**AWS Cloud Practitioner Essentials**

**MODULE 1 - THE AWS CLOUD**

***What is Cloud Computing?***

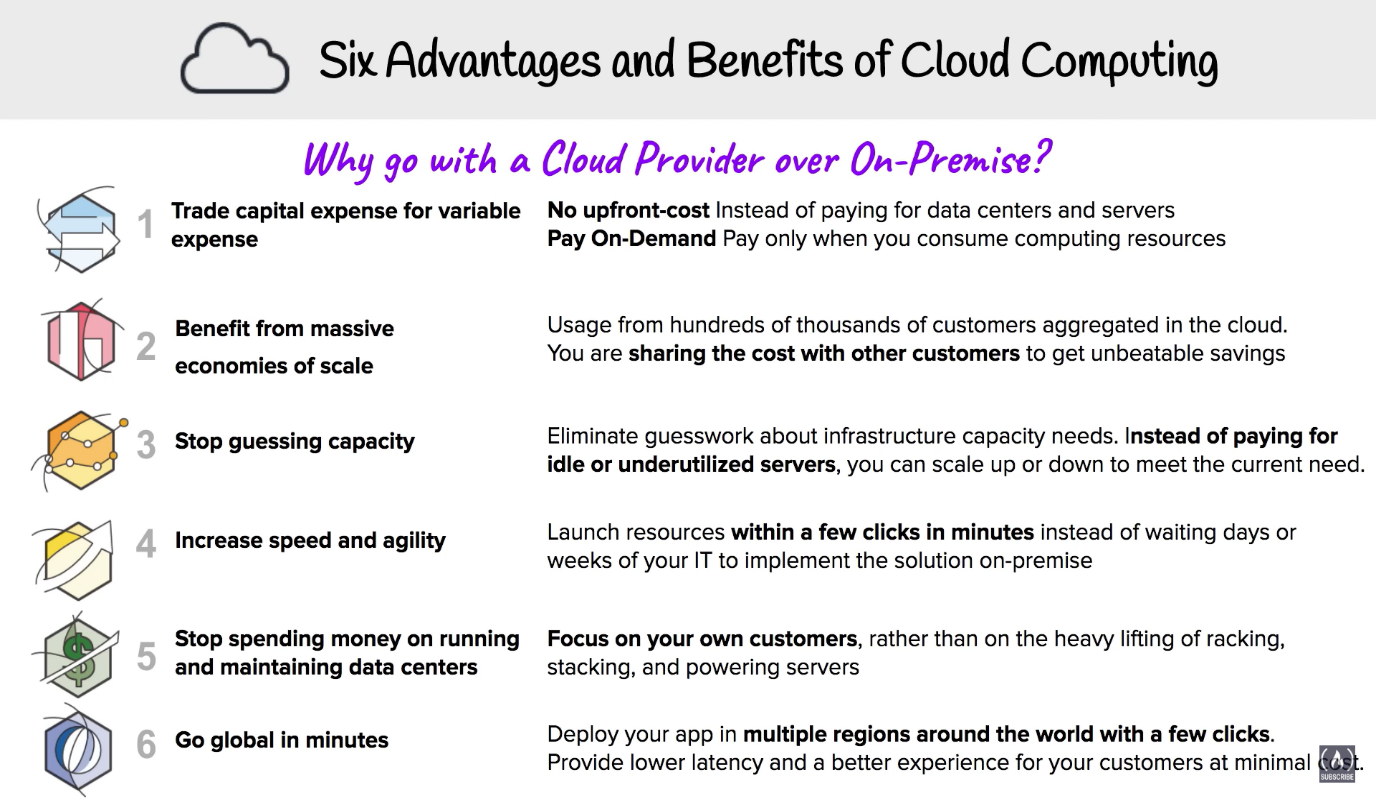
**Cloud Computing** is the **on-demand delivery of IT resources** over the internet with **pay-as-you-go pricing**.

Instead of owning your own data centers or servers, you can rent computing power, storage, and databases from a cloud provider like **AWS**.

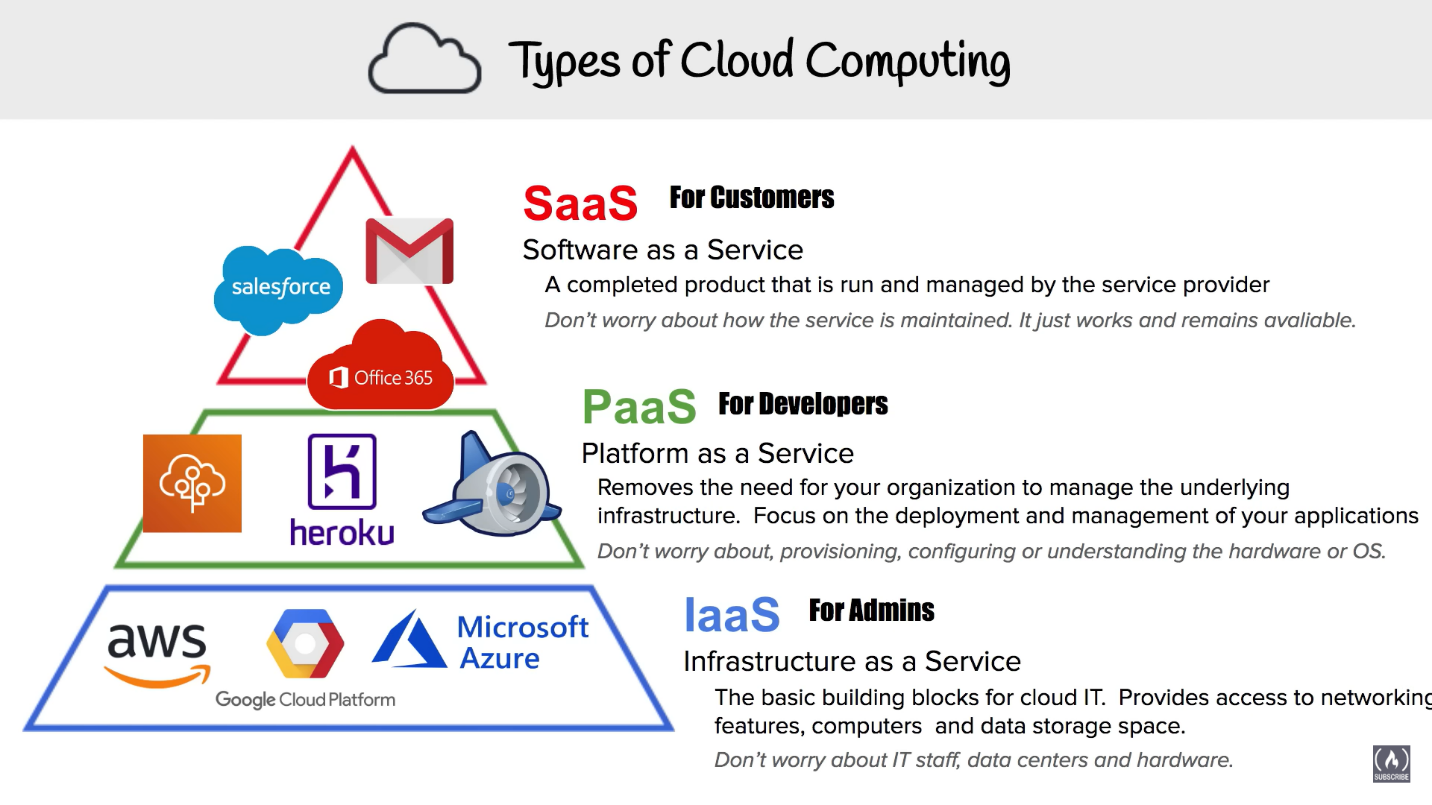
***Key Features***

* **On-demand**: You get what you need when you need it.
* **Scalable**: Easily grow (or shrink) resources.
* **Flexible**: You can run almost any kind of application.
* **Pay-as-you-go**: You pay only for what you use.

***Benefits of the AWS Cloud***



***Types of Cloud Computing***

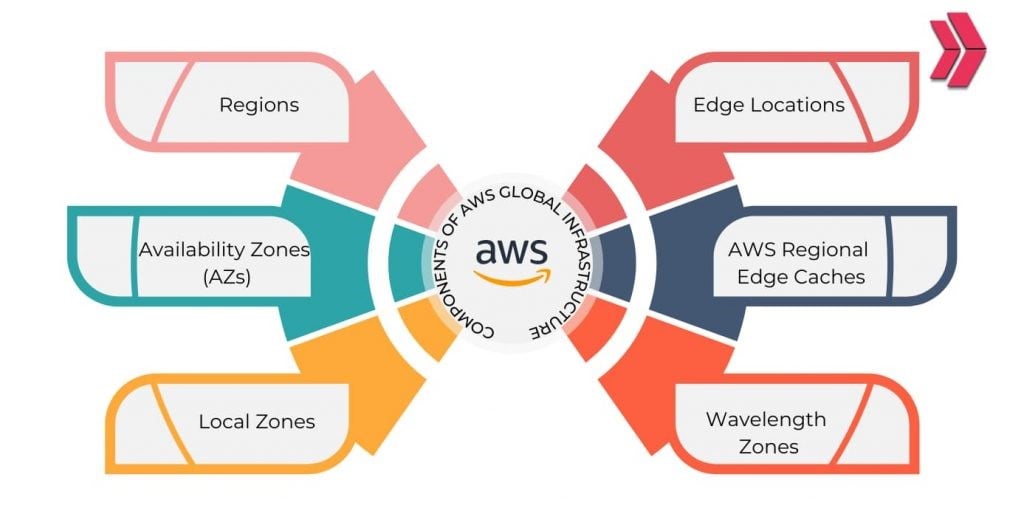


***Cloud Computing Deployment Models***

| ***Model*** | ***Description*** | ***Best For*** | ***Example*** |
| --- | --- | --- | --- |
| ***Public Cloud*** | Cloud services are hosted by third-party providers (like AWS) and shared among multiple customers. | Startups, businesses that want to reduce IT costs | AWS EC2, S3, Lambda |
| ***Private Cloud*** | Cloud infrastructure is exclusively used by one organization. It can be hosted on-premises or in a private data center. | Banks, government, companies needing high security | AWS Outposts, VMWare on AWS |
| ***Hybrid Cloud*** | Combines public and private cloud. Some workloads run on AWS, others on local servers. | Organizations with data privacy or compliance needs | AWS Direct Connect, VPN Gateway |

***Introduction to AWS Global Infrastructure***The AWS Global Infrastructure is a robust and globally distributed network designed to deliver cloud services with high speed, availability, and security. At its foundation, AWS is divided into Regions, which are physical geographic areas like Asia Pacific (Mumbai) or US East (N. Virginia). Each Region is made up of multiple Availability Zones (AZs)—these are isolated data centers, each with independent power, cooling, and networking. AZs allow businesses to run applications in a fault-tolerant way by distributing workloads across multiple data centers in the same region.

In addition to Regions and AZs, AWS also uses Edge Locations, which are data centers located closer to end users. These are primarily used for content delivery through services like Amazon CloudFront, allowing static or dynamic content (such as videos or websites) to be delivered quickly, even across continents. This global setup ensures that AWS customers can deploy applications with low latency, high availability, and geographical flexibility.

******

***The AWS Shared Responsibility Model***

The **Shared Responsibility Model** is a core security framework in AWS, clearly outlining which responsibilities are handled by AWS and which fall to the customer. AWS is responsible for **“Security *of* the cloud”**—this includes maintaining the physical infrastructure, networking, hardware, software, and facilities that run AWS services. Essentially, AWS secures the foundation and core infrastructure that supports the cloud platform.

On the other hand, customers are responsible for **“Security *in* the cloud”**. This means managing their data, configuring their applications securely, setting up proper access controls using services like **AWS IAM (Identity and Access Management)**, and enabling encryption when needed. For example, while AWS ensures your virtual server (like an EC2 instance) is running securely, it’s your responsibility to secure the data on it, patch your own software, and control who has access.

This shared approach ensures that both AWS and its users contribute to creating a **secure and resilient cloud environment**. AWS provides the tools and infrastructure, but it's up to the user to use them wisely.

**Module 2 - Compute in the Cloud**

***Introduction to Amazon EC2***

* EC2 (Elastic Compute Cloud) is a web service that provides resizable virtual servers in the cloud.
* You get full control over the OS, storage, networking, and security.
* Use Cases:
  + Hosting apps and websites
  + Running APIs, databases, and game servers
  + Machine learning and batch processing

*Why it matters: No need for physical servers. Spin up infrastructure in minutes,*

*scale easily and pay only for usage****.***

***Amazon EC2 Instance Types***

| ***Family*** | ***Use Case*** | ***Examples*** |
| --- | --- | --- |
| ***General Purpose*** | *Balanced CPU/RAM* | *t3, t4g, m6i* |
| ***Compute Optimized*** | *High CPU workloads* | *c6g, c7i* |
| ***Memory Optimized*** | *Large memory apps* | *r6g, x2idn* |
| ***Storage Optimized*** | *Fast disk performance* | *i4i, d3en* |
| ***Accelerated*** | *GPU/ML workloads* | *p4, g5* |

*Why it matters: You can choose the right tool for the job — cost-effective, performance-optimized.*

***How to Provision AWS Resources***

Provisioning means setting up and configuring an EC2 instance:

1. Choose AMI
2. Pick Instance Type
3. Configure Network & Subnet
4. Attach EBS Volumes
5. Add Tags
6. Set Security Groups
7. Generate or choose a Key Pair
8. Launch and connect via SSH

*Why it matters: AWS makes it easy to build infrastructure that's consistent and repeatable — either via console, CLI, or infrastructure-as-code.*

***Amazon EC2 Pricing***

| ***Model*** | ***Description*** | ***Best For*** | ***Savings*** |
| --- | --- | --- | --- |
| ***On-Demand*** | *Pay per hour/second* | *Testing, short-term* | *–* |
| ***Reserved*** | *1-3 year commitment* | *Long-term workloads* | *Up to 72%* |
| ***Spot*** | *Spare capacity (can be interrupted)* | *Batch jobs, ML, CI/CD* | *Up to 90%* |
| ***Savings Plans*** | *Commit to spend* | *Flexible workloads* | *Up to 72%* |

***Scaling Amazon EC2***

Auto Scaling ensures your app always has:

* Enough EC2 instances during peak load
* Minimum instances during low traffic
* High availability with health checks

Auto Scaling Group (ASG) configures:

* Min/Max/Desired capacity
* Scaling policies (based on CPU, time, predictive ML)
* Integration with CloudWatch, Load Balancers, EC2 Launch Templates

***Directing Traffic with Elastic Load Balancing***

ELB distributes traffic across EC2s to balance load and handle failures.

Features:

* Cross-AZ balancing
* Health checks
* SSL termination
* Sticky sessions
* Access logs

***Messaging and Queuing***

Allows asynchronous, decoupled communication between components:

| ***Service*** | ***Use Case*** |
| --- | --- |
| ***Amazon SQS*** | *Queue messages; workers process later (decoupling, retry logic)* |
| ***Amazon SNS*** | *Publish/subscribe notifications to multiple endpoints* |
| ***Amazon MQ*** | *Traditional protocols (AMQP, MQTT) for migrating legacy systems* |

SQS types:

* Standard – high throughput, best-effort order
* FIFO – strict order, exactly-once delivery

Use queues and topics to buffer load, retry failed jobs, broadcast events.

**Module 3 - Exploring Compute Services**

***Introduction to Serverless Computing***

Serverless computing means you focus only on your code — AWS manages the underlying infrastructure like servers, scaling, availability, and patching.

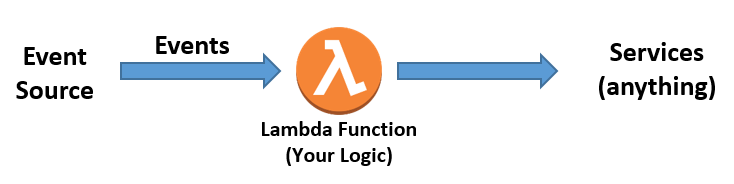
Key Features:

* Event-driven execution
* Automatic scaling (0 to 1000s of concurrent executions)
* Pay only for runtime (no idle cost)
* Stateless functions
* No server provisioning or OS management

Used for tasks that run on-demand, like processing file uploads, responding to API calls, or sending notifications — without needing servers running 24/7.

***AWS Lambda***

AWS Lambda is AWS’s serverless compute service.  
You write a function, define triggers, and Lambda executes it only when needed.

******

***Containers and Orchestration on AWS***

AWS Container Services offer fully managed container orchestration with AWS native solutions and open-source technologies. With flexible compute options that reduce operational overhead, they enable teams to focus on innovation and drive business value faster.

******

**Module 4 - Going Global**

***Introduction to Going Global***

Going global in AWS means deploying your applications and services across multiple geographic locations (Regions), to serve users around the world efficiently.

Key Benefits:

* Reduced latency
* Higher availability and fault tolerance
* Easier compliance with local data laws
* Ability to scale and replicate apps across continents

AWS handles the physical infrastructure — you just choose where to deploy.

***Choosing AWS Regions***

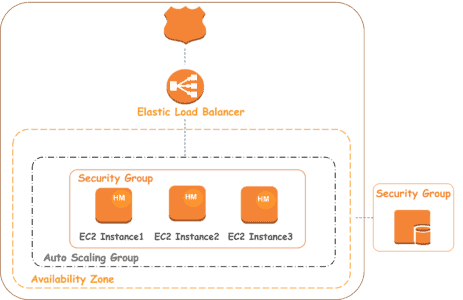
| ***Factor*** | ***Reason*** |
| --- | --- |
| ***Latency*** | *Choose regions closer to your end-users* |
| ***Cost*** | *Prices vary per region* |
| ***Service Availability*** | *Some services are limited to certain regions* |
| ***Compliance*** | *Some data must stay in specific countries (GDPR, HIPAA)* |
| ***Redundancy*** | *Use second region for disaster recovery or backup* |

***Infrastructure and Automation***

AWS allows you to **define, deploy, and manage infrastructure using code** — for speed, repeatability, and scalability.

***Benefits of Infrastructure as Code (IaC):***

* Repeatable, version-controlled setups
* One-click or script-based multi-region deployment
* Fast disaster recovery setups
* Global standardization of environments



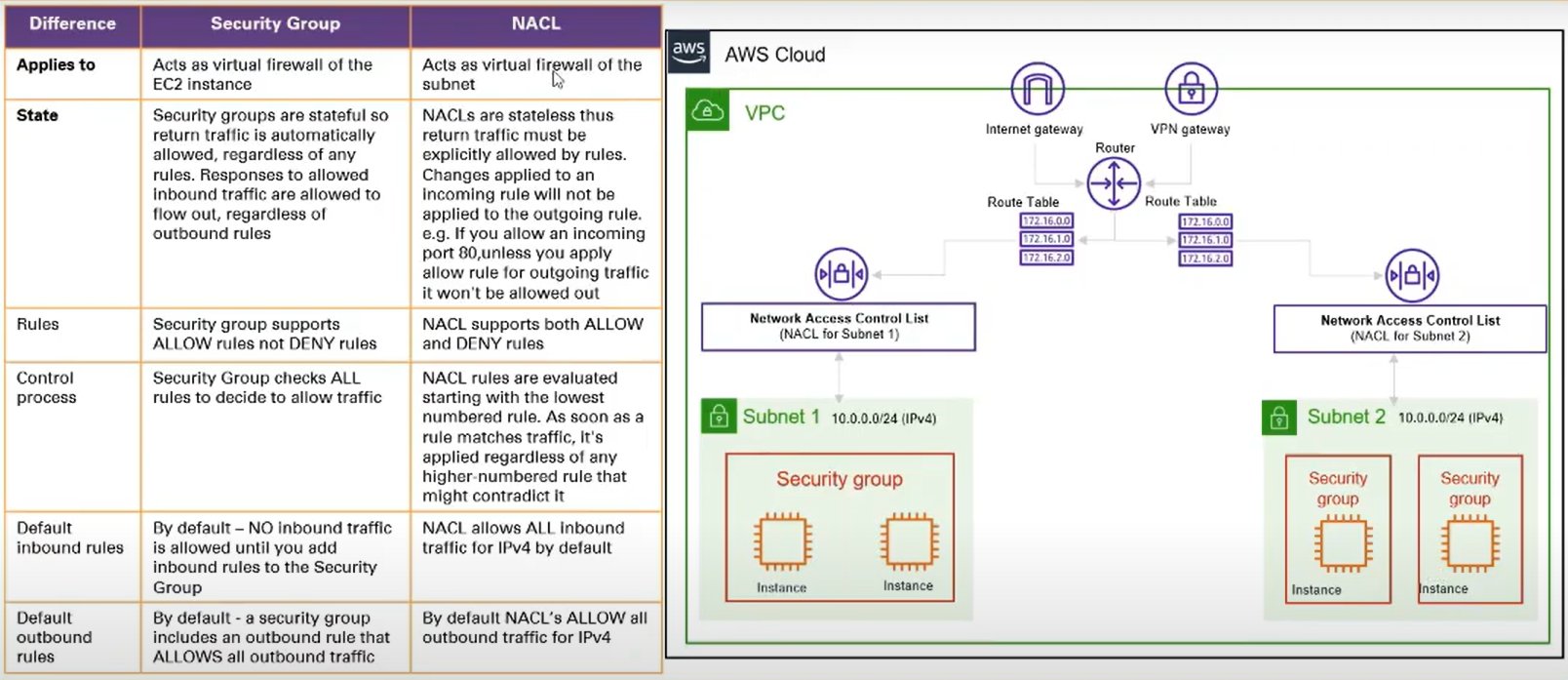
**Module 5 – Networking**

***Introduction to Networking***

Networking forms the backbone of cloud communication. AWS uses virtual networking to replicate traditional setups with better scalability, automation, and security.  
  
***Organizing AWS Cloud Resources***

* VPC (Virtual Private Cloud): Your own isolated network in AWS.
* Includes subnets, route tables, Internet Gateways, and NAT Gateways to manage traffic and segmentation.
* Resources are grouped logically for access control and security.

***Subnets, Security Groups, and Network Access Control Lists***

******

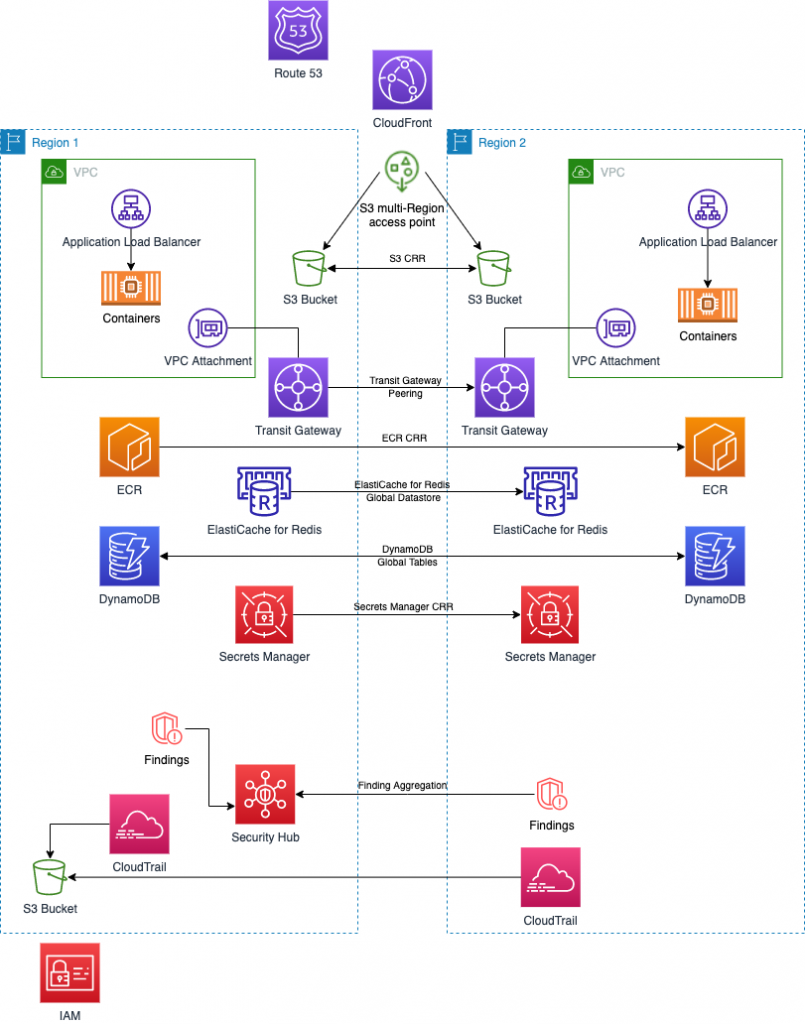
***Amazon VPC Demo***

******

***Global Networking***

* ***VPC Peering & Transit Gateway: Connect VPCs across accounts/regions.***
* ***Global Accelerator: Improves global app performance by routing user traffic via AWS backbone.***
* ***CloudFront: Delivers cached content globally.***

***Global Architectures***



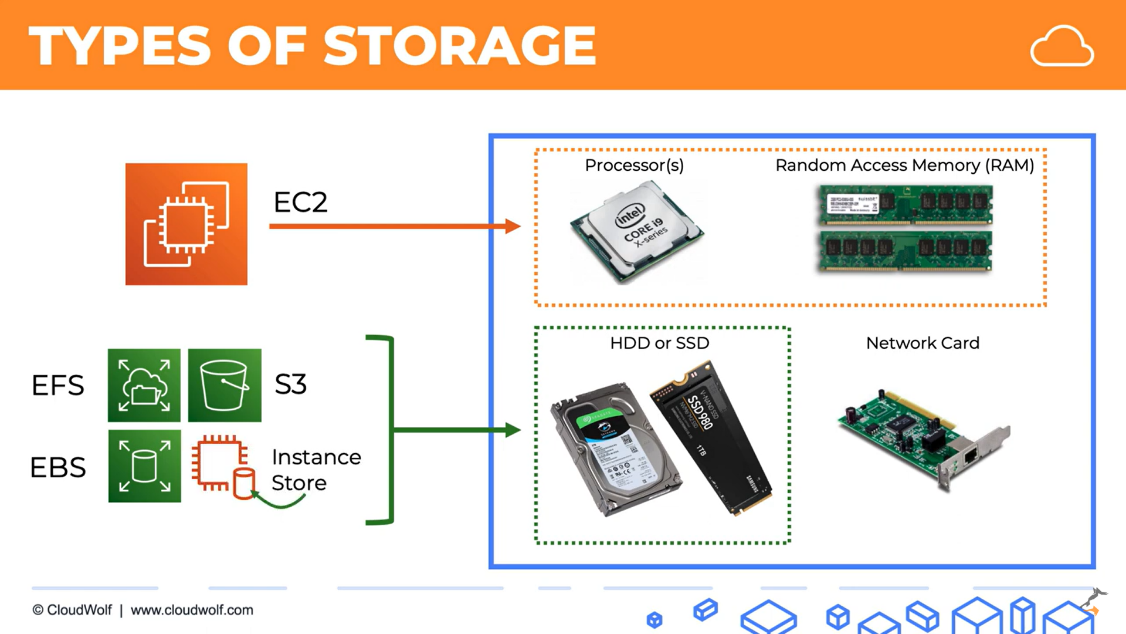
**Module 6 - Storage**

***Introduction to Storage***

* AWS offers multiple types of cloud storage: Block, Object, File, and Hybrid.
* Choice depends on use case: latency, access pattern, cost, sharing, etc.

AWS offers **diverse storage options**, each optimized for different use cases:

* **EBS** for high-performance, persistent block storage
* **S3** for scalable object storage with lifecycle controls
* **EFS & FSx** for file-based workloads with POSIX/SMB support
* **Storage Gateway** for hybrid access to S3
* **AWS DRS** for failover and recovery



***Block Storage***

**Amazon EBS & Instance Store**

* EBS: Persistent, high-performance block storage for EC2; supports backups via snapshots and encryption.
* Instance Store: Temporary block storage physically attached to the host; data lost on stop/terminate.

**EBS Data Lifecycle**

* Create → Attach → Use → Snapshot → Restore → Delete.
* Snapshots stored in S3 (incremental) and can be automated with Lifecycle Manager.
* Supports point-in-time recovery and cross-region DR.

***Object Storage***

**Amazon S3 (Simple Storage Service)**

* Stores objects (files + metadata) in **buckets**; accessible via HTTP/S.
* Extremely scalable, durable (11 9s), and used by apps, websites, and AWS services.
* Integrates with versioning, encryption, events (e.g., Lambda), and access policies.

**S3 Storage Classes & Lifecycle**

* Classes: Standard, Intelligent-Tiering, Standard-IA, One Zone-IA, Glacier, Deep Archive.
* Lifecycle Rules: Automate transition of objects to cheaper storage or delete them after X days.
* Great for cost optimization on logs, backups, static media, and archives.

**S3 Demo Recap**

* Steps: Create bucket → Upload objects → Set permissions → Enable versioning → Apply lifecycle → Trigger events (optional).
* Access via public URLs or signed requests.

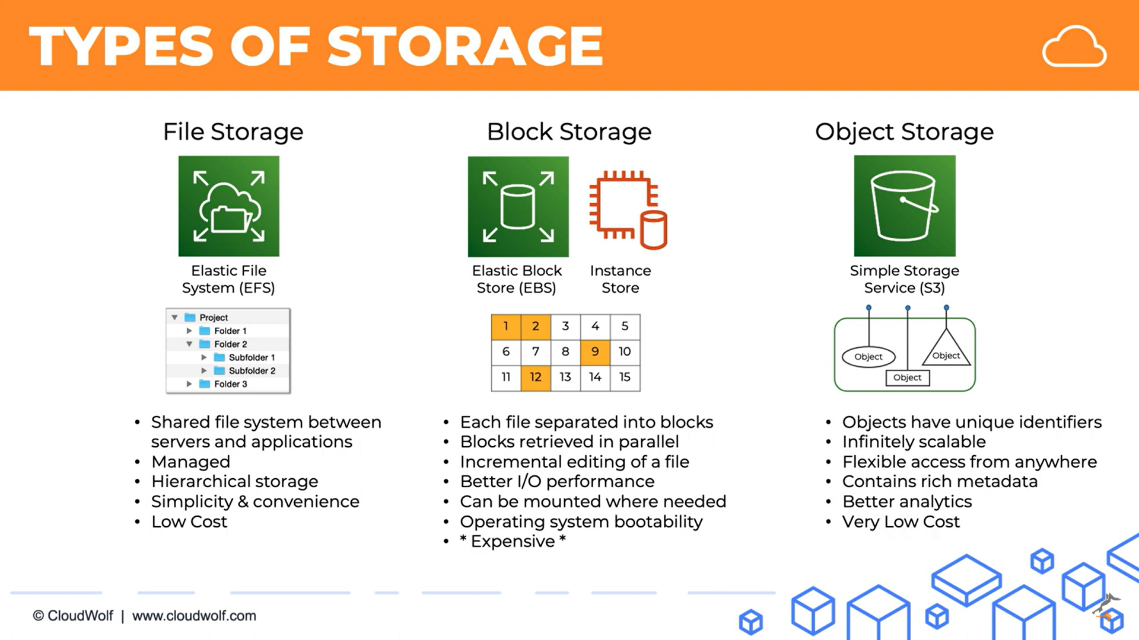
***File Storage***

**Amazon EFS (Elastic File System)**

* POSIX-compliant shared file system; accessible by multiple EC2s.
* Auto-scales with usage, offers Standard and Infrequent Access modes.
* Ideal for web servers, ML workloads, and container apps needing shared storage.

**Amazon FSx**

* Managed Windows/Linux file systems:
  + FSx for Windows File Server (SMB, AD integration)
  + FSx for Lustre (High performance for ML/analytics)
  + FSx for NetApp ONTAP (enterprise NAS)
  + FSx for OpenZFS (DevOps, ZFS-based apps)



***Hybrid & Recovery***

**AWS Storage Gateway**

* Bridges on-prem infrastructure to AWS.
* Types:
  + File Gateway → S3-backed NFS/SMB file shares
  + Volume Gateway → iSCSI block volumes backed by EBS Snapshots
  + Tape Gateway → Virtual tape backups to Glacier
* Great for gradual migration, backups, and compliance use cases.

A diagram of a storage gateway

AI-generated content may be incorrect.

***AWS Elastic Disaster Recovery (DRS)***

* Continuous replication of on-prem or cloud-based servers to AWS.
* Allows fast failover and recovery (low RTO/RPO).
* Supports physical, VMware, Hyper-V, and EC2



**Module 7 - Databases**

**Introduction to Databases**

* Databases store and manage structured or unstructured data.
* Types: **Relational**, **NoSQL**, and **In-memory**.
* Cloud databases remove manual maintenance and offer built-in scalability, backups, and security.

**Relational Database Services (Amazon RDS)**

* Fully managed relational DB supporting engines like **Aurora**, **MySQL**, **PostgreSQL**, **Oracle**, and **SQL Server**.
* Supports features like **multi-AZ failover**, **replicas**, **automated backups**, and **encryption**.
* Best for apps needing structured schema, transactions, and SQL support.

**NoSQL Database Services (Amazon DynamoDB)**

* Serverless key-value and document database.
* Auto-scales with demand and supports **microsecond latency**, **global tables**, and **DAX caching**.
* Ideal for real-time apps, gaming, IoT, and product catalogs.

A screenshot of a computer

AI-generated content may be incorrect.

***In-Memory Caching Services (Amazon ElastiCache)***

* Fully managed **Redis** and **Memcached** for high-speed caching.
* Reduces load on databases and improves app responsiveness.
* Common use cases: session storage, API rate limiting, real-time leaderboards.

