.g., air conditioning, fire suppression), and various security devices. As companies continue to grow, they may require their own data center or might lease space in an existing one to support their increased computing needs.

Challenges with associated traditional IT

- 1. Costs of Data Center Rent:** Operating a private data center requires paying rent for the physical space to house servers and other hardware.
- 2. Utility and Maintenance Expenses:** Beyond rent, there are significant ongoing costs for electricity to power servers and cooling systems to prevent overheating, as well as for general maintenance of the hardware.
- 3. Hardware Management:** The process of adding new hardware or replacing failed components can be time-consuming and often results in downtime.
- 4. Scalability Issues:** Traditional IT infrastructure can be difficult to scale quickly. Increasing capacity often means purchasing additional hardware, which is not only costly but also takes time to install and configure.
- 5. Continuous Monitoring:** A dedicated team is necessary to monitor the infrastructure around the clock to ensure its smooth operation and to address any issues promptly.
- 6. Disaster Recovery:** Developing a robust strategy for disaster recovery is complex and expensive. It requires planning for events such as earthquakes, power outages, or fires, which can cause significant disruption.
7. can these responsibilities be externalized?



Cloud Computing



- **Cloud Computing Definition:** It's explained as the on-demand delivery of compute power, database storage, applications, and other IT resources. The key feature here is "on-demand," meaning that these resources are available whenever they are needed.
- **Pay-as-you-go Pricing:** This is a flexible pricing strategy where you only pay for the cloud services you use, which helps in cost-saving and managing expenses according to usage.
- **Resource Provisioning:** You can provision the exact type and size of computing resources you need. This allows for a high degree of customization and efficiency, as you can scale up or down based on your requirements.
- **Access to Resources:** It mentions the ability to access as many resources as you need, almost instantly. This suggests the high scalability and flexibility of cloud services.
- **Simplicity in Access:** The slide emphasizes the simplicity of accessing servers, storage, databases, and a suite of application services, highlighting the user-friendly aspect of cloud computing.
- **AWS Infrastructure:** Finally, it states that Amazon Web Services owns and maintains the necessary hardware for these services, and users can provision and use what they need via a web application. This part implies that AWS takes care of the backend infrastructure, reducing the burden on users to maintain physical hardware.

Cloud Computing

The AWS logo consists of the letters "AWS" in white, bold, sans-serif font, enclosed within a stylized orange cloud shape.

AWS

1. Gmail: Identified as an email cloud service, where users can send and receive emails without having to manage any server infrastructure themselves. The implication is that users may pay only for the storage they require for their emails, rather than any additional infrastructure.



2. Dropbox: Noted as a cloud storage service, it allows users to store and share files over the internet. Dropbox was originally built on Amazon Web Services (AWS), indicating it used AWS's infrastructure to host its service.



3. Netflix: Described as being built on AWS, Netflix is a streaming service offering video on demand. It uses cloud computing to stream movies and TV shows to users, scaling its service capacity up and down as needed, based on demand.



Cloud Computing



1. Private Cloud:

1. Used exclusively by a single organization and not exposed to the public.
2. Provides complete control over the cloud infrastructure and services.
3. Offers heightened security for sensitive applications, which is crucial for organizations handling confidential data.
4. Can be tailored to meet specific business needs and requirements.



2. Public Cloud:

1. The cloud resources (like servers and storage) are owned and operated by a third-party cloud service provider and are delivered over the Internet.
2. The next slide mentions "Six Advantages of Cloud Computing" which likely refers to benefits such as cost-efficiency, scalability, performance, reliability, global reach, and the simplicity of operation.



3. Hybrid Cloud:

1. This is a combination of on-premises infrastructure (private cloud) and public cloud services.
2. Allows for keeping some servers and operations on-premises, especially those handling sensitive data or critical operations, while also leveraging the public cloud for additional capabilities.
3. Provides control over sensitive assets within the private infrastructure while still benefiting from the cloud's scalability and cost-efficiency.



Cloud Computing-5 characteristic



- 1. On-demand self-service:** This means that cloud customers can automatically provision computing resources such as server time and network storage without requiring human interaction with the service provider. Users have the ability to manage their resources on an as-needed basis.
- 2. Broad network access:** Cloud services are available over the network and can be accessed through standard mechanisms by a broad range of devices such as smartphones, tablets, laptops, and workstations.
- 3. Resource pooling:** The cloud provider's computing resources are pooled to serve multiple consumers, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. This is done with a sense of location independence, meaning the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter).
- 4. Rapid elasticity and scalability:** Capabilities can be rapidly and elastically provisioned, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.
- 5. Measured service:** Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

Cloud Computing-5 characteristic-Example

1. On-demand self-service:

- Like using a bike-sharing service, where you can pick up a bike whenever you need it without having to speak to anyone, just by using an app to unlock a bike at the nearest station.

2. Broad network access:

- Similar to being able to access buses, trains, or subways from a wide range of locations and with various devices such as a bus pass, smartphone app, or ticket kiosk.

3. Multi-tenancy and resource pooling:

- Comparable to different passengers sharing the same public transport vehicles. The bus or train has a set capacity (the pooled resource) which is shared among passengers, without each needing to own their individual vehicle.

4. Rapid elasticity and scalability:

- Analogous to the public transport authority being able to deploy more buses or trains during peak hours or a special event to accommodate the increased number of passengers, and then reduce the number of vehicles during off-peak times.

5. Measured service:

- Just like paying for a transit ticket, where you pay based on how far you travel (zone pricing) or how long you use the service (time-based fares), the usage is measured, and you pay only for what you use.

Six Advantages of Cloud Computing

1. Trade capital expense (CAPEX) for operational expense (OPEX):

- ❑ Instead of investing heavily upfront in data centers and servers, companies can pay for computing resources as they use them, shifting the financial burden from capital expenses to operational expenses.

2. Benefit from massive economies of scale:

- ❑ By leveraging services from cloud providers like AWS, costs are reduced since these providers can achieve higher economies of scale due to their large size and efficiency, passing on the savings to customers.

3. Stop guessing capacity:

- ❑ Rather than purchasing fixed amounts of computing resources in advance and risking over-provisioning or under-provisioning, companies can scale their resources up or down based on actual usage.

4. Increase speed and agility:

- ❑ Cloud computing allows businesses to quickly deploy new applications, scale up as their workload grows, and adapt faster to changing market conditions.

5. Stop spending money running and maintaining data centers:

- ❑ Companies save on the costs associated with running and maintaining data centers, such as power, cooling, and hardware maintenance.

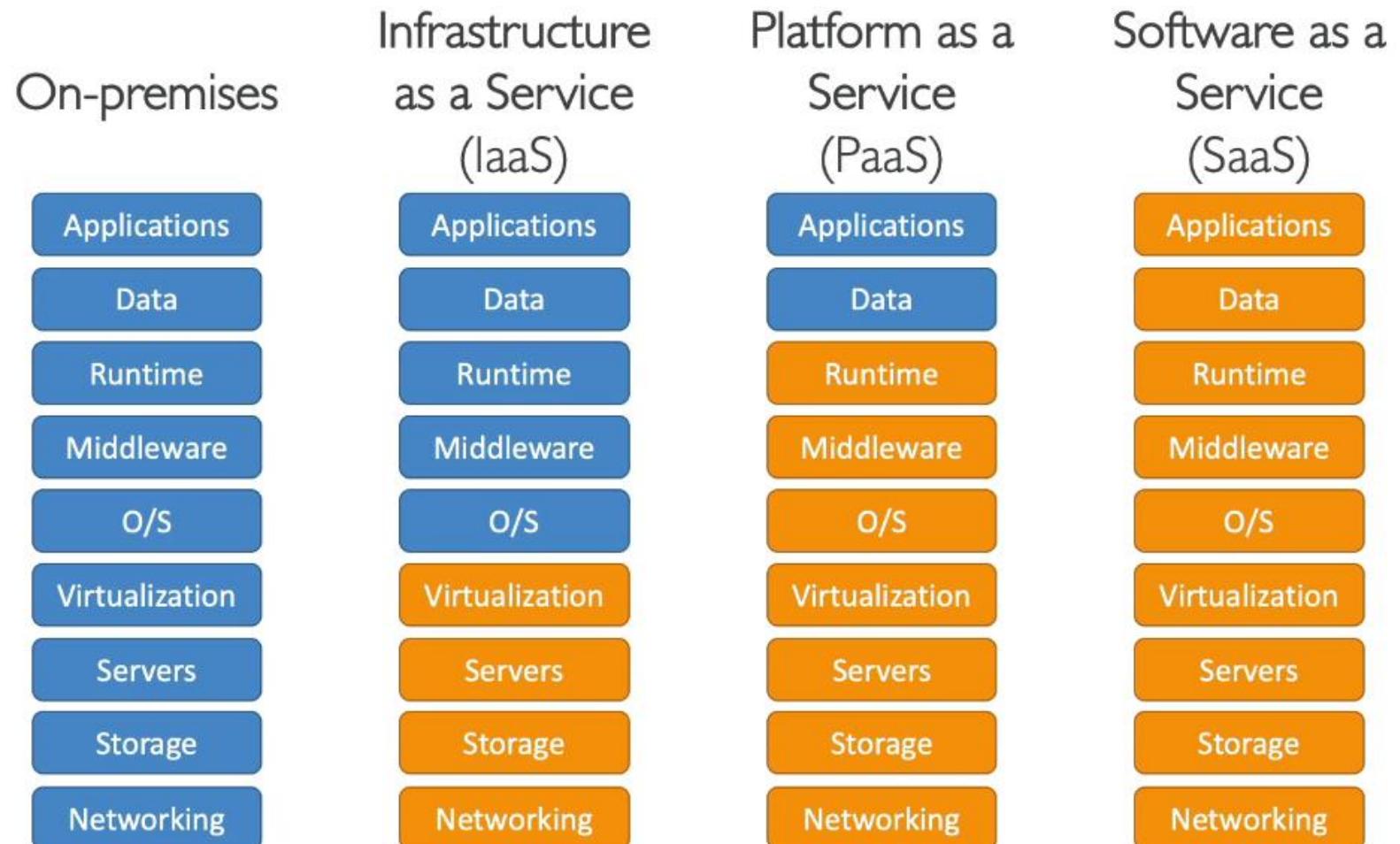
6. Go global in minutes:

- ❑ Cloud services like AWS have a global infrastructure that allows businesses to deploy their applications in multiple regions around the world with just a few clicks, providing lower latency and a better experience for their customers.

Benefits from Cloud

- 1. Flexibility:** Cloud computing provides the ability to change computing resource types based on evolving needs, without the constraints of physical hardware.
- 2. Cost-Effectiveness:** With cloud services, you pay only for the resources you consume, which can lead to significant cost savings compared to maintaining on-premises infrastructure.
- 3. Scalability:** The cloud allows for accommodating larger loads by enhancing existing hardware capabilities or adding more computing nodes, enabling growth without the need for physical hardware expansion.
- 4. Elasticity:** This is the ability to scale resources out (increase) or in (decrease) automatically in response to demand, ensuring that users have the right number of resources at any given time.
- 5. High-availability and fault-tolerance:** Cloud providers often distribute resources across multiple geographically dispersed data centers, which can reduce the risk of downtime and data loss in the event of a failure.
- 6. Agility:** Cloud computing allows organizations to rapidly develop, test, and launch software applications, which can accelerate time to market and response to market changes.

Type of cloud computing



Type of cloud computing

- 1. On-premises:** This is the traditional model where an organization owns and manages its own IT resources. All layers from networking up to applications are managed in-house. This includes the physical hardware, the virtualization technology, the operating system (O/S), middleware (like database systems), runtime environments (like Java runtime), and the actual applications and data.
- 2. Infrastructure as a Service (IaaS):** In this cloud computing model, the foundational elements of computing hardware are provided as a service. This includes servers, storage, and networking (highlighted in orange), while the management of operating systems, middleware, runtime, applications, and data (highlighted in blue) is the responsibility of the user. Examples include Amazon EC2 and Google Compute Engine.
- 3. Platform as a Service (PaaS):** This model provides a cloud-based platform that includes the infrastructure (servers, storage, networking) and also the operating system, middleware, and runtime. Users only manage the applications and data. PaaS is used for general software development, and examples include Google App Engine and Microsoft Azure.
- 4. Software as a Service (SaaS):** The most inclusive service model, where everything from the infrastructure to the applications is provided and managed by the service provider. The user simply uses the software over the internet, usually through a web browser. Examples include Google Workspace (formerly G Suite) and Salesforce.

Type of cloud computing

Infrastructure as a Service (IaaS):

Amazon EC2 (on AWS)
GCP (Google Cloud Platform)
Azure (Microsoft)
Rackspace
Digital Ocean
Linode

Platform as a Service (PaaS):

Elastic Beanstalk (on AWS)
Heroku
Google App Engine (on GCP)
Windows Azure (Microsoft)

Software as a Service (SaaS):

Many AWS services (example: Rekognition for Machine Learning)
Google Apps (Gmail)
Dropbox
Zoom

Cloud Usage Price

Compute: You pay for the compute time you use, meaning the actual time your services or applications are running on AWS servers.

- **Example:** If you run an EC2 instance (a virtual server) for web hosting, you pay for the time the instance is active. If you have an instance that's running for 10 hours, you'll only be charged for those 10 hours of usage.

Storage: The cost is based on the amount of data you store in the cloud services, such as Amazon S3 or EBS (Elastic Block Store).

- **Example:** If you use Amazon S3 to store 100 GB of backup data, you will be charged for the 100 GB storage per month.

Data Transfer OUT of the Cloud: AWS charges for data transferred "out" of AWS to the internet or to another AWS region. However, data transfer "in" to AWS is usually free.

- **Example:** If your application sends data from AWS servers to users' devices, you will pay for the data sent out. For instance, if your users download 50 GB of data in total from your application hosted on AWS, you'll pay for that outbound data transfer.

Key statistics and positioning of Amazon Web Services (AWS)

- 1. AWS Revenue:** It states that in 2019, AWS had \$35.02 billion in annual revenue.
 - 2. Market Share:** AWS accounted for 47% of the cloud market in 2019, with Microsoft being the second with 22%.
 - 3. Market Position:** AWS is recognized as a pioneer and leader in the cloud market for the ninth consecutive year.
 - 4. User Base:** AWS has over 1,000,000 active users.
- ❑ The Gartner Magic Quadrant graphic is a research methodology that provides a graphical competitive positioning of technology providers, where companies are categorized into four quadrants: Leaders, Challengers, Visionaries, and Niche Players.
 - ❑ In the Magic Quadrant portion of the slide, AWS is placed in the 'Leaders' quadrant, indicating its strong ability to execute and its completeness of vision compared to other market competitors such as Microsoft and Google

Figure 1. Magic Quadrant for Cloud Infrastructure as a Service, Worldwide



AWS (Amazon Web Services) Global Infrastructure

1. AWS Regions: Geographical locations around the world where AWS clusters data centers. Each region is a separate geographic area that has multiple, isolated locations known as Availability Zones.

2. AWS Availability Zones: Each AWS Region consists of multiple, isolated, and physically separate locations known as Availability Zones. These zones are designed to be interconnected within a region while being isolated from failures in other Availability Zones, providing redundancy and high availability.

3. AWS Data Centers: Secure facilities that house the servers and storage devices for AWS services. While the exact locations and numbers of AWS data centers are not publicly disclosed, they are spread across the Availability Zones within each region.

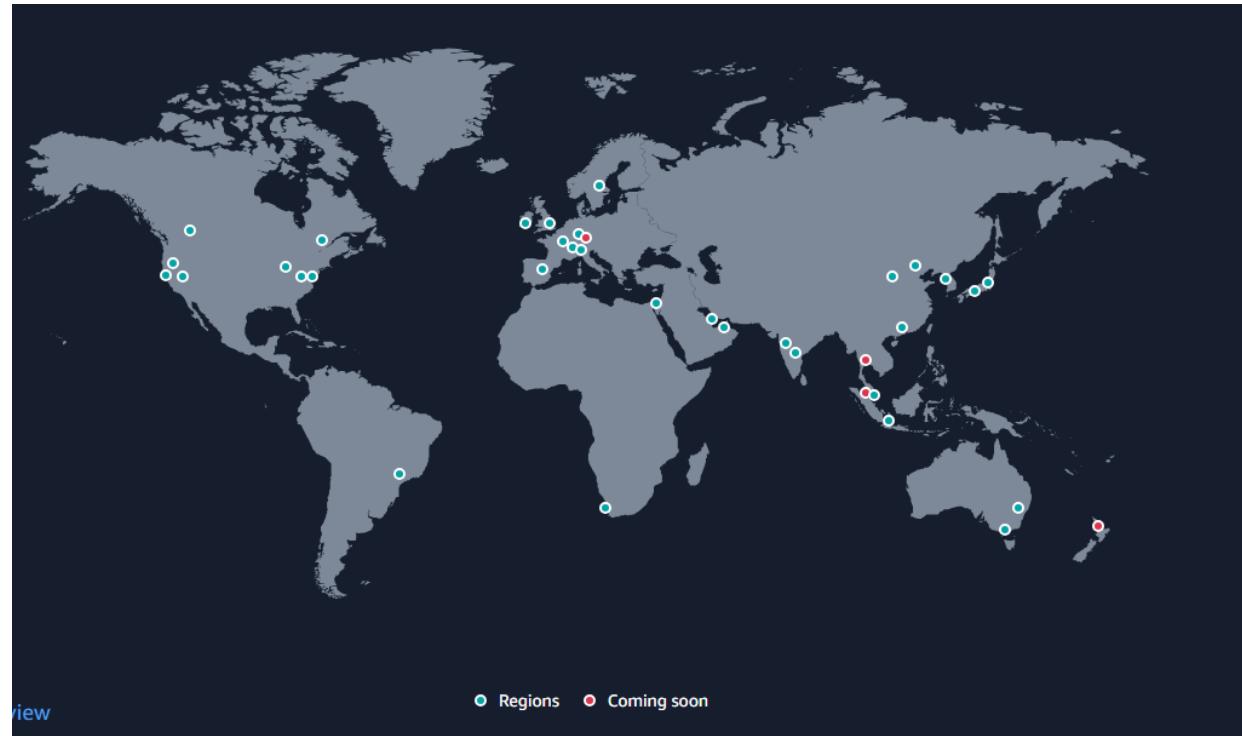
4. AWS Edge Locations / Points of Presence: These are locations that AWS has placed infrastructure to deliver services such as Amazon CloudFront (AWS's Content Delivery Network) closer to customers, to reduce latency.

- (<https://infrastructure.aws/>)



AWS (Amazon Web Services) Regions

- 1. AWS Regions:** AWS has regions all around the world, which are geographical areas that contain clusters of data centers.
- 2. Naming Convention:** The regions are named in a standardized way, such as "us-east-1" for the North Virginia region or "eu-west-3" for the Paris region.
- 3. Region as a Cluster:** Each AWS region is a cluster of data centers, which allows for the redundancy and availability of services.
- 4. Region-Scope Services:** Most AWS services are scoped to regions, meaning that the resources and services you use are contained within and operate out of a specific region.



Factors to consider for choosing an AWS Region

- 1. Compliance:** Ensuring that the region you choose complies with data governance and legal requirements that apply to your application, especially since data does not leave the chosen region without explicit permission, which is important for privacy laws and regulations.
- 2. Proximity to customers:** Selecting a region close to your user base can significantly reduce latency, leading to faster access and better performance for end-users.
- 3. Available services within a Region:** Some AWS services or features may not be available in all regions. It's essential to choose a region where all the services you plan to use are available.
- 4. Pricing:** Costs for AWS services can vary between regions. It's important to review the pricing for the services you intend to use in the region you are considering to ensure cost-effectiveness.



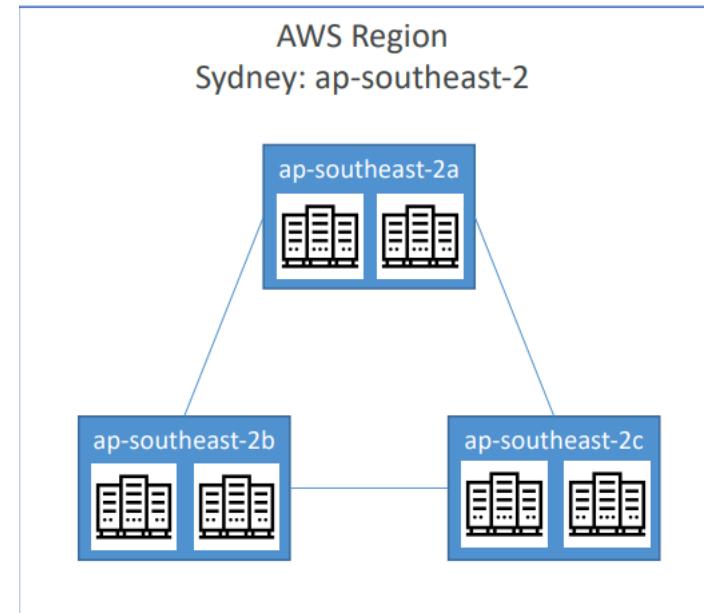
AWS Availability Zones (AZs)

1. AWS Availability Zones: Each AWS region consists of multiple availability zones, typically three or more (the minimum is three and the maximum can be up to six).

2. Examples of Availability Zones: AWS Sydney region (ap-southeast-2), specifically naming ap-southeast-2a, ap-southeast-2b, and ap-southeast-2c.

3. Characteristics of Availability Zones:

- ❑ Each AZ has one or more discrete data centers with redundant power, networking, and connectivity to ensure high availability and fault tolerance.
- ❑ The AZs are geographically separate to ensure that they are isolated from one another, which helps in protecting services from localized disasters.
- ❑ Despite being isolated, the AZs are interconnected with high bandwidth and ultra-low latency networking to provide quick communication between zones.



AWS (Amazon Web Services) Points of Presence, also known as Edge Locations

- Amazon has over 400 Points of Presence, including over 400 Edge Locations and over 10 Regional Caches.
- These Points of Presence are distributed in over 90 cities across more than 40 countries.
- The purpose of these locations is to deliver content to end-users with lower latency, meaning that data has a shorter distance to travel, resulting in faster access and download times for the user.
- Different colors indicating single Edge Locations, cities with multiple Edge Locations, and Regional Edge Caches.
- (<https://aws.amazon.com/cloudfront/features/>) which presumably provides more detailed information about the features of AWS CloudFront, AWS's CDN service.
- These Edge Locations are used to cache content closer to users, improving performance for web applications and services by delivering content from a location nearest to the user to minimize latency.



AWS (Amazon Web Services) Service Tour

1. AWS Global Services:

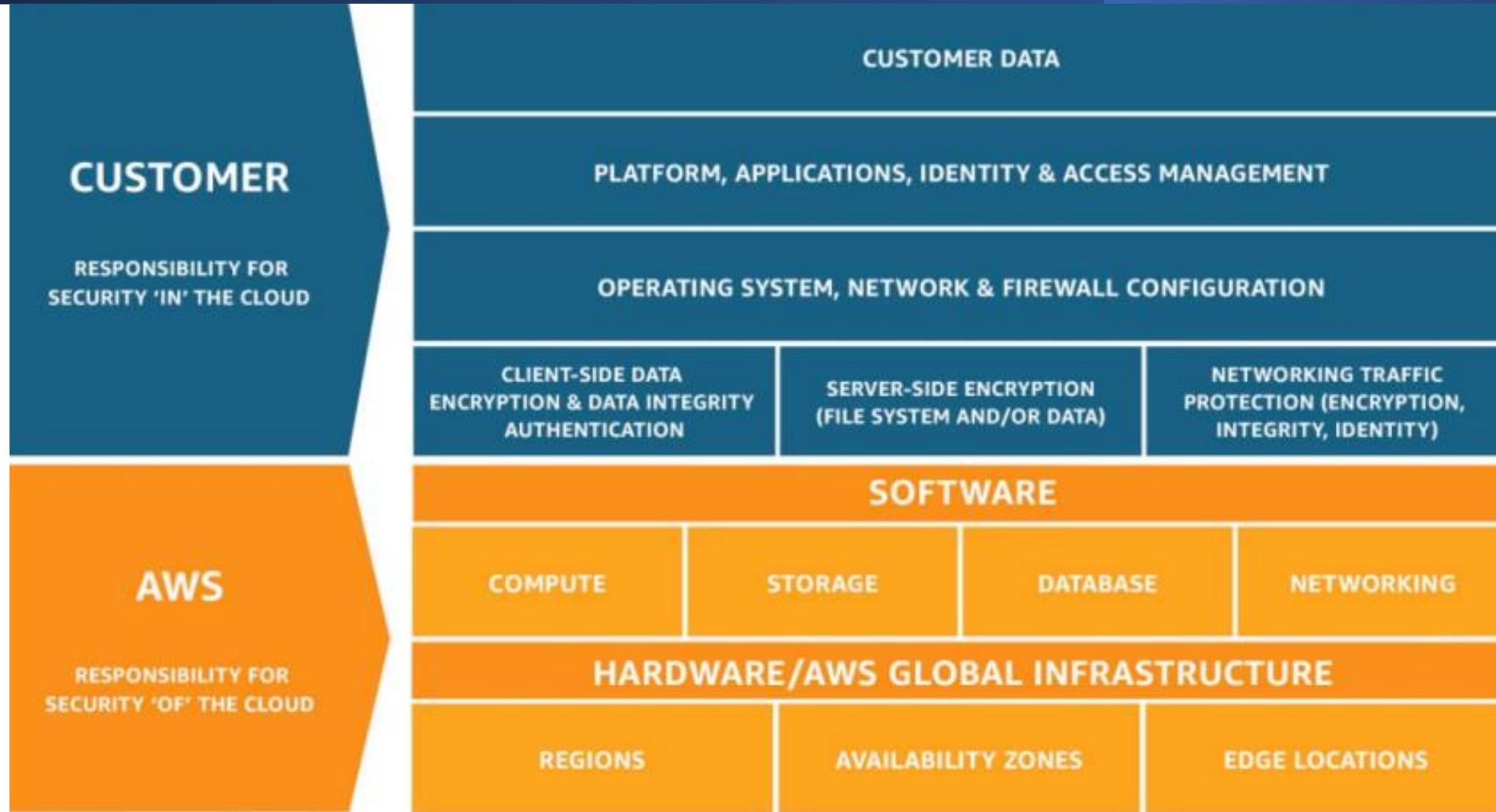
1. Identity and Access Management (IAM): A service that helps you securely control access to AWS resources.
2. Route 53: A scalable and highly available Domain Name System (DNS) web service.
3. CloudFront: Amazon's Content Delivery Network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally.
4. WAF (Web Application Firewall): A service that helps protect your web applications from common web exploits.

2. Region-scoped AWS Services:

1. Amazon EC2 (Elastic Compute Cloud): Provides scalable computing capacity in the AWS cloud.
2. Elastic Beanstalk: An easy-to-use service for deploying and scaling web applications and services.
3. Lambda: A service that lets you run code without provisioning or managing servers.
4. Rekognition: A service that makes it easy to add image and video analysis to your applications.

- (<https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services>)

Shared Responsibility Model of AWS (Amazon Web Services)



Shared Responsibility Model of AWS (Amazon Web Services)

1. AWS's Responsibility - "Security OF the Cloud":

- ❑ AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. This infrastructure is composed of the hardware, software, networking, and facilities that run AWS Cloud services.
- ❑ Key components managed by AWS include:
 - ❖ Compute ,Storage, Database, Networking, Regions, Availability Zones, Edge Locations

2. Customer's Responsibility - "Security IN the Cloud":

1. Customers are responsible for securing their data and applications that utilize AWS services. This includes managing the guest operating system (including updates and security patches), other associated application software, as well as the configuration of the AWS provided security group firewall.
 2. Customers manage aspects like:
 1. Customer Data
 2. Platform, Applications, Identity & Access Management
 3. Operating System, Network & Firewall Configuration
 4. Client-Side Data Encryption & Data Integrity Authentication
 5. Server-Side Encryption (File System and/or Data)
 6. Networking Traffic Protection (Encryption, Integrity, Identity)
- The Shared Responsibility Model emphasizes that while AWS ensures the security of the cloud infrastructure, customers must take proactive steps to secure their applications and data within the cloud. (<https://aws.amazon.com/compliance/shared-responsibility-model/>)

AWS Acceptable Use Policy

❑ <https://aws.amazon.com/aup/>

❑ Prohibited Actions and Content:

1. **No Illegal, Harmful, or Offensive Use or Content:** Users are prohibited from using AWS services for illegal, harmful, or offensive content or activities. This includes, but is not limited to, content that is illegal, infringes upon the intellectual property rights of others, or is defamatory, obscene, abusive, invasive of privacy, or otherwise objectionable.
2. **No Security Violations:** Users must not violate the security or integrity of any network, computer, or communications system, software application, or network or computing device. This includes not engaging in activities that harm or disrupt the operation of AWS or others' services and infrastructure.
3. **No Network Abuse:** The policy forbids users from engaging in network abuse, such as making network connections to any users, hosts, or networks unless you have permission to communicate with them.
4. **No E-Mail or Other Message Abuse:** Users are not allowed to distribute, publish, send, or facilitate the sending of unsolicited mass e-mail or other messages, promotions, advertising, or solicitations (like "spam"), including commercial advertising and informational announcements.

AWS Identity and Access Management (IAM) -users and groups

- ❑ **IAM (Identity and Access Management):** It is a global service within AWS that allows you to manage access to AWS services and resources securely. Using IAM, you can create and manage AWS users and groups, and use permissions to allow and deny their access to AWS resources.
- ❑ **Root Account:** The root account is created by default when you open an AWS account. It has complete access to all AWS services and resources in the account. It is recommended not to use the root account for everyday tasks, even administrative ones, and it should never be shared.
- ❑ **Users:** These are individual people or entities within your organization that you can create within IAM to represent the person or service that will interact with AWS.
- ❑ **Groups:** A way to specify permissions for a collection of users, which can make it easier to manage the permissions for those users. Groups are collections of users and only contain users, not other groups.

AWS Identity and Access Management (IAM) -users and groups

- **Group Membership:** Users can be members of zero or more groups. For example, a user can be in the "Developers" group to get certain permissions and also in the "Audit Team" group for different permissions.



IAM (Identity and Access Management) Permissions in AWS

1. **IAM Policies:** Users or groups in AWS can be assigned policies written in JSON (JavaScript Object Notation). These policies are sets of permissions that define what actions users or groups can perform on AWS resources.
2. **Policy Structure:** The JSON document typically contains the following elements:
 - **Version:** The policy language version, which is usually "2012-10-17".
 - **Statement:** An array of individual statements, where each statement includes:
 - **Effect:** Specifies whether the statement results in an "Allow" or "Deny" permission.
 - **Action:** Describes the specific actions allowed or denied, such as "ec2:Describe*" to allow actions that describe EC2 resources.
 - **Resource:** Specifies the resources to which the actions apply. An asterisk ("*") means the action applies to all resources.
3. **Least Privilege Principle:** AWS recommends applying the least privilege principle, meaning you should only give users the permissions they need to perform their tasks and no more. This reduces the risk of accidental or malicious changes to resources and increases security.

IAM (Identity and Access Management) Permissions in AWS

- In the provided policy example, the following permissions are set:

- Allow describing all EC2 resources.
- Allow describing all Elastic Load Balancing resources.
- Allow listing metrics, getting metric statistics, and describing CloudWatch resources.

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

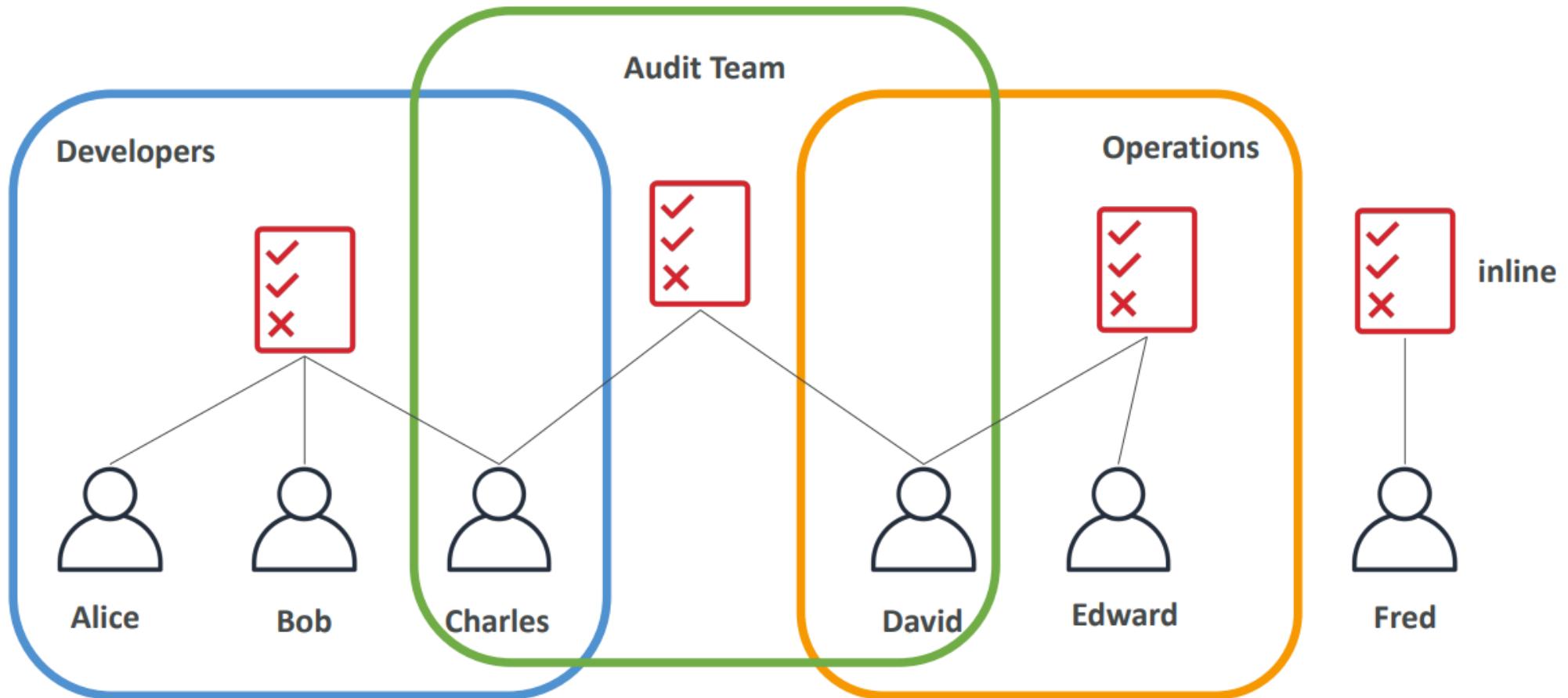




IAM

Hands on Practice

Policy inheritance within AWS Identity and Access Management (IAM)



Policy inheritance within AWS Identity and Access Management (IAM)

- **Developers Group:** Contains users Alice, Bob, and Charles. A policy is attached to the group, and all users in this group inherit the permissions specified in that policy.
- **Audit Team Group:** Contains user Charles. Charles is a member of both the Developers and Audit Team groups, so he inherits the policies from both groups.
- **Operations Group:** Contains users Edward and Fred. Just like the Developers group, a policy is attached to the Operations group, and Edward and Fred inherit those permissions. Additionally, there's an "inline" policy attached to Fred. This indicates that Fred has a policy directly attached to his user entity, which is specific to him and not shared with other users
- Key points to note:
 - Users can belong to multiple groups and inherit permissions from each group they belong to.
 - Groups can have policies attached to them, which apply to all users within the group.
 - Policies determine permissions within AWS and can be used to allow or deny access to AWS resources.
 - Inline policies are policies that are directly attached to a single IAM user, group, or role. They are a way to assign permissions that are not shared with other IAM entities.
 - This model allows for granular control over permissions and can simplify the management of user permissions by assigning common permissions to groups rather than individual users.

Policy inheritance within AWS Identity and Access Management (IAM)

```
{  
    "Version": "2012-10-17",  
    "Id": "S3-Account-Permissions",  
    "Statement": [  
        {  
            "Sid": "1",  
            "Effect": "Allow",  
            "Principal": {  
                "AWS": ["arn:aws:iam::123456789012:root"]  
            },  
            "Action": [  
                "s3:GetObject",  
                "s3:PutObject"  
            ],  
            "Resource": ["arn:aws:s3:::mybucket/*"]  
        }  
    ]  
}
```

Policy inheritance within AWS Identity and Access Management (IAM)

- IAM policies are written in JSON (JavaScript Object Notation) and define permissions within AWS.
- 1. **Version:** The version field specifies the policy language version. The current and only version supported is "2012-10-17".
- 2. **Id:** This is an optional identifier for the policy that can help you differentiate between multiple policies.
- 3. **Statement:** A policy must have at least one statement. Statements are the individual rules that describe the permissions. Within a statement, you may find:
 - **Sid** (Statement ID): An optional identifier that you provide for the policy statement. It is included for managing individual statements.
 - **Effect**: This specifies whether the statement results in an "Allow" or "Deny". "Allow" gives permissions, while "Deny" explicitly removes them.
 - **Principal**: The account, user, role, or federated user to which the policy is applied. In the context of a user's or role's policy, the principal is implicit and this element is not included.
 - **Action**: The list of actions that the policy allows or denies. Actions can be specified using wildcards to match multiple actions.
 - **Resource**: The list of resources to which the actions apply. Resources can also be specified using wildcards.
 - **Condition** (not shown in the provided policy but mentioned on the slide): These are optional conditions for when the policy is in effect. Conditions can be based on various elements such as date/time, IP address, etc.
- "**s3:GetObject- "**s3:PutObject****

IAM (Identity and Access Management) Password Policy

- 1. Strong Passwords:** The policy emphasizes that strong passwords contribute to higher security for AWS accounts.
- 2. Password Policy Configuration:** It outlines the steps and criteria that can be set up in AWS for password policies, which include:
 - 1. Minimum Password Length:** Setting the minimum number of characters that a password must contain.
 - 2. Character Type Requirements:** Enforcing the inclusion of different character types in the password, such as:
 1. Uppercase letters (A-Z)
 2. Lowercase letters (a-z)
 3. Numbers (0-9)
 4. Non-alphanumeric characters (e.g., !, \$, #, %)
- 3. User Password Changes:** The policy can be configured to:
 1. Allow IAM users to change their own passwords, which helps with user self-management.
 2. Require users to change their password after a certain period (known as password expiration), enhancing security by ensuring that passwords are regularly updated.
- 4. Password Reuse:** Implementing measures to prevent password reuse can be part of the policy to ensure that users do not recycle old passwords, which can mitigate the risk of unauthorized access from compromised credentials.

Multi-Factor Authentication (MFA)

- 1. Access and Security:** It states that users with access to an AWS account can change configurations or delete resources, which underscores the need for robust security measures.
- 2. Protection of Accounts:** Using MFA to protect root accounts and IAM users. The root account is the primary account with full access to all AWS services and resources, and IAM users are individuals with permission to use specific AWS resources.
- 3. MFA Definition:** MFA is defined as a combination of something you know (like a password) and a security device you own (like a mobile device or hardware token).
- 4. Benefit of MFA:** The main benefit highlighted is that even if a password is compromised, MFA ensures that the account remains secure because the attacker would still need the second factor, which they are unlikely to have.



Multi-Factor Authentication (MFA) Options

Virtual MFA device



Google Authenticator
(phone only)



Authy
(multi-device)

Universal 2nd Factor (U2F) Security Key



YubiKey by Yubico (3rd party)

Multi-Factor Authentication (MFA) Options

- 1. Virtual MFA Device:** This typically refers to software that generates time-based, one-time passcodes. like Google Authenticator, which works on phones, and Authy, which is available on multiple devices. These applications can support multiple tokens for different accounts on a single device.
- 2. Universal 2nd Factor (U2F) Security Key:** This is a hardware device that provides a strong form of MFA. YubiKey by Yubico as an example. Such keys are usually inserted into a USB port or connected via NFC (Near Field Communication) and tapped or touched to generate an authentication code. The key advantage of U2F keys is that they support multiple root and IAM users on a single device, and they're not as easily phished as other forms of MFA because they require physical interaction.

Access Amazon Web Services (AWS)

1. **AWS Management Console:** This is a web interface accessible through a browser. Users can log in using their username and password, and it is recommended to secure the account further with Multi-Factor Authentication (MFA).
 2. **AWS Command Line Interface (CLI):** This is a tool that enables users to interact with AWS services using commands in their command-line shell. Access to the AWS CLI is secured using access keys.
 3. **AWS Software Developer Kit (SDK):** The SDK is used for coding and allows developers to access AWS services from their application code. It is also secured with access keys.
- Access Keys, which are credentials used to authenticate and authorize access to AWS services. An Access Key consists of an Access Key ID (akin to a username) and a Secret Access Key (akin to a password). These keys are generated through the AWS Console, and it is crucial for users to manage their keys securely. The keys should remain confidential and never be shared, as they provide programmatic access to AWS resources.

Amazon Web Services (AWS) CLI

```
→ ~ aws s3 cp myfile.txt s3://ccp-mybucket/myfile.txt
upload: ./myfile.txt to s3://ccp-mybucket/myfile.txt
→ ~ aws s3 ls s3://ccp-mybucket
2021-05-14 03:22:52          0 myfile.txt
→ ~
```

- **AWS CLI:** It's a unified tool to manage AWS services directly from the command line. This tool allows you to issue commands to perform various AWS functions within your terminal or script.
- **Direct API Access:** The CLI provides direct access to the public APIs of AWS services, enabling automated scripts and commands to interact with AWS resources.
- **Scripting:** The CLI is useful for developing scripts to manage resources, automate tasks, and batch operations.
- **Open-Source:** The AWS CLI is an open-source application, and its source code is available on <https://github.com/aws/aws-cli>
- **Alternative to Management Console:** The CLI is an alternative to the AWS Management Console, providing a way to interact with AWS services through scripts and commands instead of the web interface.

Amazon Web Services (AWS) CLI- Hands on Practice

1. Open Command Prompt or PowerShell on your Windows machine.
2. If you haven't already configured your AWS CLI with your credentials, you would use the **aws configure** command. When you run this command, it will prompt you to enter your AWS Access Key ID, Secret Access Key, default region name, and default output format one after the other:

```
aws configure
```

- ❑ You would then enter your credentials and preferences when prompted:

AWS Access Key ID [None]: Enter your access key ID here

AWS Secret Access Key [None]: Enter your secret access key here

Default region name [None]: Enter your default region (e.g., us-west-1)

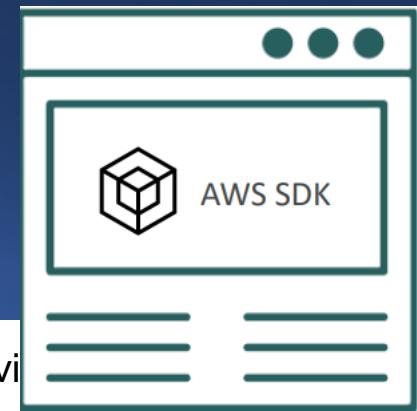
Default output format [None]: Enter your preferred output format (e.g., json)

- ❑ This will output a list of IAM users in your AWS account in the format you specified during configuration. If you didn't specify an output format, it will default to JSON.

```
aws iam list-users
```

- ❑ Remember, the above steps assume that you have the necessary permissions to list IAM users and that you have already installed AWS CLI on your Windows machine.

Amazon Web Services (AWS) SDK



- **AWS SDK:** This is a collection of software tools that allow developers to write applications using AWS services by providing language-specific APIs and libraries.
- **Language-Specific APIs:** The AWS SDK comes with a set of libraries for different programming languages, enabling developers to access and manage AWS services in a language that they are comfortable with.
- **Programmatic Access:** With the AWS SDK, developers can programmatically access AWS services, which allows for automation and integration within applications.
- **Embedded Within Applications:** The SDK is meant to be embedded within your own applications, providing a seamless way to interact with AWS services.
- **Supports Various Languages and Platforms:** The AWS SDK supports a wide range of programming languages and platforms, including but not limited to:
 - Standard SDKs for JavaScript, Python, PHP, .NET, Ruby, Java, Go, Node.js, and C++.
 - Mobile SDKs that cater to Android and iOS.
 - IoT Device SDKs for embedded C, Arduino, and other platforms.
- **Example:** It's mentioned that the AWS CLI, a command-line tool for AWS, is built on the AWS SDK for Python, also known as Boto3.

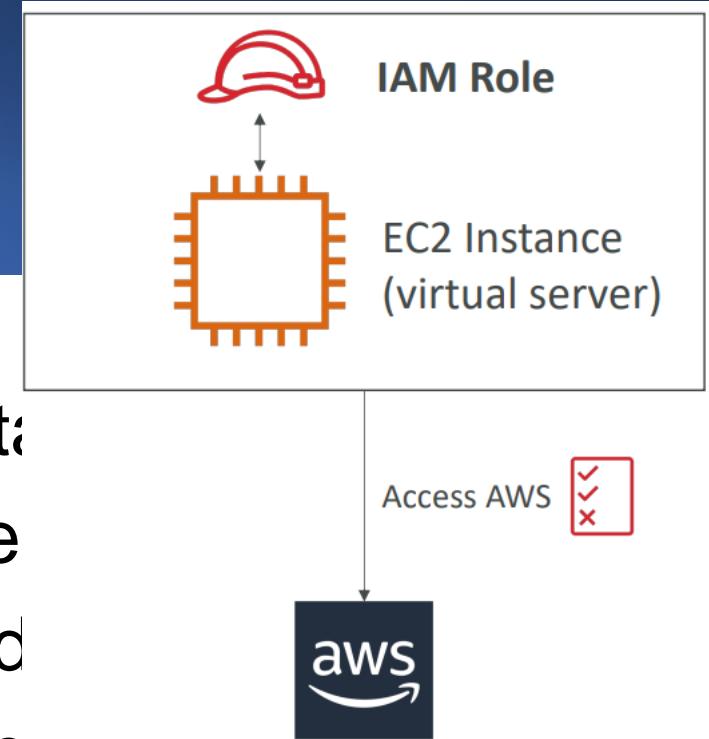
Amazon Web Services (AWS) SDK- Example



- An analogous example of the AWS SDK (Software Development Kit) could be compared to a universal remote control for entertainment systems. Here's how they are similar:
- **Universal Remote Control:** A device that consolidates control of all home entertainment equipment (TV, DVD player, streaming devices) into a single interface.
- **AWS SDK:** A toolkit that consolidates access to various AWS services (computing power, storage, databases) into a single programming interface.
- Just as a universal remote allows you to manage multiple devices without needing a separate remote for each one, the AWS SDK allows developers to interact with multiple AWS services using one set of tools. Each button on the universal remote can be thought of as a function or method in the SDK: you press a button to perform an action, just as you would call a function to perform an operation in AWS.
- For example, using the SDK might look like this in practice:
- Just as you might press the 'power' button on a universal remote to turn on your TV, you might call a method in the AWS SDK to start a virtual server instance.
- Changing the channel with the remote is like using the SDK to switch between different AWS resources.
- Adjusting the volume can be likened to scaling the capacity of an AWS service up or down.

IAM roles for services

- The diagram seems to be showing that an EC2 instance (virtual server) has an IAM role attached to it. This IAM role grants permissions to access other AWS services, denoted by a checkmark. The implication is that the EC2 instance, by assuming this role, can perform actions as allowed by the role's permissions without needing to store AWS credentials on the instance. This practice enhances security by avoiding the need to manage static credentials.



IAM roles for services

- **IAM Roles for Services:** IAM stands for Identity and Access Management, and it is a feature of AWS that helps securely control access to AWS resources. Roles are a secure way to grant permissions that applications can assume when making AWS API calls.
- **Performing Actions on Your Behalf:** Some AWS services require permissions to interact with other services on your behalf. For example, you may have a Lambda function that needs to access an S3 bucket. Instead of embedding credentials within the service, you assign an IAM role with the necessary permissions.
- **Assignment of Permissions:** This is done through IAM Roles, which are entities that define a set of permissions for making AWS service requests. IAM roles are not associated with a specific user or group. Instead, trusted entities assume roles, such as AWS services, applications, or users.
- **Common Roles:**
 - **EC2 Instance Roles:** These are roles that you attach to your EC2 instances to grant permissions to the applications running on them without using static credentials.
 - **Lambda Function Roles:** These are roles that grant AWS Lambda functions permissions to call other AWS services.
 - **Roles for CloudFormation:** These roles are used by AWS CloudFormation to call AWS services on your behalf.

IAM Security tools

- 1.IAM Credentials Report (account-level):** This is a comprehensive report that lists all the IAM users in your AWS account and the status of their various credentials. This report helps you audit the security status of your users' credentials, including passwords, access keys, MFA (Multi-Factor Authentication) devices, and when the credentials were last used.
- 2.IAM Access Advisor (user-level):** The IAM Access Advisor tool shows the service permissions granted to a user and records when those services were last accessed. This can help you identify unused permissions and apply the principle of least privilege, tightening security by revoking unnecessary permissions. It's a way to ensure users have only the permissions they actually use or need, reducing the risk of unauthorized access or actions within your AWS environment.

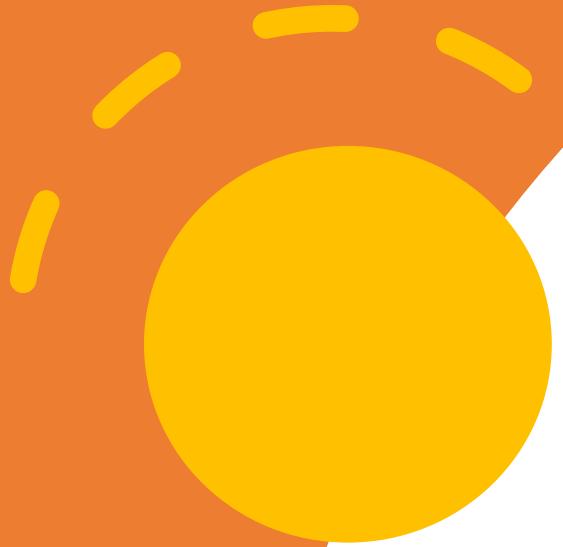
Shared Responsibility Model IAM

AWS's responsibilities include:

- **Infrastructure:** AWS is responsible for the security of the cloud infrastructure, which includes the global network security that spans all the regions, availability zones, and edge locations.
- **Configuration and vulnerability analysis:** AWS handles the configuration of its managed services and performs vulnerability analyses on them.
- **Compliance validation:** AWS ensures that its infrastructure complies with various industry standards and certifications.

The user's responsibilities involve:

- **Users, Groups, Roles, Policies management and monitoring:** Customers must manage their users, groups, roles, and IAM policies, which includes granting the minimum necessary permissions and monitoring activities.
- **Enable MFA (Multi-Factor Authentication) on all accounts:** It's the user's responsibility to enhance security by enabling MFA on their accounts.
- **Rotate all your keys often:** Users should regularly rotate their AWS access keys to minimize risks associated with key leakage or theft.
- **Use IAM tools to apply appropriate permissions:** Leveraging IAM tools to assign and manage permissions accurately is crucial for maintaining the security and functionality of AWS resources.
- **Analyze access patterns & review permissions:** Users should regularly review and adjust permissions based on usage patterns to ensure that the principle of least privilege is followed.

A yellow circular icon resembling a bomb or a sun, positioned on the left side of the slide. It has a yellow circle with a black outline and several short yellow lines radiating from its top edge.

EC2

EC2 instance in AWS

- **Renting virtual machines (EC2):** This allows users to run applications on a virtual server in the AWS cloud, effectively renting the computing power without having to invest in physical hardware.
- **Storing data on virtual drives (EBS):** Amazon Elastic Block Store (EBS) provides block-level storage volumes for persistent data storage for use with EC2 instances.
- **Distributing load across machines (ELB):** Elastic Load Balancing (ELB) distributes incoming application traffic across multiple EC2 instances, improving the scalability and reliability of applications.
- **Scaling the services using an auto-scaling group (ASG):** Auto Scaling helps to maintain application availability and allows the user to scale EC2 capacity up or down automatically according to defined conditions.

Various configuration and sizing options for Amazon EC2 instances

- **Operating System (OS):** You can choose between Linux, Windows, or Mac OS for your EC2 instance depending on your application needs.
- **Compute Power & Cores (CPU):** This refers to the processing power and number of CPU cores. AWS offers a range of EC2 instance types with different CPU configurations to suit various workloads.
- **Random-access Memory (RAM):** The amount of memory available for your instance. Different instance types offer varying amounts of RAM to support applications from small-scale to memory-intensive tasks.
- **Storage Space:**
 - **Network-attached:** This includes Elastic Block Store (EBS) for persistent block storage, and Elastic File System (EFS) for file storage.
 - **Hardware (EC2 Instance Store):** Temporary block-level storage located on disks that are physically attached to the host server.
- **Network Card:** Refers to the networking capabilities of the instance, such as the speed of the network card and the assignment of a Public IP address.
- **Firewall Rules (Security Group):** Security groups act as a virtual firewall for your instance to control inbound and outbound traffic.
- **Bootstrap Script (EC2 User Data):** This is a script that runs when the instance is first launched to perform initial configuration tasks automatically.

EC2 instance User Data

- EC2 User Data allows the execution of a script or a set of commands when an EC2 instance is first launched. This process is known as **bootstrapping**. The user data script runs only once, during the initial start of the instance, and it is typically used for tasks such as:
- Installing updates to the operating system or software packages.
- Installing specific software required for the operation of the instance.
- Downloading files from the internet that may be needed immediately upon startup.
- The EC2 User Data Script is executed with root user privileges, which means it has the ability to perform tasks that require administrative permissions without any restrictions. This feature is powerful for automating the setup process of EC2 instances, ensuring that they are configured correctly and uniformly as soon as they become operational.

EC2 instance User Data

- Hands on practice

EC2 Instance User Data

- EC2 User Data allows the execution of a script or a set of commands when an EC2 instance is first launched. This process is known as **bootstrapping**. The user data script runs only once, during the initial start of the instance, and it is typically used for tasks such as:
- Installing updates to the operating system or software packages.
- Installing specific software required for the operation of the instance.
- Downloading files from the internet that may be needed immediately upon startup.
- The EC2 User Data Script is executed with root user privileges, which means it has the ability to perform tasks that require administrative permissions without any restrictions. This feature is powerful for automating the setup process of EC2 instances, ensuring that they are configured correctly and uniformly as soon as they become operational.

EC2 instance Type

m5.2xlarge

- '**mm**' class instances are General Purpose instances, providing a balance of compute, memory, and networking resources, and can be used for a variety of diverse workloads.
- '**5- '**2xlargemicro**' instances to '**12xlarge**' instances to accommodate different workloads and performance requirements.
- The instance types are optimized for different use cases, such as **compute optimized, memory optimized, accelerated computing, storage optimized, and more.**
[\(https://aws.amazon.com/ec2/instance-types/\)](https://aws.amazon.com/ec2/instance-types/)**

EC2 Instance Types-General Purpose

- **General Purpose EC2 Instances:** These instances are designed to provide a balance between compute, memory, and networking resources. They are suitable for a variety of workloads, including web servers and code repositories, making them a versatile choice for many applications.
- **Workloads:** The slide mentions that General Purpose instances are great for diverse workloads, specifically highlighting web servers or code repositories.
- **Balance of Resources:** It emphasizes that General Purpose instances offer a balance between the following:
 - Compute power
 - Memory
 - Networking capabilities
- **t2.micro:** For the context of the course or presentation, the **t2.micro** instance type is mentioned as the specific General Purpose EC2 instance that will be used. The **t2.micro** is known for being a cost-effective option for small-scale applications and is part of the free tier that AWS offers, making it a common choice for educational purposes and small-scale experiments.
- **Instance Classes:** **T4g, T3, T3a, T2, M6g, M5, M5a, M5n, M5zn, M4, and A1.** These represent different subcategories within the General-Purpose instance family, each with its own set of capabilities and price points.
- [Link to EC2 Instances General-Purpose](#)

PAGE CONTENT

General Purpose

Compute Optimized

Memory Optimized

Accelerated Computing

Storage Optimized

General Purpose

General purpose instances provide a balance of compute, memory and networking resources, and can be used for a variety of diverse workloads. These instances are ideal for applications that use these resources in equal proportions such as web servers and code repositories.

M7g	M7i	M7i-flex	M7a	Mac	M6g	M6i	M6in	M6a	M5	M5n	M5zn	M5a
M4	T4g	T3	T3a	T2								

[Amazon EC2 T2 instances](#) are Burstable Performance Instances that provide a baseline level of CPU performance with the ability to burst above the baseline.

EC2 Instance Types- Compute optimized

- **Compute Optimized EC2 Instances:** These instances are ideal for compute-bound applications that benefit from high-performance processors. They are optimized for tasks that require more processing capability.
- **Suitable Workloads:** Various compute-intensive tasks that Compute Optimized instances are well-suited for, including:
 - Batch processing workloads
 - Media transcoding
 - High performance web servers
 - High performance computing (HPC)
 - Scientific modeling and machine learning
 - Dedicated gaming servers
- **Instance Classes:** C6g, C6gn, C5, C5a, C5n, C4. Each of these instance classes offers different specifications and pricing to cater to the specific needs of compute-intensive applications.
- Compute Optimized instances are particularly useful for applications that need a lot of CPU power. Examples include scientific simulations, gaming servers, and machine learning inference tasks that can benefit from the high-performance computing capabilities these instances provide.
- [Link to EC2 Instances Computed-Optimized](#)

Compute Optimized

Compute Optimized instances are ideal for compute bound applications that benefit from high performance processors. Instances belonging to this category are well suited for batch processing workloads, media transcoding, high performance web servers, high performance computing (HPC), scientific modeling, dedicated gaming servers and ad server engines, machine learning inference and other compute intensive applications.

C7g	C7gn	C7i	C7a	C6g	C6gn	C6i	C6in	C6a	C5	C5n	C5a	C4
Amazon EC2 C7g instances are powered by Arm-based AWS Graviton3 processors. They deliver the best price performance in Amazon EC2 for compute-intensive applications.												

EC2 Instance Types- Memory optimized

- **Memory Optimized EC2 Instances:** These instances are designed to deliver fast performance for workloads that require large data sets to be processed in memory.
- **Use Cases:**
 - High performance for both relational and non-relational databases.
 - Suitable for distributed web-scale cache stores, which require large amounts of memory to store temporary data for fast access.
 - Ideal for in-memory databases optimized for BI (Business Intelligence), allowing for quick retrieval and analysis of data.
 - Well-suited for applications that perform real-time processing of big unstructured data, which demands significant memory resources for efficient processing.
- **Instance Types:** R6g, R5, R5a, R5b, R5n, R4, X1e, X1, have High Memory instances and z1d. Each of these classes is tailored to various memory-intensive applications and has different specifications and pricing.
- Memory Optimized instances are particularly useful for applications that need to process large amounts of data quickly and efficiently in RAM, as opposed to relying on disk-based storage which is comparatively slower. This includes applications such as real-time big data analytics, high-performance.
- [Link to EC2 Instances Memory-optimized](#)

Memory Optimized

Memory optimized instances are designed to deliver fast performance for workloads that process large data sets in memory.

R8g	R7g	R7i	R7iz	R7a	R6g	R6i	R6in	R6a	R5	R5n	R5b	R5a	R4
X2gd	X2idn	X2iedn	X2iezn	X1	X1e	High Memory	z1d						

[Amazon EC2 R8g instances](#), now available in preview, are powered by AWS Graviton4 processors. They are ideal for memory-optimized applications.

EC2 Instance Types- Memory optimized

- **Memory Optimized EC2 Instances:** These instances are designed to deliver fast performance for workloads that require large data sets to be processed in memory.
- **Use Cases:**
 - High performance for both relational and non-relational databases.
 - Suitable for distributed web-scale cache stores, which require large amounts of memory to store temporary data for fast access.
 - Ideal for in-memory databases optimized for BI (Business Intelligence), allowing for quick retrieval and analysis of data.
 - Well-suited for applications that perform real-time processing of big unstructured data, which demands significant memory resources for efficient processing.
- **Instance Types:** R6g, R5, R5a, R5b, R5n, R4, X1e, X1, have High Memory instances and z1d. Each of these classes is tailored to various memory-intensive applications and has different specifications and pricing.
- Memory Optimized instances are particularly useful for applications that need to process large amounts of data quickly and efficiently in RAM, as opposed to relying on disk-based storage which is comparatively slower. This includes applications such as real-time big data analytics, high-performance.
- [Link to EC2 Instances Memory-optimized](#)

Memory Optimized

Memory optimized instances are designed to deliver fast performance for workloads that process large data sets in memory.

R8g	R7g	R7i	R7iz	R7a	R6g	R6i	R6in	R6a	R5	R5n	R5b	R5a	R4
X2gd	X2idn	X2iedn	X2iezn	X1	X1e	High Memory	z1d						

[Amazon EC2 R8g instances](#), now available in preview, are powered by AWS Graviton4 processors. They are ideal for memory-optimized applications.