



Topic 1

Cloud Principles

Lesson Objectives

1

Objective 1:

Understand the difference between cloud computing and on-premise computing.

2

Objective 2: Learn

how to access major cloud platforms via portals, APIs, and SDKs.

3

Objective 3:

Explore the definition, characteristics, and advantages of cloud computing.

4

Objective 4:

Understand the relationship between Cloud Computing and Service-Oriented Architecture (SOA) .

5

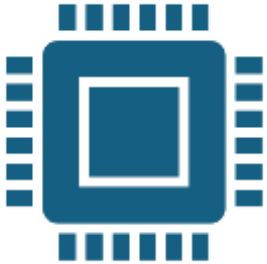
Objective 5:

Discuss enterprise cloud drivers and adoption trends.

What is Cloud Computing?

"Cloud Computing is the delivery of computing services (servers, storage, databases, networking, software, analytics, etc.) over the internet ('the cloud')."

On-Premise vs Cloud



On-Premise Computing:

Infrastructure managed internally.

Requires significant upfront capital expenditure.

Scalability is limited by hardware availability.



Cloud Computing:

Infrastructure managed externally by cloud providers.

Follows a pay-as-you-go model.

High scalability and flexibility.

Cloud vs On-Premises

CLOUD	FEATURE	ON-PREMISES
Software is hosted and managed by a cloud provider.	Deployment	Software is installed and executed on the organization's servers.
Typically, pay-as-you-go subscription model.	Cost	Upfront capital expenditures for hardware and software, ongoing maintenance, and support costs.
It is easy to scale resources up or down as needed.	Scalability	More complicated and expensive to scale.
Applications and data can be accessed from anywhere with an internet connection.	Accessibility	Applications and data are only accessible from within the organization's network.
Cloud providers offer a wide range of security features and certifications.	Security	Security is the responsibility of the organization.
Cloud providers offer compliance solutions for a variety of industry regulations.	Compliance	The organization is responsible for ensuring compliance with all applicable regulations.
Cloud providers have a team of experts who manage and maintain the cloud infrastructure.	Expertise	The organization needs an IT team to manage and maintain its on-premises infrastructure.

Cloud Access Methods



Portals: Web-based dashboards for cloud service management (e.g., AWS Management Console).



APIs: Programmatic access for automated tasks and integrations.



SDKs: Language-specific libraries (e.g., AWS SDK for Python, Azure SDK for Java).

Cloud Computing Characteristics

Elasticit
y

Multi-
Tenancy

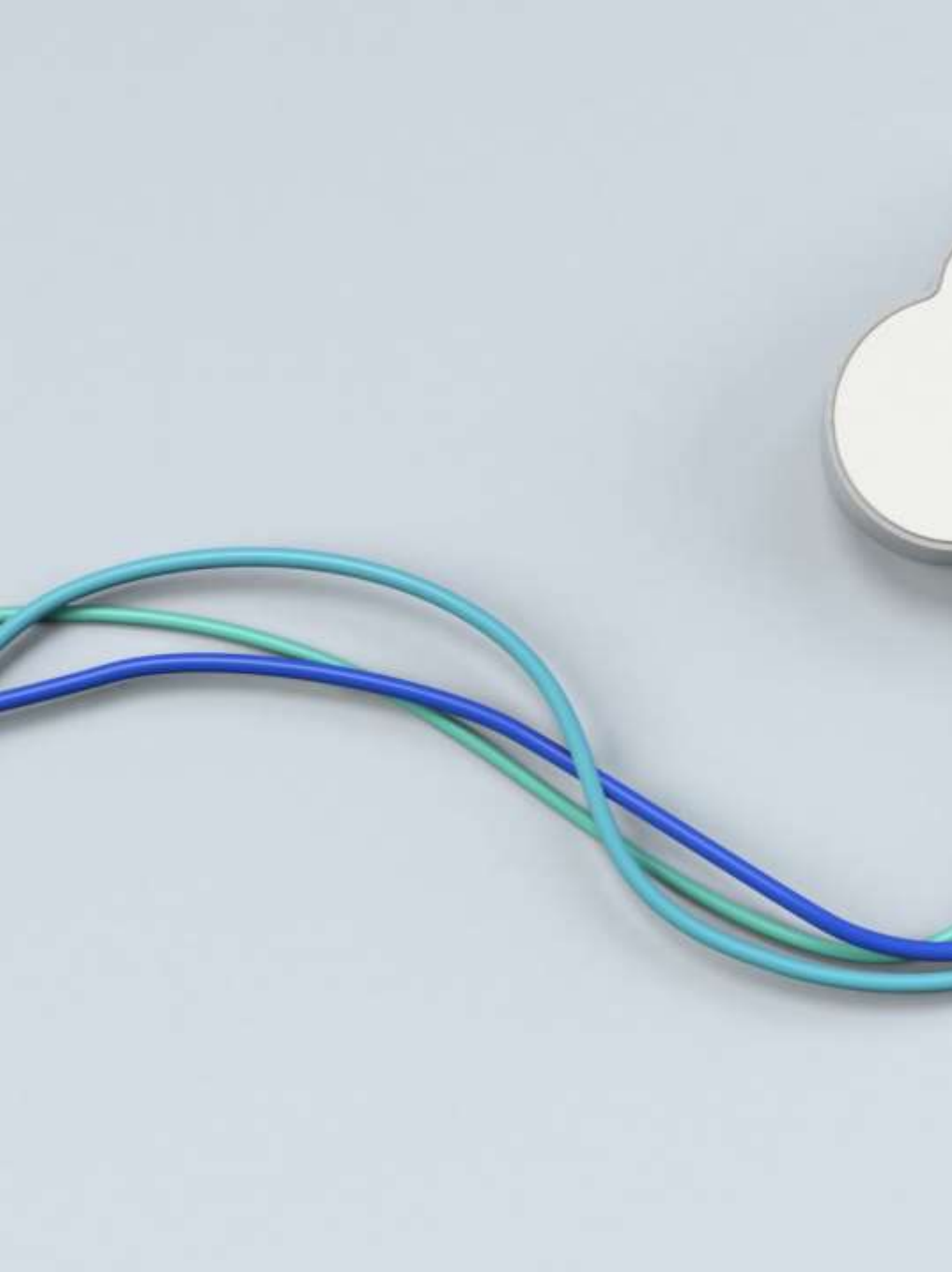
On-Demand
Access

Ubiquitou
s Access

Usage
Metering

Self-
Service
Capabilit
y

SLA
Monitorin
g

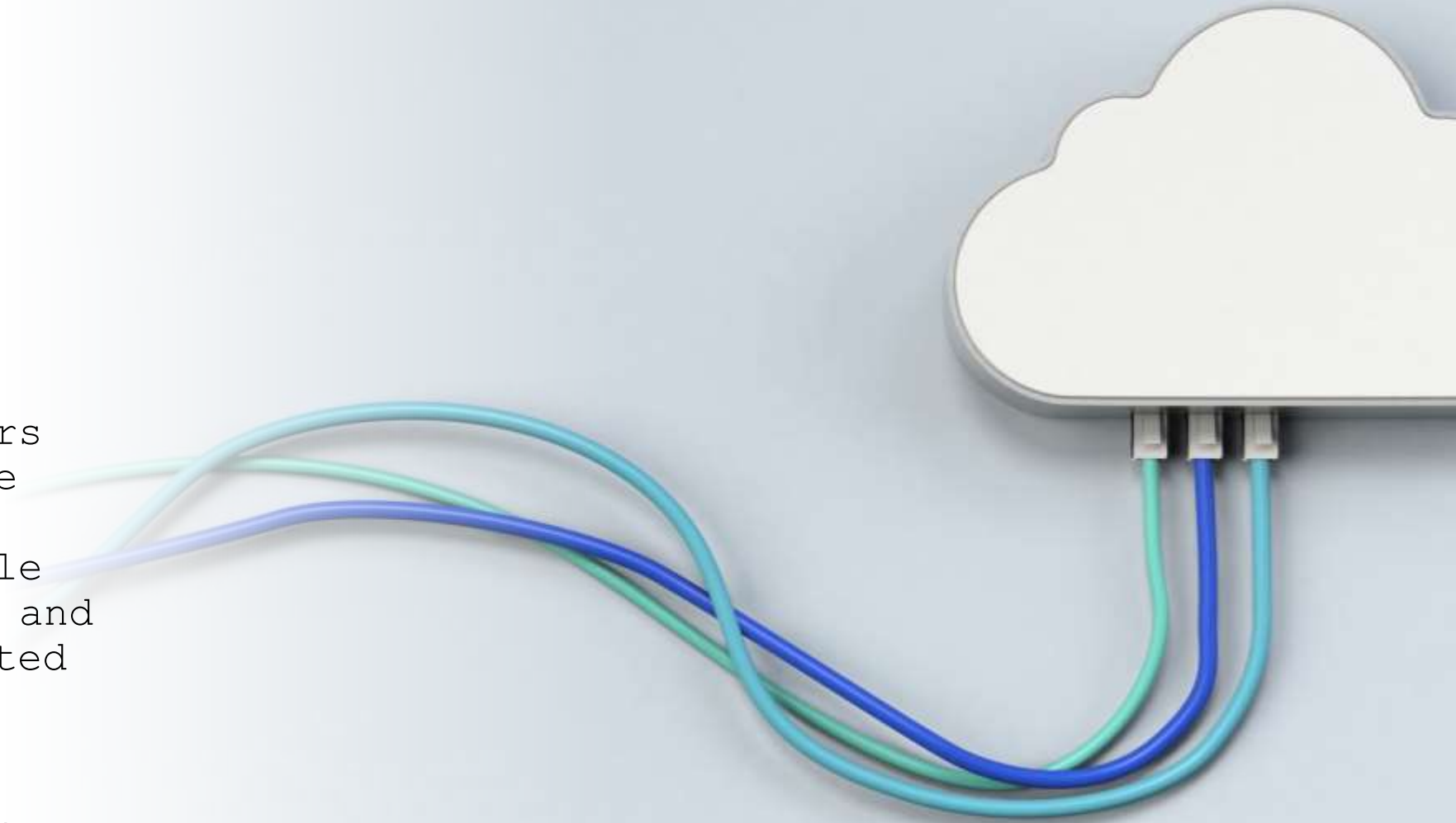


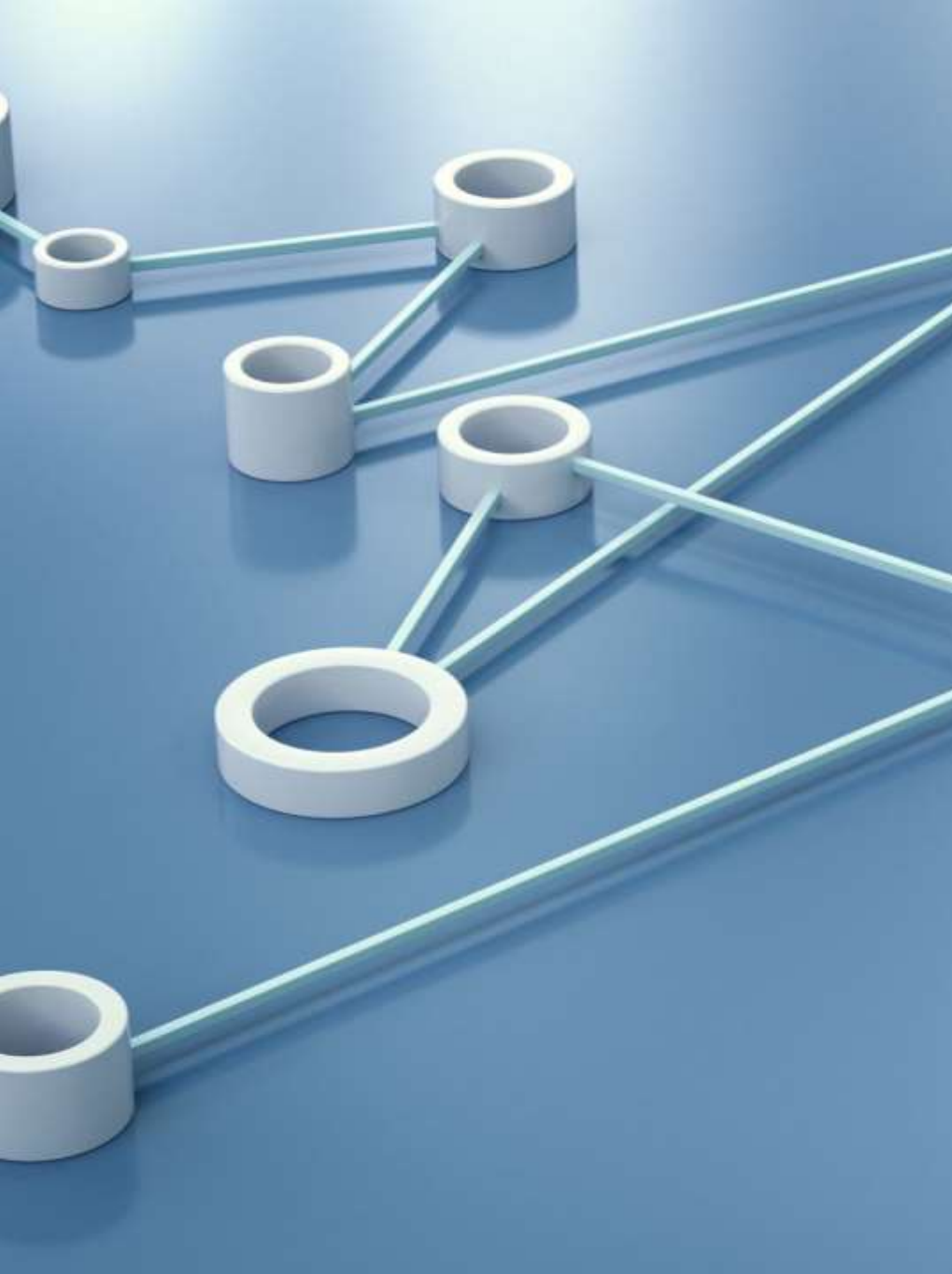
Elasticity

Elasticity refers to the ability to scale computing resources up or down dynamically based on current demand. This means if you have a sudden spike in user activity, the cloud can allocate additional resources automatically, and when demand drops, it reduces the resources to avoid wastage.

Multi-Tenancy

- Multi-tenancy is a cloud architecture where multiple users (tenants) share the same physical infrastructure while keeping their data and applications isolated and secure.
- A cloud provider like AWS hosts data for multiple organizations on the same server, but each organization's data is accessible only





On-Demand Access

- Users can access and use computing resources whenever they need them, without waiting for manual provisioning.
- A developer can instantly spin up a virtual server or database instance with just a few clicks in a cloud portal or an API call.
- Immediate resource availability.
- Faster deployment of applications.
- Improved flexibility and agility for businesses.



Ubiquitous Access

- Cloud resources can be accessed from anywhere with an internet connection, using any compatible device, like a laptop, tablet, or smartphone.
- Employees working remotely can access cloud-based productivity tools like Google Workspace or Microsoft 365 from any location.

Usage Metering

- Cloud providers use metering systems to track resource usage, such as CPU, memory, storage, and bandwidth. Users are billed based on their actual consumption.
- A cloud provider charges a company for storage in GB, compute power in hours, and data transfer in TB.



Self-Service Capability

Users can independently provision and manage resources through a self-service portal without needing intervention from the service provider's

support team. A developer can create a virtual machine, configure it, and deploy an application directly from the cloud dashboard or CLI.

SLA Monitoring

Service Level Agreements (SLAs) define the expected level of service, such as uptime, performance, and response times. Cloud providers continuously monitor their systems to meet these commitments. AWS provides an SLA of 99.99% uptime for certain services. If the uptime drops below this threshold, customers may receive compensation as per the SLA terms.

Service-Oriented Architecture (SOA)

Service-Oriented Architecture (SOA) :

Framework for designing and developing software as modular services.

Relationship with Cloud Computing:

SOA principles enhance cloud computing by enabling service reuse, scalability, and flexibility.

Example:

Microservices architecture commonly deployed in cloud environments.

Feature	SOA	Monolithic	Microservice	Event-Driven	Layered
Definition	Design paradigm structuring software as modular services communicating via protocols.	Entire application built as a single, unified unit.	Breaks down applications into small, highly specific services.	System communicates using asynchronous events.	Applications divided into layers like presentation, business logic, and data.
Modularity	Medium - Coarse-grained services focused on business processes.	Low - Single deployable codebase.	High - Each microservice handles a single functionality.	High - Components react to and produce events.	Medium - Organized into distinct functional layers.
Scalability	Scalable but may require more effort for inter-service communication.	Limited; scaling requires duplicating the entire application.	Highly scalable; services scale independently.	Highly scalable; can handle real-time workloads efficiently.	Limited; scalability depends on specific layer bottlenecks.
Interdependence	Loosely coupled services with some interdependencies.	Highly interdependent; components tightly coupled.	Loosely coupled with minimal interdependencies.	Minimally coupled; services interact indirectly via events.	Layers interact sequentially; moderately coupled.
Technology Flexibility	Supports heterogeneous technologies with protocols like SOAP/REST.	Requires uniform technology stack.	High flexibility; services can use different technologies.	Flexible; integrates with various messaging technologies (e.g., Kafka).	Relatively inflexible; layers depend on each other.
Use Cases	Enterprise systems, reusable services.	Small-scale applications or systems with simple requirements.	Large-scale, complex systems requiring agility (e.g., Netflix, Uber).	IoT, real-time notifications, stock trading platforms.	Traditional enterprise systems like CRM or ERP.
	Reusability, Scalability	Easy to develop	Independent	Low latency,	Clear separation of

Enterprise Cloud Drivers



Cost reduction (CAPEX to OPEX shift).



Improved scalability and flexibility.



Faster time-to-market for applications.




Enhanced security features provided by cloud providers.



Adoption Trend

- Increasing multi-cloud strategies.
- Migration to hybrid cloud setups.
- Growing reliance on AI and ML services in the cloud.



End of Topic 1

