MOD I

NIST (5-4-4)

1) Essetial characteristics

- On demand self service - Requires No human interaction with each service provider.
- Rapid elasticity - elasicity provisioned & released, in some cases automatically, to scale rapid outward & inward commensurat
- Measured service - Resource usage can be monitored, controlled & reported
- Resource pooling - Resources are pooled to serve multiple consumers using multitenant model.
- Broad Network access - Available over the network & accessed through standard mechanisms that promote use by hetro tun or thick client platforms

2) Service Model

Software as a Service (SaaS) - The capability provided to the consumer is to use the providers applications running on a cloud infrastructure.

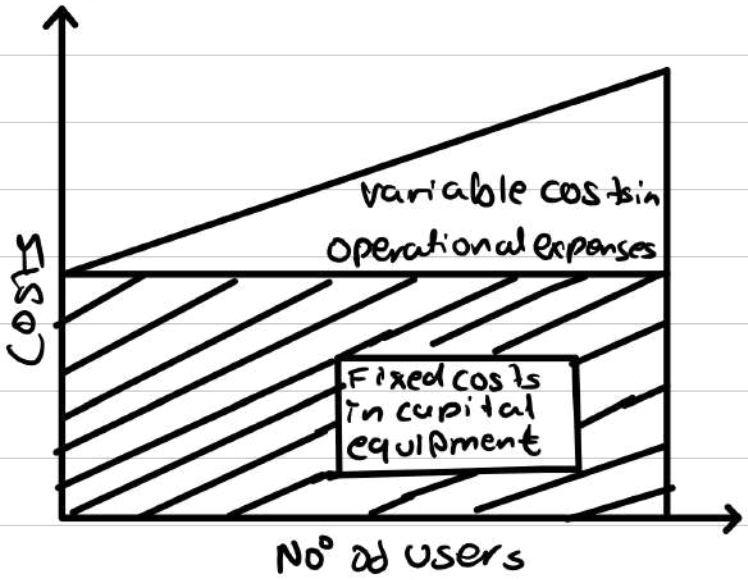
- Platform as a Service (PaaS) - The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired application created using programming languages, lib, services & tools supported by the provider.

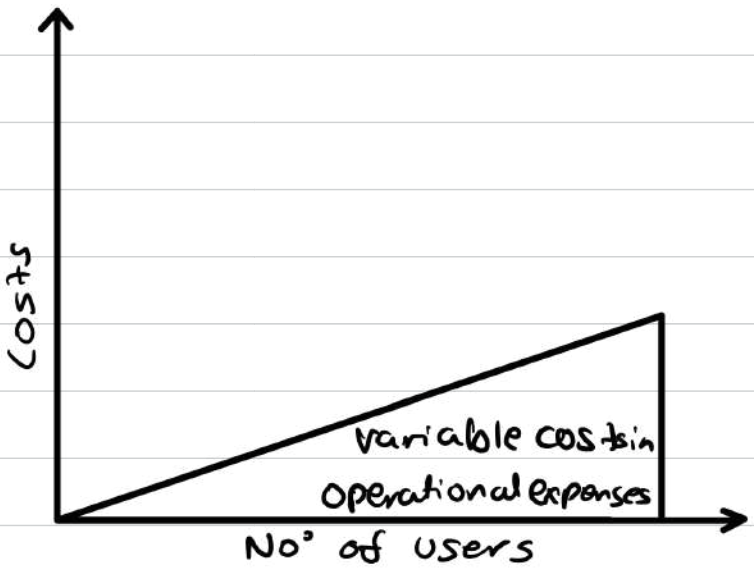
- Infrastructure as a Service (IaaS) - The capability provided to the consumer is to provision processing, storage, networks & other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include OS & applications.

3) Deployment Models

- Private cloud
- Community cloud
- Public cloud
- Hybrid cloud.

Cost model





a) Traditional

- User must acquire their own computer & peripheral equipment as capital expenses.
- In addition, they have to face operational expenditures

b) Cloud Computing

- CC applies a pay per use business model, in which user jobs are outsourced to data centers
- To use, one has no up-front hardware cost, only variable costs are experienced.

Design Objective

Shifting computing from desktops to data centers

- Computer processing, storage, and software delivery is shifted away from desktops & local servers & toward data centers over the internet.

Service provisioning & cloud economics

- Providers supply cloud services by signing SLAs with consumers and end users
- The services must be efficient in terms of computing, storage, and power consumption.
- Pricing is based on a pay as you go policy

Cloud Model	Public Cloud	Private Cloud	Community Cloud	Hybrid Cloud
Ease of setup	Very easy to setup	Very hard to setup as your team creates the system	Easy to setup because of community practices	Complex to setup due to interconnected systems
Ease of use	Very easy to use	Complex and requires on in-house team	Relatively easy to use as members help solve problems and protocols	Difficult to use if the system was not setup properly
Data control	Low-provider has complete control	Very high- ownership is with your system	High - if members collaborate	Very high- with right setup
Reliability	Prone to outrages and failures	High with right team	Depends on the community policy	High- with the setup
Scalability	Low, most providers offer limited resources and predefined setups	Very high- there are no other tenants	Fixed capacity tends to limit scalability	High - with right setup
Cost	Inexpensive	Very expensive	Members share cost	Cost-effective
Demand for in-house hardware	No	In-house hardware is preferable	No	In-house hardware is not a must

Scalability in Performance

- The cloud platforms & software & infrastructure services must be able to scale in performance as the no° of users increase.

Data Privacy Protection

High quality of Cloud Services

New Standards & interfaces

- This refers to solving the data lock-in problem associated with data centers or cloud providers

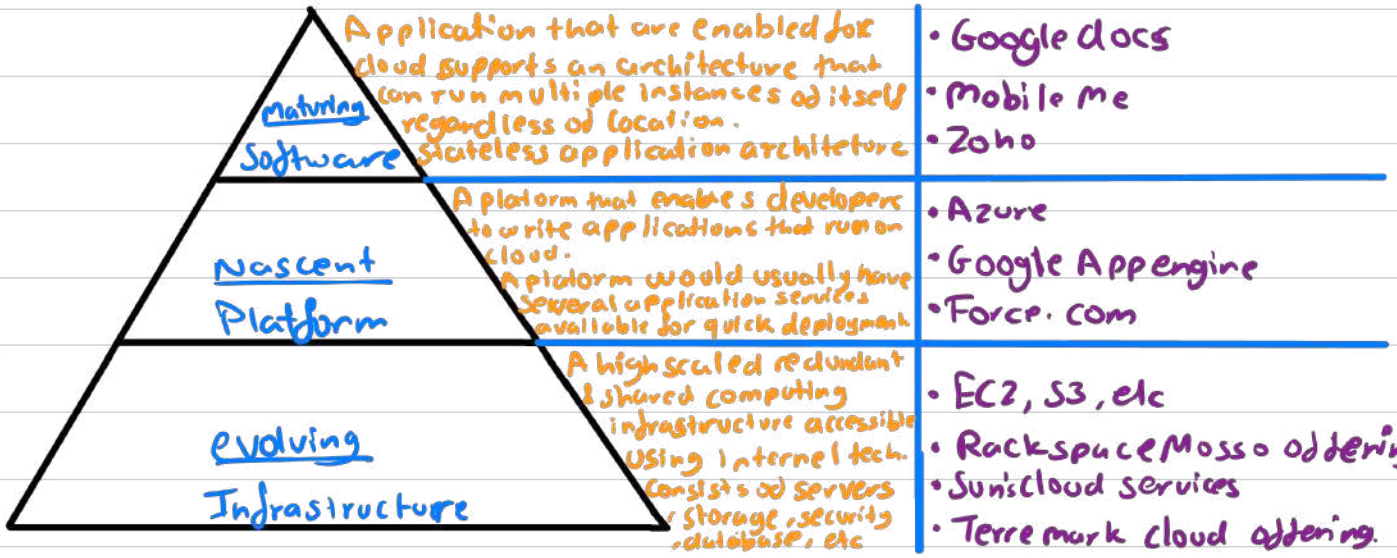
Advantage

- lower computer Costs - No need of high powered & high priced computer
- Improved Performance
- Reduced Software Costs
- Instant Software updates
- Improved document format Compatibility.
- Unlimited Storage capacity
- Increased data reliability
- Universal Info. access
- Latest version availability
- Easier Group Colab
- Device Independence

Disadvantage

- Requires a constant Internet connection
- Does not work well with Low-speed connections.
- Features might be limited
- Can be Slow
- Stored data might not be secured.
- Stored data can be lost
- HPC systems
- May not be possible to run applications between cloud based systems

Service Models (Xaas)



Application where (is used)

1) SaaS

- Where there is significant interplay between org. & outside world
- Need for web or mobile access.
- Only used for a short term.
- Demand spikes significantly

2) PaaS

- Multiple developers will be working on a development project.
- Wish to automate testing & deployment services.
- Where agile software development is used.

3) IaaS

- Where demands are volatile
- With new org without the capital to invest in hardware.
- Org is growing rapidly & scaling hardware would be problematic
- Pressure on the org. to limit expenditure

Application where (is not used)

1) SaaS

- Extremly fast processing of real time data is needed.
- Where legislation or other regulation does not permit data being hosted externally
- Where an existing on-premise solⁿ exists

2) PaaS

- Where proprietary languages or approaches would impact on the development process
- Where proprietary languages would hinder later moves to another provider
- Where application performance requires customization of the underlying hardware and software.

3) IaaS

- Where regulatory compliance makes the offshoring or outsourcing of data storage and processing is difficult.
- Where the highest levels of performance are required, and on-premise or dedicated hosted infrastructure has the capacity

Cloud bursting for load balancing

- It is a configuration with setup between public & private cloud to deal with peaks in IT demand.
- If an org. using private cloud reaches 100% of it's capacity the overflow traffic is directed to a public cloud so there is no interuption of services.

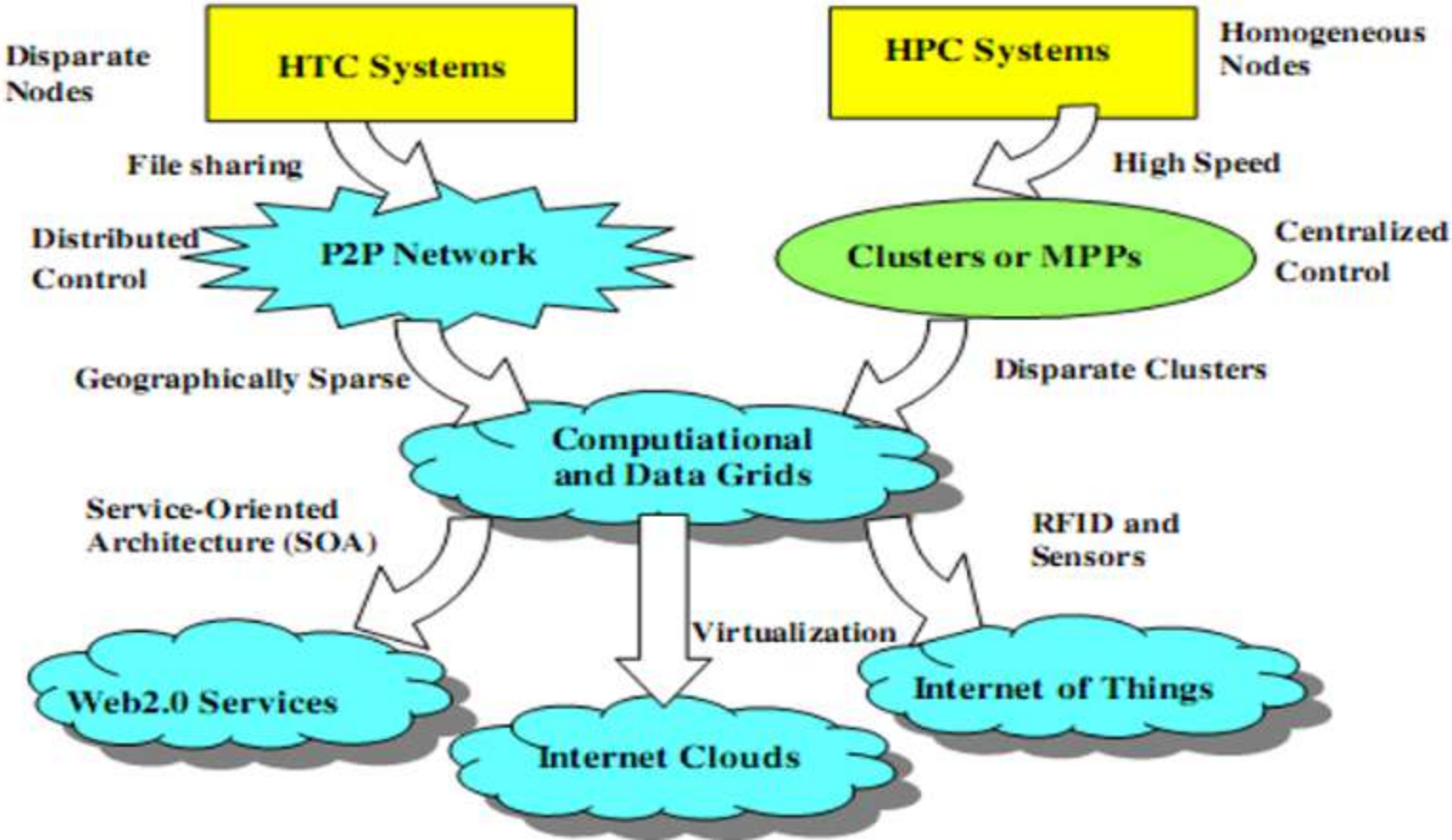
Feature comparison of IaaS providers

Provider	Geographic distribution of data centers	User interfaces and APIs	Hardware capacity	Guest operating systems	Smallest billing unit
Amazon E2C	US Europe	CLI, WS, Portal	CPU: 1_20 EC2 compute units Memory: 1.7-15 GB Storage: 160-1690 GB, 1 GB – 1 TB (per ESB units)	Linux Windows	Hour
Flexiscale	UK	Web console	CPU: 1-4 Memory: 0.5-16 GB Storage: 20-270 GB	Linux, Windows	Hour
GoGrid		REST, Java, PHP, Python, Ruby	CPU: 1-6 Memory: 0.5-8 GB Storage: 30-480 GB	Linux, Windows	Hour
Joyent	US		CPU: 1/16-8 Memory: 0.25-32.5 GB Storage: 5-100GB	OpenSolaris	Month
RackSpace	US	Portal, REST, Python, PHP, Java, .NET	CPU: Quad-core Memory: 0.25-16 GB Storage: 10-620 GB	Linux	Hour

Feature comparison of PaaS providers

Provider	Target to Use	Programming language, Frameworks	Programming Models	Persistence options
Aneka	.NET enterprise applications, Web applications	.NET	Threads, Task, MapReduce	Flat files, RDBMS
AppEngine	Web applications	Python, Java	Request-based Web programming	BigTable
Force.com	Enterprise applications	Apex	Workflow, Request-based Web programming, Excel-like formula language	Own object database
Azure	Enterprise applications, Web applications	.NET	Unrestricted	Table/BLOB/queue storage, SQL Services
Heroku	Web applications	Ruby on Rails	Request-based Web programing	PostgreSQL, Amazon RDS
Amazon Elastic MapReduce	Data processing	Hive and Pirg, Cascading, Java, Ruby, Perl, Python, PHP, C++	MapReduce	Amazon S3

Evolutionary trend



HPC vs HTC

- Needs large amt of computing power for short periods of time	- Needs large amt of computing power for longer periods of time
- Measured in terms of FLOPS	- Measured in operations per month or per year.
- Supercomputers or MPPs (Massively parallel Processors) are gradually replaced by clusters of co-operative computers out of desire to share computing resources	- P2P are formed for distributed file sharing & content delivery. It's built over many client machines & are globally distributed in nature.

Compute grid vs data grid

- Allows you to take a computation, optionally split it into multiple parts, and execute them on different grid nodes in parallel	- Allows you to distribute your data across the grid.
- The computation will perform faster as it can now use resources from all grid nodes in parallel	- The main goal of data grid is to provide as much data as possible from memory on every grid node to ensure data coherency
- Features: 1) load balancing, 2) fail-over 3) grid events, 4) Node metrics	- Features: 1) data replication, 2) data backups, 3) data partitioning / affinity

How is Grid Computing Used?

All computing resources do not have to work on the same specific task, but can work on sub-tasks that collectively make up the end goal.

It especially useful when different subject matter experts need to collaborate on a project but do not necessarily have the means to immediately share data and computing resources in a single site

3 new Computing Paradigms

- The maturity of radio frequency identification (RFID), GPS & sensor Tech. has triggered the development of the IOT.
- With SOA, web services became available
- Advances in Virtualization led to the growth of internet clouds.

Building Cloud computing environments

Application Development

- One class of applications is that of Web application
- Their performance is mostly influenced by the workload generated by varying user demands.
- Another class of application is that of resource intensive application
- These can be either data intensive or compute intensive applications. In both cases, a lot is required.
- But large amt of resources are not needed constantly or for a long duration.
- These are not interactive & they are mostly characterized by batch processing.

Infrastructure & system development

- Providing methods for renting computer power, storage & networking
- offering runtime enviroments designed for scalability & dynamic sizing
- Providing application services that mimic the behavior of desktop applications but that are completely hosted & managed on the provider side,
- All these leverage service orientated architecture.
- Accessed via REST web services.

- The core Tech :- 1) Distributed computing, 2) Virtualization
3) Service orientation, 4) Web2.0 / web services

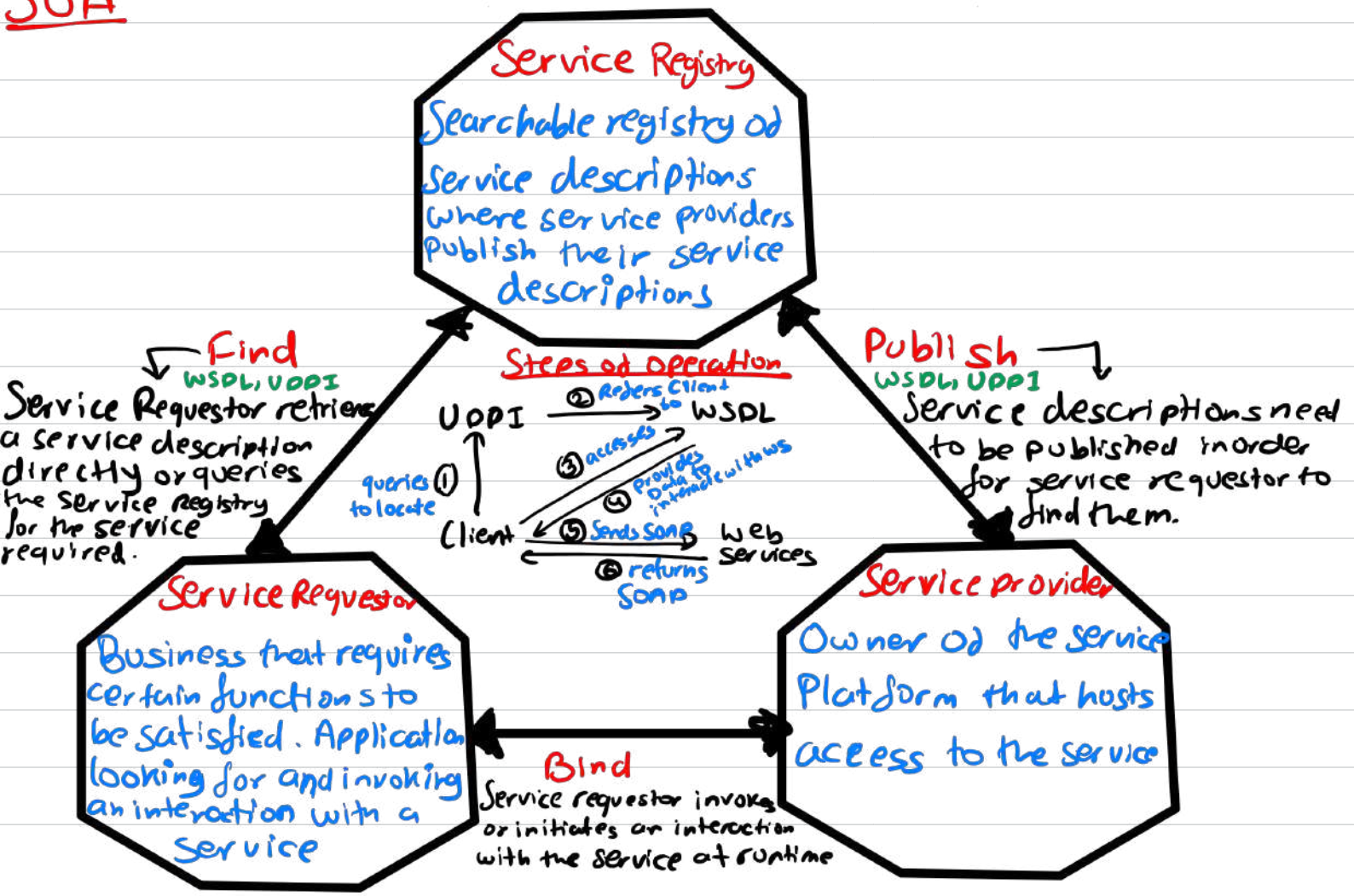
Web Services

- It is any piece of software that makes itself available over the internet & uses a standardized XML messaging system.
- As all comm. is in XML, web services are not tied to any one OS or programming language.

- Components

- XML - eXtensible markup Language
- SOAP - Simple object access Protocol, (Astd way for comm.)
- WSDL - web services Description Language, (meta Language)
- UDDI - Universal Description, Discovery & Integration specification (A mechanism to register & locate WS based appli)

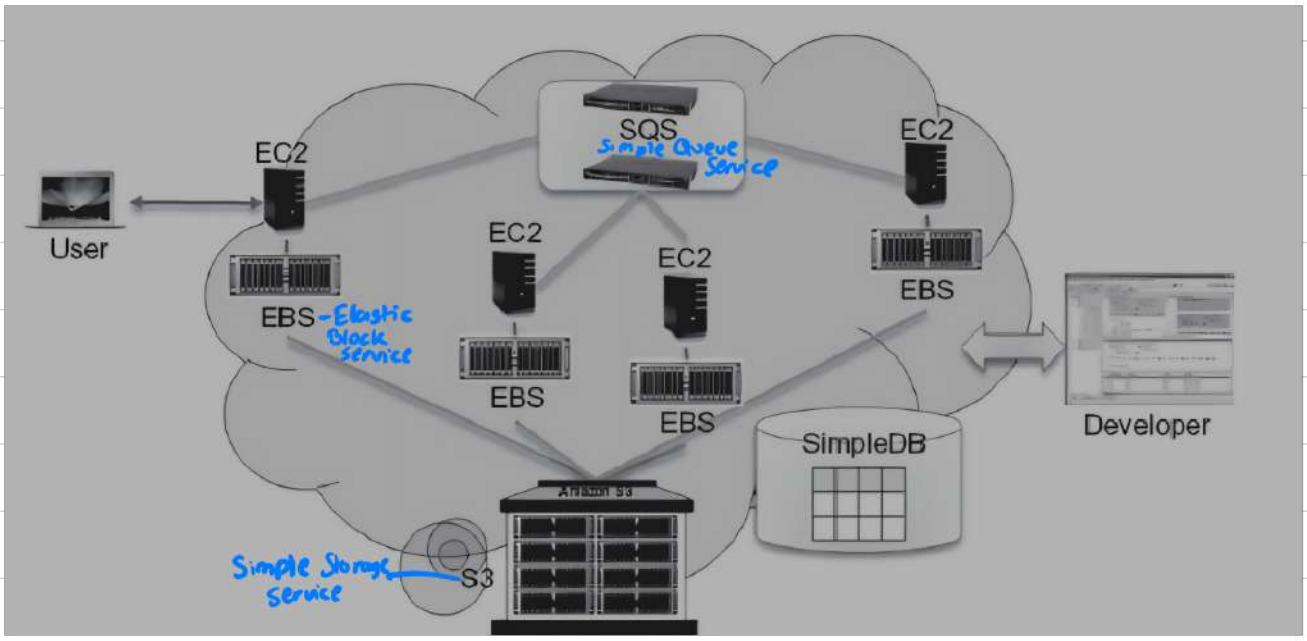
SOA



	Web Service	Website
1.	A web service doesn't have a user interface	A website has a user interface or GUI
2.	Web services are meant for other applications to be interacted with over internet	Websites are meant for use by humans
3.	Web services are platform independent as they use open protocols	Websites are cross platform as they require tweaking to operate on different browsers, operating systems etc.
4.	Web services are accessed by HTTP methods - GET, POST, PUT, DELETE etc	Websites are accessed by using their GUI components - buttons, text boxes, forms etc.
5.	E.g. Google maps API is a web service that can be used by websites to display Maps by passing coordinates to it	E.g. ArtOfTesting.com is website that has collection of related web pages containing tutorials

AWS

- It is a cloud computing platform that offers a variety of services, including compute, storage, and databases.
- IAAS.
- Architecture



EC2 - provides the virtualized platforms to the host VMs where the cloud application can run.
S3 - Provides OO storage service for users
EBS - provides the block storage interface which can be used to support traditional applicatn
SQS - It's job is to ensure a reliable message service between two processes. The message can be kept reliably even when the receiver processes are not running.
Users - can access their object through SOAP with either browsers or other client programs which support the SOAP std.

- EC2 also provides the capability to save a specific running instance as an image
- The templates are stored in S3 that delivers

Persistent storage on demand which is organized into buckets
- these are containers of objects that are stored in binary form & can be enriched with attributes.
- Users can store objects of any size, from simple files to entire disk images, and have them accessible from everywhere

Google App Engine (GAE)

- PaaS
- Provides both a secure execution environment & a collection of services that simplify the development of scalable and high-performance Web applications

- The Datastore offers OO, distributed, structured data storage services based on Big Table.
- The application runtime environment offers a platform for scalable web programming & execution in python & Java.
- Developers can build & test applications on their own machines using SDK.
- Once complete, developers can easily migrate their application.

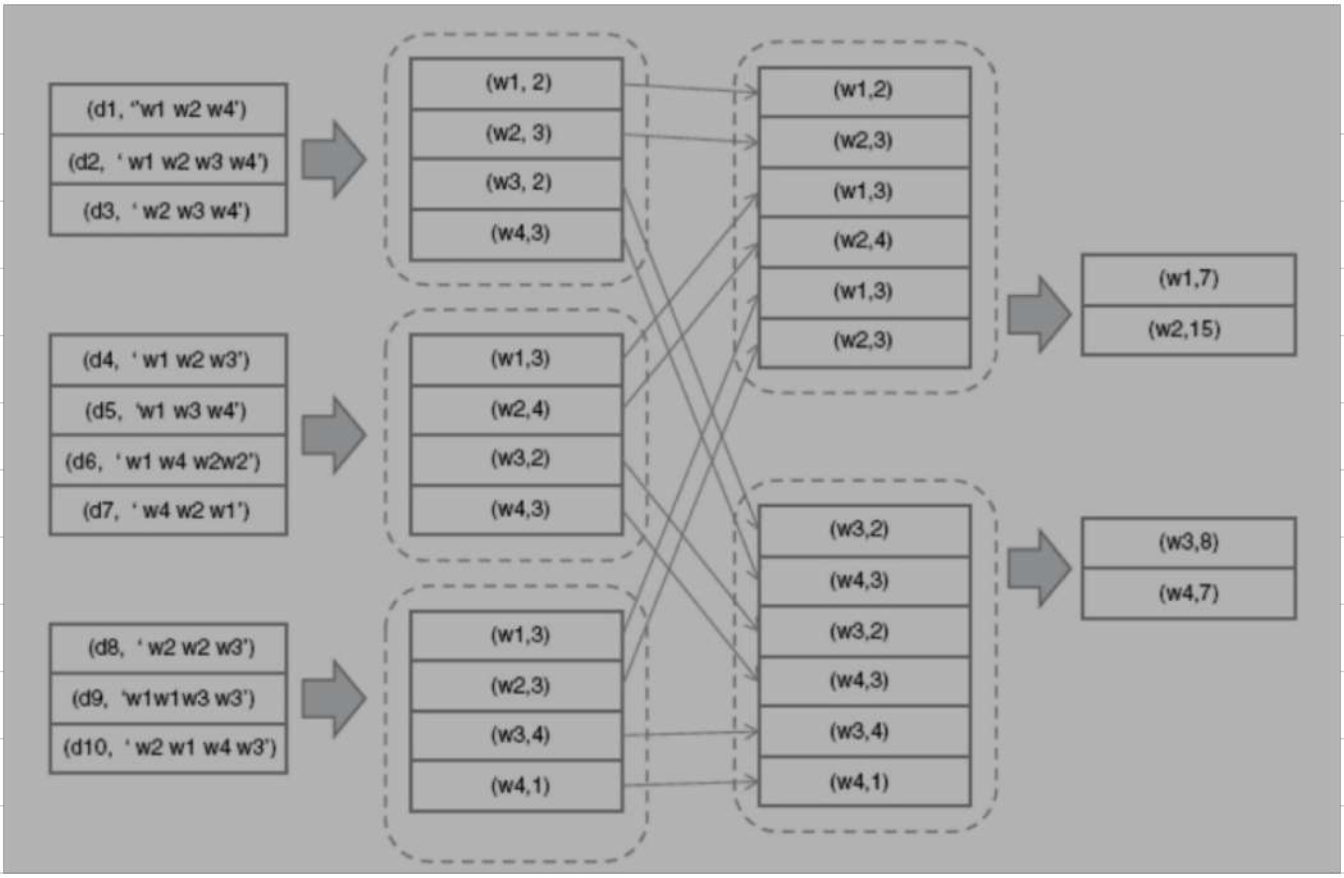
Microsoft Azure

- It is a platform for developing applications in the cloud
- It provides a scalable runtime environment for Web applications & distributed applications in general

- These are organized around the concept of roles, which identify a distribution unit for applications & embody the application's logic.
- Roles are:
- Web role: Is designed to host a web application
- Worker role: Is a more generic container of application & can be used to perform workload processing.
- Virtual machine role: provides a virtual environment in which the computing stack can be fully customized, including the OS.
- Support for storage, networking, caching, and others also.

Map Reducer (Google)

- Used in web-scale search & cloud computing applications
- Web programming model for scalable data processing on larger clusters over large data sets.
- Two phases:
- Map Phase
  - Each mapper reads approx 1/M of the inputs from the global file system
  - Map operation consists of transforming one set of key-value pairs to another
    $(K_1, V_1) \rightarrow [(K_2, V_2)]$
- Reduce Phase
  - The reducers make remote procedure call requests to the mappers to fetch the files.
  - Each reducer group results of the map step using the same key & performs a function f on the list of values: $(K_2, [V_2]) \rightarrow (K_2, f([V_2]))$.



Hadoop Library (Yahoo)

- Open Source implementation
- Provides the runtime environment
- A software platform to write & run applications over the vast amts of distributed data.

Feature	Force.com	Salesforce.com	Microsoft Aneka
Definition	Force.com is a **Platform as a Service (PaaS)** that allows developers to build and deploy applications within the **Salesforce ecosystem**.	Salesforce.com is a **Software as a Service (SaaS)** solution offering **Customer Relationship Management (CRM)** and other cloud-based business applications.	Microsoft Aneka is a **Platform as a Service (PaaS)** designed for **parallel computing and distributed cloud computing** applications.
Purpose	Used for developing **custom business applications** that integrate with Salesforce services.	Provides businesses with **ready-to-use CRM, marketing automation, and customer service solutions**.	Enables **distributed computing** by allowing users to execute parallel workloads across different cloud environments.
Primary Users	Developers building **enterprise applications** within Salesforce.	Businesses and organizations needing **CRM and sales automation** tools.	Researchers, engineers, and businesses that require **high-performance computing (HPC)**.

Hosting & Deployment	Hosted on **Salesforce's cloud infrastructure** and runs within the Salesforce ecosystem.	Runs on **Salesforce's cloud**, accessible via web browsers and mobile apps.	Can be deployed on **public, private, and hybrid cloud environments**, including AWS and Azure.
Customization	Developers can **create, modify, and integrate apps** tailored to business needs.	Businesses can **customize CRM workflows, dashboards, and reports**.	Users can customize **how computing tasks are distributed** across cloud resources.
Security	Provides **enterprise-level security**, including **role-based access control and data encryption**.	Follows **strict security and compliance standards** to protect customer data.	Ensures **secure computing environments** for distributed workloads.
Integration	Integrates seamlessly with **Salesforce products and third-party APIs**.	Supports integrations with **ERP systems, marketing tools, and third-party apps**.	Works with **various cloud providers**, including AWS, Azure, and private clouds.

Advantages	- Fast **app development** with pre-built Salesforce tools. - Secure and **easily scalable**. - Fully **integrated with Salesforce services**.	- Leading **CRM solution** with AI-powered insights. - Easy-to-use and **highly customizable**. - Large **ecosystem of apps** in AppExchange.	- Supports **hybrid cloud deployment**. - Efficient for **data-intensive computing**. - Works across **multiple cloud providers**.
Limitations	- Tied to **Salesforce ecosystem**. - Requires **Apex knowledge** for advanced development.	- Expensive for **small businesses**. - **Limited offline functionality**.	- **Less widely adopted** compared to other cloud platforms. - Requires **knowledge of parallel computing**.

MOD II

- The software methods used to allow multiple virtual computing resources to run on single hardware platform. → Virtualization

What is VM?

- A software virtualization of a computer hardware that executes program like a physical computer and can be interacted with like a physical COmuter
- Allows to run Multiple OS as VMs on a single computer.
- Main Files
  - Configuration
  - Virtual disk
  - NVRAM settings
  - Log
- The purpose of VM is to enchance resource sharing by many users & improve computer performance in terms of resource utilization & application flexibility

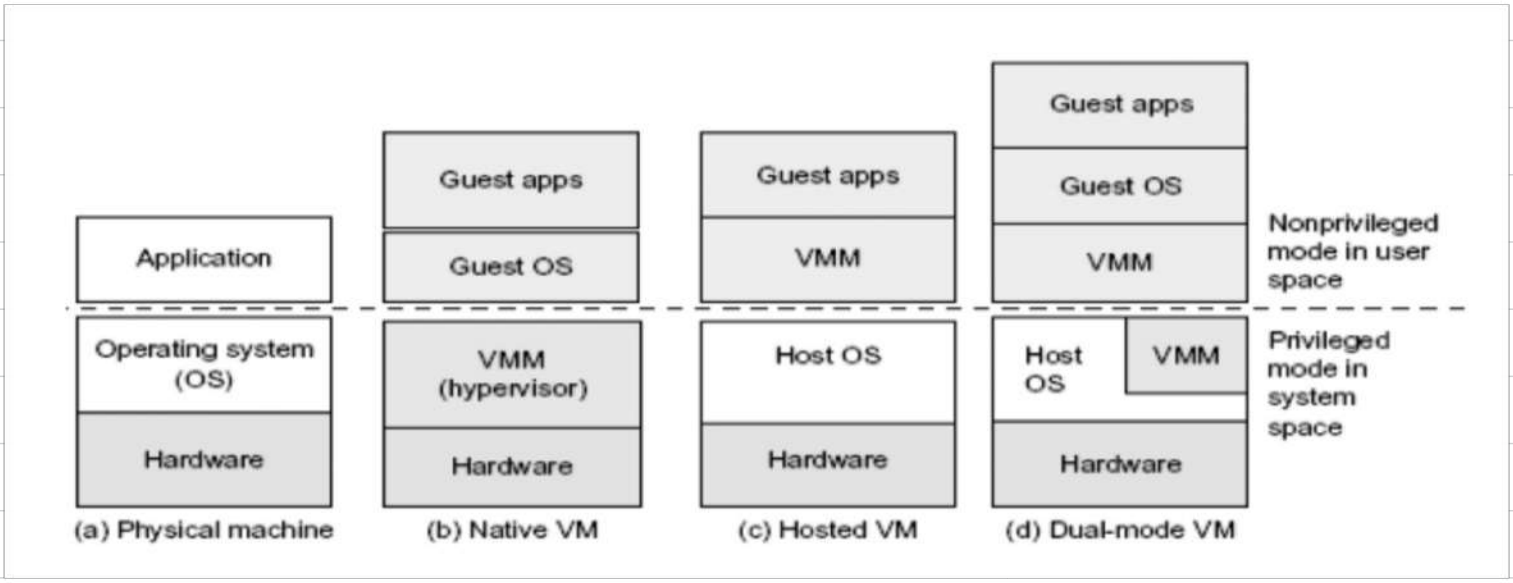
Difference between Cloud Computing and Traditional Computing

Aspect	Cloud Computing	Traditional Computing
Definition	Cloud Computing refers to delivery of different services such as data and programs through internet on different servers.	Traditional Computing refers to delivery of different services on local server.
Infrastructure Location	Cloud Computing takes place on third-party servers that is hosted by third-party hosting companies.	Traditional Computing takes place on physical hard drives and website servers.
Data Accessibility	Cloud Computing is ability to access data anywhere at any time by user.	User can access data only on system in which data is stored.
Cost Effectiveness	Cloud Computing is more cost effective as compared to tradition computing as operation and maintenance of server is shared among several parties that in turn reduce cost of public services.	Traditional Computing is less cost effective as compared to cloud computing because one has to buy expensive equipment's to operate and maintain server.
User-Friendliness	Cloud Computing is more user-friendly as compared to traditional computing because user can have access to data anytime anywhere using internet.	Traditional Computing is less user-friendly as compared to cloud computing because data cannot be accessed anywhere and if user has to access data in another system, then he need to save it in external storage medium.
Internet Dependency	Cloud Computing requires fast, reliable and stable internet connection to access information anywhere at any time.	Traditional Computing does not require any internet connection to access data or information.
Storage and Computing Power	Cloud Computing provides more storage space and servers as well as more computing power so that applications and software run must faster and effectively.	Traditional Computing provides less storage as compared to cloud computing.

Benifits of Virtualization

- Scalable
- Cost effective
- Better security
- Better performance
- Flexible

VM architecture

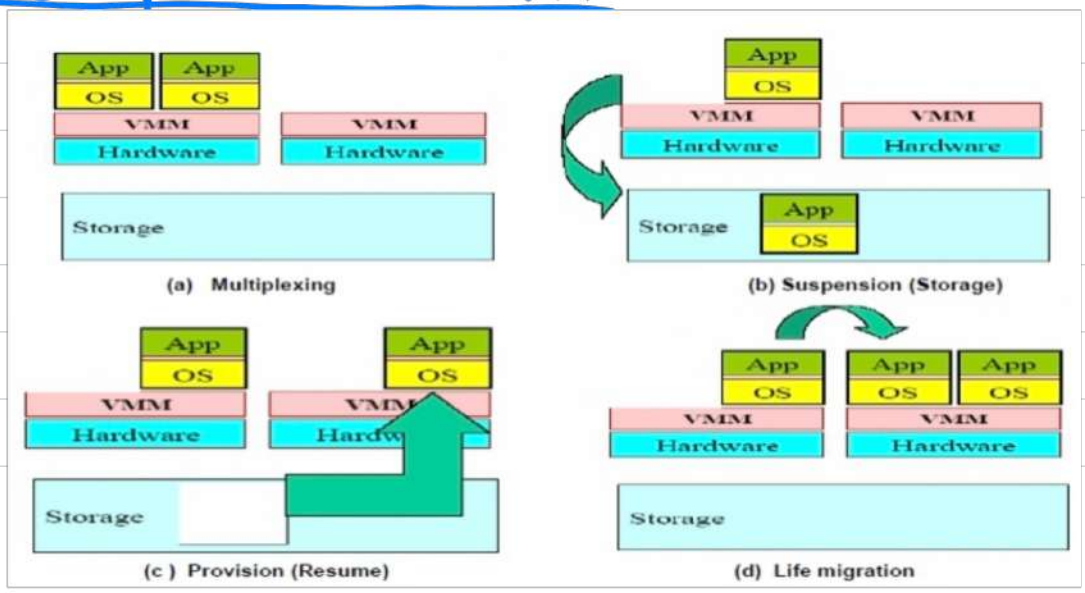


b) The hypervisor handles the bare hardware directly

c) VMM runs in non priviledge mode. The host OS need not to be modified

d) One part of the VMM runs at the user level & another part runs at the supervisor level

Primitive Operations in VM

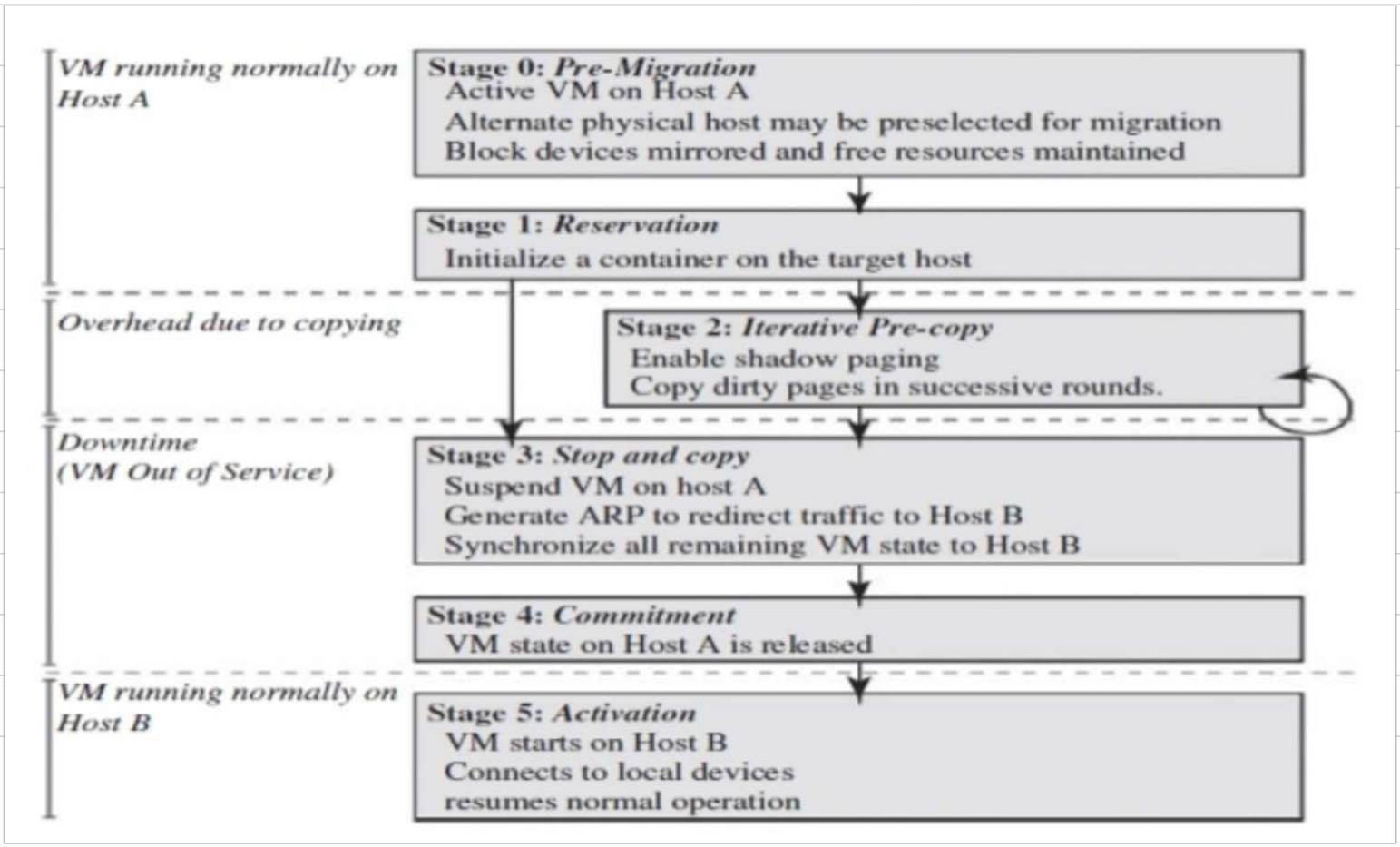


Provisioning VM

1) You need to select a server from a pool of available servers along with appropriate OS template you need to provision the VM
2) You need to load the, appropriate software (OS, device drivers, middleware & need application)
3) You need to customize and configure the M/C to configure an associated n/w & storage resources
4) Finally, the VM is ready to start with it's newly loaded software

Live Migration (Hot or Real time)

- It is the process of moving a running VM from one host to another without interrupting the VM access
- When it is properly carried out this process takes place without any noticeable effect from the end users point of view
- Also can be used for load balancing



Stage 0 - Pre Migration

- An active VM exists on host A
- Alternate physical host may be selected for migration

Stage 1 - Reservation

- A request is issued to migrate an OS from host A to host B
- Precondition is that the necessary resources exist on B & on a VM container of that size

Stage 2 - Iterative Precopy

- During the 1st iteration all memory pages are transferred from A to B
- Subsequent iteration copy only those memory pages dirtied (updated) during the previous transfer phase

Stage 3 - Stop & Copy

- Running OS instance at A is suspended & it's network traffic redirected to B
- CPU state & any remaining inconsistent memory pages are then transferred.
- At the end there is a consistent suspended copy of the VM at both A & B, the copy of A is considered primary & resumed in case of failure.

Stage 4-Commitment

- Host B indicates to A that it has successfully recieved a consistent OS image
- Host A acknowledges this message as commitment of the migration.
- Host A is discarded & Host B is now the primary host.

Stage 5 - Activation

- The migrated VM on B is activated.
- Post migration code runs to reattach the device drivers to the new m/c & advertise moved IP address
- Atleast one host has a consistent VM image at all times during migration

Regular / cold migration

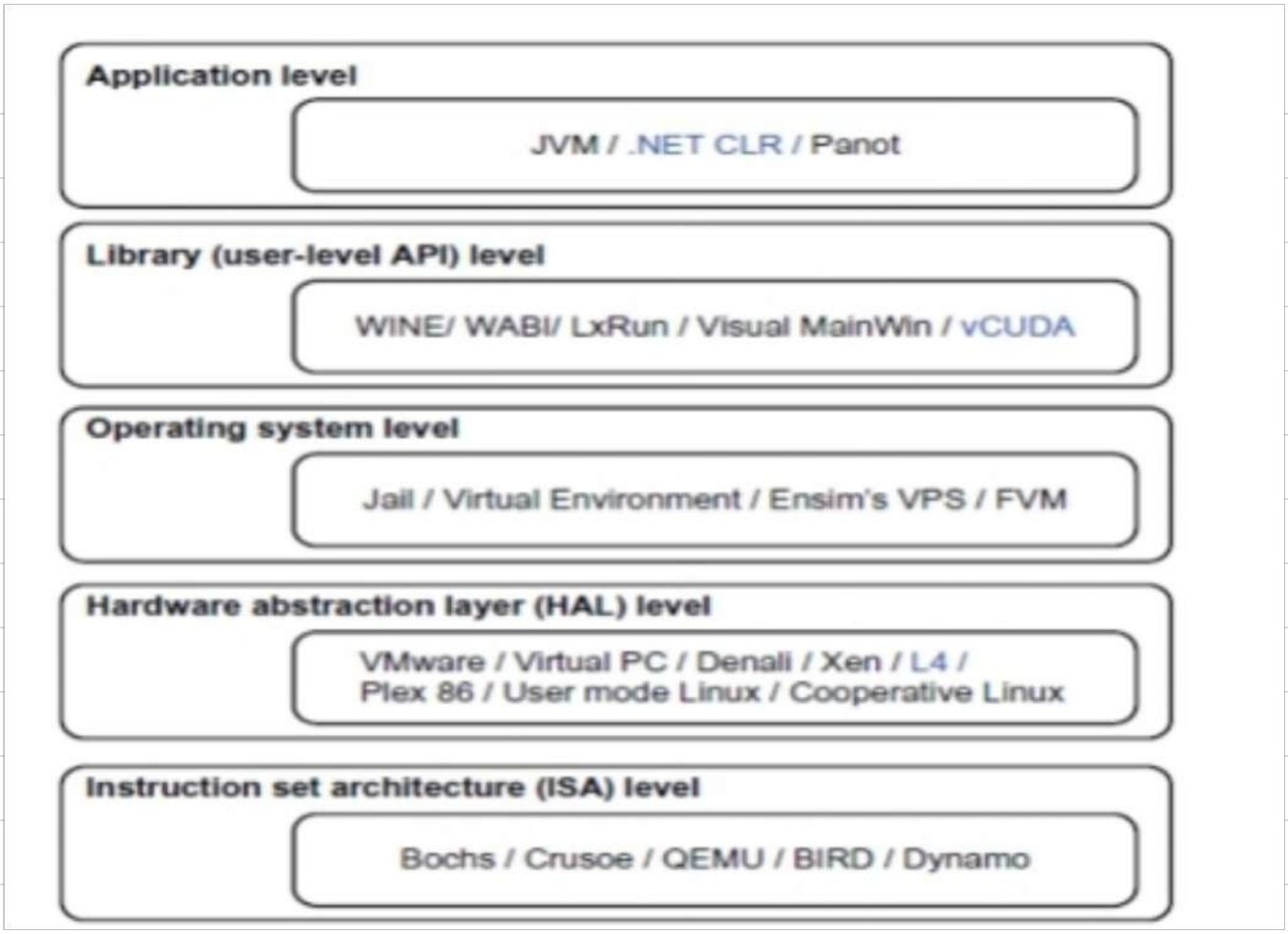
- It is the process of moving a VM from one host to another when powered off.
- You have the option of moving the associated disks from one data store to another
- The VM's are not required to be on a shared storage
- Process
  1) The configuration file, including the NVRAM file, log files, as well as the disks of the VM are moved for src host to destination Host associated storage area.
  2) The VM is registered with the New host.
  3) After the migration is completed, the old version is deleted.
- 2 main difference b/w hot & cold migration
  1) Hot needs shared storage for VM in the server's pool, but cold does not
  2) In Hot for a VM between 2 host's there would be certain CPU compatibility checks to be applied.

Application of Migration

- Load balancing
- Maintenance
- Recovery from host failure.

Level of Virtualization

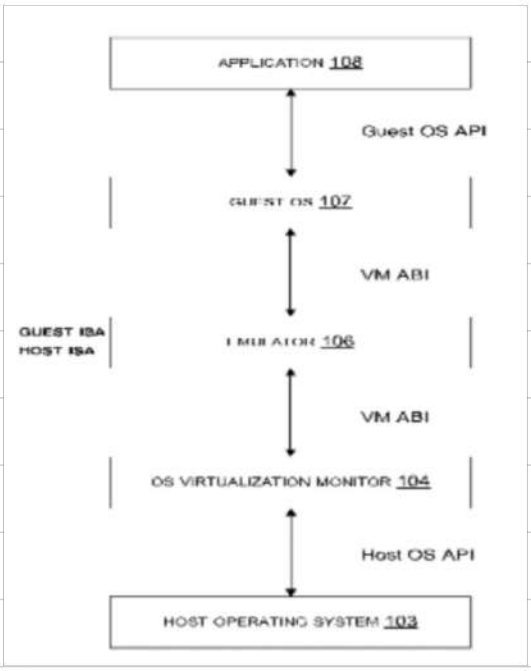
- The main function of the software layer for virtualization is to virtualize the physical hardware of a host machine into virtual resources to be used by the VMs, exclusively.
- The virtualization software creates the abstraction of VMs by interposing a virtualization layer at various levels of a computer system.



ISA Level

- The instruction set provides commands to the processor to tell it what to do
- Consists of : 1) addressing mode, 2) Operations, 3) native data types, 4) registers, 5) memory architecture , 6) interrupt & exception handling , 7) external I/O.
- Ex: x86 instruction set.
- Different computer processors can use almost the same instruction set while still having very different internal design.
- Virtulization is done by transforming physical architecture of the system's Instruction set completely into software.
- The guest system issue instruction for the emulator to process & execute
- The instructions are recieved by the emulator, which transforms them into a native instruction set.
- The native instructions are run on the host m/c hardware
- It included processor-oriented instructions & the I/O specific ones.

- The basic emulation is through code interpretation
- An interpreter program interprets the source instructions to target instructions one by one
- One src instruction may require tens or hundreds od native target instructions to perform it's function.
- Instruction set emulation requires binary translation & optimization



- Advantage
- Enables the host to adjust to a change in the architecture od the guest system
- The infrastructure provided can be use forcreating VM's on a platform.
- It is possible to run alarge amt od legacy binary code written for various processors on any given new hardware host mlc.
- Disadvantage
- The instructions need to be interpreted bedore being executed.
- Shows poor performance

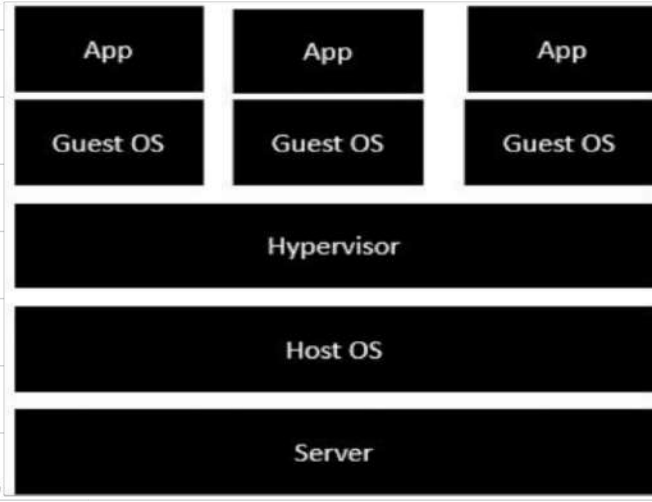
HAL Level
- This approach generates a virtual hardware env for a VM.
- It also manages the underlying hardware through virtualization.
- The idea is to virtualize computer's resources, such as it's processors, memory, I/o device.
- The intention is to upgrade the hardware utilization rate by multiple users concurrently
- Priviledge Instruction
  - The instructions that can run only in Kernel mode
  - It needs full attention od CPU for execution.
  - If not managed properly by the VMM, will raise an exception resulting into system crash.

- Trapping & forwarding the priviledged instructions to VMM helps in managing a system properly, thereby avoiding various risks keeping indivitual virtual m/c isolated.
- This required because of the possible existence of multiple VMs

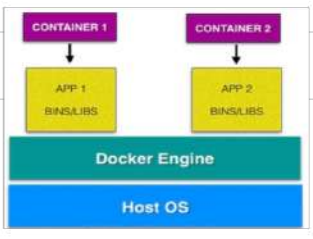
- It supports multiple OS's & application to be run simultaneously without system reboot.
- It gives appearance of multiple seperate m/c
- Degree of isolation is high with less implementation risky & easy maintenance.
- But it requires you to access to raw computer.

OS Level

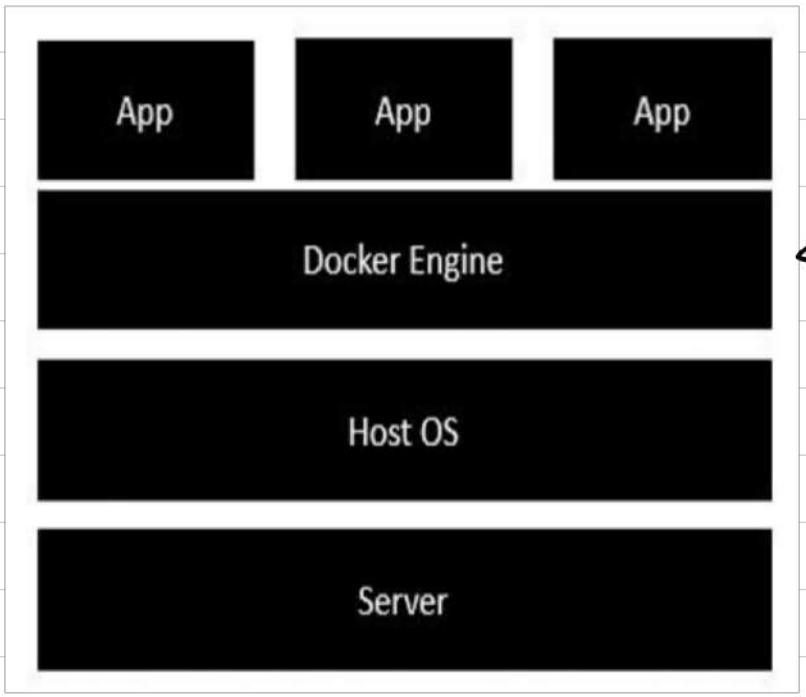
- To overcome the issue of redundancy & timeconsumption we implement virtualization at a higher Level.
- It is an abstraction layer between traditional OS & use applications.
- Traditional Virtualization
  - Servers is the physical server that is used to host multiple VMs.
  - Host os is the base machine.
  - Hypervisor is either VMware or Windows Hyper V that is used to host VMs.
  - One would then install multi OS as VMs on top of the existing hypervisor as GuestOs



- Dockers
  - They are analogous to physical containers that you can use to store, package & trarsport goods.
  - A container is an env. that runs an application that is not dependent on the OS

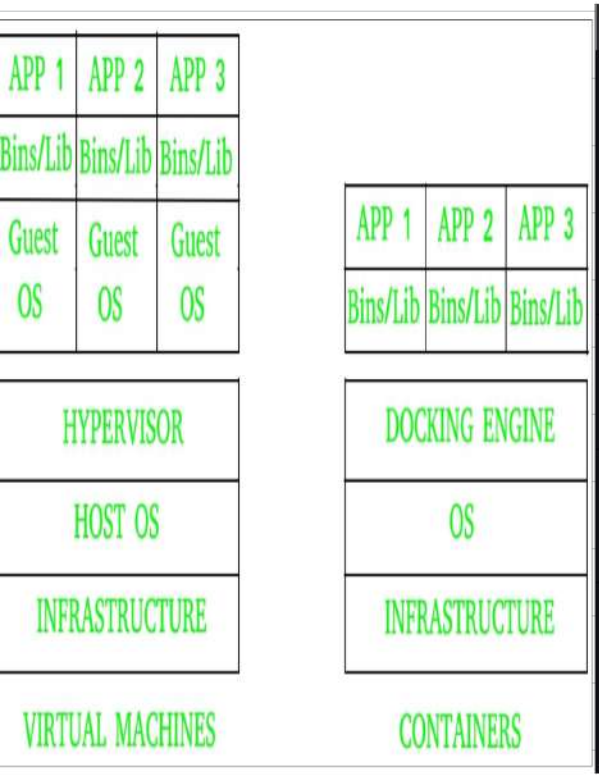


- It is a portable unit of software that has the application along with the associated dependency & config.
- The kernel of the host OS serves the need of running different functions of an app, seperated into containers.
- Each runs isolated tasks
- It cannot harm the host machine nor come in conflict with other apps.
- Architecture



← Is used to virtualize the guest OS which earlier used to run in VMs.

- The containers behave like real servers.



Virtual Machines(VM)	Containers
VM is a piece of software that allows you to install other software inside of it so you control it virtually as opposed to installing the software directly on the computer.	While a container is software that allows different functionalities of an application independently.
Applications running on a VM system, or hypervisor, can run different OS.	While applications running in a container environment share a single OS.
VM virtualizes the computer system, meaning its hardware.	While containers virtualize the operating system, or the software only.
VM size is very large, generally in gigabytes.	While the size of the container is very light, generally a few hundred megabytes, though it may vary as per use.
VM takes longer to run than containers, the exact time depending on the underlying hardware.	While containers take far less time to run.
VM uses a lot of system memory.	While containers require very less memory.
VM is more secure, as the underlying hardware isn't shared between processes.	While containers are less secure, as the virtualization is software-based, and memory is shared.
VMs are useful when we require all of the OS resources to run various applications.	While containers are useful when we are required to maximize the running applications using minimal servers.
Examples of Type 1 hypervisors are KVM, Xen, and VMware. Virtualbox is a Type 2 hypervisor	Examples of containers are RancherOS, PhotonOS, and Containers by Docker.

Library Support Level

- Most applications use APIs exported by user level lib. rather than using lengthy system calls by the OS
- API working
  - API architecture is usually explained interms of client & server.
  - The application sending the request is called client & the application sending the response is called server.
  - Types
  1) SOAP APIs: Stands for Simple Object access protocol, the client & server exchange messages using XML. Least Flexible API.
  2) RPC APIs: Stands for Remote procedure Calls, the client completes a function on the server, and the server sends the output back to the client.
  3) Web socket APIs: Uses JSON obj to pass data.
  4) REST APIs: Most Popular & flexible APIs found on the web today. It defines set of CRUD functions & uses HTTP for data exchange.
- also called user-level Application Binary Interface (ABI)
- It is done by API call interception & remapping.
- Create execution env. for running alien programs on a platform rather than creating a VM to run the entire OS.
- Ex. Wine, WAB, LxRun.
- Virtualization is possible by controlling the communication link between application & the rest of the system through API hooks.
- WINE
  - A free & open source compatibility Layer that aims to allow comp. programs developed for Microsoft windows, to run on UNIX-like OS.
  - It translates windows API calls into POSIX.
  - Eliminating the performance & memory penalties of other methods & allowing you to cleanly integrate windows applications into your desktop

- Another ex. vCUDA which allows application executing within VMs to leverage GPU hardware acceleration.
- Advantage: it has very low implementation effort.
- Disadvantage: poor application flexibility & isolation.

Application Level

- On traditional OS, an application often runs as a process, known as Process-Level virtualization
- The virtualization layer sits as an application program on top of an OS & exports an abstraction of a VM that can run programs which are written & compiled to that particular abstract machine definition.
- Any program written in the HLL & compiled for this VM will run on it.
- JVM
  - It is a VM that enables a computer to run Java programs as well as programs written in other lang. but compiled to Java bytecode.
  - It is responsible for interpreting Java bytecode & translating this into actions or OS sys calls.
- Stack based VMs
  - Both JVM & .Net CLI are examples
  - It is based on execution stack that is used to perform operations.
  - The bytecode generated by compiler for these architecture contains a set of instructions that load operand on the stack, perform some operations with them and put the result on the stack.
- Register based VMs
  - Ex: Parrot
- Application VM
  - A process VM also called application VM or MRE runs as a normal application inside a host OS & supports a single process.
  - It is created when ne process is started & destroyed when it exits.
  - Its purpose is to provide a platform independent programming env. & allows a program to execute in the same way on any platform.

- Other forms are application isolation, application streaming.
- The process involves wrapping the application in a layer that is isolated from the host os & other applications.
- The result is an application that is much easier to distribute & remove from user workstations.
- Advantage: has the best application isolation
- Disadvantage: low performance, low application flexibility & high implementation complexity.

Table 3.1 Relative Merits of Virtualization at Various Levels (More "X"'s Means Higher Merit, with a Maximum of 5 X's)

Level of Implementation	Higher Performance	Application Flexibility	Implementation Complexity	Application Isolation
ISA	X	XXXX	XXX	XXX
Hardware-level virtualization	XXXX	XXX	XXXX	XXXX
OS-level virtualization	XXXX	XX	XXX	XX
Runtime library support	XXX	XX	XX	XX
User application level	XX	XX	XXXX	XXXX

Virtualization Design Requirements

1) Equivalence Requirement

- A m/c that is developed through virtualization must have a logical equivalence with the real m/cs need to match the capabilities of physical system in it's computational performance & to execute all the applications & programs that are designed to execute on the real m/cs.

2) Efficiency Requirement

- While taking the route of virtualization, the virtual m/c must be as efficient in it's performance as a real system.
- Virtualization is primarily done with a purpose of getting efficient software without the physical hardware.

3) Resource Control requirement

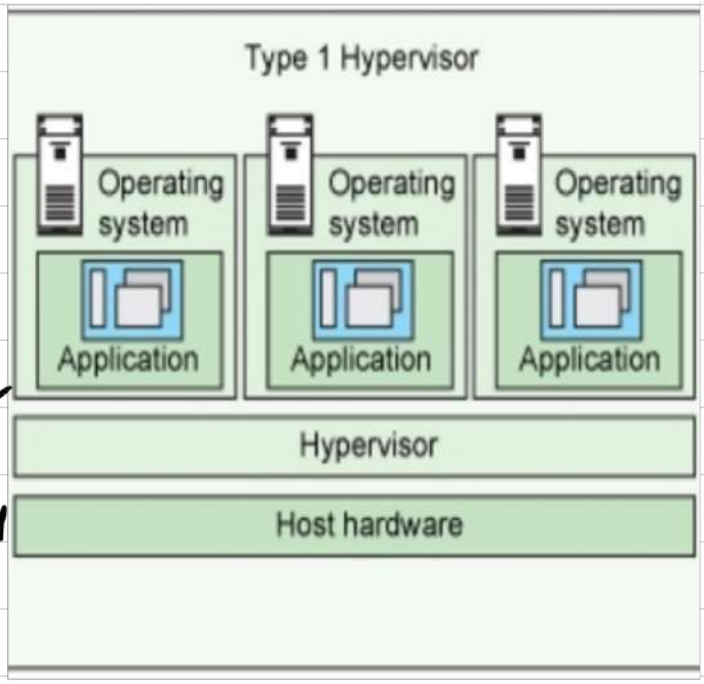
- A typical computer system is a combination of various resources including processors, memory & I/O devices.
- All these resources must be managed & controlled effectively by the VMM.
- The VMM must be in a state of enforcing isolation between the virtualized systems.

Hypervisors

- It is a hardware virtualization technique allowing multi OS, called guests to run on a host machine.
- Essentially it must be able to convert physical devices into virtual resources dedicated for the deployed VM to use.
- Types

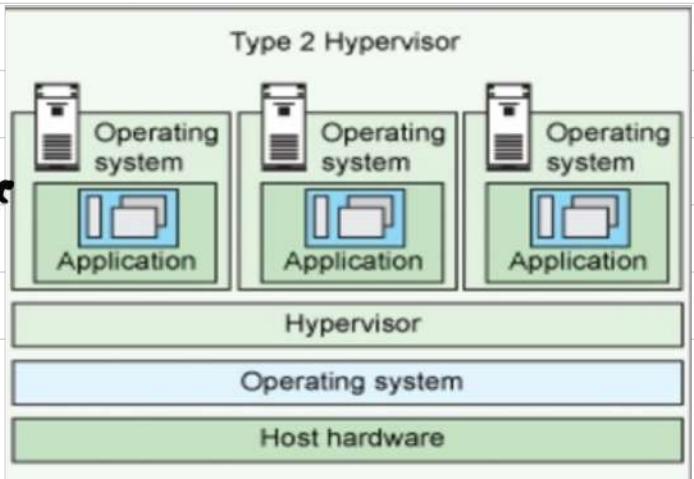
1) Type 1

- Sits on the bare metal computer hardware like the CPU, memory, etc.
- All the guest OS are a layer above the hypervisor.
- So the hypervisor is the first layer over the hardware.
- The orginal CP/CMS hypervisor developed by IBM was of this kind
- Examples are Microsoft Hyper-V



2) Type 2

- Runs over a host OS
- It is the second layer over the hardware
- The guest OS runs a layer over the hypervisor & so they form the third laye
- Examples are Free BSD.



- Architecture

1) Micro-Kernel
- includes only the basic & unchanging functions
- The device drivers & other changeable components are outside the hypervisor.

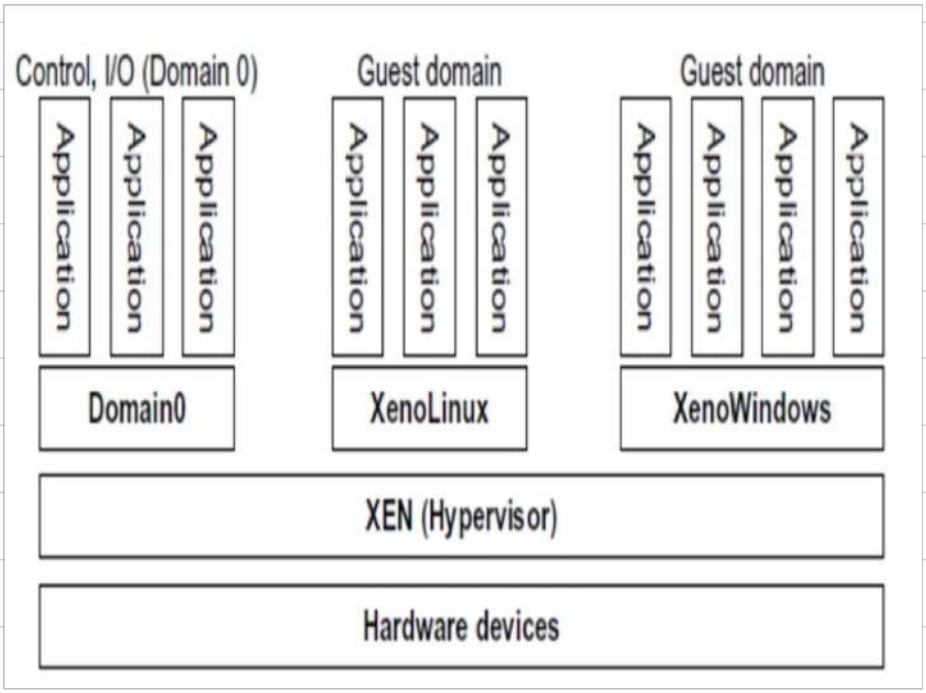
2) Monolithic
- implements all the mentioned functions including those of the device drivers.
- Size of the hypervisor code of a microkernel is smaller than that of a monolithic.

Feature	Monolithic Hypervisor	Microkernel Hypervisor
Definition	A hypervisor where most services (e.g., device drivers, memory management) run in kernel mode.	A hypervisor with minimal core functions, relying on user-space services for most operations.
Architecture	Large, single-layered kernel with all necessary components.	Small core with separate user-space services handling device drivers, networking, etc.
Performance	Faster due to direct kernel execution.	Slightly slower due to additional inter-process communication (IPC).
Security	Less secure as all components run in kernel mode.	More secure since only the core runs in kernel mode, reducing attack surface.
Complexity	More complex due to tightly integrated services.	Simpler core, but managing inter-service communication adds complexity.
Fault Isolation	Poor fault isolation; a failure in any component can crash the entire system.	Strong fault isolation; failures in user-space services don't affect the hypervisor.

Xen Architecture
- It is a micro kernel hypervisor.
- It seperates the policy from the mechanism
- Xen hypervisor implements all the mechanisms Leaving the policy to be handled by Domain 0.
- It is used for either:

1) Desktop virtualization
2) Server "
3) Recently for cloud computing solo with Xen Cloud Platform.

- Primarily based on Para virtualization.



- Many Guest Os can run on top of the hypervisor.
- Not all guest os are created equal, & one in particular controls the other
- Domains

Types

1) Domain 0

- Specific control Software: controls all the other guest Oss, is executed in a special domain called Domain 0.
- Privileged guest Os of xen
- Also called management VM, as it has the privilege to manage other VMs implemented on the same host.
- It first loaded when xen boots without any file system drivers being available.
- It is designed to access hardware directly & manage devices.
- So, one of the responsibilities is to allocate & map hardware resources for Domain U

- Working
  - The first one that is loaded once VMM has been completely booted
  - It hosts an HTTP server that servers requests for VM creation, config, migrate, suspension, resume & termination.

- It allows users to create, copy, save, read, modify, share, migrate & roll back VMs as easily as manipulating a file
- Security
  - If it gets compromised, the hacker can control the entire system
  - Security policy are needed to improve the security.

2) Domain U - All the other domains running guest OS.

Hyper Calls

- It is a special instruction used by VM to directly communicate with the hypervisor.
- Sensitive syscalls need to be re-implemented with hypercalls.

Full Virtualization

- non critical instructions run on the hardware directly while critical instructions are discovered & replaced with traps into the VMM to be emulated by software.
- The critical instructions are trapped as they threaten the security of the system and control hardware.
- And running non critical promotes efficiency also ensures system security.
- It relies on binary translation to trap & to virtualize the execution of certain sensitive, non-virtualizable instructions.
- FV does not need to modify guest OS & critical instructions are emulated by software through the use of binary translation
- Advantage
  - No need to modify OS
- However, this approach of binary translation slows down the performance alot & incur a large performance overhead.

Popek and Goldberg Instructions

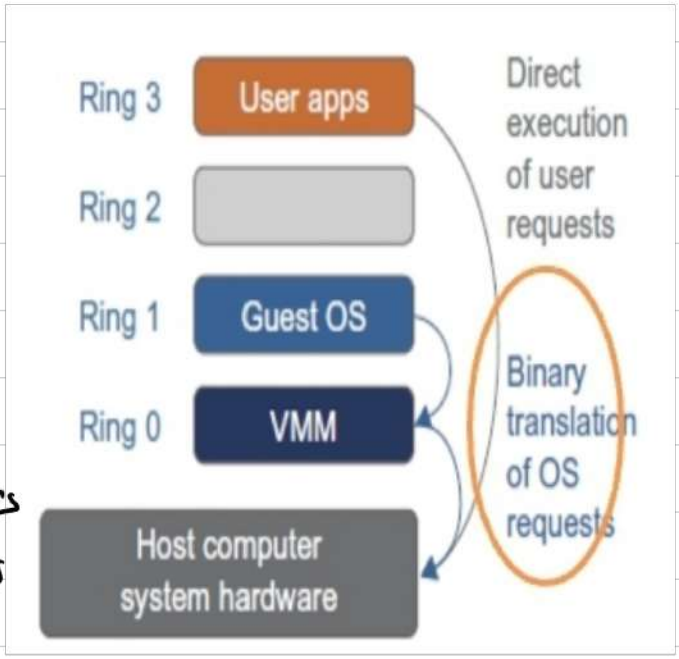
- The Popek and Goldberg virtualization requirements are **a set of conditions sufficient for a computer architecture to support system virtualization efficiently.**
- They were introduced by Gerald J. Popek and Robert P. Goldberg in their 1974 article "Formal Requirements for Virtualizable Third Generation Architectures".
- A convenient way of determining whether a computer architecture supports efficient virtualization and provide guidelines for the design of virtualized computer architectures.

Popek and Goldberg Instructions

- Privileged instructions
  - Those that trap if the processor is in user mode and do not trap if it is in system mode (supervisor mode).
- Control sensitive instructions
  - Those that attempt to change the configuration of resources in the system.
- Behavior sensitive instructions
  - Those whose behavior or result depends on the configuration of resources (the content of the relocation register or the processor's mode).

Binary Translation

- VMware puts the VMM at ring 0 & the guest OS at ring 1.
- The VMM scans the instruction stream & identifies the privileged control & behavior sensitive instructions
- They are trapped into the VMM, which emulates the behavior of these instructions
- This method used in this emulation is called binary translation

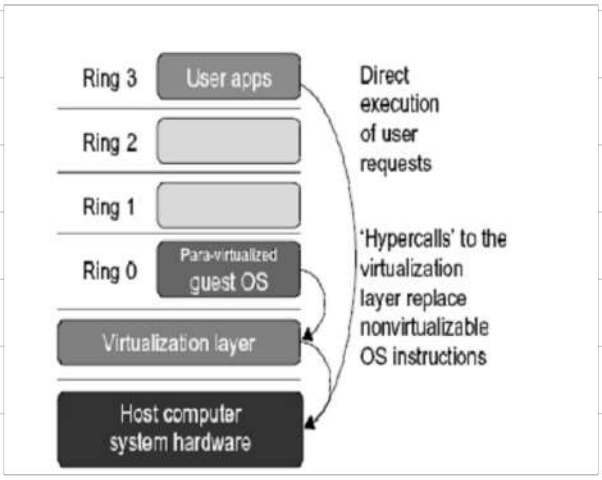


- **Translation**
  - The trap triggers the translation of the offending instructions into an equivalent set of instructions that achieve the same goal without generating exceptions.
- Thus FV combines binary translation & direct execution.
- The guest OS is completely decoupled from the underlying hardware and it is unaware that it is being virtualized.
- **Caching**
  - Binary translation employs a code cache.
  - To store translated hot instructions to improve performance.
  - To improve efficiency, the equivalent set of instruction is cached, so that for further occurrences of the same instructions

, translation is not necessary.

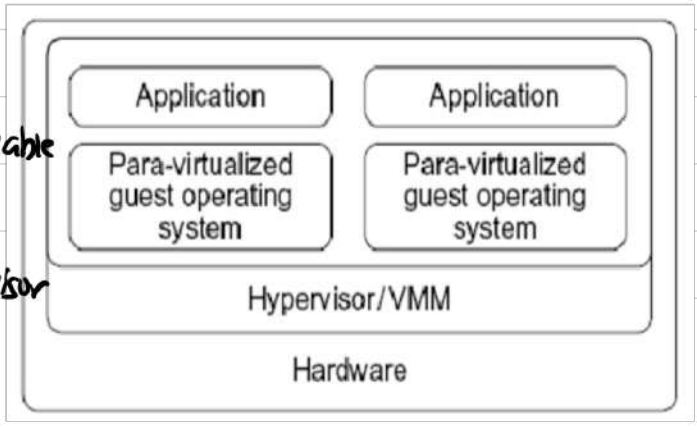
- But it ↑cost of memory usage.
- Advantage
  - Guest can run unmodified in a virtualized env.
  - Binary Translation is applied to a subset of instruction set, while others are managed through direct execution on the underlying hardware
  - The reduces the impact on performance.
- Disadvantage
  - It introduces additional overhead that is not present in other approaches.
  - Rather time Consuming
  - The code cache to store translated hot instructions to improve performance but ↑cost of memory usage.

Para virtualization



- Needs to modify the guest OS
- Provides special APIs requiring substantial OS modifications in user applications
- However when the guest OS kernel is modified for virtualization, it can no Longer run on the hardware directly
- Performance degradation is a critical issue of a virtualized system.
- Attempts to reduce the virtualization overhead.

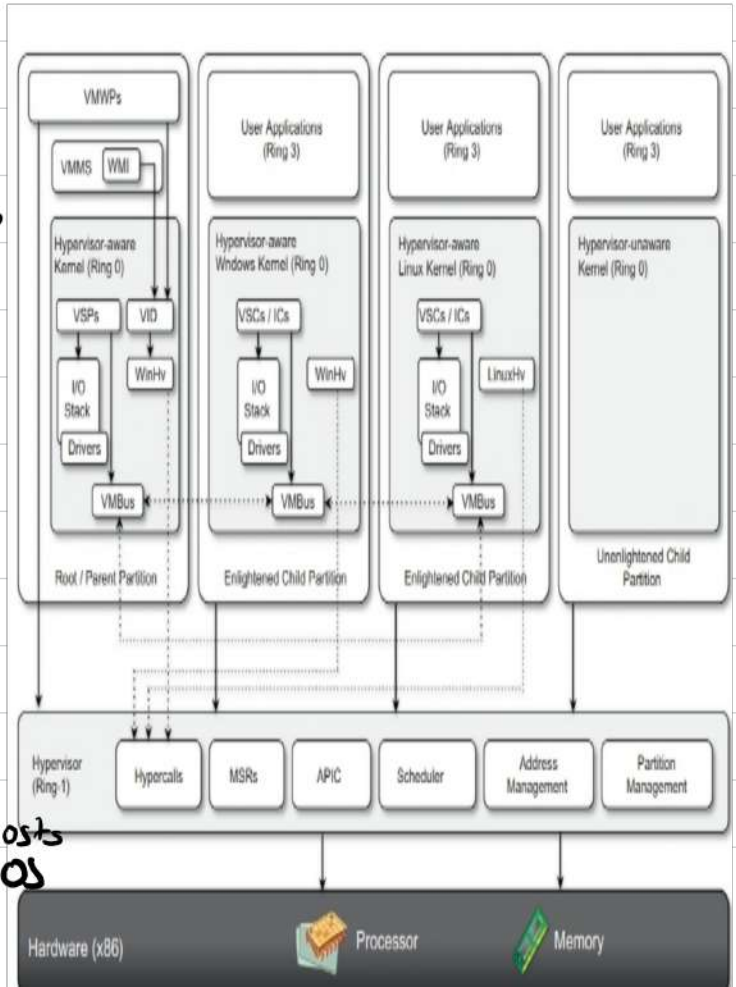
- They are assisted by an intelligent compiler to replace the non-virtualizable OS instruction by hypercalls that directly communicates with the hypervisor or VMM.



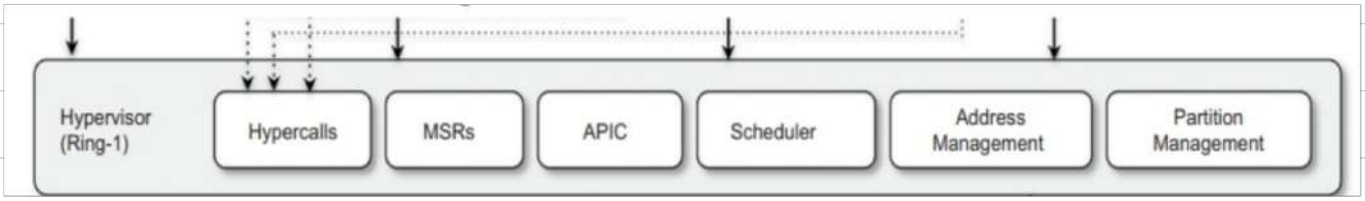
S.No.	Full Virtualization	Paravirtualization
1.	In Full virtualization, virtual machines permit the execution of the instructions with the running of unmodified OS in an entirely isolated way.	In paravirtualization, a virtual machine does not implement full isolation of OS but rather provides a different API which is utilized when OS is subjected to alteration.
2.	Full Virtualization is less secure.	While the Paravirtualization is more secure than the Full Virtualization.
3.	Full Virtualization uses binary translation and a direct approach as a technique for operations.	While Paravirtualization uses hypercalls at compile time for operations.
4.	Full Virtualization is slow than paravirtualization in operation.	Paravirtualization is faster in operation as compared to full virtualization.
5.	Full Virtualization is more portable and compatible.	Paravirtualization is less portable and compatible.
6.	Examples of full virtualization are Microsoft and Parallels systems.	Examples of paravirtualization are Microsoft Hyper-V, Citrix Xen, etc.
7.	It supports all guest operating systems without modification.	The guest operating system has to be modified and only a few operating systems support it.
8.	The guest operating system will issue hardware calls.	Using the drivers, the guest operating system will directly communicate with the hypervisor.
9.	It is less streamlined compared to para-virtualization.	It is more streamlined.
10.	It provides the best isolation.	It provides less isolation compared to full virtualization.

Microsoft Hyper-V

- Hyper-V is an infrastructure virtualization SOI developed by Microsoft.
- It uses hypervisor-based approach to hardware virtualization, which leverages several Techniques to support variety of guest OS.
- Type 1 Hypervisor.
- **Architecture**
  - Despite it's straightforward installation as a component of the host OS
  - Hyper-V takes control of the hardware, and the operating system becomes a VM instance with special privileges, called the parent partition
- Is the only one that has direct access to the hardware
- It runs the virtualization stack, hosts all the drivers required to config guest OS & creates child partitions through the hypervisor.



- The child partitions have guest OS & do not have access to the underlying hardware
- but their interaction with it is controlled by either the parent partition or the hypervisor itself.



Hypercall interface

- This is the entry point for all the partitions for the execution of sensitive instruction.
- This is an implementation of the paravirtualization approach.
- This interface is used by drivers in the parititioned OS to contact the hypervisor using the std windows calling convention.
- The parent partition also uses this interface to create child partitions.

Memory service routines (MSRs)

- These are the set of functionalities that control the memory & its access from partitions.
- By leveraging hardware-assisted virtualization, the hypervisor uses IOMMU to fast-track access to devices from partitions by translating virtual memory address

Advance Programmable interrupt controller (APIC)

- This component represents the interrupt controller, which manages the signals coming from the underlying hardware when some event occurs.
- Each virtual processor is equipped with a synthetic interrupt controller, which constitutes an extension of the local APIC
- The hypervisor is responsible of dispatching the physical interrupts to the synthetic interrupt controllers.
- Its services are available through the hypercall interface API previously discussed

Scheduler

- This component schedules the virtual processors to run on available physical Processors.
- It is controlled by policies that are set by the parent partition

Address Manager

- This component is used to manage the virtual network addresses that are allocated to each guest OS

Partition Manager

- This component is in charge of performing partition, creation, finalization destruction, enumeration & config.
- The hypervisor runs in Ring 1 & requires corresponding hardware tech that enables such a cond°.
- By executing in this highly privileged mode, the hypervisor can support legacy OS that has been designed for x86 hardware
- OS of newer gen can take advantage of the new specific architecture of Hyper-V.

VMware

- It offers a collection of virtualization sol^ covering the entire range of the market.
- Based on FV.
- Implement either in the desktop env (Type II) or server env (Type I)
- Supports virtualization of OS errs on end user comp.
- **Desktop virtualization (end-user)**
  - Specific VMware software (Workstation for Windows, Fusion for Mac) are installed in the host OS.
  - To create VM & manage their execution.
  - It is done by installing a specific driver in the host OS that provides 2 main services:
    1) To deploy a VM manager that can run in privileged mode.
    2) It provides hooks for the VMware application to process specific I/O reques eventually by relaying such requests to the host OS via syscalls

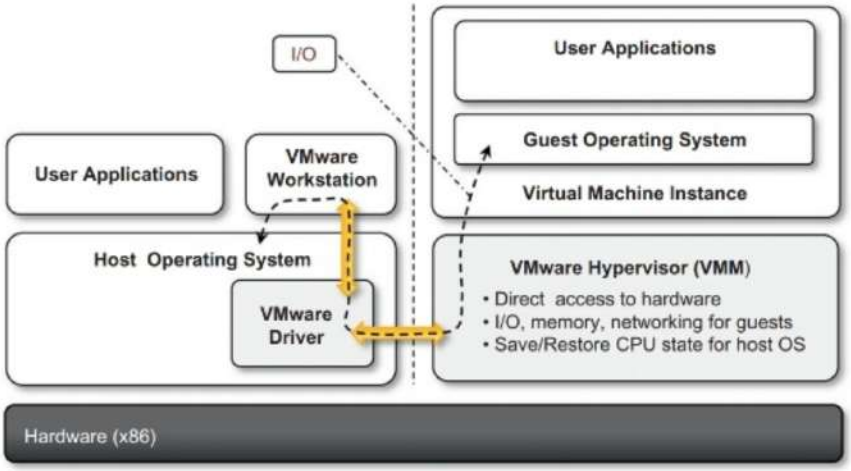


FIGURE 3.13
VMware workstation architecture.

Server Virtualization

- Initial support came from GSX server.
- It replicates the approach used for end-user comp. and introduces remote management & scripting capabilities.
- It uses a client-server model, allowing remote access to VM, at the cost of some graphical performance.
- The architecture is mostly designed to serve the virtualization of web servers.
- A daemon process called serverd, controls & manages VMware application processes.
- The remote host runs a VMware server authentication daemon.
- This applications are then connected to the VM instance by the VMware driver installed on the host OS
- VM instances are managed by the VMM
- User requests for VM management & provisioning are routed from the Web Server through the VMM by means of serverd.

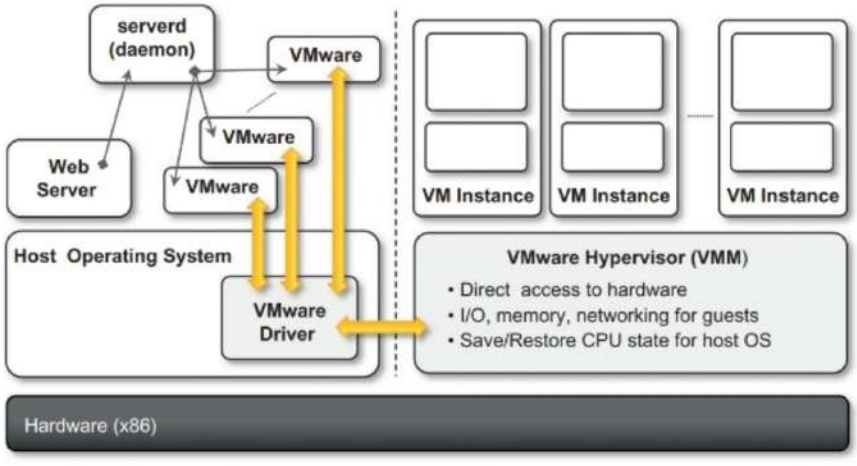


FIGURE 3.14
VMware GSX server architecture.

Infrastructure virtualization and cloud computing sol^n

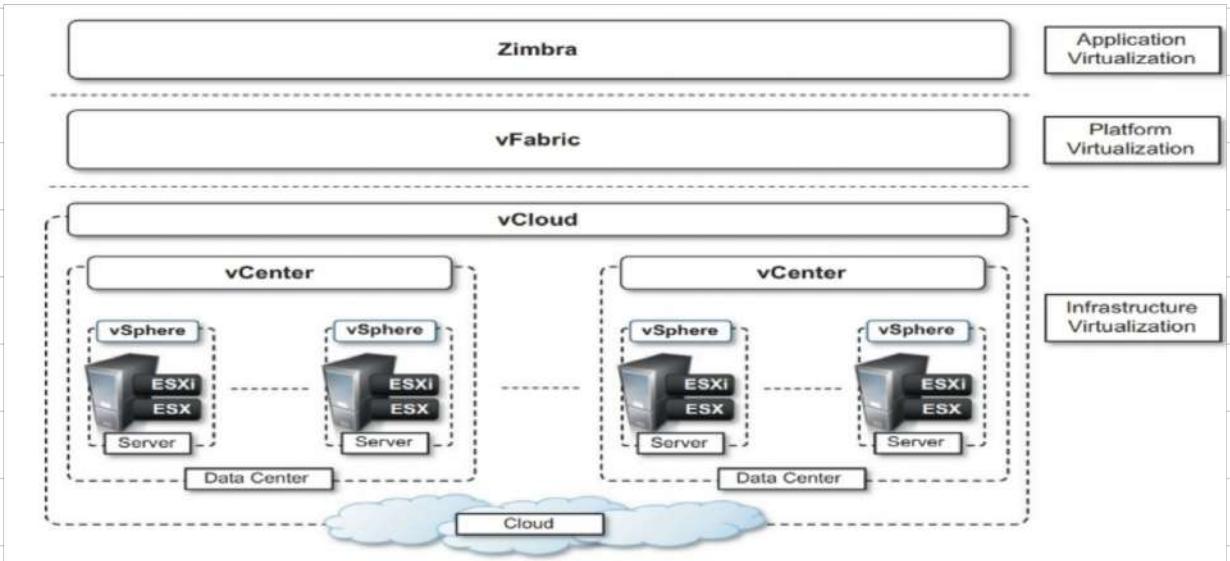


FIGURE 3.16
VMware Cloud Solution stack.

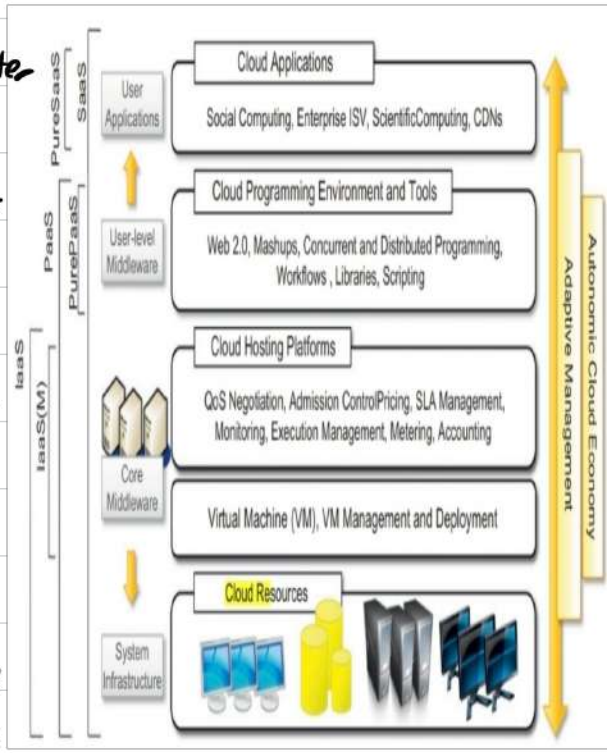
1) ESX & ESXi constitute the building blocks of the sol^n for virtual infrastructure management.
2) A pool of virtualized servers is tied together & remotely managed as a whole by vSphere.
- vSphere as a virtualization platform it provides a set of basic services beside virtual comp. services: 1) Virtual file system, 2) Virtual Storage 3) Virtual Network
- Application services such as VM migration, Storage migration, data recovery & security zone, complete the services offered by vSphere
3) vCenter provides centralized administration & management of vSphere installations in a data center env.
4) A collection of virtualized data centers are turn into a Iaas cloud by vCloud.
- Which allows to make available to end users virtual computing env on demand on a pay-per-use basis
- A web portal provides access to the provisioning services of vCloud & end users.
5) Sol^n for application development in the cloud with vfabric.
- It is a set of components that facilitate development of scalable web applications → PaaS

- It is a collection of components for application monitoring, scalable data management & scalable execution & provisioning of Java Web application.

6) Finally, at the top, Zimbra, a soln for office automation, messaging, and collaboration that is completely hosted in the cloud
- It is an SaaS soln that integrates various features into a single software platform providing email & collaboration management

The Cloud Reference Model

Architecture



- Cloud resources are harnessed to offer computer HP required for providing services
- Often, this layer is implemented using a datacenter in which 100s & 1000s of nodes are stack together
- Cloud infrastructure can be heterogenous in nature, due to clusters & networked PCs moreover, database systems & other storage serivces can also be part of it.
- The physical infrastructure is managed by the core middleware used to provide an runtime env for application & to best utilize resources
- Core Middleware at the bottom of the stack, virtualization tech are used to guarantee runtime env customization, application isolation, Sandbox & QoS.
- Hypervisors manage the pool of resources.
- We can organize IaaS into 2 categories:
  1) both management & physical infrastructure.
  2) Only management (IaaS(M)).
- In this second case management layer is often integrated with other IaaS soln.
- A new layer is formed by cloud programming env. & tools, for offering users a development platform for applications.

- The range of tools include Web-based interfaces, command line tools & framework.
- PaaS solns generally include infrastructure as well.
- Pure PaaS, only the user-level middleware is offered.
- The top layer is services delivered at the application level.
- These are mostly referred to as SaaS.

Economics of Cloud

- Uses the pay-as-you go model.
- In particular, cloud computing allows:
  1) Reducing the capital cost.
  2) Eliminate the depreciation or lifetime cost
  3) Replace software licensing with subscriptions
  4) Cutting maintenance & admin cost.

Capital cost

- It is the cost occurred in purchasing an asset that is useful in the production of goods or rendering of services, it is a one time expense paid up front.
- IT infrastructure & software are capital assets.
- It is good practice to try to keep capital costs as low as possible.
- As they are associated with material things they are subject to depreciation over time, which in the end reduces the profit.
- In case of IT capital cost, the depreciation costs are represented by the loss of value of the hardware & the aging of software products
- Many enterprises owned small or medium-sized datacenter that introduces several operational costs in terms of maintenance, electricity & cooling.
- Pricing models introduced are
  1) Tiered pricing - offered in several tiers, each of which offers a fixed computing specification & SLA at a specific price per unit of time, used by Amazon (EC2 services)
  2) Pre-unit pricing → GoGrid
  3) Subscription-based pricing → used by mostly SaaS providers

Open Challenges

1) Cloud Definition

i) Definition by NIST (mod 1)

ii) Definition by UCSB, which departs from the Xaas concept & tries to define an ontology for CC.

- In their work the concept of cloud is dissected into five main layers: 1) application, 2) Software env, 3) Software Infrastructure 4) Software Kernel, 5) hardware.
- Each layer address the needs of different class of users within the cloud computing community.

2) Cloud interoperability & Stds

- Vendor lock-in constitutes one of the major strategic barriers against the seemless adoption of CC at all stages.
- It can prevent customer from switching to another competitor's sol^n
- This can occur either because the customer wants to find a more suitable sol^n for customer needs or because the vendor is no longer able to provide the required service.
- The first steps toward a standardization process have been made done by, CC interoperability forum, Open CC & DMTF Cloud Standard Incubator.

3) Scalabity & fault tolerance

- The ability to scale on demand constitutes one of the most attractive features of CC.
- To implement such a capability, the cloud middlewave has to be designed with the principle of scalability along different dimensions in mind.

4) Security, trust and Privacy

- Trad. crypographic tech. are used to prevent data tampering & access to sensitive data
- The massive use of virtualization tech exposes the existing system to new threats.

- On one side we need to decide whether to trust the provider itself
- On the other side, specific regulation can simply prevail over the agreement the provider is willing to establish with us.
- Moreover, cloud services delivered to the end user can be the result of complex stack obtained by 3rd parties.
- In this case there is a chain of responsibilities in terms of service delivery that can introduce more vulnerability.

5) Organization aspects

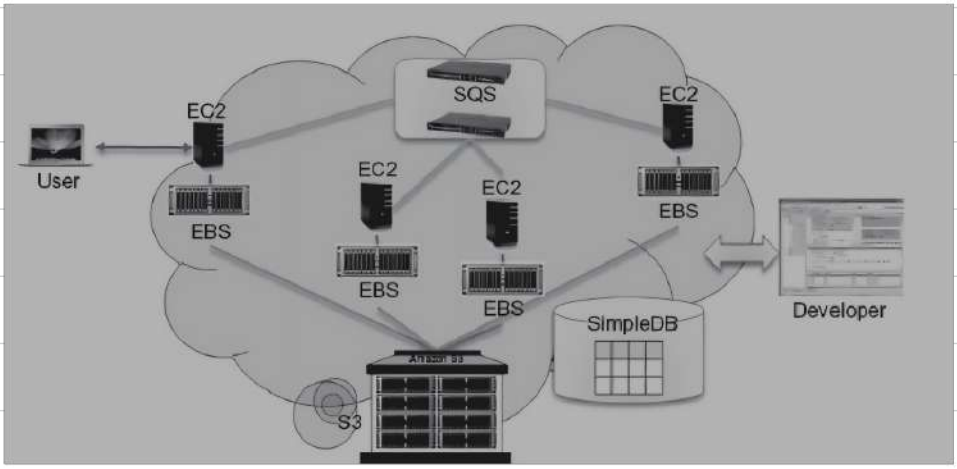
- CC introduces a billing model that is new within typical enterprise IT departments, which require a certain level of cultural & organizational process maturity.
- A wide acceptance of CC will require significant change to business process & org. boundaries.

- In particular, the following questions have to be considered:
  - What is the **new role of the IT department in an enterprise that completely or significantly relies on the cloud?**
  - **How will the compliance department perform its activity when there is a considerable lack of control over application**
  - **What are the implications (political, legal, etc.) for organizations that lose control over some aspects of their services?**
  - What will be the perception of the end users of such services workflows?

Mod III

AWS

- IaaS model



- The figure shows the AWS architecture

1) EC2, 2) S3, 3) EBS, 4) SQS, 5) Users.

Amazon Web Services

Compute Services	Storage Services	Communication Services	Additional Services
Amazon Elastic Compute Cloud (EC2)	Amazon Simple Storage Service (S3)	Amazon Simple Queue Service (SQS)	Amazon GovCloud
Amazon Elastic MapReduce	Amazon Elastic Block Store (EBS)	Amazon Simple Notification Service (EBS)	Amazon CloudWatch
AWS Elastic Beanstalk	Amazon ElastiCache	Amazon Simple Email Service (SES)	Amazon Flexible Payment Service (FPS)
AWS Cloudformation	Amazon SimpleDB	Amazon Route 53	Amazon DevPay
Autoscaling	Amazon Relational Database Service (RDS)	Amazon Virtual Private Cloud (VPC)	Amazon Fullfillment Web Service (FWS)
	Amazon CloudFront	Amazon Direct Connect	Amazon Mechanical Turk
	Amazon Import/Export	Amazon Elastic Load Balancing	Alexa Web Information Service
			Alexa Top Sites

Amazon AWS Platform

FIGURE 9.1

Amazon Web Services ecosystem.

Amazon machine image

- It is a template for which it is possible to create a VM
- They are stored in S3
- Identified by a unique identifier form of ami-xxxxx & manifest XML file.
- It contains a physical file system layout with a predefined OS installed
- They are specified by the Ramdisk Image & Kernel Image
- Which are part of the config of the template.
- They are either created from scratch or bundled from existing EC2 instances.

- Prepare a new AMIs

1) Create an instance from pre-existing AMI
2) log into it once it is booted & running & install all the software needed , customize the instance.
3) Then save this as a new custom AMI.
4) Instance launched from this new custom AMI include the customizati that you made when you created the AMI.
5) Once created it is then stored in S3.

- The user can decide whether to make it available to other users or keep it for personal use.
- Finally, it also possible to associate a product code with a given AMI, thus allowing the owner of AMI to get revenue every time this AMI is used to creat EC2 instances

EC2

- The EC2 compute unit (ECU) is a measure of the computing power of virtual core, it is used to express a predictable quantity of real CPU power that is allocated to an instance.
- By using compute units instead of real frequency values, Amazon can change over time the mapping of such units to the underlying real amt of computing power allocated.

- We can identify 6 major categories

1) Standard instances: This class offers a set of config, that are suitable for most applications, EC2 provides 3 different categories of increasing computing power, storage & memory

2) Micro instances: This class is suitable for those applications that consume a limited amt of computing power, memory & occasionally need bursts in CPU cycles to process surges in the workload, used for small web app. with limited traffic.

3) High-memory instance:
- This class targets app. that need to process huge workloads & require large amts of memory
- 3-tier Web app characterized by high traffic are the target Profile
- 3 categories of ↑ memory & CPU are available, with memory Proportionally larger than computing power.

4) High-CPU instances: This class compute-intensive app., 2 config are available where computing power proportionally ↑ more than memory.

5) Cluster Compute instances: This class is used to provide virtual cluster services, instances in this category are characterized by high CPU compute power & large memory & an extremely high I/O and network performance.

6) Cluster GPU instances: This class provides instances featuring GPU & high compute power, particularly suited for cluster app. that perform heavy graphic computations.

- EC2 instances are priced hourly according to the category they belong to
- It can be run either by CLI tool or via the AWS console.
- Environment
  - It is in charge of allocating addresses, attaching storag Vol. and config security in terms of access control & network connectivity

- By default, instance is created with an internal IP addresses which makes them capable of communicating within the EC2 network and accessing the internet as clients.
- It is possible to associate an Elastic IP to each instance, which can be remapped to different instance over time.
- The Public IP auto assigns every time I stop & start. to avoid this Elastic IP can be used.
- Together with an external IP, EC2 instances are also given a domain name.
- Owners can partially control where to deploy instances, they can associate a key pair to one or more instances when created.
- EC2 controls the accessibility of a virtual instance with basic firewall configuration
- Rules can also be attached to security groups, and instances can be made part of one or more groups before their deployment.
- Security groups & firewall rules constitute a flexible way of providing basic security of EC2 instance.

Storage Services AWS

- collection of services for data storage

S3

- It is a distributed object store that allows users to store info. in different formats
- It is accessible through REST interface, which is quite similar to a distributed filesystem but with some imp. differences:

1) The storage is organized in 2-level hierarchy
   - Buckets & Objects
   - It organizes its storage space into buckets that cannot be further partitioned, which means not possible to create directories or other kinds of physical grouping of obj. stored.

2) Stored Obj. cannot be manipulated like std. files

- Designed to provide storage for objects that will not change over time; it does not allow renaming, modifying or relocating an obj.
- Once an obj. has been added to a bucket its content & position is immutable & only way to change it is to remove the obj from the store & add it again.

3) Content is not immediately available to users

- because it is a large distributed storage facility, change are not immediately reflected.
- It uses replication to provide redundancy & efficiently serve obj. across the globe, which intro. latencies when adding obj to the store.

4) Request will occasionally fail.

- Due to a large distributed infrastructure being managed requests for obj may occasionally fail
- It can decide to drop a request by returning an Internal server error
- ∴ it is expected to have a small failure rate during day to day operations.

- Core Components

1) Buckets

- It is a container of obj.
- It provides users with a flat store to which they can add objects.
- Buckets are top level elements of the S3 storage architecture & do not support nesting
- It is located in a specific geo. location & eventually replicated for fault tolerance.
- Users can select the location at which to create buckets. (by default US)
- Once created, all the obj. that belong to the bucket will be stored in the same availability zone of the bucket.
- Made accessible by REST interface.
- ∴ Represented by URIs

- All the operations are performed by expressing the entity they are directed to in the form of a request for a URI.
- Users create a bucket by sending a PUT request to https://s3.amazonaws.com/ with the bucket name & availability zone.
- Content can be listed by GET
- Once created it cannot be renamed or relocated
- bucket can be deleted by DELETE.

2) Objects
- It constitute the content elements stored in S3
- An obj. is identified by a name that needs to be unique within the bucket.
- Either store files or push to the S3 text stream.
- Name cannot be longer than 1024 bytes and encoded in UTF-8 & allows almost any characters.
- Since buckets do not support nesting, even characters normally used as path separators.
- Created via PUT request that specifies the name of obj with bucket name, its content & additional properties.
- Max size 5GB & immutable
- Retrieved via GET & deleted via DELETE

3) Metadata
- Objects can be tagged with metadata, which passed as properties of PUT request.
- Retrieved by either GET or HEAD request.

- Access Control & security of Buckets and Objects

1) READ → allows to retrieve an obj & it's metadata.
2) WRITE → allows to add an obj. to a bucket as well as remove it
3) READ-ACP → allows to read ACP of a resource.
4) WRITE-ACP → allows to modify the ACP of a resource.
5) FULL-CONTROL → grants all.

Feature	Amazon EBS (Elastic Block Store)	Amazon S3 (Simple Storage Service)
Type	Block Storage	Object Storage
Use Case	Best for low-latency, high-performance storage like databases, boot volumes, and applications requiring frequent updates.	Ideal for storing large amounts of unstructured data, backups, archives, media files, and static content.
Data Access	Can be attached to an EC2 instance; only accessible within the same Availability Zone (AZ).	Accessible over the internet using REST API, CLI, SDKs, and web console.
Scalability	Limited by EC2 instance type; needs manual resizing.	Virtually unlimited storage with automatic scaling.
Performance	Provides high IOPS and low latency. Suitable for transactional workloads.	Designed for high throughput, not optimized for low-latency transactions.
Backup & Recovery	Snapshots can be taken and stored in S3, but restoring requires creating a new volume.	Supports versioning, lifecycle policies, and cross-region replication.
Data Modification	Allows block-level modifications.	Immutable storage—files must be re-uploaded to modify.
Size Limits	- **Max volume size:** 1 TiB per volume - **Max volumes:** 20 per account	- **Max object size:** 5 TiB - **Max buckets:** 100 per account - **Total storage:** Unlimited
Security	- No built-in access management tools. - If an attacker gains access to an EC2 instance, they can access the attached EBS volume.	- Prevents unauthorized access using access management tools and encryption policies.
Applications	- Databases (MySQL, PostgreSQL, MongoDB, etc.). - Virtual machines and boot volumes. - Real-time transactional applications.	- Data backup and archiving. - Media and content distribution. - Big data analytics and machine learning datasets.
Advantages	- High-performance, low-latency storage. - Supports database workloads. - Persistent storage for EC2 instances.	- Highly durable and scalable. - Cost-effective for large data storage. - Accessible from anywhere via API.
Limitations	- Tied to a single AZ, requiring extra configuration for redundancy. - Limited scalability compared to S3. - Charges for provisioned storage, even if unused.	- Not ideal for frequently changing data due to immutability. - Higher latency compared to EBS. - Extra cost for frequent data retrieval.

Amazon ElastiCache

- based on a cluster of EC2 instances running the caching software
- It is an implementation of an elastic in-memory cache based on a cluster of EC2 instances.
- Auto. patch management & failure detection & recovery of cache nodes allow the cache cluster to keep running.
- Users only have to elastically size the cluster when needed.
- It can be dynamically resized according to the demand of the client applications.
- Priced according to EC2 model.

Structured Storage Solutions

i) Preconfigured EC2 AMIs

- Are predefined templates featuring an installation of a given data base management system.
- EC2 instances created from these AMIs can be completed with an EBS volume for storage persistence.
- Instance are priced hourly according to the EC2 cost model.
- This soft poses most of the admin burden on the EC2 user.
- But offers the greatest variety of products to choose from.

2) Amazon RDS

- It relies on the EC2 infrastructure.
- Devs do not need to worry about config the storage for high availability, designing failover strategies.
- Users are not responsible for any admin work
- Service provides user with auto backups, snapshots, point-in-time recoveries, and facilities for implementing replications.
- It's a managed database service, it's responsible for most managment tasks.
- Advantage
  - manages backups, software patching, autofailure detection & recovery
  - You can use the backups to restore a DB.
  - Restore process is reliable & efficient.
  - Users can get high availability with a primary instance and a sync. secondary instance, they can also use read replicas to Pread Scaling
  - Addition to the security you can control the access.
  - You can use DB such as MySQL, Oracle, MariaDB, Microft SQL Server and PostgreSQL

3) Amazon Dynamo DB

- Fully Managed NoSQL DB services that provides fast & predictable Performance with seemless scalability.
- It lets you offload the adminstorative burden od operating & Scaling a distributed database.
- It also offers encryption, which eliminates the Operational burden & complexity involved in protecting Sensitive Data.

4) Amazon Simple DB

- It is a light weight, highly scalable, & flexible data storage SOL for applications that do not require a fully relational model for their data
- It provides support for semistructured data, the model for which is based on the concept of domains, items & attributes

- With respect to the relational model, this model provides fewer constraints on the structure of data entries.
- It is a highly available NoSQL DB that offloads the work of database admin.

Data base

Instance

- basic building block of RDS
- An isolated DB env. in the cloud.
- It can contain one or more user created DBs.
- It can be accessed by using the same tools & applications that you use with a std alone DB.
- You can create & modify by CLI, RDS API or Management console.
- Types

1) General Purpose, 2) Memory optimized, 3) Burstable-performance 4) Optimized Reads.

- Classes
  - It determines the computation & memory capacity of Amazon RDS DB instance types.
  - It depends on your processing power & memory requirements.
  - It consists of both type and size.

Engines

- It is the specific RDB software that runs on your DB instance.
- Each DB engine has it's own supported features, and each version of a DB engine can include specific features.
- Additionally, each has a set of parameters, that control the behavior of the DBs that it manages.

Communication Services

1) Virtual Networking

- It comprises a collection of services that allow AWS users to control the connectivity to & between compute & storage services.

a) Amazon Virtual Private Cloud (AVPC)
- It provides a great degree of flexibility in creating VPNs within the amazon infrastructure & beyond.
- The service providers prepare either templates or a fully custom network service.
- The template include, public subnets, isolated networks, NAT, hybrid network.
- Also it is possible to control connectivity between different services. by using IAM service.

b) Amazon Direct Connect (ADC)
- It allows aws users to create a dedicated networks between the user private network & ADC locations called ports.
- The advantage is the consistent performance of the connection.
- It is compatable with other services
- Used when it requres high bandwidth between Amazon network & the outside world.

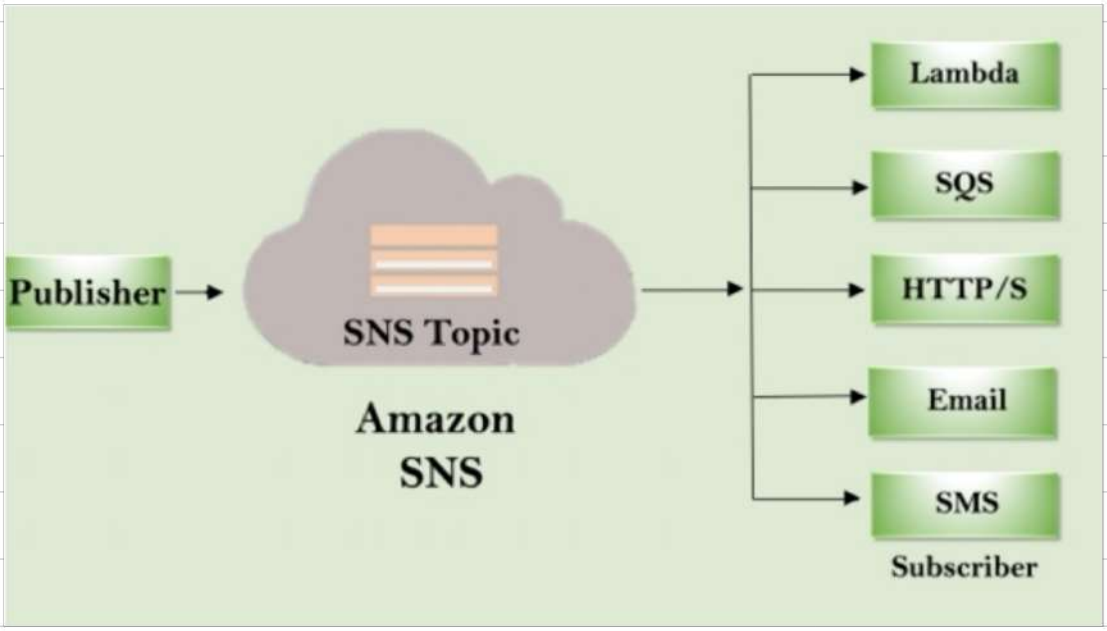
c) Amazon Route 53
- Implements dynamic DNS services that allow aws resources to be reached through domain names different from the amazon.com domain.

2) Messaging

a) Amazon Simple Queue Service
- Constitues disconnected model for exchaning messages between application.
- Users can create an unlimited no° of message queues & config them to control their access
- Applications can send messages to any queue they have access to.
- These messages are securly & reduntantly stored within the Aws infrastructure for a limited period of time, and they can be accessed by other applications
- While a message is being read, it is kept locked to avoid spurious processing from other applications, such a lock will expire after a given period.

b) Amazon Simple Notification Service

- Provides a publish-subscribe method for connecting heterogeneous applications
- It allows applications to be notified when new content of interest is available
- User can create a topic, which other applications can sub to
- At any time, applications can publish content on a given topic & subscribers can be automatically notified.
- There are two clients of SNS



c) Amazon Simple Email Service

- Provides AWS users with a scalable email services.
- It is a cloud email service provider that can integrate into any application for bulk email sending.
- Also provides a wide range of statistics that help users to improve their email campaigns.

1) **Once users are signed up for the service, they have to provide an email id that SES will use to send emails on their behalf.**
2) To activate the service, **SES will send an email to verify the given address and provide the users with the necessary information for the activation.**
3) Upon verification, **the user is given an SES sandbox to test the service**

Additional Services

1) Amazon CloudWatch

- It provides a comprehensive set of statistics
- It observes & monitors resources & applications on AWS on premises & on other clouds.
- It collects info. from several other AWS services.
- Using this devs can see a detailed breakdown of their usage of the services they are renting on AWS.
- The home page auto. displays metrics about every AWS service you use.
- You can additionally create custom dashboards.
- You can create alarms that watch metrics & send notifications or auto make changes to the resources you are monitoring.

2) Amazon Flexible Payment Service

- It allows users to leverage Amazon's billing infrastructure to sell goods & services to other AWS users.
- It is the first payment service designed from the ground up.
- Web service APIs allows movement of money.
- It is 100% focus on the needs of devs.
- Payment instructions give devs the flexibility to build multiple changing models that exactly meets their needs