

# Cloud Fundamentals

## On this page

[What is Cloud? What are its advantages?](#)

[Region vs Zones vs EdgeLocations](#)

[On-premise vs Hybrid Cloud vs Multi Cloud](#)

[IaaS vs PaaS vs SaaS vs Serverless](#)

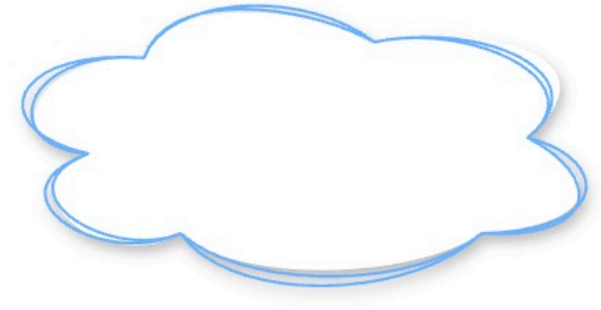
[What is Virtualization?](#)

[How can you create Virtual Machines in the cloud?](#)

[What are some of the architectural aspects to think about when creating virtual machines in the cloud?](#)

[What are the Compute Options in Cloud?](#)

## What is Cloud? What are its advantages? <#>

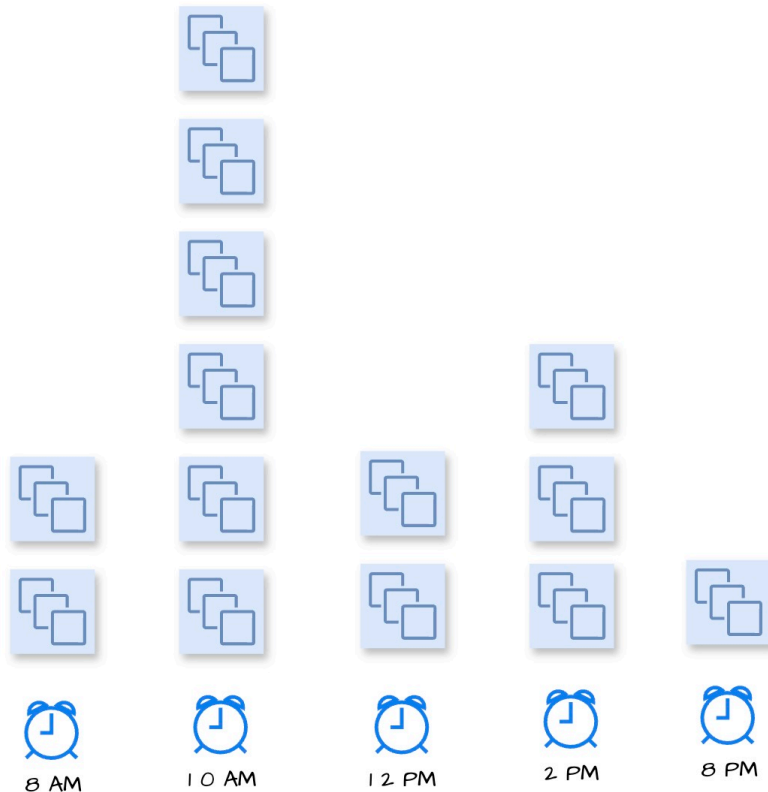


### What is Cloud?

- **Scenario:** Imagine you need to setup 100 servers for a website you are building. Can we rent instead of buy?
- **Cloud:** Rent computing, storage, database, and other services from companies like AWS, Azure, Google Cloud, .. – whenever you need them.

### Cloud is On-Demand

- **No Upfront Investment:** No need to buy servers or data centers
- **Faster Start:** Launch apps in minutes, not months
- **Pay-As-You-Go:** Pay only for what you use



### Cloud is Scalable

- **Elastic - Scale Up/Down:** Add or remove resources as needed

- **Automatic Scaling:** Many services scale based on demand
- **Global Reach:** Deploy apps closer to your users across the world



### Benefits of Cloud

- **Faster Delivery:** Launch apps in minutes
- **Cost-Efficient:** Pay only for what you use
- **Scalable:** Auto-scale to handle traffic spikes
- **Global:** Deploy close to users, anywhere in the world
- **Secure & Compliant:** Built-in security and certifications

## Region vs Zones vs EdgeLocations #

## Region vs Zones vs Edge Locations

- **Scenario:** You want your app to be fast, reliable, and globally available. How does the cloud provider structure its infrastructure to help you do that?
- **Answer:** Using **Regions**, **Zones**, and **Edge Locations** – each plays a different role.

### Region

- **What It Is:** A **geographical area** (e.g., Mumbai, US-East)
- **Use Case:** Choose a region close to your users or data for low latency and compliance



### Region Advantages

- **High Availability:** Even if a region is down, you can serve from other regions
- **Low Latency:** Deploy closer to users for faster response
- **Global Footprint:** Reach users worldwide without owning infrastructure
- **Adhere to government regulations:** Keep data in-region to meet compliance laws



### Zone

- **What It Is:** One or more **data centers** inside a region
- **Isolated but Connected:** Physically separated but connected with low latency
- **Use Case:** Deploy across zones for **high availability and fault tolerance** (while being in same region)

### Edge Location

- **What It Is:** A **content delivery endpoint** near users
- **Key Thing To Remember:** Count of Edge Locations is far greater than Count of Regions
- **Smaller than AZs:** Focused on caching and request routing
- **Use Case:** Deliver content faster via **CDN** (e.g., CloudFront)



#### Edge Location Use Case – Step by Step

- **Step 01:** A user requests a video or image from your website (e.g., `https://yourapp.com/logo.png`)
- **Step 02:** The request is routed to the nearest **edge location** (a local server near the user)

- **Step 03:** If the content is cached at the edge, it is served instantly — no need to reach the main server
- **Step 04:** If not cached, the edge location fetches it from the origin server
- **Step 05:** The content is delivered to the user and simultaneously **cached at the edge** for future requests
- **Step 06:** Next time another user in the same region makes the same request, it's delivered directly from the edge — faster and more efficient
- Edge locations improve **performance**, **reduce latency**, and **offload origin servers**, especially for static content and media files

#### What are Multi Regions?

- **Multi Regions:** Multiple geographically separate cloud regions are logically linked to enable **geo-redundancy**, **disaster recovery**, and **resilient architecture**

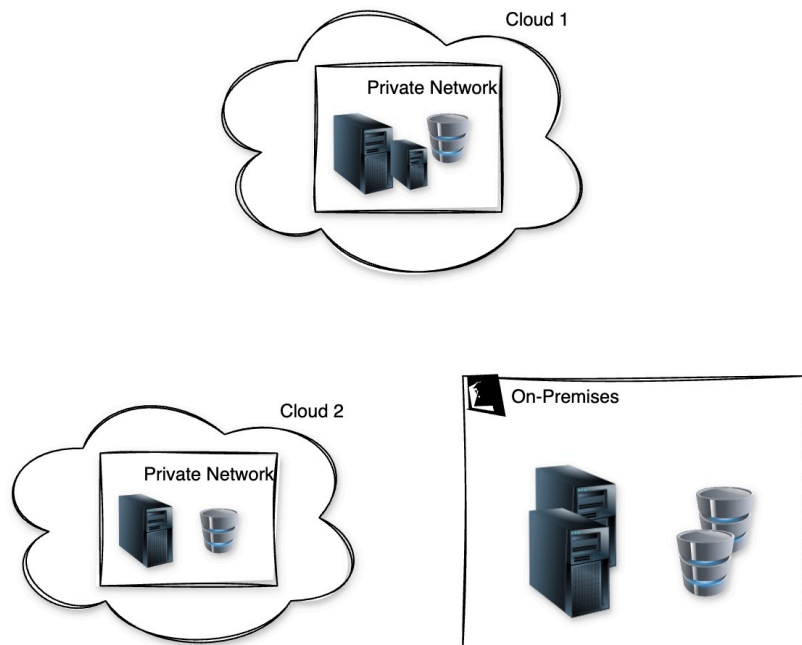
#### Key Benefits of Multi Regions

- **High Availability:** Data is replicated to multiple regions
- **High Durability:** Multiple copies of data are stored across both regions

| Concept                    | AWS  | Azure  | Google Cloud (GCP)  |
|----------------------------|--|--|---|
| Region                     | <b>Region</b><br>E.g., us-east-1   | <b>Region</b><br>E.g., East US ,<br>West Europe                                      | <b>Region</b><br>E.g., us-central1 ,<br>europe-west1                  |
| Zone                       | <b>Availability Zone</b><br>E.g., us-east-1a   | <b>Availability Zone</b><br>E.g., East US 2<br>Zone 1                                | <b>Zone</b><br>E.g., us-central1-a                                    |
| Edge Location              | <b>Edge Location</b><br>(CloudFront POP)<br>E.g., Hyderabad                                | <b>Point of Presence (POP)</b> used in<br>Azure Front Door /<br>Azure CDN            | <b>Edge Location</b><br>/ <b>POP</b> used in<br>Cloud CDN             |
| Multi-Region / Dual-Region | ✗ Not explicitly named; use features like <b>Cross-Region Replication, Global DynamoDB</b> | ✓ <b>Region Pairs</b><br>(Dual Regions)<br>Automatically paired for <b>DR</b> , etc. | ✓ <b>Multi-Region</b> is natively supported<br>E.g., us , eu , asia - |

| Concept | AWS | Azure | Google Cloud (GCP)   |
|---------|-----|-------|----------------------|
|         |     |       | <b>BigQuery, GCS</b> |

## On-premise vs Hybrid Cloud vs Multi Cloud <#>



## 1. On-Premise

- **What It Is:** All infrastructure is **owned and operated by you**
- **Where It Runs:** In your **own data center**
- **Pros:**
  - Full control over hardware and security

- No dependency on external providers

- **Cons:**

- Expensive upfront cost
- Hard to scale quickly
- Slower to adopt modern tools

## 2. Hybrid Cloud

- **What It Is:** A **mix of on-prem + public cloud**
- **Where It Runs:** Some workloads on-prem, some in cloud
- **Pros:**
  - Flexibility to keep sensitive data on-prem
  - Leverage cloud scale for web apps or backups
  - Step-by-step cloud adoption
- **Cons:**
  - Complex setup and integration
  - Needs strong network and security architecture
  - Harder to manage consistently

## 3. Multi-Cloud

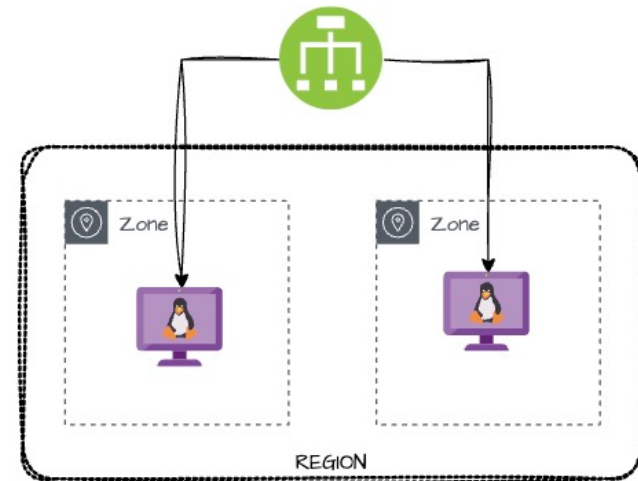
- **What It Is:** Use of **2 or more public cloud providers** (e.g., AWS + Azure) with or without on-premises
- **Why Use It:** Avoid vendor lock-in, optimize cost/performance

- **Pros:**
  - Use best features from each provider
  - Increase availability and redundancy
  - Competitive pricing and negotiation
- **Cons:**
  - Higher complexity in architecture and management
  - Requires multi-skilled teams
  - Data sync challenges

## IaaS vs PaaS vs SaaS vs Serverless #

### IaaS vs PaaS vs SaaS vs Serverless

- **Scenario:** You want to build, deploy, or use an application. How much do you want to manage yourself vs how much should the cloud provider handle for you?
- **Goal:** Choose the right cloud service model based on **control**, **convenience**, and **responsibility**.

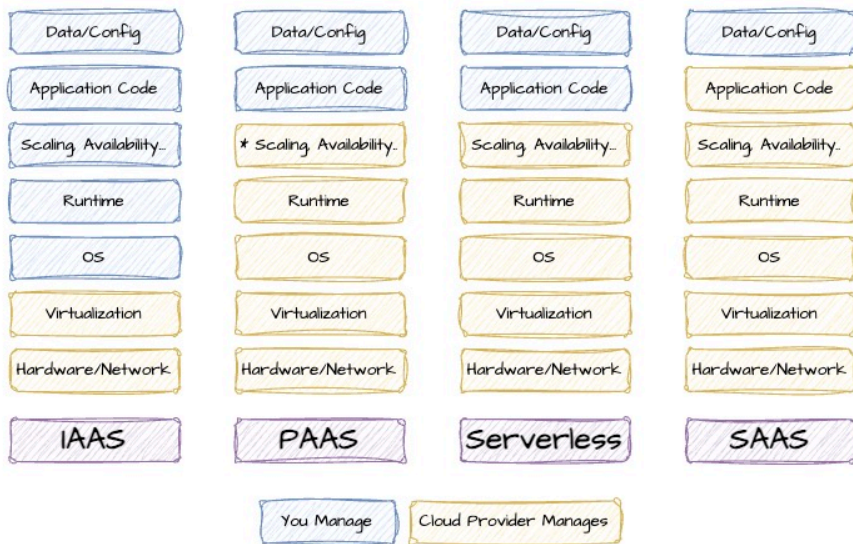


### IaaS (Infrastructure as a Service)

- Use **only infrastructure** from cloud provider
  - **Ex:** Using VM service to deploy your apps/databases
- **Cloud provider** is responsible for:
  - Hardware, Networking & Virtualization
- You are responsible for:
  - OS upgrades and patches
  - Application Code and Runtime
  - Configuring load balancing
  - Auto scaling

- Availability
- etc.. ( and a lot of things!)

- **Examples:** Amazon EC2, Azure Virtual Machines, Google Compute Engine



## PaaS (Platform as a Service)

- Use a platform provided by the cloud
  - **Cloud provider** is responsible for:
    - Hardware, Networking & Virtualization
    - OS (incl. upgrades and patches)

- Application Runtime
- Auto scaling, Availability & Load balancing etc..

- **You** are responsible for:

- Configuration (of Application and Services)
- Application code (if needed)

- **Examples:**

- **Compute:** AWS Elastic Beanstalk, Azure App Service, Google App Engine
- **Databases:** Relational (Amazon RDS, Google Cloud SQL, Azure SQL Database etc)
- And a lot of others!

## SaaS (Software as a Service)

- **Centrally hosted software** (mostly on the cloud)
  - Offered on a subscription basis (pay-as-you-go)
  - Examples:
    - Email, calendaring & office tools (such as Outlook 365, Microsoft Office 365, Gmail, Google Docs)
    - Customer relationship management (CRM), enterprise resource planning (ERP) and document management tools
- **Cloud/Service provider** is responsible for:
  - OS (incl. upgrades and patches)
  - Application Runtime
  - Auto scaling, Availability & Load balancing etc..

- Application code and/or
- Application Configuration (How much memory? How many instances? ..)
- **Customer** is responsible for:
  - Configuring the software!

## Serverless

- What do **we think about** when we develop an application?
  - Where to deploy? What kind of server? What OS?
  - How do we take care of scaling and availability of the application?
- What if **you don't worry about servers and focus ONLY on code**?
  - Enter **Serverless**
    - Remember: **Serverless does NOT mean "No Servers"**
- **Serverless for me:**
  - You **don't worry** about infrastructure (ZERO visibility into infrastructure)
    - Flexible scaling and automated high availability
  - Most Important: **Pay for use**
    - Ideally ZERO USAGE => ZERO COST
- **You focus on code** and the cloud managed service takes care of all that is needed to scale your service/code to serve millions of requests!
  - And you pay for usage and NOT servers!

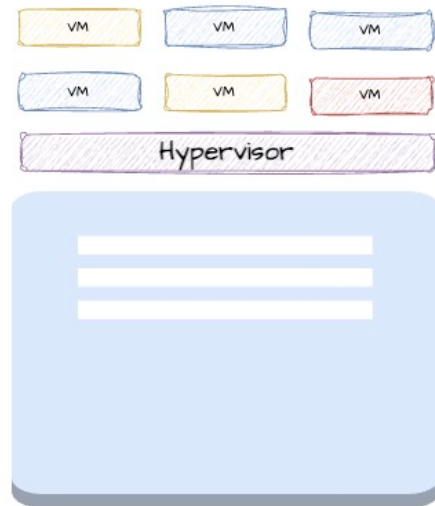
## Serverless Examples

| Category                         | AWS                        | Azure                                       | Google Cloud                     |
|----------------------------------|----------------------------|---|----------------------------------|
| Compute                          | AWS Lambda,<br>AWS Fargate | Azure Functions,<br>Azure Container<br>Apps | Cloud<br>Functions,<br>Cloud Run |
| Storage                          | Amazon S3                  | Azure Blob Storage                          | Cloud Storage                    |
| Databases and a<br>lot of others | ..                         | ..  | ..                               |

## What is Virtualization? <#>

### What is Virtualization?

- **Scenario:** In the past, running one app meant using one full physical server—even if the app used only 10% of the CPU. This wasted resources and increased costs.
- **Goal:** Use one physical server to run multiple virtual systems efficiently.



## Virtualization – The Basics

- **Definition:** Creating virtual versions of computing resources (like virtual machines) on a single physical server
- **How It Works:**
  - A **Hypervisor** (software layer) sits on top of the physical server
  - It splits the hardware into multiple **Virtual Machines (VMs)**
  - Each VM acts like a separate computer with its own OS and applications

## Why Use Virtualization?

- **Better Utilization:** Run many apps on fewer servers
- **Cost Efficiency:** Lower hardware and maintenance costs
- **Flexibility:** Quickly create, start, stop, or move VMs
- **Isolation:** Each VM is independent – safer and easier to manage

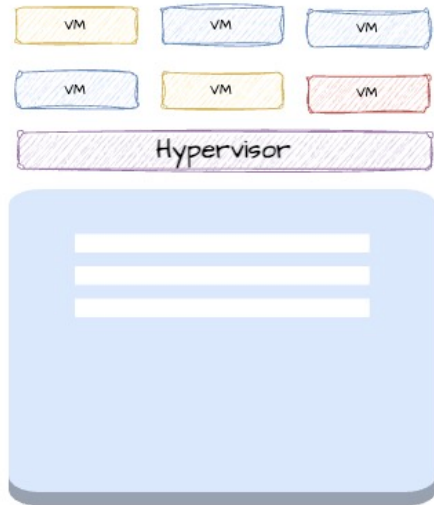
## Foundation of Cloud Computing

- Public cloud providers (like AWS, Azure, Google Cloud) use virtualization to offer shared infrastructure
- You get a VM (virtual server), not a dedicated physical machine
- Containers, serverless functions and almost everything you see in the cloud are built on top of virtualization layers

## How can you create Virtual Machines in the cloud? <#>

### What is a Virtual Machine (VM)?

- **Definition:** A **software-based (virtual) machine** running inside a physical server
- **Goal:** Run multiple isolated systems on a single physical machine
- **Benefit:** Full control over OS, software, and security settings while sharing hardware



### Cloud VMs are Flexible

- **Choose What You Need:** Select CPU, RAM, storage, and OS
- **Multiple OS Options:** Run Linux, Windows, or custom images
- **Custom Machine Types:** Match VMs to your workload and budget

### Cloud VMs are Scalable

- **Scale Up/Down:** Change number of VMs based on traffic
- **Auto Scaling:** Add or remove VMs automatically
- **Global Deployment:** Launch in regions close to users

### Cloud VMs are Cost Efficient

- **Pay-As-You-Go:** Billed by second, minute, or hour
- **Reserved/Spot Pricing:** Save cost with long-term reservations or using spare capacity
- **Release When Not Needed:** Reduce waste by releasing unused VMs

### Reviewing Important Virtual Machine Concepts

| Feature         | Explanation                                     | AWS                                | Azure                  | Google Cloud          |
|-----------------|---|------------------------------------|------------------------|-----------------------|
| Managed Service | Create VMs                                      | Amazon EC2 (Elastic Compute Cloud) | Azure Virtual Machines | Google Compute Engine |
| Image           | What OS and software should be on the instance? | AMI (Amazon Machine Image)         | VM Image               | Image                 |
|                 |   |                                    |                        |                       |

| Feature         | Explanation   | AWS                                | Azure                  | Google Cloud                            |
|-----------------|---|------------------------------------|------------------------|---|
| Instance Family | Type of hardware: General, High CPU or High Memory or Storage Optimized | Instance Family                    | VM Series              | Machine Family                          |
| Instance Size   | Amount of vCPU and memory   | Instance Type - t3.micro, m5.large | VM Sizes - B2s,B2ms .. | Machine Type - e2-medium, n2-standard-2 |
| Attached Disks  | Block storage volumes for VMs   | Elastic Block Store                | Managed Disks          | Persistent Disks                        |

### Networking for VMs

| Feature                       | Explanation   | AWS                | Azure              | Google Cloud        |
|-------------------------------|---|--------------------|--------------------|---------------------|
| Permanent Internal IP Address | Permanent Internal IP Address that does not change during the lifetime of an instance | Private IP Address | Private IP Address | Internal IP Address |

| Feature                       | Explanation  | AWS                | Azure                        | Google Cloud                     |
|-------------------------------|--|--------------------|------------------------------|----------------------------------|
| Ephemeral External IP Address | Ephemeral External IP Address that changes when an instance is stopped | Public IP Address  | Public IP Address            | External or Ephemeral IP Address |
| Permanent External IP Address | Permanent External IP Address that can be attached to a VM             | Elastic IP Address | Static IP Address            | Static IP Address                |
| Firewall Rules                | Control incoming/outgoing traffic                                      | Security Group     | Network Security Group (NSG) | Firewall Rules                   |

### Managing Costs

| Feature                | Explanation                             | AWS            | Azure    | Google Cloud |
|------------------------|---|----------------|----------|--------------|
| Cheaper temp instances | Create cheaper, temporary instances for | Spot instances | Spot VMs | Spot VMs     |

| Feature                             | Explanation  | AWS                | Azure                          | Google Cloud                                    |
|-------------------------------------|--|--------------------|--------------------------------|---|
|                                     | non critical workloads   |                    |                                |   |
| Reservations/Usage Discounts        | Reserve compute instances ahead of time/Get discounts for using resources for long periods of time | Reserved instances | Reserved instances             | Committed use discounts/Sustained use discounts |
| Reservation based on dollar amounts | Reserve based on monetary commitment (\$100 per hour, for example)                                 | AWS Savings Plans  | Azure Savings Plan for Compute | -   |

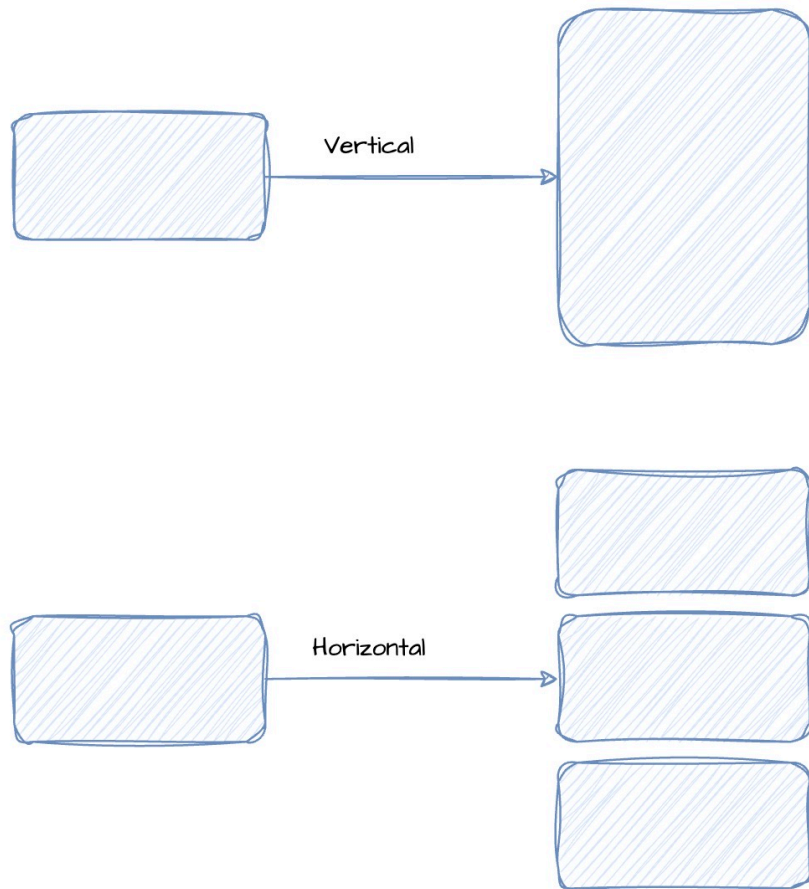
| Feature         | Explanation       | AWS           | Azure         | Google Cloud  |
|-----------------|-------------------|---------------|---------------|---------------|
| Managing Budget | Budget Management | Budget alerts | Budget alerts | Budget alerts |

## What are some of the architectural aspects to think about when creating virtual machines in the cloud? <#>

- Scaling
- Resiliency
- Observability
- Tenancy

### What is Scaling?

- **Scenario:** Your app is growing – more users, more traffic, more data. One server is not enough anymore. You need to scale.
- **Scaling:** Expanding system capacity to handle more load.



### Vertical Scaling (Scale Up)

- **What It Is:** Increase the power of a single machine
  - More CPU, RAM, disk, or network
  - Example: Upgrade a small server to a large one
- **Benefits:**
  - Simple and fast to implement
  - No code changes or architectural updates
- **Limitations:**
  - Hardware has physical limits
  - Downtime may be required
  - Higher cost for high-end machines

### Horizontal Scaling (Scale Out)

- **What It Is:** Add more machines or instances to share the load
  - Run multiple copies of the application or database
- **Benefits:**
  - Improved availability and fault tolerance
  - Scale without downtime
- **Challenges:**
  - Needs infrastructure like load balancers

### Horizontal Scaling in Practice

- **Auto Scaling:** Automatically add or remove instances based on traffic
- **Load Balancing:** Evenly distribute requests across all instances
- **Deployment Models:**
  - Within a single zone
  - Across multiple zones

**Resiliency for Virtual Machines and Load Balancing (Cloud Neutral)**

- **Scenario:** You're running your application on virtual machines. What happens if a VM crashes, a zone goes down, or your app faces a sudden surge in traffic?
- **Goal:** Design your architecture to **handle failures gracefully** and continue to serve users reliably — that's **resiliency**.

**What is Resiliency?**

- **Definition:** The ability of a system to provide **acceptable performance** and recover quickly even when **parts of the system fail**
- **Why It Matters:**
  - Failures are inevitable (hardware, software, network)
  - Users expect high uptime and reliability

**Building Resilient Architectures**

| Design Element                        | Purpose   |
|---------------------------------------|---|
| Auto-Healing Groups / Instance Groups | Automatically detect and replace failed VMs                         |
| Load Balancing                        | Distribute traffic across healthy instances to ensure availability  |
| Multi-Zone Deployment                 | Avoid single point of failure by spreading instances geographically |
| Health Checks                         | Detect and route traffic only to healthy instances                  |
| Disaster Recovery Planning            | Keep up-to-date VM images and backups in multiple regions or zones  |

**Creating Multiple VMs**

| Feature      | Explanation                    | AWS              | Azure             | Google Cloud       |
|--------------|--------------------------------|------------------|-------------------|--------------------|
| VM Templates | Templates to simplify creation | Launch Templates | - (ARM Templates) | Instance templates |

| Feature        | Explanation  | AWS                      | Azure                             | Google Cloud                   |
|----------------|--|--------------------------|-----------------------------------|--------------------------------|
|                | of Virtual Machines  |                          |                                   |                                |
| Auto Scaling   | Automatically add or remove VMs based on usage (load, time, events)          | Auto Scaling Group (ASG) | Virtual Machine Scale Sets (VMSS) | Managed Instance Groups (MIGs) |
| Load Balancing | Load Balancing   | Elastic Load Balancer    | Azure Load Balancer (& others)    | Cloud Load Balancing           |
| VM Management  | Simplify management (software, OS patches etc) of 1000's of Virtual Machines | Systems Manager          | Azure Update Manager              | VM Manager                     |

### Observability

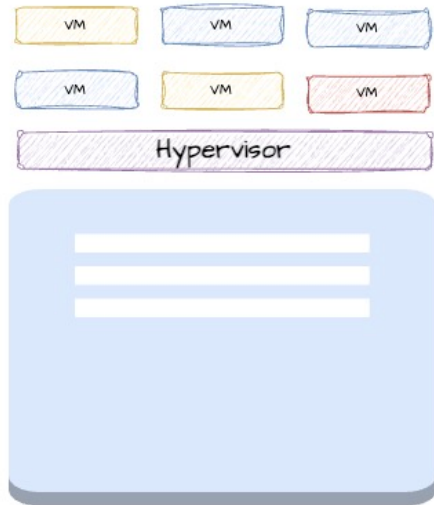
- Collect metrics like CPU, memory, network usage, disk I/O

- Capture application and system logs for troubleshooting
- Set alerts for failures

| Question  | AWS                    | Azure  | Google Cloud     |
|---|------------------------|--|------------------|
| How do you monitor metrics around your applications         | Amazon CloudWatch      | Azure Monitor                                  | Cloud Monitoring |
| How do you manage application and service logs?             | Amazon CloudWatch Logs | Azure Monitor Logs                             | Cloud Logging    |
| How do you trace requests across applications and services? | AWS X-Ray              | Azure Application Insights Distributed Tracing | Cloud Trace      |

### Why Tenancy Matters?

- **Scenario:** You're deploying an app with licensing restrictions or regulatory needs. You need control over the hardware it runs on.
- **Tenancy:** Determines *who shares* the physical server where your virtual machines (VMs) run.



### Shared Tenancy (Default)

- **What It Is:** Multiple customers' instances run on the same physical server
- **Limitations:**
  - Not suitable for strict compliance or licensing needs

### Dedicated Hosts

- **What It Is:** Entire physical server reserved for your use
- **Benefits:**
  - Required for some licensing models (e.g., Windows Server, SQL Server)

- Helps meet compliance and audit requirements

- **Limitations:**

- More expensive

| Feature                        | Explanation                              | AWS                 | Azure                 | Google Cloud      |
|--------------------------------|--|---------------------|-----------------------|-------------------|
| Host dedicated to one customer | Physical hosts dedicated to one customer | EC2 Dedicated Hosts | Azure Dedicated Hosts | Sole-tenant nodes |

## What are the Compute Options in Cloud? #

| Category          | AWS                   | Azure             | GCP                   |
|-------------------|-----------------------|-------------------|-----------------------|
| IaaS              | Amazon EC2            | Azure VMs         | Google Compute Engine |
| PaaS              | AWS Elastic Beanstalk | Azure App Service | App Engine            |
| FaaS - Serverless | AWS Lambda            | Azure Functions   | Cloud Functions       |

| Category                       | AWS                 | Azure   | GCP   |
|--------------------------------|---------------------|---|---|
| CaaS - Serverless              | AWS Fargate         | Azure Container Instances                         | Cloud Run, GKE Autopilot                          |
| CaaS - Kubernetes              | Amazon EKS          | Azure Kubernetes Service, Azure Red Hat OpenShift | Google Kubernetes Engine                          |
| CaaS - Custom                  | Amazon ECS          |   |   |
| Multi-cloud Container services | Amazon EKS Anywhere | Azure Arc-enabled Kubernetes                      | Google Kubernetes Engine (GKE) Enterprise edition |
| VMWare                         | VMware Cloud on AWS | Azure VMware Solution                             | VMware Engine                                     |

Next [Cloud Storage and Databases Fundamentals](#) →

## Keep Learning

Home ↗  
 Springboot ↗  
 Cloud ↗

## Our Products

Roadmaps ↗  
 Flashcards ↗  
 Bookshelf ↗

← Previous  
[DevOps Interview - Diagrams](#)