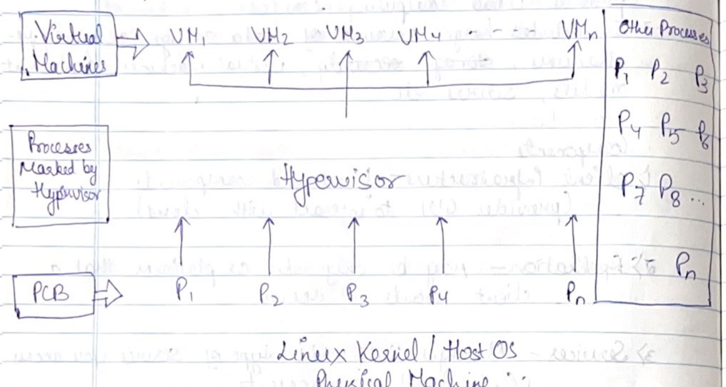
(*) Compute Node Architecture



In cloud computing, a computer node architecture refers to the design and configuration of individual computing units or nodes within a cloud infrastructure. These nodes are the fundamental building block of cloud environments, and their architecture plays a crucial role in defining the performance, scalability, and reliability of cloud services.

1) Basic Node Components

- Hardware: Each computer node is a physical or virtual machine within its own set of hardware resources, including CPU, RAM, storage and network interfaces.

- Operating System: Nodes typically run on a specific operating system to manage hardware resources and execute software applications.

- Hypervisor or Container Runtime: In virtualized cloud environments, a hypervisor or a container runtime may be used to create and manage multiple isolated instances on a single physical server.

2) Node Types

- Compute Nodes: These nodes are primarily responsible for processing tasks and running applications. They have a significant amount of CPU and memory resources to handle computing workloads.

- Storage Nodes: These nodes focus on storing and managing data. They often have large-capacity storage devices and optimized for data-intensive operations.

- Networking Nodes: In some cloud architectures, specific nodes are dedicated to managing network traffic, routing, and load balancing. They ensure efficient data transmission within cloud infrastructure.

- Management Nodes: These nodes handle orchestration, monitoring, and management tasks for the entire cloud environment. They help coordinate the activities of other nodes, ensuring that resources are allocated efficiently.

3) Scalability and Elasticity: Cloud computing allows for easy scaling of nodes to accomodate changing workload. You can add or remove nodes dynamically based on demand, achieving elasticity and cost optimization.

4) High Availability: Cloud providers often implement redundancy and failover mechanisms to ensure high

availability. This involves deploying multiple nodes in different availability zones or regions to mitigate the impact of hardware failures or outages.

5) Security: Security measures are crucial in node architecture. Access controls, firewalls, and encryption are often implemented at the node level to protect data and resources.

6) Load Balancing: Load balancers distribute incoming traffic across multiple nodes to ensure even resource utilization and prevent overloading of individual nodes.

7) Data Replication and Backup: Storage nodes may employ data replication and backup strategies to ensure data durability, and recoverability, in case of failures.

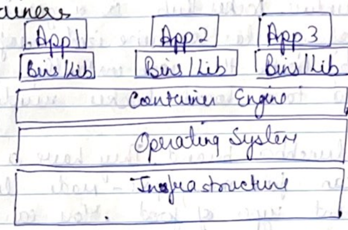
8) Resource Allocation and Scheduling: Resource management tools and schedulers are used to allocate computing resources efficiently among various nodes, optimizing performance and cost.

9) Monitoring and Management: Node health, performance, and utilization are continuously monitored, Management nodes and cloud management platforms provide visibility and control over the entire infrastructure.

It is a blueprint for designing, deploying and managing individual computing units within a cloud infrastructure.

(2) Types of Compute nodes
A compute node provides the storage, networking, memory, and processing resources that can be consumed by virtual machine instances.
Types of compute nodes are
1) Virtual Machines
2) Containers

Containers



Containers are much more lightweight than VMs because they share the host OS's kernel and resources. This results in faster startup times and lower overhead.

Containers are highly portable as they package the application its dependencies into a single unit. This makes it easier to move applications between environments such as from development to production.

Containers are well-suited for microservice architecture and can be easily scaled up or down based on demand.

Since they share resources with the host OS, containers are more resource efficient than VMs, especially while running multiple instances on the same host.

Container orchestration platforms like kubernetes provide advanced features for managing containerized applications, including automatic scaling, load balancing and rolling updates.

Popular container Providers

1) Docker

Docker is the most popular and widely used container runtime. Docker Hub is a giant public repository of popular containerized software application. Containers on Docker Hub can instantly downloaded and deployed to a local Docker runtime.

It is like lunchbox brand. They have a big store where you can find many pre-made lunchboxes with different types of food. You can pick one you like and it's ready to eat. Docker makes it easy to get these lunchboxes and use them on your computer.

2) RKT (Rocket)

It is a security-first focused container system. It does not allow insecure container functionality unless user explicitly enables insecure features. It aim to address the underlying cross containamination exploitive security issues that other container runtime systems suffer from.

It is a special type of lunchbox that is very focused on making sure the food inside is safe. It doesn't allow any unsafe food unless you really really want it. RKT aims to prevent any food from one lunchbox mixing with the food in another, which could be a problem with some other lunchboxes.

3) Linux Containers (LXC) (open source Linux container runtime system)

LXC is like a plain, reusable lunchbox. It's open for you to put your own food in it. Docker actually uses LXC behind the scenes to make its lunchboxes. LXC is like a basic, open-source lunchbox that anyone can use and customize.

4) CRI-O

It is an implementation of the Kubernetes Container Runtime Interface that allow the use of Open Containers Initiative (OCI) compatible runtimes. It is a lightweight alternative to using Docker as the runtime for Kubernetes.

Virtual Machine

- Virtual machine provide complete isolation from the underlying hardware and the host OS. Each VM runs its own guest OS making them independent environments.

Virtual machine can run a wide range of OS, including Windows, Linux, and others. They are suitable for running legacy applications that may require specific OS versions.

Virtual machine are allocated a predefined amount of CPU, memory, and storage, making it easier to manage resource allocation.

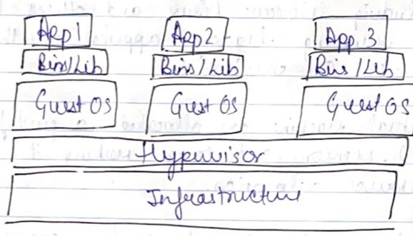
Virtual machine can be easily snapshotted and migrate to different hosts, allowing for backup, recovery, and mobility.

Popular Virtual Machine Providers

1) VirtualBox
Think of VirtualBox as a free tool that lets you create and run virtual computers inside your real computers. It's like having multiple computers in one and you can use it for different tasks.

2) VMware
It is a company that makes software for virtualization. It helps you manage and run several virtual computers on a single physical computer. It's popular for businesses and has user friendly interface.

3) QEMU (Quick Emulator)
It is a powerful tool that can ~~ukka~~ simulate almost any type of computer. However it doesn't have a fancy visual interface like the others it's more like a command line tool but it's really fast.



Virtual Machine	Containers
Applications running on VM can run different OS.	Applications running in a container share a single OS
VM virtualizes the computer system	Containers virtualize the OS only.
VM size is very large	Container is very light i.e few megabytes
VM takes minutes to run, due to large size	Containers take a few seconds to run
VM uses a lot of system memory	Containers require very less memory
VM is more secure	Containers are less secure
VM are useful when we require all of OS resources to run various applications	Containers are useful when we are required to maximise the running applications using minimal servers.

✻ Configuration of Compute node
(configuring a compute node for optimal performance especially in the context of NUMA (Non-Uniform Memory Access) setups, can significantly improve the efficiency of virtualized workloads, including NFV and HPC. Here are the steps and considerations for configuring a compute node for such deployments.

1) Evaluate NUMA Topology
Understand the NUMA topology of your host system. NUMA systems have multiple processor nodes, each with its own local memory. To optimize performance you need to work within this architecture

2) Allocate CPU cores
Decide how many CPU cores should be reserved for host processes and how many can be used by virtual machine instances. This should be based on observed performance to balance host and VM efficiency

3) Create an Environment file
Generate an environment file with settings for CPU pinning, emulator thread pinning, huge pages, core reservations, RAM allocations, and host process isolation. The specifics will depend on your cloud management platform.

4) CPU Pinning
Bind virtual CPUs to Physical CPUs. This minimizes CPU scheduling overhead, especially important for latency-sensitive workloads.

5) Emulator Thread Pinning
Pin emulator thread to specific physical CPUs to reduce contention and enhance performance.

6) Huge Pages
Configure memory allocation policies for both regular and huge page (for better memory performance in large workloads).

7) Reserve CPU Cores for Dedicated instances
Specify which CPU cores are exclusively reserved for specific VMs to ensure predictable performance.

8) Reserve CPU Cores for Shared instances
Designate CPU cores for shared VMs that don't require dedicated pinning but still need good performance

9) Allocate RAM for Host Processes
Define the amount of RAM reserved for host processes to ensure the host has adequate memory for its operations without competing with VMs

10) Isolate Host Processes
Prevent host processes from running a CPU cores designated for VMs. Isolation minimizes interference between host and VM workloads.

11) Deploy Configuration
Incorporate the environment file into your cloud deployment setup, along with other configuration files, and then deploy the system with these settings.