



**Sixth Semester B.E./B.Tech. Degree Examination, June/July 2025**  
**Cloud Computing and Security**

BIS613D

Max. Marks: 100

Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.  
 2. M : Marks, L: Bloom's level, C: Course outcomes.

Module - 1				M	L	C
Q.1	a.	Describe the vision introduced by cloud computing?		06	L2	CO1
	b.	Provide brief characteristics of distributed system with examples?		06	L3	CO1
	c.	What is the major revolution introduced by web 2.0?		08	L2	CO1
OR						
Q.2	a.	Describe the main characteristic of service orientation of cloud computing with examples.		06	L3	CO1
	b.	What is the major distributed computing technology that led to cloud computing.		06	L2	CO1
	c.	Briefly summarize the challenges still open in cloud computing.		08	L1	CO1
Module - 2						
Q.3	a.	What is Xen? Discuss its elements for virtualization.		06	L2	CO2
	b.	Discuss the reference module of full virtualization.		06	L3	CO2
	c.	List and discuss different types of virtualization.		08	L2	CO2
OR						
Q.4	a.	How is cloud development different from traditional software development.		06	L2	CO2
	b.	Discuss the architecture of Hyper - V. Discuss its use in cloud computing.		06	L2	CO2
	c.	Discuss classification on taxonomy of visualization of different levels?		08	L3	CO2
Module - 3						
Q.5	a.	Explain the three primary cloud service models; IaaS PaaS, and SaaS with examples.		06	L2	CO3
	b.	Differentiate between Public, Private and hybrid cloud with advantages and limitation.		06	L3	CO3
	c.	What is a warehouse - scale data center? How does it differ from modular data centers.		08	L2	CO3
OR						
Q.6	a.	Compare the services and target applications of GAE, AWS and Microsoft Azure.		06	L3	CO3
	b.	Describe the fat-free topology and explain its use in data center network architecture.		06	L2	CO3
	c.	Explain the key requirements for an efficient cloud data center interconnection network.		08	L2	CO3
1 of 2						

BIS613D

## Module - 4

Q.7	a.	What is a trusted Hypervisor? Explain the mobile devices face a range of security challenges?	06	L2	CO4
	b.	Discuss the Security Risks posed by shared images and management os?	06	L2	CO4
	c.	Describe the Hidden Risk in the cloud computing.	08	L3	CO4

## OR

Q.8	a.	Explain the best Top 5 Cloud Security Best practices.	06	L2	CO4
	b.	What are the most important advantages of cloud technology for social network.	06	L2	CO4
	c.	Describe the key features of Google?	08	L2	CO4

## Module - 5


Q.9	a.	What are the benefits and challenges of using server less computing in the cloud?	06	L2	CO5
	b.	How does grid computing differ from cloud computing.	06	L2	CO5
	c.	What are the key features of cloud computing platforms.	08	L2	CO5

## OR

Q.10	a.	How do multi cloud and hybrid cloud strategies differ.	06	L2	CO5
	b.	What is infrastructure as code (IaC) and how if it used in the cloud.	06	L2	CO5
	c.	What are emerging cloud environments.	08	L2	CO5

\*\*\*\*\*



USN										<div><div>CELEBRATING 25 YEARS</div><div></div><div>CMRIT</div><div>CMR INSTITUTE OF TECHNOLOGY, BENGALURU.</div><div>ACCREDITED WITH A++ GRADE BY NAAC</div></div>				
VTU EXAMINATION JUN/JUL - 2025														
SCHEME OF EVALUATION														
Sub:	Cloud Computing & Security					Sub Code:	BIS613D	Branch:	ISE					
Exam Date:	07/07/2025	Duration:	3 Hrs	Max Marks:	100	Sem/Sec:	VI/ A, B & C					OBE		
Answer any FIVE FULL Questions choosing ONE full question from each Module									MARKS	CO	RBT			
MODULE-1														
1.a	<p><b>Describe the vision introduced by cloud computing?</b></p> <p><b>Scheme: list and explanation 2+4 marks</b></p> <p><b>Solution:</b></p> <p><b>The key aspects of this vision are:</b></p> <ol style="list-style-type: none"><li>1. On-Demand Services: Cloud computing enables users to access computing resources like storage, processing power, and software applications over the Internet on demand, without owning or managing the infrastructure.</li><li>2. Pay-as-You-Go Model: It introduces an economical model where users pay only for the resources they consume, thereby eliminating the need for large capital investments in hardware and software.</li><li>3. Service-Oriented Platform: The vision includes shifting computing from personal desktops to service-oriented platforms hosted in large data centers. This facilitates better scalability, resource utilization, and maintenance.</li><li>4. Elastic and Scalable Infrastructure: Cloud platforms are designed to scale seamlessly based on demand, offering flexibility for running diverse workloads across distributed data centers.</li><li>5. Virtualized Resource Pool: It envisions a pool of virtualized resources that can be rapidly provisioned and released to match user requirements, improving efficiency and agility.</li><li>6. Global Accessibility and Collaboration: By using the Internet as a backbone, cloud computing supports ubiquitous access, enabling collaboration across different geographic regions.</li></ol>								[06]	1	L2			
1.b	<p><b>Provide brief characteristics of distributed systems with examples</b></p> <p><b>Scheme: list and explanation 2+4 marks</b></p> <p><b>Solution:</b></p> <p>A distributed system is a collection of independent computers that appears to the users as a single coherent system.</p> <p>The following are the key characteristics of distributed systems, along with examples:</p> <ol style="list-style-type: none"><li>1. Resource Sharing Distributed systems allow sharing of hardware, software, and data resources. Example: Google File System enables distributed storage sharing among data centers.</li><li>2. Concurrency Multiple processes run concurrently on different machines, improving performance. Example: Online multiplayer gaming platforms where many users play simultaneously.</li><li>3. Scalability Systems can scale horizontally (adding more nodes) to handle increased workload. Example: Amazon Web Services (AWS) adds virtual machines to balance user load.</li><li>4. Fault Tolerance The system continues functioning even if some components fail. Example: Hadoop replicates data blocks across multiple nodes to ensure availability during node failures.</li><li>5. Transparency Users experience the system as a single unit despite its distributed nature.<ul style="list-style-type: none"><li>o Access Transparency: Users don't need to know where a resource is located.</li><li>o Location Transparency: Resources can be moved without affecting users. Example: Distributed databases like Cassandra or MongoDB.</li></ul></li></ol>								[06]	1	L3			

	<p>6. Heterogeneity The system supports different hardware, operating systems, and networks. Example: A web application using Windows servers for storage and Linux servers for computation.</p>			
1.c	<p><b>What is the major revolution introduced by Web 2.0?</b> <b>Scheme: list and explanation 2+6 marks.</b> <b>Solution:</b> Web 2.0 marks a significant revolution in how users interact with the internet. It shifted the web from a static content delivery system (Web 1.0) to a dynamic, interactive, and user-driven platform. The key aspects of this revolution include:</p> <ol style="list-style-type: none"> <li><b>1. User Participation and Collaboration</b> Web 2.0 empowered users not only to consume content but also to contribute, share, and collaborate. Social media platforms like Facebook and Twitter are prime examples of this transformation.</li> <li><b>2. Mashup Applications</b> A major innovation of Web 2.0 is the creation of mashups, which combine services and data from multiple web sources to deliver integrated, enhanced functionalities. For example, combining Flickr images with Google Maps allows for interactive photo geolocation.</li> <li><b>3. Service-Oriented Web Architecture</b> Web 2.0 applications are built on loosely coupled services using RESTful APIs and web services. This allowed seamless integration and scalability across platforms.</li> <li><b>4. Rich User Experience (RUX)</b> Through AJAX and modern UI technologies, Web 2.0 delivers responsive and interactive user experiences. Users can interact with content in real-time without full page reloads, e.g., live chat systems or collaborative editing tools.</li> <li><b>5. Social Networking and Microblogging</b> Platforms like Facebook and Twitter revolutionized communication and media sharing. Facebook introduced the concept of social graphs, while Twitter popularized real-time updates and micro-content (140-character messages).</li> <li><b>6. Community-Driven Content</b> Websites like YouTube, Wikipedia, and Reddit allowed users to contribute, curate, and rate content, changing how knowledge is shared and validated online.</li> <li><b>7. Enhanced Accessibility and Mobility</b> Web 2.0 applications are accessible across devices, supporting mobile interaction and cloud-based services. This mobility greatly enhances user engagement and convenience.</li> <li><b>8. APIs and Extensibility</b> The use of APIs enables third-party developers to extend functionalities. For example, Facebook's API allows app developers to integrate games, surveys, and utilities directly into the platform.</li> </ol>	[08]	1	L2
<b>OR</b>				
2.a	<p><b>Describe the main characteristic of service orientation in cloud computing with examples.</b> <b>Scheme: Definition + Explanation – 2 + 4 Marks</b> <b>Solution :</b> The key characteristic of service orientation in cloud computing is the delivery of computing functionalities as modular, reusable services. This is realized through Service-Oriented Architecture (SOA), where services are discoverable, loosely coupled, and accessible over a network.</p> <ol style="list-style-type: none"> <li><b>1. Service as the Basic Unit of Functionality</b> Services encapsulate business or computational logic and expose it through standard interfaces. These can be accessed remotely by users or applications without knowing the internal implementation.</li> <li><b>2. Loose Coupling and Interoperability</b> Service orientation promotes loose coupling between service providers and consumers. This allows systems built on different platforms or programming languages to communicate effectively. Example: A Java-based application accessing a .NET-based web service using SOAP/REST.</li> <li><b>3. Platform and Language Independence</b> Services use platform-neutral communication protocols like XML or JSON, which ensure</li> </ol>	[06]	1	L3

	<p>interoperability across heterogeneous systems. Example: A cloud service providing weather data can be used by mobile apps (iOS/Android), web applications, or embedded devices.</p> <p><b>4. Discoverability and Reusability</b> Services are published in a service registry, making them discoverable and reusable by other components or applications. This promotes code reuse and modular development. Example: AWS Simple Queue Service (SQS) can be reused in different workflows to manage asynchronous tasks.</p> <p><b>5. Message-Oriented Communication</b> Services interact using standardized message formats (e.g., HTTP, SOAP), ensuring seamless communication between distributed components. Example: Web services using WSDL and SOAP for formal interaction between client and server components.</p> <p><b>6. Real-Life Cloud Examples</b> Amazon Web Services (AWS) uses service orientation to offer compute (EC2), storage (S3), and messaging (SQS) as modular services. Google App Engine allows developers to build cloud applications using standardized services like user authentication and datastore APIs.</p>			
2.b	<p><b>What is the major distributed computing technology that led to cloud computing ?</b> <b>Scheme: Definition + Explanation – 2 + 4 Marks</b> <b>Solution:</b></p> <p>Cloud computing emerged from a blend of several distributed computing paradigms. The most significant technology that led to cloud computing is Grid Computing, along with other enabling models such as utility computing and peer-to-peer (P2P) systems.</p> <p><b>The key contributors are described below:</b></p> <p><b>1. Grid Computing</b></p> <ul style="list-style-type: none"> <li>• Grid computing provides an infrastructure to couple computers, software, sensors, and users across organizational boundaries.</li> <li>• It supports resource sharing, collaborative processing, and data integration across distributed locations.</li> <li>• Grid systems evolved to support large-scale data analysis, scientific simulations, and collaborative applications.</li> <li>• Example: NSF TeraGrid and European DataGrid are prominent grid infrastructures that paved the way for modern cloud platforms.</li> </ul> <p><b>2. Utility Computing</b></p> <ul style="list-style-type: none"> <li>• Utility computing offers computing as a metered service, similar to electricity or water.</li> <li>• It emphasized on-demand provisioning and pay-per-use models, both of which are central to cloud computing today.</li> <li>• Grid and cloud platforms are both regarded as utility service providers, but cloud computing is broader in scope.</li> </ul> <p><b>3. Virtualization</b></p> <ul style="list-style-type: none"> <li>• Virtualization allows efficient utilization of hardware by running multiple virtual machines (VMs) on a single physical server.</li> <li>• This made cloud computing flexible, cost-effective, and scalable, enabling the dynamic provisioning of resources.</li> </ul> <p><b>4. Convergence of Technologies</b></p> <ul style="list-style-type: none"> <li>• The rise of cloud computing is the result of a convergence of grid computing, virtualization, web services, SOA (Service-Oriented Architecture), and autonomic computing.</li> </ul>	[06]	1	L2
2.c	<p><b>Briefly summarize the challenges still open in cloud computing.</b> <b>Scheme: List + Explanation – 2 + 6 Marks</b> <b>Solution:</b></p> <p>Despite its benefits, cloud computing still faces several unresolved challenges that affect its widespread and secure adoption. The major challenges include:</p> <p><b>1. Service Availability and Data Lock-In</b></p> <ul style="list-style-type: none"> <li>• Relying on a single cloud provider introduces a risk of service failure or vendor lock-in.</li> <li>• Proprietary APIs make it difficult to migrate services across platforms.</li> <li>• <b>Solution:</b> Use multiple providers and standardize APIs to support "surge computing" and avoid data loss during outages.</li> </ul>	[08]	1	L1

	<p><b>2. Data Privacy and Security</b></p> <ul style="list-style-type: none"> <li>Cloud platforms operate over public networks, making them vulnerable to attacks such as DDoS, VM hijacking, or man-in-the-middle during VM migration.</li> <li>Compliance with data protection laws (e.g., storing data within national boundaries) is a challenge.</li> <li><b>Solution:</b> Use encryption, firewalls, VPNs, and apply strict access control measures.</li> </ul> <p><b>3. Unpredictable Performance and Bottlenecks</b></p> <ul style="list-style-type: none"> <li>Sharing I/O between VMs can result in performance degradation.</li> <li>Disk and network I/O become major bottlenecks under heavy load.</li> <li><b>Solution:</b> Improve I/O virtualization and design better hardware/software systems.</li> </ul> <p><b>4. Distributed Storage and Software Bugs</b></p> <ul style="list-style-type: none"> <li>Large-scale distributed storage systems are prone to software bugs and inconsistencies.</li> <li>Data consistency and replication management across global data centers are still difficult.</li> <li><b>Solution:</b> Use robust error-checking, software validation, and consistent data replication models.</li> </ul> <p><b>5. Scalability, Interoperability, and Standardization</b></p> <ul style="list-style-type: none"> <li>Lack of uniform standards hinders interoperability between cloud providers.</li> <li>As demand scales, maintaining performance and compatibility becomes complex.</li> <li><b>Solution:</b> Develop industry-wide standards for APIs, data formats, and cloud interoperability.</li> </ul> <p><b>6. Software Licensing and Reputation Sharing</b></p> <ul style="list-style-type: none"> <li>Licensing software for dynamic and scalable environments is still under legal and financial ambiguity.</li> <li>Providers must also build mechanisms for trust and reputation sharing among clouds.</li> <li><b>Solution:</b> Adapt licensing policies for virtualized environments and build reliable trust models.</li> </ul>			
<b><u>Module-2</u></b>				
3.a	<p><b>What is Xen? Discuss its elements for virtualization.</b>  <b>Scheme: Definition + Elements + Explanation – 2 + 2 + 2 Marks</b>  <b>Solution:</b>  <b>Xen</b> is an <b>open-source hypervisor</b> developed by Cambridge University. It is used to enable virtualization by allowing multiple operating systems to run on the same physical hardware concurrently.</p> <p><b>Main Features of Xen:</b></p> <ul style="list-style-type: none"> <li>Xen is a <b>microkernel hypervisor</b>, which means it separates <b>policy from mechanism</b>.</li> <li>It provides only the minimal mechanism needed for virtualization, and the rest of the management is handled by a special guest OS known as <b>Domain 0 (Dom0)</b>.</li> <li>Xen offers a virtual environment between the <b>hardware and the guest operating systems</b>, but it does <b>not include native device drivers</b>, making it lightweight.</li> </ul> <p><b>Key Elements of Xen Virtualization:</b></p> <ol style="list-style-type: none"> <li><b>Xen Hypervisor</b> <ul style="list-style-type: none"> <li>The core component installed directly on the hardware.</li> </ul> </li> </ol>	[06]	2	L2

	<ul style="list-style-type: none"> <li>Responsible for abstracting and controlling access to hardware resources.</li> <li>Provides low-level CPU scheduling, memory management, and device I/O.</li> </ul> <p><b>2. Domain 0 (Dom0)</b></p> <ul style="list-style-type: none"> <li>A <b>privileged guest operating system</b> that is booted first.</li> <li>Has direct access to the hardware and contains the <b>device drivers</b>.</li> <li>Responsible for managing other virtual machines (guest domains).</li> <li>Handles VM lifecycle tasks such as creation, deletion, and migration.</li> </ul> <p><b>3. Domain U (DomU)</b></p> <ul style="list-style-type: none"> <li><b>Unprivileged guest domains</b> which run user applications.</li> <li>These VMs do not directly access hardware and depend on Dom0 for I/O and device access.</li> <li>Each DomU is isolated from others and runs its own guest OS.</li> </ul> <p><b>4. Hypercalls</b></p> <ul style="list-style-type: none"> <li>These are special calls made by guest OSes to interact with the Xen hypervisor, similar to system calls to an OS.</li> </ul> <p><b>5. Paravirtualization Support</b></p> <ul style="list-style-type: none"> <li>Xen primarily uses <b>paravirtualization</b>, which means guest operating systems are modified to communicate efficiently with the hypervisor using hypercalls.</li> </ul>			
3.b	<p><b>Discuss the reference model of full virtualization.</b></p> <p><b>Scheme: Definition + Diagram + Explanation – 2+2+2 Marks</b></p> <p><b>Solution:</b></p> <div data-bbox="529 972 994 1274" data-label="Diagram"> <pre> graph TD     subgraph Rings         R3[Ring 3]         R2[Ring 2]         R1[Ring 1]         R0[Ring 0]     end     UA[User apps]     GO[Guest OS]     VMM[VMM]     HCSH[Host computer system hardware]      UA --- R3     GO --- R1     VMM --- R0      UA -- "Direct execution of user requests" --&gt; HCSH     GO -- "Binary translation of OS requests" --&gt; VMM     VMM -- "Binary translation of OS requests" --&gt; HCSH   </pre> <p>The diagram illustrates the reference model of full virtualization across four privilege rings. Ring 3 contains User apps, Ring 2 is empty, Ring 1 contains the Guest OS, and Ring 0 contains the VMM. The Host computer system hardware is at the base. Arrows show that User apps can execute requests directly on hardware, while the Guest OS requests are intercepted by the VMM for binary translation before reaching the hardware.</p> <p><b>Full virtualization</b> is a virtualization technique that enables the execution of unmodified guest operating systems by using a <b>Virtual Machine Monitor (VMM)</b> or <b>hypervisor</b> to fully abstract the underlying hardware.</p> <p><b>Key Components and Working:</b></p> <ol style="list-style-type: none"> <li><b>Virtual Machine Monitor (VMM) or Hypervisor</b> <ul style="list-style-type: none"> <li>Sits between hardware and guest OS.</li> <li>Controls and manages guest access to physical resources.</li> </ul> </li> <li><b>Binary Translation</b> <ul style="list-style-type: none"> <li>The VMM uses <b>binary translation</b> to intercept and emulate <b>privileged instructions</b> (e.g., control- and behavior-sensitive instructions).</li> <li><b>Noncritical instructions</b> are executed directly on the hardware to boost performance.</li> </ul> </li> <li><b>Guest Operating System Execution</b> <ul style="list-style-type: none"> <li>The guest OS is unaware it's running in a virtual environment.</li> <li>Runs in a lower privilege ring (e.g., <b>Ring 1</b>), while the VMM operates in <b>Ring 0</b>, maintaining security and control.</li> </ul> </li> <li><b>Performance Optimization</b> <ul style="list-style-type: none"> <li>Uses <b>code caching</b> to store frequently translated instructions (hot paths) and improve performance.</li> <li>Despite this, binary translation can still result in overhead, especially for I/O-intensive workloads</li> </ul> </li> </ol> </div>	[06]	2	L3

3.c	<p><b>List and discuss different types of virtualization.</b></p> <p><b>Scheme: List + Explanation – 2 + 6 Marks</b></p> <p><b>Solution:</b></p> <p>Virtualization refers to the creation of a virtual version of hardware, OS, storage, or network resources. Various types of virtualization have emerged depending on where the virtualization layer is inserted in the system.</p> <p><b>1. Instruction Set Architecture (ISA) Level Virtualization</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Virtualization is performed by emulating one ISA on top of another.</li> <li>• <b>Example:</b> Running MIPS code on an x86 machine using emulation.</li> <li>• <b>Tool:</b> Bochs, QEMU.</li> <li>• <b>Use Case:</b> Supports legacy binaries on modern platforms.</li> </ul> <p><b>2. Hardware-Level Virtualization</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Virtualization is performed directly on the bare hardware.</li> <li>• <b>Key Feature:</b> Provides VMs with direct access to hardware.</li> <li>• <b>Example:</b> IBM VM/370, Xen hypervisor.</li> <li>• <b>Use Case:</b> High-performance computing, cloud infrastructure.</li> </ul> <p><b>3. Operating System-Level Virtualization</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Inserts a virtualization layer inside the OS kernel to create isolated containers.</li> <li>• <b>Example:</b> OpenVZ, Linux vServer.</li> <li>• <b>Advantages:</b> Fast VM startup, low overhead, high density.</li> <li>• <b>Limitation:</b> All containers share the same OS kernel.</li> <li>• <b>Use Case:</b> Web hosting, SaaS deployment.</li> </ul> <p><b>4. Library Support-Level Virtualization</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Virtualizes at the API level using libraries instead of OS calls.</li> <li>• <b>Example:</b> WINE (Windows apps on UNIX), vCUDA (GPU access inside VM).</li> <li>• <b>Use Case:</b> Cross-platform application execution.</li> </ul> <p><b>5. Application-Level Virtualization</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Encapsulates applications from the underlying OS in isolated environments.</li> <li>• <b>Example:</b> Java Virtual Machine (JVM), .NET CLR.</li> <li>• <b>Use Case:</b> Platform-independent execution, sandboxing.</li> </ul> <p><b>6. Para-Virtualization</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Requires modification of the guest OS to replace sensitive instructions with hypercalls.</li> <li>• <b>Example:</b> Xen, KVM.</li> <li>• <b>Advantage:</b> Better performance than full virtualization.</li> <li>• <b>Limitation:</b> Guest OS must be modified.</li> </ul> <p><b>7. Full Virtualization</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Allows unmodified guest OS to run using binary translation of sensitive instructions.</li> <li>• <b>Example:</b> VMware ESX, VirtualBox.</li> <li>• <b>Advantage:</b> Supports any OS without changes.</li> <li>• <b>Limitation:</b> Higher overhead due to instruction emulation.</li> </ul> <p><b>8. Host-Based Virtualization</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Virtualization layer runs on top of a host OS.</li> <li>• <b>Example:</b> VMware Workstation, Virtual PC.</li> <li>• <b>Advantage:</b> Easy to install and use on desktop systems.</li> <li>• <b>Limitation:</b> Reduced performance due to multiple layers.</li> </ul>	[08]	2	L2
-----	---	------	---	----

**OR**

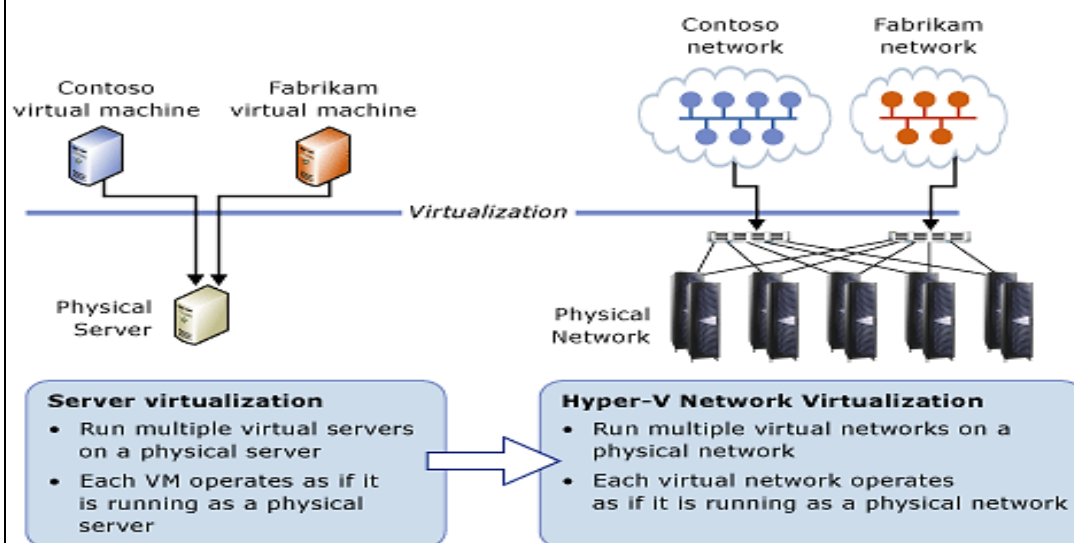
4.a	<p><b>How is cloud development different from traditional software development ?</b>  <b>Scheme: Definition + explanation with example– 2+4 Marks</b>  <b>Solution :</b>  Cloud development introduces a paradigm shift from traditional software development by leveraging scalable, on-demand, and service-oriented platforms.  <b>The key differences are:</b>  <b>1. Development Platform</b> <ul style="list-style-type: none"> <li>Traditional: Requires purchasing hardware, installing system software, and configuring environments manually.</li> <li>Cloud: Offers Platform as a Service (PaaS) which includes preconfigured hardware and software tools like Java, Python, .NET, and IDEs. Developers can directly focus on application logic without managing infrastructure.</li> </ul> <b>2. Cost Model</b> <ul style="list-style-type: none"> <li>Traditional: High-priced capital investment in hardware and software licensing. Cost increases with the number of users.</li> <li>Cloud: Follows a pay-as-you-go model. Resources are leased, and users only pay for actual usage, significantly reducing operational and capital costs.</li> </ul> <b>3. Application Deployment</b> <ul style="list-style-type: none"> <li>Traditional: Requires local deployment and manual scaling.</li> <li>Cloud: Supports automated deployment, scaling, and load balancing. For example, Google App Engine provides tools to upload, scale, and monitor applications across distributed systems.</li> </ul> <b>4. Scalability</b> <ul style="list-style-type: none"> <li>Traditional: Limited by physical infrastructure.</li> <li>Cloud: Offers elastic scalability, allowing resources to grow or shrink dynamically based on demand.</li> </ul> <b>5. Collaboration and Integration</b> <ul style="list-style-type: none"> <li>Traditional: Localized development with manual integration.</li> <li>Cloud: Enables collaborative development, third-party integrations, and remote team workflows.</li> </ul> <b>6. Development Environment</b> <ul style="list-style-type: none"> <li>Traditional: Code, test, and deploy all on local machines.</li> <li>Cloud: Provides local emulators (e.g., App Engine SDK) for development and testing, and seamless deployment to cloud infrastructure after testing.</li> </ul> </p>	[06]	2	L2
4.b	<p><b>Discuss the architecture of Hyper-V. Discuss its use in cloud computing.</b>  <b>Scheme : Definition + explanation + Diagram + uses – 2+2+2 Marks</b>  <b>Solution :</b>    <b>Hyper-V is a micro-kernel-based hypervisor</b> developed by Microsoft. It provides <b>hardware-level virtualization</b>, allowing multiple virtual machines (VMs) to run simultaneously on a physical host. Hyper-V is part of Windows Server and provides an environment for deploying and managing VMs efficiently.    <b>Architecture of Hyper-V</b>  Hyper-V uses a <b>micro-kernel hypervisor architecture</b>, which is characterized by the following components: <ul style="list-style-type: none"> <li><b>Microkernel Core:</b> <ul style="list-style-type: none"> <li>Includes <b>essential and fixed functions</b> like CPU scheduling and physical memory management.</li> <li>Does <b>not include device drivers</b> or dynamic components inside the hypervisor, reducing its size and attack surface.</li> </ul> </li> <li><b>Parent Partition (Management OS):</b> <ul style="list-style-type: none"> <li>Also known as <b>Domain 0</b> in similar architectures like Xen.</li> <li>This is a privileged VM that runs the full Windows Server OS and contains all the device drivers.</li> <li>Responsible for creating, managing, and controlling other guest VMs.</li> </ul> </li> <li><b>Child Partitions (Guest VMs):</b> <ul style="list-style-type: none"> <li>These are <b>unprivileged virtual machines</b> created and managed by the parent</li> </ul> </li> </ul> </p>	[06]	2	L2



- partition.
  - Each runs its own guest OS and applications.
- **VMBus:**
  - A communication channel between the parent and child partitions to facilitate efficient data transfer.
- **Hypercalls:**
  - Provide a mechanism for guest VMs to request privileged operations from the hypervisor.

#### Use of Hyper-V in Cloud Computing

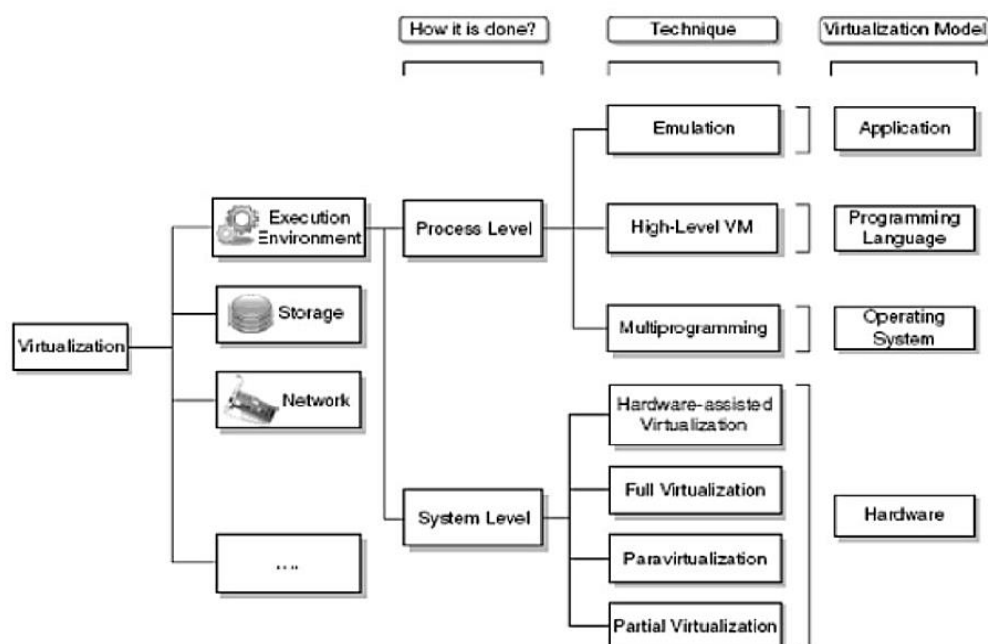
- **Virtual Infrastructure Foundation:**
  - Hyper-V serves as the core for Microsoft's **private and hybrid cloud solutions** (e.g., Azure Stack).
  - It allows the **dynamic provisioning of VMs**, enabling Infrastructure as a Service (IaaS).
- **Isolation and Security:**
  - Hyper-V's architecture ensures isolation between VMs, which is critical in multi-tenant cloud environments.
- **Scalability and Flexibility:**
  - It supports **live migration**, **dynamic memory**, and **resource pooling**, all essential for scalable cloud infrastructure.



4.c Discuss classification and taxonomy of virtualization at different levels.

**Scheme : Definition + explanation + Diagram – 2+4+2 Marks**

**Solution :**



[08]

2

L3

Virtualization can be implemented at multiple levels of a computer system. Each level offers different abstractions, use cases, and performance characteristics. The classification of virtualization is based on where the virtualization layer is inserted in the system architecture.

### 1. Instruction Set Architecture (ISA) Level Virtualization

- **Description:** Emulates the instruction set of one platform on another using code interpretation or dynamic binary translation.
- **Example:** Running MIPS code on an x86 machine using QEMU or Bochs.
- **Use Case:** Legacy code execution across different platforms.
- **Merit:** High application flexibility.
- **Limitation:** Slow performance due to instruction translation.

### 2. Hardware-Level Virtualization

- **Description:** Virtualization layer is inserted directly on top of the physical hardware.
- **Example:** Xen, VMware ESX, Microsoft Hyper-V.
- **Use Case:** Cloud platforms, server consolidation.
- **Merit:** High performance and isolation.
- **Limitation:** Requires hardware support and complex implementation.

### 3. Operating System-Level Virtualization

- **Description:** Inserts a virtualization layer within the OS kernel to provide isolated containers or virtual environments.
- **Example:** OpenVZ, Linux vServer.
- **Use Case:** Lightweight VM creation for hosting services.
- **Merit:** Fast startup, low overhead.
- **Limitation:** All containers must use the same OS kernel type.

### 4. Library Support Level Virtualization

- **Description:** Virtualization is achieved by intercepting library/API calls at the user level.
- **Example:** WINE (for Windows apps on Linux), vCUDA (GPU virtualization).
- **Use Case:** Cross-platform compatibility without full OS emulation.
- **Merit:** Application portability.
- **Limitation:** Limited scope and platform dependency.

### 5. User/Application-Level Virtualization

- **Description:** Runs applications in a virtual environment, isolated from the host OS.
- **Example:** JVM, .NET CLR, LANDesk.
- **Use Case:** Platform-independent deployment, application sandboxing.
- **Merit:** Easy distribution and isolation.
- **Limitation:** Not suitable for full OS virtualization.

## Module-3

5.a	<p>Explain the three primary cloud service models IaaS, PaaS, and SaaS with examples.  <b>Scheme : Definition + explanation – 2+2+2 Marks each</b>  <b>Solution :</b></p> <p>Cloud computing offers three primary service models—<b>Infrastructure as a Service (IaaS)</b>, <b>Platform as a Service (PaaS)</b>, and <b>Software as a Service (SaaS)</b>—each delivering a different layer of services to end users.</p> <p><b>1. Infrastructure as a Service (IaaS)</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Provides <b>virtualized computing resources</b> such as CPU, storage, and networking over the internet.</li> <li>• <b>User Role:</b> Users manage the OS, applications, and storage but not the underlying hardware.</li> <li>• <b>Example:</b> <b>Amazon EC2</b> provides virtual servers (VMs), and <b>Amazon S3</b> offers scalable cloud storage.</li> <li>• <b>Use Case:</b> Startups and enterprises use IaaS for scalable infrastructure without upfront hardware investments.</li> </ul> <p><b>2. Platform as a Service (PaaS)</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Provides a <b>software development platform</b> with tools, runtime environments, and middleware.</li> <li>• <b>User Role:</b> Developers build, test, and deploy applications without managing infrastructure.</li> <li>• <b>Example:</b> <b>Google App Engine</b> supports Java, Python, and other tools for web application deployment. It includes features like auto-scaling and load balancing.</li> <li>• <b>Use Case:</b> Ideal for developers who want to focus on application logic without worrying about servers.</li> </ul> <p><b>3. Software as a Service (SaaS)</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Delivers <b>application software over the web</b>, accessible through browsers.</li> <li>• <b>User Role:</b> End users access applications without managing any underlying infrastructure or platforms.</li> <li>• <b>Example:</b> <b>Google Gmail</b>, <b>Google Docs</b>, and <b>Salesforce CRM</b> offer business and productivity software to users via the cloud.</li> <li>• <b>Use Case:</b> Commonly used by businesses and individuals for day-to-day operations like email, document editing, and customer relationship management.</li> </ul>	[06]	3	L2
5.b	<p>Differentiate between Public, Private, and Hybrid cloud with advantages and limitation.  <b>Scheme : Definition + explanation of each – 2+2+2 Marks</b>  <b>Solution :</b></p> <p>Cloud deployment models differ based on ownership, access control, resource management, and use cases. The three major types are <b>Public</b>, <b>Private</b>, and <b>Hybrid</b> clouds.</p> <p><b>1. Public Cloud</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Cloud infrastructure and services offered over the internet to general users on a pay-per-use basis.</li> <li>• <b>Ownership:</b> Owned and managed by third-party cloud providers.</li> <li>• <b>Examples:</b> Amazon Web Services (AWS), Microsoft Azure, Google App Engine, IBM Blue Cloud.</li> </ul> <p><b>Advantages:</b></p> <ul style="list-style-type: none"> <li>• No capital investment required.</li> <li>• Highly scalable and elastic.</li> <li>• Easy and quick to access services globally.</li> </ul> <p><b>Limitations:</b></p> <ul style="list-style-type: none"> <li>• Less control over data and infrastructure.</li> <li>• Security and compliance concerns in shared environments.</li> </ul> <p><b>2. Private Cloud</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Cloud environment operated solely for a single organization, hosted either on-premises or by a third-party provider.</li> <li>• <b>Ownership:</b> Owned, managed, and accessed only by the organization and its authorized users.</li> <li>• <b>Example:</b> IBM Research Compute Cloud (RC2).</li> </ul>	[06]	3	L3



	<p><b>Advantages:</b></p> <ul style="list-style-type: none"> <li>• Greater security, privacy, and control.</li> <li>• Customization based on organizational needs.</li> </ul> <p><b>Limitations:</b></p> <ul style="list-style-type: none"> <li>• High setup and maintenance costs.</li> <li>• Limited scalability compared to public clouds.</li> </ul> <p><b>3. Hybrid Cloud</b></p> <ul style="list-style-type: none"> <li>• <b>Definition:</b> Combines both public and private clouds to allow data and applications to be shared between them.</li> <li>• <b>Usage:</b> Frequently used to handle burst workloads and maintain sensitive tasks internally.</li> </ul> <p><b>Advantages:</b></p> <ul style="list-style-type: none"> <li>• Balance between cost-efficiency and control.</li> <li>• Flexibility to scale with public cloud while keeping sensitive data private.</li> </ul> <p><b>Limitations:</b></p> <ul style="list-style-type: none"> <li>• Complexity in integration and management.</li> <li>• Security and compliance issues in data flow between environments.</li> </ul>			
5.c	<p>What is a warehouse-scale data center? How does it differ from modular data centers?</p> <p><b>Scheme: Definition + explanation with comparison of each – 2+3+3 Marks</b></p> <p><b>Solution:</b></p> <p><b>1. Warehouse-Scale Data Center (WSDC)</b></p> <ul style="list-style-type: none"> <li>• A <b>warehouse-scale data center</b> is a massive, centralized data center built in a single facility.</li> <li>• Designed like a <b>shopping mall or factory floor</b>, it can house <b>hundreds of thousands to millions of servers</b>.</li> <li>• Example: Google and Microsoft data centers are built to warehouse scale, often 11 times the size of a football field.</li> <li>• Offers <b>economies of scale</b>, reducing the cost per unit as the size increases.</li> <li>• Efficient for <b>centralized cloud services</b>, large-scale storage, and compute operations.</li> </ul> <p><b>Key Features:</b></p> <ul style="list-style-type: none"> <li>• Centralized management of resources.</li> <li>• High operational efficiency and performance.</li> <li>• Better suited for large-scale, high-throughput cloud services.</li> <li>• Lower network/storage costs due to scale.</li> </ul> <p><b>2. Modular Data Center</b></p> <ul style="list-style-type: none"> <li>• A <b>modular data center</b> consists of <b>containerized units</b>—server clusters housed inside <b>truck-towed containers</b>.</li> <li>• Example: SGI ICE Cube container can house <b>46,080 processing cores</b> or <b>30 PB of storage</b> per container.</li> <li>• These containers form a <b>shipping yard-style data center</b> and can be moved as needed.</li> <li>• Modular design enables <b>rapid deployment, mobility, and energy efficiency</b>.</li> </ul> <p><b>Key Features:</b></p> <ul style="list-style-type: none"> <li>• Better <b>cooling efficiency</b> (up to 80% less than traditional warehouses).</li> <li>• Suitable for <b>remote locations</b> with lower power, cooling, and housing costs.</li> <li>• Scalable by adding more containers.</li> <li>• Easier maintenance and relocation of infrastructure.</li> </ul>	[08]	3	L2

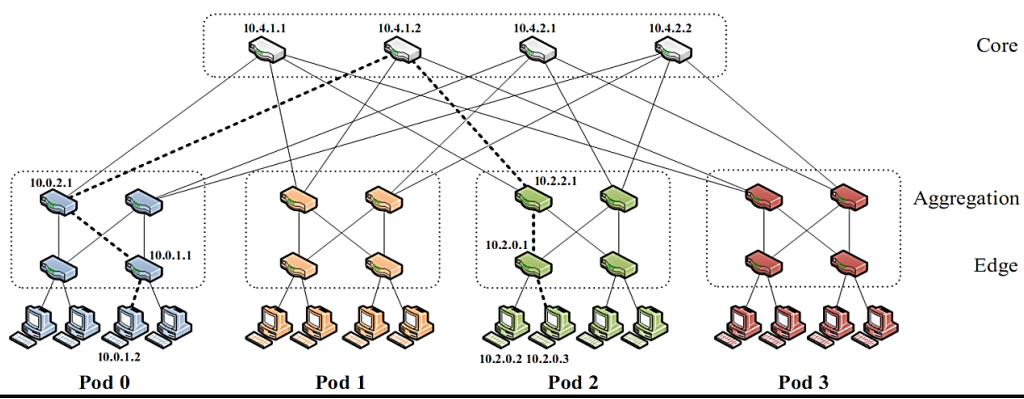
Comparison Table																																									
Feature	Warehouse-Scale Data Center	Modular Data Center																																							
Scale	400,000 to 1 million servers	Up to tens of thousands per container																																							
Structure	Centralized building	Truck-towed containers																																							
Deployment	Fixed, permanent	Rapid, flexible, mobile																																							
Cooling	Raised-floor CRAC systems	Heat exchangers and chilled water pipes																																							
Cost Efficiency	Lower operational costs at massive scale	Reduced cooling and installation costs																																							
Flexibility	Low mobility	High mobility and modularity																																							
OR																																									
6.a	<p>Compare the services and target applications of GAE, AWS, and Microsoft Azure.</p> <p><b>Scheme: Definition + explanation of each – 2+4 Marks</b></p> <p><b>Solution :</b></p> <p><b>GAE (Google App Engine):</b> A <b>Platform-as-a-Service (PaaS)</b> offering from Google Cloud that allows developers to build, deploy, and scale applications without managing underlying infrastructure.</p> <p><b>AWS (Amazon Web Services):</b> A comprehensive and widely adopted <b>cloud computing platform</b> by Amazon that provides <b>IaaS, PaaS, and SaaS</b> solutions for computing, storage, databases, AI, and more.</p> <p><b>Microsoft Azure:</b> A <b>cloud computing platform and service</b> created by Microsoft that supports a range of services including <b>virtual computing, analytics, storage, networking, and AI tools</b>, for building, testing, deploying, and managing applications</p> <table><tr><th>Feature / Criteria</th><th>Google App Engine (GAE)</th><th>Amazon Web Services (AWS)</th><th>Microsoft Azure</th></tr><tr><td>Service Type</td><td>PaaS</td><td>IaaS + PaaS + SaaS</td><td>IaaS + PaaS + SaaS</td></tr><tr><td>Primary Focus</td><td>Simplifying app development and deployment</td><td>Full-suite cloud infrastructure and services</td><td>Enterprise-grade cloud services, hybrid integration</td></tr><tr><td>Programming Languages</td><td>Java, Python, Go, Node.js, PHP, Ruby, .NET</td><td>Supports most languages: Java, Python, C#, PHP, Go, Ruby, etc.</td><td>C#, .NET, Java, Python, Node.js, and more</td></tr><tr><td>Compute Services</td><td>App Engine Standard &amp; Flexible Environment</td><td>EC2 (Elastic Compute Cloud), Lambda, ECS, EKS</td><td>Azure Virtual Machines, Functions, App Services</td></tr><tr><td>Storage Services</td><td>Cloud Datastore, Cloud SQL, Cloud Storage</td><td>S3, DynamoDB, RDS, EBS</td><td>Blob Storage, Azure SQL, Cosmos DB</td></tr><tr><td>Target Applications</td><td>Web apps, mobile backends, REST APIs</td><td>Web apps, mobile apps, big data, machine learning, enterprise apps</td><td>Enterprise apps, hybrid cloud, .NET-based solutions, DevOps</td></tr><tr><td>Scalability &amp; Management</td><td>Auto-scaling, serverless, fully managed</td><td>Auto-scaling, configurable management</td><td>Auto-scaling, integrated management tools</td></tr><tr><td>Use Case Examples</td><td>Hosting serverless web apps and APIs</td><td>Enterprise-scale apps, ML pipelines, IoT, data lakes</td><td>Business intelligence apps, hybrid environments, MS 365 integrations</td></tr></table>		Feature / Criteria	Google App Engine (GAE)	Amazon Web Services (AWS)	Microsoft Azure	Service Type	PaaS	IaaS + PaaS + SaaS	IaaS + PaaS + SaaS	Primary Focus	Simplifying app development and deployment	Full-suite cloud infrastructure and services	Enterprise-grade cloud services, hybrid integration	Programming Languages	Java, Python, Go, Node.js, PHP, Ruby, .NET	Supports most languages: Java, Python, C#, PHP, Go, Ruby, etc.	C#, .NET, Java, Python, Node.js, and more	Compute Services	App Engine Standard & Flexible Environment	EC2 (Elastic Compute Cloud), Lambda, ECS, EKS	Azure Virtual Machines, Functions, App Services	Storage Services	Cloud Datastore, Cloud SQL, Cloud Storage	S3, DynamoDB, RDS, EBS	Blob Storage, Azure SQL, Cosmos DB	Target Applications	Web apps, mobile backends, REST APIs	Web apps, mobile apps, big data, machine learning, enterprise apps	Enterprise apps, hybrid cloud, .NET-based solutions, DevOps	Scalability & Management	Auto-scaling, serverless, fully managed	Auto-scaling, configurable management	Auto-scaling, integrated management tools	Use Case Examples	Hosting serverless web apps and APIs	Enterprise-scale apps, ML pipelines, IoT, data lakes	Business intelligence apps, hybrid environments, MS 365 integrations	[06]	3	L3
Feature / Criteria	Google App Engine (GAE)	Amazon Web Services (AWS)	Microsoft Azure																																						
Service Type	PaaS	IaaS + PaaS + SaaS	IaaS + PaaS + SaaS																																						
Primary Focus	Simplifying app development and deployment	Full-suite cloud infrastructure and services	Enterprise-grade cloud services, hybrid integration																																						
Programming Languages	Java, Python, Go, Node.js, PHP, Ruby, .NET	Supports most languages: Java, Python, C#, PHP, Go, Ruby, etc.	C#, .NET, Java, Python, Node.js, and more																																						
Compute Services	App Engine Standard & Flexible Environment	EC2 (Elastic Compute Cloud), Lambda, ECS, EKS	Azure Virtual Machines, Functions, App Services																																						
Storage Services	Cloud Datastore, Cloud SQL, Cloud Storage	S3, DynamoDB, RDS, EBS	Blob Storage, Azure SQL, Cosmos DB																																						
Target Applications	Web apps, mobile backends, REST APIs	Web apps, mobile apps, big data, machine learning, enterprise apps	Enterprise apps, hybrid cloud, .NET-based solutions, DevOps																																						
Scalability & Management	Auto-scaling, serverless, fully managed	Auto-scaling, configurable management	Auto-scaling, integrated management tools																																						
Use Case Examples	Hosting serverless web apps and APIs	Enterprise-scale apps, ML pipelines, IoT, data lakes	Business intelligence apps, hybrid environments, MS 365 integrations																																						
6.b	<p>Describe the fat-free topology and explain its use in data center network architecture.</p> <p><b>Scheme: Definition + explanation of each – 2+4 Marks</b></p> <p><b>Solution :</b></p> <p><b>Fat-free topology</b> is a <b>variant of the Fat-Tree topology</b> used in data center networks. It aims to optimize the <b>network performance and resource usage</b> by eliminating unnecessary redundant links and switches while still maintaining <b>high bandwidth, fault tolerance, and scalability</b>.</p> <p>Explanation: Fat-free topology is derived from the Fat-Tree architecture, which in turn is based on the concept of Clos networks commonly used in telecommunication systems. The primary goal of the fat-free topology is to reduce hardware costs and power consumption while still maintaining near-optimal bandwidth and fault tolerance. Structurally, it resembles the Fat-Tree</p>		[06]	3	L2																																				

design but removes or compresses underutilized paths and links, resulting in a leaner and more efficient version.

The topology follows a hierarchical layout with core, aggregation, and edge layers, but includes fewer redundant paths compared to a full Fat-Tree. It intelligently reduces links based on utilization, ensuring better resource efficiency. In data centers, this topology is used to design cost-effective, energy-efficient, and scalable network architectures with reduced switches and cables.

Some notable benefits of fat-free topology include lower deployment costs, decreased energy consumption, simpler management, and performance that is still comparable to more complex topologies. However, it does come with certain limitations such as slightly reduced redundancy and fault tolerance. Additionally, it requires traffic analysis to determine which links can be safely removed without affecting overall performance.

This topology finds real-world applications in cloud data centers, enterprise networks, and AI training clusters, where resource optimization is critical to performance and cost management.



6.c Explain the key requirements for an efficient cloud data center interconnection network.

[08] 3 L2

**Scheme: Definition + explanation of each – 2+6 Marks**

**Solution :**

An **interconnection network** in a cloud data center is the communication backbone that connects **thousands of servers and storage devices** within and across multiple data centers. An **efficient interconnection network** ensures high-speed, low-latency, reliable, and scalable data transfer to support modern cloud applications and services.

The key requirements for an efficient cloud data center interconnection network are:

- 1. High Bandwidth**  
Cloud workloads such as big data analytics, machine learning, and content delivery require the **ability to transfer large volumes of data** quickly. The interconnection network must support **high-throughput links** across all nodes to prevent bottlenecks.
- 2. Low Latency**  
To ensure real-time responsiveness in applications such as video conferencing, online gaming, or stock trading, the network must provide **low end-to-end latency**. Low latency is especially crucial for distributed applications that require quick synchronization.
- 3. Scalability**  
The network architecture should support **scaling from hundreds to thousands (or millions) of servers** without degradation in performance. Technologies like **Fat-Tree, Clos, or Spine-Leaf** topologies help maintain efficiency as the system grows.
- 4. Fault Tolerance and Reliability**  
An efficient network must tolerate hardware failures (switches, links, etc.)



	<p>without interrupting service. Redundancy in paths, <b>multi-path routing (like ECMP)</b>, and failover mechanisms are vital to ensure <b>high availability</b>.</p> <p>5. <b>Efficient Load Balancing</b> Traffic should be evenly distributed across network links to <b>maximize utilization</b> and avoid congestion. Protocols and algorithms must dynamically reroute traffic in case of uneven load or link failure.</p> <p>6. <b>Energy Efficiency</b> Data centers consume vast amounts of power. The interconnection network must be <b>designed to minimize power consumption</b>, using techniques like power-aware routing, energy-efficient switches, and consolidating traffic during low usage.</p> <p>7. <b>Cost Effectiveness</b> Building and maintaining the network infrastructure must be <b>economically viable</b>. Use of <b>commodity switches</b>, intelligent topologies (e.g., Fat-Free), and optimized cabling help reduce CAPEX and OPEX.</p> <p>8. <b>Support for Virtualization and SDN</b> Modern data centers heavily use virtualization and <b>Software Defined Networking (SDN)</b>. The interconnection network must support <b>dynamic reconfiguration, network slicing, and multi-tenancy</b> to meet varied tenant requirements.</p>			
--	---	--	--	--

<b>MODULE-4</b>
-----------------

7.a	<p>What is a trusted Hypervisor? Explain the mobile devices face a range of security challenges?</p> <p><b>Scheme: Definition + explanation of each – 2+4 Marks</b></p> <p><b>Solution:</b> A trusted hypervisor is a secure and verified virtualization layer that manages virtual machines (VMs) while enforcing strong isolation and protection mechanisms between them. It is designed to be minimal, formally verified, and resistant to attacks, making it suitable for secure environments such as cloud data centers, military applications, and secure mobile platforms.</p> <div style="text-align: center;"> </div> <p>Mobile devices face a wide range of <b>security threats</b> due to their <b>portability, wireless connectivity, and open application ecosystem</b>. Key challenges include:</p> <ol style="list-style-type: none"> <li><b>Malware and Untrusted Apps</b> Apps from third-party stores can contain <b>malware, spyware, or ransomware</b>. Users often unknowingly grant unnecessary permissions.</li> <li><b>Data Leakage</b> Personal and sensitive data (contacts, messages, photos) can be exposed due to <b>poor app design</b> or malicious intent.</li> <li><b>Network-based Attacks</b> Mobile devices are frequently exposed to <b>Wi-Fi spoofing, man-in-the-middle attacks, and packet sniffing</b>, especially on public networks.</li> <li><b>Device Theft and Loss</b> Physical access to a lost/stolen phone can allow attackers to <b>bypass security</b>,</li> </ol>	[06]	4	L2
-----	--	------	---	----

	<p>especially if encryption or screen lock is weak.</p> <p>5. <b>Outdated Software and OS Vulnerabilities</b> Delayed updates or unsupported devices lead to unpatched vulnerabilities that attackers can exploit.</p> <p>6. <b>Insecure Communication</b> Lack of <b>end-to-end encryption</b> in messaging, emails, or calls increases risk of interception.</p> <p>7. <b>Permissions Misuse</b> Apps may request <b>excessive or unnecessary permissions</b>, potentially leading to privacy invasions.</p> <p>8. <b>Rooting/Jailbreaking</b> Users who root or jailbreak their devices disable default security layers, making them <b>highly vulnerable to attacks</b>.</p>			
7.b	<p>Discuss the security risks posed by shared images and management OS.</p> <p><b>Scheme: Definition + explanation of each – 2+2+2 Marks</b></p> <p><b>Solution:</b></p> <ul style="list-style-type: none"> <li>Shared Images refer to virtual machine (VM) or container images that are reused or distributed across multiple systems or tenants in a cloud environment.</li> <li>Management OS (Operating System) is the underlying operating system responsible for managing virtualization platforms, storage, network, and administrative functions.</li> </ul> <p>1. Risks from Shared Images</p> <ul style="list-style-type: none"> <li><b>Embedded Malware or Backdoors</b> Shared or public images may contain malicious code left by the creator—either unintentionally or intentionally—leading to system compromise once deployed.</li> <li><b>Residual Sensitive Data</b> Images may contain leftover credentials, SSH keys, API tokens, or configuration files, which attackers can exploit to gain access.</li> <li><b>Unpatched Vulnerabilities</b> If shared images are outdated or not regularly patched, they may include known vulnerabilities that attackers can easily exploit.</li> <li><b>Privilege Escalation</b> Improper image configuration can allow users or attackers to gain elevated privileges, leading to unauthorized access or control.</li> <li><b>Image Spoofing</b> In public repositories, attackers can upload malicious images disguised as popular ones, tricking users into deploying compromised software.</li> </ul> <p>2. Risks from Management OS</p> <ul style="list-style-type: none"> <li><b>Single Point of Failure</b> If the management OS is compromised, all VMs or containers running on that system can be exposed to security breaches.</li> <li><b>Hypervisor or API Exploits</b> Vulnerabilities in the management OS or hypervisor APIs can be used by attackers to escape the VM and access other parts of the infrastructure (e.g., hypervisor escape attacks).</li> <li><b>Unauthorized Access or Misconfiguration</b> Weak passwords, open ports, or poorly configured firewall rules can allow attackers</li> </ul>	[06]	4	L2

	<p>to remotely access and control the management OS.</p> <ul style="list-style-type: none"> <li>Malicious Insider Threats Administrators or users with access to the management OS can intentionally or unintentionally cause data breaches, leakage, or service disruptions.</li> <li>Insufficient Logging and Monitoring Without proper logging, detecting abnormal activity in the management OS becomes difficult, allowing attacks to go unnoticed.</li> </ul>			
7.c	<p>Describe the hidden risks in cloud computing.</p> <p><b>Scheme: Definition + explanation of each – 2+2+4 Marks</b></p> <p><b>Solution:</b></p> <p><b>Hidden risks in cloud computing</b> refer to the <b>less visible or indirect threats</b> that may not be immediately apparent during cloud adoption or use. These include risks related to <b>data control, compliance, vendor lock-in, and multi-tenancy</b> that could lead to <b>security breaches, data loss, or legal issues</b> if not properly addressed.</p> <ul style="list-style-type: none"> <li>Cloud computing, while offering numerous advantages, also carries several hidden risks that organizations must carefully consider.</li> <li>One major concern is <b>data breaches and leakage</b>, which may occur due to misconfigured storage or weak access controls, allowing unauthorized users to access sensitive information.</li> <li><b>Vendor lock-in</b> is another risk, where organizations become highly dependent on a single cloud provider's tools, making it difficult and costly to switch platforms, thus limiting flexibility.</li> <li><b>Insider threats</b> pose significant danger as cloud provider employees may have privileged access to customer data, potentially leading to data manipulation or theft through either malicious intent or human error.</li> <li><b>Compliance and legal issues</b> also arise due to the global nature of cloud storage; storing data across different countries may violate local regulations like GDPR or HIPAA if not managed properly.</li> <li>Additionally, there is often <b>inadequate visibility and control</b>, meaning customers cannot fully monitor or audit the underlying infrastructure, making it hard to detect or respond to security incidents.</li> <li><b>Insecure APIs and interfaces</b>, which serve as the primary gateways to cloud services, can become easy targets for attackers if poorly designed or unprotected.</li> <li>In <b>multi-tenant environments</b>, risks of <b>resource exhaustion and abuse</b> are also present—one tenant's excessive usage or a DDoS attack can degrade service for others.</li> <li>Lastly, the rise of <b>Shadow IT and misconfigurations</b>, where employees use unauthorized services or deploy systems without proper oversight, introduces vulnerabilities and unmonitored entry points into the organization.</li> </ul>	[08]	4	L3
OR				



8.a	<p>Explain the best top 5 cloud security best practices.</p> <p><b>Scheme: Definition + explanation of each – 2+2+2 Marks</b></p> <p><b>Solution:</b></p> <ol style="list-style-type: none"> <li>1.Enable Strong Identity and Access Management (IAM) <ul style="list-style-type: none"> <li>• Use Multi-Factor Authentication (MFA) to add an extra layer of protection.</li> <li>• Apply the Principle of Least Privilege (PoLP) to ensure users and services only have the access they need.</li> <li>• Regularly review and rotate access keys and credentials.</li> </ul> </li> <li>2.Encrypt Data at Rest and in Transit <ul style="list-style-type: none"> <li>• Protect sensitive information using AES-256 encryption for stored data and TLS/SSL protocols for data in motion.</li> <li>• Manage encryption keys securely using Key Management Services (KMS) provided by cloud vendors.</li> </ul> </li> <li>3.Regularly Monitor and Audit Cloud Resources <ul style="list-style-type: none"> <li>• Enable <b>logging and monitoring tools</b> like AWS CloudTrail, Azure Monitor, or Google Cloud Logging.</li> <li>• Use <b>Security Information and Event Management (SIEM)</b> systems for centralized threat detection and response.</li> </ul> <p>Conduct <b>regular security audits and compliance checks</b>.</p> <li>4.Secure APIs and Endpoints <ul style="list-style-type: none"> <li>• Use authentication tokens, rate limiting, and input validation to protect APIs.</li> <li>• Regularly scan for vulnerabilities and misconfigurations in exposed services and interfaces.</li> </ul> </li> <li>5.Implement Robust Backup and Disaster Recovery Plans <ul style="list-style-type: none"> <li>• Ensure regular and automated backups are in place.</li> <li>• Test disaster recovery procedures to ensure data availability and business continuity during incidents.</li> <li>• Use geo-redundant storage to protect against regional failures.</li> </ul> </li> </li></ol>	[06]	4	L2
8.b	<p>What are the most important advantages of cloud technology for social networks?</p> <p><b>Scheme: Definition + explanation of each – 2+4 Marks</b></p> <p><b>Solution:</b></p> <p><b>Key Advantages of Cloud Technology for Social Networks</b></p> <ol style="list-style-type: none"> <li>1. <b>Scalability</b> <p>Cloud infrastructure allows social networks to <b>scale up or down automatically</b> based on user demand. This is crucial during <b>traffic spikes</b>, like viral posts or global events.</p> </li> <li>2. <b>High Availability and Reliability</b> <p>Cloud platforms ensure <b>continuous uptime</b> using <b>redundant systems and data replication</b>, keeping social networks accessible 24/7.</p> </li> <li>3. <b>Cost Efficiency</b> <p>Instead of investing in expensive hardware, social media companies can use <b>pay-as-you-go models</b>, reducing <b>capital expenditure (CAPEX)</b> and optimizing operational costs.</p> </li> <li>4. <b>Global Accessibility</b> <p>With <b>cloud data centers spread worldwide</b>, content delivery is faster, enabling users from different regions to experience <b>low latency and quick access</b>.</p> </li> <li>5. <b>Big Data Storage and Analytics</b> <p>Social networks generate vast amounts of data. Cloud platforms provide <b>massive storage</b> and <b>real-time analytics tools</b> to process user behavior, preferences, and trends efficiently.</p> </li> </ol>	[06]	4	L2

	<p><b>6. Security and Compliance</b></p> <p>Cloud providers offer <b>built-in security features</b> like encryption, DDoS protection, and access control, helping social networks <b>protect user data</b> and comply with regulations.</p> <p><b>7. Rapid Deployment and Innovation</b></p> <p>Developers can use cloud services to <b>quickly test and deploy new features</b>, improving user engagement and experience without long setup times</p>			
8.c	<p>Describe the key features of Google.</p> <p><b>Scheme: Features + explanation of each – 2+6 Marks</b></p> <p><b>Solution:</b></p> <ul style="list-style-type: none"> <li>Google is a global technology leader known primarily for its powerful and fast search engine, which delivers relevant and ranked results using advanced algorithms.</li> <li>Beyond search, Google offers a wide range of services under Google Workspace, including Gmail for email, Google Drive for cloud storage, and productivity tools like Google Docs, Sheets, and Slides for online collaboration.</li> <li>Google Maps is another standout feature, providing real-time navigation, traffic updates, and satellite views, making it an essential tool for travel and local discovery.</li> <li>Google Chrome, the company's web browser, is known for its speed, security, and seamless syncing across devices.</li> <li>For mobile users, the Google Play Store serves as a hub for downloading Android apps, games, books, and media content.</li> <li>Additionally, Google Photos offers cloud-based image and video storage with features like automatic backup and AI-powered organization.</li> <li>Google Translate supports real-time translation of text, speech, and images in over 100 languages, making global communication more accessible.</li> <li>Google Assistant, the company's virtual AI helper, allows users to perform tasks, ask questions, and control smart devices using voice commands.</li> <li>Google also owns YouTube, the world's largest video-sharing platform, offering content creation, streaming, and monetization options.</li> <li>Across all its services, Google emphasizes user privacy and security through features like two-step verification, data encryption, and account monitoring, ensuring a safer online experience.</li> </ul>	[08]	4	L2
<b><u>MODULE-5</u></b>				
9.a	<p>What are the benefits and challenges of using serverless computing in the cloud?</p> <p><b>Scheme: Definition + explanation of each – 2+2+2 Marks</b></p> <p><b>Solution:</b></p> <p><b>Benefits of Serverless Computing</b></p> <ol style="list-style-type: none"> <li><b>Cost Efficiency</b> <ul style="list-style-type: none"> <li>Pay only for the execution time of code (no charge when code is idle).</li> </ul> </li> <li><b>Automatic Scalability</b> <ul style="list-style-type: none"> <li>Functions scale up or down automatically based on demand.</li> </ul> </li> <li><b>Faster Development and Deployment</b></li> </ol>	[06]	5	L2

	<ul style="list-style-type: none"><li>○ Developers focus on writing code without worrying about infrastructure setup.</li></ul> <p>4. <b>Simplified Operations</b></p> <ul style="list-style-type: none"><li>○ No need to manage or maintain servers.</li></ul> <p>5. <b>Built-in Availability and Fault Tolerance</b></p> <ul style="list-style-type: none"><li>○ Cloud providers handle high availability and redundancy.</li></ul> <p><b>Challenges of Serverless Computing</b></p> <p>1. <b>Cold Start Latency</b></p> <ul style="list-style-type: none"><li>○ Delay in execution when functions are invoked after a period of inactivity.</li></ul> <p>2. <b>Execution Limits</b></p> <ul style="list-style-type: none"><li>○ Functions may have restrictions on memory, duration, and concurrent executions.</li></ul> <p>3. <b>Vendor Lock-in</b></p> <ul style="list-style-type: none"><li>○ Difficult to migrate code and architecture across different cloud platforms.</li></ul> <p>4. <b>Limited Debugging and Monitoring</b></p> <ul style="list-style-type: none"><li>○ It is harder to trace bugs or analyze performance due to lack of server access.</li></ul> <p>5. <b>Security and Compliance Concerns</b></p> <ul style="list-style-type: none"><li>○ Managing secure data flow between stateless functions can be complex.</li></ul>																																	
9.b	<p>How does grid computing differ from cloud computing?</p> <p><b>Scheme: Features + explanation of each – 2+2+2 Marks</b></p> <p><b>Solution:</b></p> <table><tr><th>Feature</th><th>Grid Computing</th><th>Cloud Computing</th></tr><tr><td>Definition</td><td>A distributed system where multiple computers work together to complete large tasks.</td><td>A service model delivering computing resources (e.g., storage, servers, databases) over the internet.</td></tr><tr><td>Resource Ownership</td><td>Resources are usually shared across multiple organizations.</td><td>Resources are owned and managed by a cloud provider (like AWS, Azure, or GCP).</td></tr><tr><td>Scalability</td><td>Limited scalability based on the physical infrastructure of the grid.</td><td>Highly scalable; resources can be added or removed on demand.</td></tr><tr><td>Flexibility</td><td>Less flexible; often requires manual configuration.</td><td>Highly flexible with on-demand provisioning and automation.</td></tr><tr><td>Cost Model</td><td>Often free or fixed-cost usage across institutions.</td><td>Pay-as-you-go or subscription-based pricing model.</td></tr><tr><td>Use Case</td><td>Best for scientific and research-based distributed computing tasks.</td><td>Ideal for web hosting, SaaS, big data, AI, and enterprise apps.</td></tr><tr><td>Accessibility</td><td>Usually limited to authorized users within organizations.</td><td>Accessible over the internet from anywhere.</td></tr><tr><td>Fault Tolerance</td><td>Less robust; may require manual recovery.</td><td>High availability and fault tolerance are built-in.</td></tr><tr><td>Virtualization</td><td>Rarely used; typically physical machines.</td><td>Heavily reliant on virtualization and containerization.</td></tr></table>	Feature	Grid Computing	Cloud Computing	Definition	A distributed system where multiple computers work together to complete large tasks.	A service model delivering computing resources (e.g., storage, servers, databases) over the internet.	Resource Ownership	Resources are usually shared across multiple organizations.	Resources are owned and managed by a cloud provider (like AWS, Azure, or GCP).	Scalability	Limited scalability based on the physical infrastructure of the grid.	Highly scalable; resources can be added or removed on demand.	Flexibility	Less flexible; often requires manual configuration.	Highly flexible with on-demand provisioning and automation.	Cost Model	Often free or fixed-cost usage across institutions.	Pay-as-you-go or subscription-based pricing model.	Use Case	Best for scientific and research-based distributed computing tasks.	Ideal for web hosting, SaaS, big data, AI, and enterprise apps.	Accessibility	Usually limited to authorized users within organizations.	Accessible over the internet from anywhere.	Fault Tolerance	Less robust; may require manual recovery.	High availability and fault tolerance are built-in.	Virtualization	Rarely used; typically physical machines.	Heavily reliant on virtualization and containerization.	[06]	5	L2
Feature	Grid Computing	Cloud Computing																																
Definition	A distributed system where multiple computers work together to complete large tasks.	A service model delivering computing resources (e.g., storage, servers, databases) over the internet.																																
Resource Ownership	Resources are usually shared across multiple organizations.	Resources are owned and managed by a cloud provider (like AWS, Azure, or GCP).																																
Scalability	Limited scalability based on the physical infrastructure of the grid.	Highly scalable; resources can be added or removed on demand.																																
Flexibility	Less flexible; often requires manual configuration.	Highly flexible with on-demand provisioning and automation.																																
Cost Model	Often free or fixed-cost usage across institutions.	Pay-as-you-go or subscription-based pricing model.																																
Use Case	Best for scientific and research-based distributed computing tasks.	Ideal for web hosting, SaaS, big data, AI, and enterprise apps.																																
Accessibility	Usually limited to authorized users within organizations.	Accessible over the internet from anywhere.																																
Fault Tolerance	Less robust; may require manual recovery.	High availability and fault tolerance are built-in.																																
Virtualization	Rarely used; typically physical machines.	Heavily reliant on virtualization and containerization.																																
9.c	<p>What are the key features of cloud computing platforms?</p> <p><b>Scheme: Features + explanation of each – 2+2+4 Marks</b></p> <p><b>Solution:</b></p> <p><b>Key Features of Cloud Computing Platforms</b></p> <p>1. <b>On-Demand Self-Service</b></p> <ul style="list-style-type: none"><li>○ Users can provision computing resources like storage, server time, and applications as needed, without requiring human interaction with the provider.</li></ul>	[08]	5	L2																														

	<div>2. <b>Broad Network Access</b></div> <div>○ Services are available over the network and accessible through standard devices such as laptops, mobiles, and tablets.</div> <div>3. <b>Resource Pooling</b></div> <div>○ Computing resources are pooled to serve multiple users using a multi-tenant model, with different physical and virtual resources dynamically assigned.</div> <div>4. <b>Rapid Elasticity</b></div> <div>○ Resources can be quickly scaled up or down automatically, depending on user demand.</div> <div>5. <b>Measured Service (Pay-as-You-Go)</b></div> <div>○ Resource usage is monitored, controlled, and billed based on consumption — promoting efficient use of resources.</div> <div>6. <b>Scalability and Flexibility</b></div> <div>○ Easily accommodate growth or changes in workload without service interruption.</div> <div>7. <b>High Availability and Reliability</b></div> <div>○ Built-in redundancy and failover mechanisms ensure services are continuously available.</div> <div>8. <b>Automatic Updates and Maintenance</b></div> <div>○ The cloud provider handles system updates, software patches, and routine maintenance.</div>																								
OR																									
10. a	<div>How do multi-cloud and hybrid cloud strategies differ?</div> <div>Scheme: differences with explanation of each – 2+4 Marks</div> <div>Solution:</div> <table><tr><th>Aspect</th><th>Multi-Cloud</th><th>Hybrid Cloud</th></tr><tr><td>Cloud Types</td><td>Multiple public clouds</td><td>Combination of private (on-premises) and public clouds</td></tr><tr><td>Goal</td><td>Avoid vendor lock-in, optimize cost and performance</td><td>Balance control, security, and scalability</td></tr><tr><td>Integration Level</td><td>Looser integration, separate environments</td><td>Tight integration between private and public clouds</td></tr><tr><td>Data Location</td><td>Spread across multiple public cloud providers</td><td>Data split between on-premises/private and public cloud</td></tr><tr><td>Use Case Examples</td><td>Using AWS for storage, Azure for AI services</td><td>On-premises data center with AWS for overflow or backup</td></tr><tr><td>Focus</td><td>Flexibility, redundancy, best-of-breed services</td><td>Combining control with cloud scalability</td></tr></table>	Aspect	Multi-Cloud	Hybrid Cloud	Cloud Types	Multiple public clouds	Combination of private (on-premises) and public clouds	Goal	Avoid vendor lock-in, optimize cost and performance	Balance control, security, and scalability	Integration Level	Looser integration, separate environments	Tight integration between private and public clouds	Data Location	Spread across multiple public cloud providers	Data split between on-premises/private and public cloud	Use Case Examples	Using AWS for storage, Azure for AI services	On-premises data center with AWS for overflow or backup	Focus	Flexibility, redundancy, best-of-breed services	Combining control with cloud scalability	[06]	5	L2
Aspect	Multi-Cloud	Hybrid Cloud																							
Cloud Types	Multiple public clouds	Combination of private (on-premises) and public clouds																							
Goal	Avoid vendor lock-in, optimize cost and performance	Balance control, security, and scalability																							
Integration Level	Looser integration, separate environments	Tight integration between private and public clouds																							
Data Location	Spread across multiple public cloud providers	Data split between on-premises/private and public cloud																							
Use Case Examples	Using AWS for storage, Azure for AI services	On-premises data center with AWS for overflow or backup																							
Focus	Flexibility, redundancy, best-of-breed services	Combining control with cloud scalability																							
10. b	<div>What is Infrastructure as Code (IaC) and how is it used in the cloud?</div> <div>Scheme: Definition + explanation of each – 2+2+2 Marks</div> <div>Solution:</div>	[06]	5	L2																					

	<p><b>Infrastructure as Code (IaC)</b> is a practice where you manage and provision computing infrastructure (like servers, networks, storage) through machine-readable configuration files, rather than manual hardware setup or interactive configuration tools.</p> <p><b>How is IaC used in the cloud?</b></p> <ul style="list-style-type: none"> <li>• <b>Automated Provisioning:</b> You define cloud resources (VMs, networks, storage, load balancers, etc.) in code, and tools automatically create and configure them in your cloud environment.</li> <li>• <b>Version Control &amp; Collaboration:</b> Infrastructure definitions can be stored in Git or similar systems, allowing teams to collaborate and track changes.</li> <li>• <b>Consistency &amp; Repeatability:</b> Ensures environments (dev, test, prod) are consistent and can be recreated easily.</li> <li>• <b>Disaster Recovery &amp; Scaling:</b> Quickly rebuild infrastructure or scale up/down by applying the code.</li> <li>• <b>Examples of IaC Tools:</b> Terraform, AWS CloudFormation, Azure Resource Manager (ARM) templates, Google Cloud Deployment Manager, Ansible (for configuration management).</li> </ul>			
10. c	<p>What are emerging cloud environments?</p> <p><b>Scheme: Definition + explanation of each – 2+6 Marks</b></p> <p><b>Solution:</b></p> <p><b>Emerging cloud environments</b> refer to new or evolving types of cloud computing platforms and models that go beyond traditional public, private, and hybrid clouds. These environments address specific needs, leverage advanced technologies, and expand cloud capabilities in innovative ways.</p> <p>Here are some key <b>emerging cloud environments</b>:</p> <p><b>1. Edge Cloud / Edge Computing</b></p> <ul style="list-style-type: none"> <li>• <b>Description:</b> Cloud resources and services are deployed closer to data sources or end users (e.g., IoT devices, local edge servers) instead of centralized data centers.</li> <li>• <b>Purpose:</b> Reduce latency, improve real-time data processing, and optimize bandwidth usage.</li> <li>• <b>Use cases:</b> Autonomous vehicles, smart cities, augmented reality, industrial automation.</li> </ul> <p><b>2. Fog Computing</b></p> <ul style="list-style-type: none"> <li>• <b>Description:</b> Extends cloud computing to the network edge but includes multiple layers between the cloud and devices, creating a decentralized infrastructure.</li> <li>• <b>Purpose:</b> Enhance data processing closer to devices while maintaining cloud connectivity.</li> </ul>	[08]	5	L2



	<ul style="list-style-type: none"> <li>● <b>Use cases:</b> IoT applications requiring quick local decision-making with cloud backup.</li> </ul>			
	<p><b>3. Serverless Computing (Function as a Service - FaaS)</b></p> <ul style="list-style-type: none"> <li>● <b>Description:</b> Abstracts server management entirely, letting developers deploy functions triggered by events without managing infrastructure.</li> <li>● <b>Purpose:</b> Simplify development, scale automatically, and optimize resource usage.</li> <li>● <b>Use cases:</b> APIs, microservices, real-time data processing.</li> </ul> <p><b>4. Multi-Access Edge Computing (MEC)</b></p> <ul style="list-style-type: none"> <li>● <b>Description:</b> Similar to edge computing but specifically focused on telecom networks, allowing cloud computing at the edge of mobile networks.</li> <li>● <b>Purpose:</b> Support low-latency 5G applications, mobile gaming, and IoT.</li> </ul> <p><b>5. Cloud-native Environments</b></p> <ul style="list-style-type: none"> <li>● <b>Description:</b> Cloud platforms optimized for microservices, containers, and orchestration (e.g., Kubernetes).</li> <li>● <b>Purpose:</b> Enable flexible, scalable, and resilient application deployment and management.</li> <li>● <b>Use cases:</b> Modern app development and DevOps pipelines.</li> </ul> <p><b>6. Distributed Cloud</b></p> <ul style="list-style-type: none"> <li>● <b>Description:</b> Cloud services distributed across multiple physical locations but operated and managed as a single cloud.</li> <li>● <b>Purpose:</b> Combine benefits of centralized management with localized service delivery.</li> <li>● <b>Use cases:</b> Data sovereignty, compliance, disaster recovery.</li> </ul>			

Faculty Signature

CCI Signature

HOD Signature