Assignment : Cloud Computing

**Submitted By :- Purv Patel**

**Module 2: Virtualization And Storage Management**

**1-What is virtualization and virtualization type?**

## > **What is Virtualization?**

### **Definition**

Virtualization is the **process of creating virtual instances** of computing resources such as servers, storage, networks, or operating systems. It allows multiple **virtual machines (VMs)** to run on a single physical machine, optimizing hardware usage and improving efficiency.

### **Key Benefits**

✅ **Efficient Resource Utilization** – Maximizes hardware performance  
✅ **Cost Savings** – Reduces the need for physical hardware  
✅ **Scalability** – Easily add or remove virtual machines as needed  
✅ **Isolation** – Each VM operates independently, improving security  
✅ **Simplified Management** – Centralized administration and automation

## **Types of Virtualization**

### **1. Server Virtualization**

✅ Divides a **physical server** into multiple **virtual machines (VMs)**  
✅ Allows multiple OS environments to run on a single server  
✅ Uses **Hypervisors** like VMware ESXi, Microsoft Hyper-V, KVM

🔹 **Example:** Running Windows Server and Linux on the same physical server

### **2. Desktop Virtualization**

✅ Hosts **virtual desktops** on a centralized server  
✅ Users access their desktops remotely from any device  
✅ Improves security and reduces hardware costs

🔹 **Example:** Virtual Desktop Infrastructure (VDI) solutions like Citrix, Microsoft RDS

### **3. Network Virtualization**

✅ Combines multiple **network resources** into a single software-based network  
✅ Enables **software-defined networking (SDN)** for better control  
✅ Increases flexibility and security

🔹 **Example:** VLANs (Virtual LANs), VMware NSX, Cisco ACI

### **4. Storage Virtualization**

✅ Combines multiple physical storage devices into a single **virtual storage pool**  
✅ Improves storage management, redundancy, and performance

🔹 **Example:** Software-Defined Storage (SDS), SAN (Storage Area Network)

### **5. Application Virtualization**

✅ Runs applications in **isolated environments** without installing them locally  
✅ Improves compatibility and security

🔹 **Example:** Microsoft App-V, VMware ThinApp

### **6. Data Virtualization**

✅ Integrates data from multiple sources into a **single virtual view**  
✅ Simplifies data access without requiring data movement

🔹 **Example:** Cloud data virtualization, IBM Cloud Pak

**2-Type of hypervisor and how to manage it?**

## > **Types of Hypervisors and How to Manage Them**

### **What is a Hypervisor?**

A **hypervisor** is software that enables **virtualization** by allowing multiple virtual machines (**VMs**) to run on a single physical machine. It manages hardware resources and isolates each VM for secure and efficient operations.

## **Types of Hypervisors**

### **1. Type 1 Hypervisor (Bare-Metal)**

✅ **Installed directly on hardware** (no operating system required)  
✅ **More secure and efficient** than Type 2 hypervisors  
✅ Used in **enterprise and data center environments**

🔹 **Examples:**

* VMware ESXi
* Microsoft Hyper-V (on Windows Server)
* KVM (Kernel-based Virtual Machine)
* Citrix XenServer

### **2. Type 2 Hypervisor (Hosted) 💻**

✅ Runs on top of an **existing operating system** (Windows, macOS, Linux)  
✅ Easier to install but less efficient than Type 1  
✅ Used for **testing, development, and personal use**

🔹 **Examples:**

* VMware Workstation
* Oracle VirtualBox
* Microsoft Hyper-V (on Windows 10/11 Pro)
* Parallels Desktop (for macOS)

## **How to Manage a Hypervisor**

### **1. Using Management Interfaces**

🔹 **VMware ESXi** – Managed via **vSphere Client or vCenter**  
🔹 **Microsoft Hyper-V** – Managed via **Hyper-V Manager or PowerShell**  
🔹 **KVM** – Managed via **Virtual Machine Manager (VMM) or CLI**

### **2. Managing VMs with CLI & PowerShell**

✅ **List Virtual Machines (VMs) in Hyper-V**

Get-VM

✅ **Create a New VM in Hyper-V**

New-VM -Name "TestVM" -MemoryStartupBytes 2GB -Generation 2 -NewVHDSizeBytes 50GB -Path "C:\VMs"

✅ **Start a VM in VMware ESXi**

vim-cmd vmsvc/power.on <VM\_ID>

✅ **List KVM Virtual Machines**

virsh list --all

### **3. Network & Storage Management**

✅ Configure **virtual switches** for networking  
✅ Allocate **virtual storage (SAN, NAS, or local storage)**  
✅ Set up **VM snapshots and backups** for recovery

### **4. Monitoring & Optimization**

✅ Use **vCenter, Hyper-V Performance Monitor, or Prometheus** for monitoring  
✅ Optimize resource allocation (**CPU, RAM, storage**)  
✅ Ensure **regular updates and patches** for security

**3-Roles of virtualization in cloud computing?**

## > **Roles of Virtualization in Cloud Computing**

Virtualization plays a **key role** in cloud computing by enabling efficient resource management, scalability, and cost savings. It allows multiple **virtual instances** (VMs, networks, storage) to run on shared physical hardware, forming the foundation of cloud services.

### **1. Resource Optimization & Efficiency**

✅ Virtualization **divides hardware resources** among multiple virtual machines (VMs)  
✅ Maximizes **CPU, RAM, storage, and network utilization**  
✅ Reduces hardware waste, making cloud computing more **cost-effective**

🔹 **Example:** A single physical server can host multiple virtual machines, each running a different OS or application.

### **2. Scalability & Flexibility**

✅ Virtual machines can be **quickly deployed, resized, or migrated**  
✅ Supports **dynamic scaling** based on demand (e.g., auto-scaling in AWS, Azure)  
✅ Ideal for businesses that require **on-demand resources**

🔹 **Example:** Cloud providers like AWS, Azure, and Google Cloud allow users to scale VM instances up or down automatically.

### **3. Cost Savings**

✅ Reduces the need for **physical hardware**, lowering capital expenses  
✅ Cloud providers use **multi-tenant virtualization** to serve multiple users efficiently  
✅ Organizations **pay only for what they use** (Pay-as-you-go model)

🔹 **Example:** Instead of buying multiple physical servers, businesses can run multiple VMs on a single machine.

### **4. Disaster Recovery & High Availability**

✅ Virtualization enables **automated backups, failover, and recovery**  
✅ VMs can be **replicated** across different data centers for redundancy  
✅ Ensures **minimal downtime** in case of failures

🔹 **Example:** VMware vSphere and Microsoft Hyper-V offer live migration and fault tolerance features.

### **5. Security & Isolation**

✅ Each VM operates in an **isolated environment**, preventing security breaches  
✅ Virtual **firewalls, network segmentation, and encryption** enhance security  
✅ Enables **sandboxing** for testing applications safely

🔹 **Example:** Cloud providers use hypervisors to separate customer workloads for security and compliance.

### **6. Multi-Tenancy Support**

✅ Virtualization allows multiple users (**tenants**) to share cloud resources securely  
✅ Ensures **efficient workload distribution** while maintaining user isolation  
✅ Helps cloud providers **serve multiple clients from a single infrastructure**

🔹 **Example:** AWS EC2 instances allow multiple users to run independent VMs on shared physical hardware.

**4-What is container?**

## > **What is a Container?**

A **container** is a lightweight, standalone, and executable software package that includes everything needed to run an application: **code, runtime, libraries, and dependencies**. Containers **isolate applications** from the underlying system, ensuring consistency across different computing environments.

### **Key Characteristics of Containers**

✅ **Lightweight** – Shares the host OS kernel, reducing overhead  
✅ **Portable** – Runs consistently across different environments (development, testing, production)  
✅ **Fast Deployment** – Starts in seconds compared to traditional virtual machines  
✅ **Isolated & Secure** – Each container runs independently without interfering with others  
✅ **Efficient** – Requires fewer system resources than VMs

### **Containers vs. Virtual Machines (VMs)**

| Feature | Containers | Virtual Machines |
| --- | --- | --- |
| **OS Dependency** | Shares the host OS kernel | Each VM runs its own OS |
| **Size** | Small (MBs) | Large (GBs) |
| **Performance** | Fast startup | Slower startup |
| **Resource Usage** | Uses fewer resources | Requires more resources |
| **Isolation** | Process-level isolation | Full system isolation |

🔹 **Example:** A VM needs an entire OS to run an application, while a container runs the application with only its dependencies, sharing the host OS kernel.

### **Popular Container Platforms**

✅ **Docker** – Most widely used containerization platform  
✅ **Kubernetes** – Manages and orchestrates containerized applications  
✅ **Podman** – Alternative to Docker, focused on security  
✅ **LXC (Linux Containers)** – System-level containerization

### **Use Cases of Containers**

🚀 **Microservices Architecture** – Containers break down applications into smaller, independent services  
🚀 **DevOps & CI/CD** – Enables **continuous integration & deployment**  
🚀 **Cloud-Native Applications** – Run seamlessly on cloud platforms like AWS, Azure, and GCP  
🚀 **Testing & Development** – Developers can create **consistent environments** across different stages

**5-What is high availability and live migration in virtualization?**

## > **High Availability (HA) & Live Migration in Virtualization**

### **1. High Availability (HA) in Virtualization**

**Definition:**  
High Availability (**HA**) ensures that virtual machines (VMs) remain operational **with minimal downtime** even in the event of hardware or software failures. HA solutions automatically detect failures and restart VMs on another healthy host.

✅ **Ensures continuous availability of applications**  
✅ **Prevents downtime** due to hardware or software failures  
✅ **Automatically restarts VMs on another host** if one fails

🔹 **Example:**

* **VMware vSphere HA** – Automatically restarts VMs on another ESXi host if a server fails
* **Microsoft Hyper-V Failover Clustering** – Moves VMs to another node in case of failure
* **KVM with Pacemaker & Corosync** – Ensures automatic failover in Linux environments

### **2. Live Migration in Virtualization**

**Definition:**  
Live Migration allows moving a running **virtual machine (VM) from one physical host to another** **without downtime** or service interruption. It is useful for load balancing, hardware maintenance, and failover prevention.

✅ **No downtime** – Users don’t experience interruptions  
✅ **Resource balancing** – Move VMs to balance CPU, RAM, or network load  
✅ **Hardware maintenance** – Perform server updates without stopping VMs

🔹 **Example:**

* **VMware vMotion** – Transfers VMs between ESXi hosts seamlessly
* **Microsoft Hyper-V Live Migration** – Moves VMs between Hyper-V servers with no service impact
* **KVM Live Migration** – Uses virsh migrate command for seamless VM transfer

### **Key Differences: HA vs. Live Migration**

| Feature | High Availability (HA) | Live Migration |
| --- | --- | --- |
| **Purpose** | Ensures VM uptime after failure | Moves running VMs without downtime |
| **Trigger** | Failure detection (hardware, network, OS) | Manual or automatic workload balancing |
| **Downtime** | **Minimal** (VM restarts) | **Zero** (VM keeps running) |
| **Example** | Failover cluster, automatic VM restart | Maintenance, load balancing |

**6-Storage configuration –describe block storage, file storage and object storage---DAS NAS and SAN**

## > **Storage Configuration: Block, File, and Object Storage**

Storage in IT infrastructure is categorized into **Block Storage, File Storage, and Object Storage**, each serving different needs. Additionally, **DAS, NAS, and SAN** are different storage architectures used in enterprises.

## **1. Types of Storage**

### **A. Block Storage**

✅ **Stores data in fixed-sized blocks**  
✅ Each block has a unique address but no metadata  
✅ Used in **high-performance applications** like databases and virtual machines

🔹 **Example Storage Solutions:**

* Amazon Elastic Block Store (EBS)
* SAN (Storage Area Network)
* VMware vSAN

🔹 **Use Cases:**  
✅ Databases (MySQL, Oracle)  
✅ Virtual Machines (VMware, Hyper-V)  
✅ High-speed transactional applications

### **B. File Storage**

✅ Data is stored in a **hierarchical folder structure**  
✅ Users access files using **protocols like NFS, SMB, CIFS**  
✅ Easy to use but **scalability can be a challenge**

🔹 **Example Storage Solutions:**

* Windows File Server
* Network Attached Storage (NAS)
* AWS Elastic File System (EFS)

🔹 **Use Cases:**  
✅ File sharing across teams  
✅ Web servers and content storage  
✅ Backup and archiving

### **C. Object Storage**

✅ Stores data as **objects with metadata and unique identifiers**  
✅ Scales infinitely, making it ideal for cloud storage  
✅ Uses HTTP-based APIs for access (REST, S3)

🔹 **Example Storage Solutions:**

* Amazon S3
* Google Cloud Storage
* OpenStack Swift

🔹 **Use Cases:**  
✅ Cloud storage (Dropbox, Google Drive)  
✅ Backup and archival  
✅ Storing media files (videos, images, logs)

## **2. Storage Architectures: DAS, NAS, and SAN**

| **Storage Type** | **DAS (Direct Attached Storage)** | **NAS (Network Attached Storage)** | **SAN (Storage Area Network)** |
| --- | --- | --- | --- |
| **Definition** | Directly connected to a single server | A dedicated storage device connected via a network | A high-speed network of storage devices |
| **Connectivity** | SATA, SCSI, NVMe | Ethernet (NFS, SMB, CIFS) | Fiber Channel (FC), iSCSI |
| **Performance** | Fast but limited to one server | Slower than SAN, good for file sharing | High-speed and scalable |
| **Scalability** | Limited | Moderate | Highly scalable |
| **Use Case** | Personal computers, single-server applications | File sharing in offices, small businesses | Enterprise storage, databases, virtualization |

🔹 **Example Solutions:**  
✅ **DAS** – Internal HDDs, SSDs, USB drives  
✅ **NAS** – Synology, QNAP, NetApp NAS  
✅ **SAN** – Dell EMC PowerMax, NetApp ONTAP, HPE 3PAR

**7-Describe storage allocation and provisioning. Storage Allocation**

## > **Storage Allocation and Provisioning**

### **1. Storage Allocation**

**Storage allocation** refers to the process of **assigning storage resources** (disk space, volumes, or logical units) to a server, virtual machine, or application. It ensures that each system gets the required storage capacity for optimal performance.

🔹 **Key Aspects of Storage Allocation:**  
✅ **Static Allocation** – Predefined storage space is assigned manually  
✅ **Dynamic Allocation** – Storage is allocated on demand based on usage  
✅ **Quota Management** – Limits are set to prevent excessive storage use

🔹 **Example:**

* Assigning **100GB of storage** to a virtual machine in a VMware environment
* Allocating **500GB** for a database server in a **SAN** environment

### **2. Storage Provisioning**

**Storage provisioning** is the process of preparing and configuring storage resources before allocation. It involves **creating, managing, and distributing storage capacity** based on organizational needs.

🔹 **Types of Storage Provisioning:**

### **A. Thick Provisioning (Pre-allocated Storage)**

✅ Allocates the full amount of storage **immediately**  
✅ Prevents storage oversubscription  
✅ Ensures **predictable performance**

🔹 **Example:** A **100GB** volume is fully allocated upfront, whether used or not.

### **B. Thin Provisioning (On-Demand Allocation)**

✅ Allocates storage **dynamically as needed**  
✅ Optimizes storage usage, reducing waste  
✅ Risk of **over-provisioning** if actual storage exceeds physical capacity

🔹 **Example:** A 500GB storage pool is created, but only **used space is allocated**.

### **C. Dynamic Provisioning (Automated Scaling)**

✅ Uses automation to adjust storage based on real-time demand  
✅ Ideal for **cloud and virtualized environments**  
✅ Reduces manual intervention and improves efficiency

🔹 **Example:** AWS, Azure, and VMware **auto-expand storage** when needed.

## **3. Storage Allocation vs. Storage Provisioning**

| **Feature** | **Storage Allocation** | **Storage Provisioning** |
| --- | --- | --- |
| **Definition** | Assigning storage to users, applications, or servers | Creating and configuring storage before allocation |
| **Types** | Static, Dynamic | Thick, Thin, Dynamic |
| **Purpose** | Manages who gets how much storage | Ensures efficient storage availability |
| **Example** | Assigning 200GB to a VM | Creating a 1TB storage pool with thin provisioning |