Santa Clara University

Computer Networks

COEN 233 Spring 2020

Virtualization & Cloud Computing

By: Vaishali Gupta (W1588183)

Audience

This research project report is for Engineers working in Network and Virtualization fields who are trying to build secure applications in the virtualized and cloud environment. Additionally, it can help individuals with basic knowledge of networking and operating systems, who are interested in modern cloud technology, several virtualization techniques, and development and deployment of applications. This Study will touch upon topics like cloud architecture, types, virtualization techniques in cloud computing, which are needed to support new generation of real-time-managed IT service use cases in the cloud computing industry.

Table of Content

Section No.	Topic
1	Introduction
2	Features of Cloud Computing
3	Types of Cloud Services
3.1	Infrastructure as a Service
3.2	Platform as a Service
3.3	Software as a Service
4	Cloud Computing Deployment Models
4.1	Public Cloud
4.2	Private Cloud
4.3	Hybrid Cloud
5	Why not Cloud Computing?
6	Virtualization
7	Brief History of Virtualization
8	Physical Machine vs Virtual Machine
8.1	What is Physical Server?
8.2	What is Virtual Machine?
9	Characteristics of Virtualization
9.1	Increased Security
9.2	Managed Execution
9.3	Portability
10	Types of Virtualization
10.1	Process Level Virtualization
10.1.1	Language Level Virtualization
10.1.2	Application Level Virtualization
10.1.3	Operating System Level Virtualization
10.2	System Level Virtualization
10.2.1	Hardware Assisted Virtualization
10.2.2	Full Virtualization
10.2.3	Para Virtualization
10.2.4	Partial Virtualization
11	Virtual Machine Monitor
11.1	Processor Virtualization
11.2	Memory Virtualization
11.3	Device and I/O Virtualization
12	VMM Storage Management
13	Live Migration
14	Dark Side of Virtualization
15	Cloud Computing and Virtualization
16	Improvements to Cloud Computing and Virtualization
16.1	Docker and X11 protocol
16.2	Edge Computing
17	Conclusion
18	References

1. Introduction

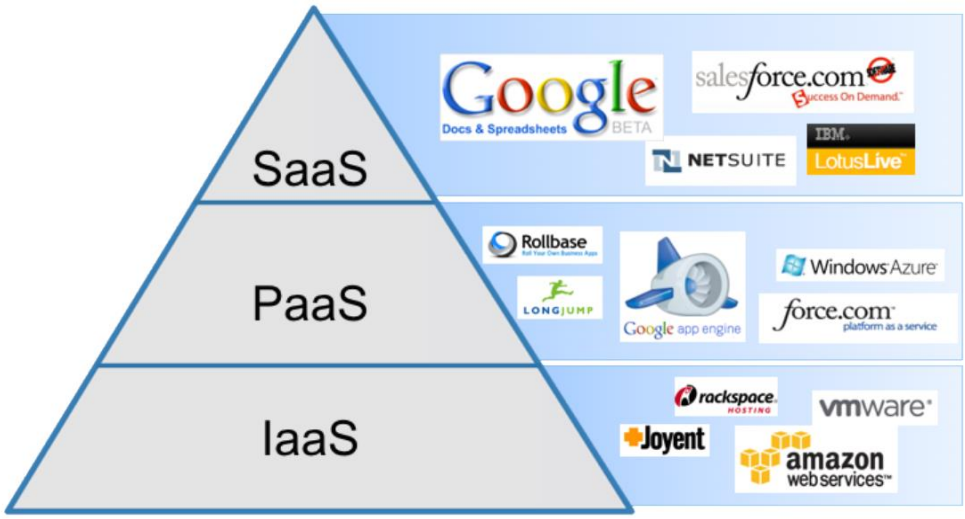
Like clouds in nature are the collection of water molecules, the 'cloud' in cloud computing is the collection of networks. Cloud computing is a model which enables on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services). Instead of setting up a whole new physical infrastructure the users ordinarily prefer a mediator provider for the service over internet in cloud computing. They typically are charged only for the cloud services they use, which lowers their operating costs, helps them run their infrastructure more efficiently, and scale it according to their business needs.

2. Features of Cloud Computing

Services handled by latest and upgraded components of cloud heavily reduces the requisition of hardware and software at the user side. All they need to have is a web browser to access Internet. Following are key characteristics of cloud computing:

- On-demand self-service
- Secure and Reliable network access
- Resource pooling and Rapid elasticity
- Measured service
- Reduced pricing
- Quality of service

3. Types of Cloud Services



Most cloud computing services fall into three broad categories:

- Infrastructure as a service (IaaS)
- Platform as a service (PaaS)
- Software as a service (SaaS)

3.1. Infrastructure as a Service (IaaS)

Infrastructure as a Service, abbreviated as IaaS, provides the basic building blocks for cloud IT. IaaS provides resources in the form of storage, network, operating system, hardware, and storage devices on demand. Users can access the services using internet. For example, the user can create virtual machines(VMs) to log in to the IaaS platform; install operating systems in each VM; deploy middleware, such as databases; create storage buckets for workloads and backups; and install the enterprise workload into that VM. Some examples of independent IaaS providers are Amazon Web Services (AWS) and Google Cloud Platform (GCP).

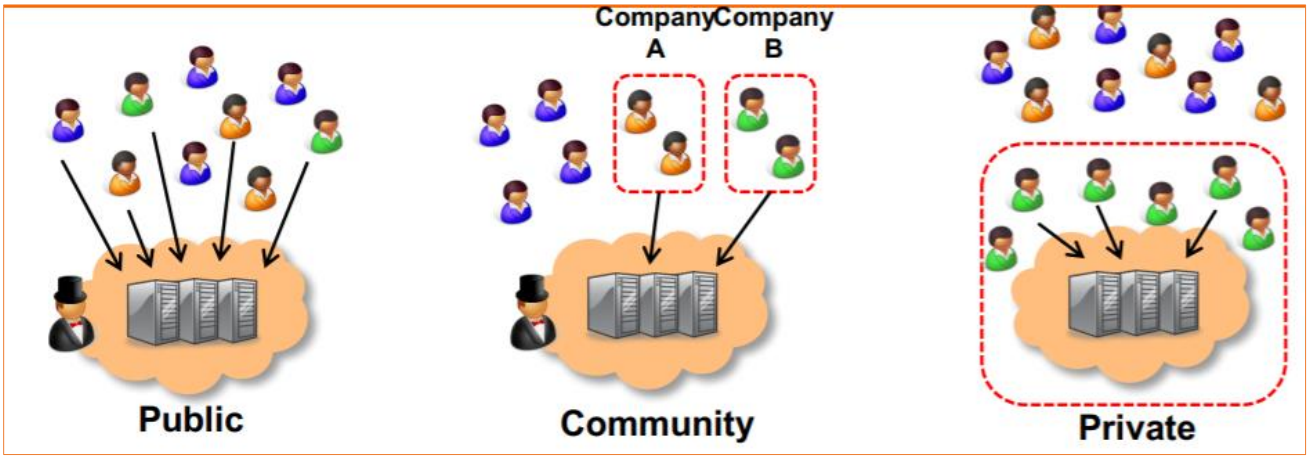
3.2. Platform as a Service (PaaS)

Platform as a Service, abbreviated as PaaS, allows organizations to focus on the deployment and management of their applications by removing the need to manage the underlying infrastructure (usually hardware and operating systems). Under PaaS, a development environment or platform is provided as a service to the consumers upon which they deploy their own software and coding. Platform provided includes a predefined composition of operating system and application server to obtain the management capacity for the applications. This helps developers build applications efficiently as there is no load of resource procurement, capacity planning, software patching, maintenance, or any of the other undifferentiated heavy lifting involved in running your application. Google App Engine, Windows Azure, Heroku are some examples of PaaS providers.

3.3. Software as a Service (SaaS)

Software as a Service, abbreviated as SaaS, is a model of software delivery and licensing. SaaS allows users to access software online via subscription, rather than purchasing and installing on individual computers. SaaS users do not have to worry about the software or hardware maintenance and update. Service provider run and manages the complete product including the underlying infrastructure. A web-based email is a common example of a SaaS application where user can send and receive email without worrying about the management of additional features to the product or maintenance of servers and operating systems on which the email program runs. Microsoft Office 365, Google Apps are examples of SaaS providers.

4. Cloud Computing Deployment Models



No cloud is same and therefore several different models, types, and services have evolved to help offer the right solution for customer needs. Cloud deployment models categorizes cloud environment based on proprietorship, size, and access and describes the nature and purpose of the cloud. There are mainly three different ways to deploy cloud services

- Public Cloud
- Private Cloud
- Hybrid Cloud

4.1. Public Cloud

Public clouds are owned and operated by a third-party cloud service providers. Microsoft Azure, AWS are examples of public cloud. The service providers manage all the hardware, software and supporting infrastructure in this model and deliver their computing resources like storage and servers, over the Internet. User access these services and manage their account using a web browser. They are charged only for the services they consumed.

4.2. Private Cloud

A private cloud refers specifically to resources used by one organization or business. A private cloud can either be hosted by a third-party service provider (Virtual Private Cloud) or can be physically located at company's on-site datacenter. The services and infrastructure are always maintained on a secure private network which makes it easier for an organization to customize its resources to meet specific IT requirements. Government agencies, financial institutions, any other mid-to large-size organizations are main users of private clouds.

4.3. Hybrid Cloud

Often called "the best of both worlds," hybrid cloud deployment is a way to connect on-premises infrastructure, or private clouds with public clouds. In the hybrid cloud, each cloud can be managed independently however data and applications can move between on-premises infrastructure, or private clouds and public clouds for greater scalability, flexibility, and more deployment options. For example, user can use the private cloud for sensitive, business-critical operations and the public cloud for high-volume, lower-security needs. "Cloud bursting" is also an option in hybrid cloud
in which an application or resource runs in a private cloud until there is a spike in demand, at which point it bursts into a public cloud utilizing additional computing resources.

5. Why not Cloud Computing?

There are some limitations with cloud computing implementation.

- *Vulnerability to attacks*: Storing confidential data on cloud is not always safe even best organizations have suffered security breach which is a potential risk in the cloud as well. Although advanced encryption techniques are used it can be hacked sometimes.
- *Network connectivity dependency***:** To reap benefits of Cloud Computing organization needs to have reliable and consistent internet service as well as fast connection and bandwidth. Any fault in network connections leads to downtime.

- *Technical Issues and Downtime*: Due to Loss of power, poor Internet connection, data centers going out of service for maintenance cloud providers face technical outages leading to a temporary downtime in the cloud service.

6. Virtualization

Virtualization is the inseparable element of cloud computing. Without virtualization, cloud computing would leave the data uncontrolled, unstable, and unsafe. Virtualization allows unification of parallel and distributed computing, which then on higher levels enables harness of networked and heterogeneous computational nodes and present them as a unified resource. Virtualization strategies allow the infrastructure to be completely virtualized and controlled. In computing, virtualization means creation of virtual version of a resource or device, such as a server, storage device, network or even an operating system which is then divided into one or more execution environments by the framework.
It is the single most effective way to boost agility and efficiency for all size businesses while reducing IT expenses.
Following are few benefits of Virtualization in Cloud Environment:

- Automatic Protection of Applications from System and Server Failures
- Maximized Uptime
- Hassle-free Migration of Workloads as Needs Change
- Firewall and Security
- Resource Optimization and Smooth IT Operations
- Protect Investment in Existing, Legacy Systems

7. Brief History of Virtualization

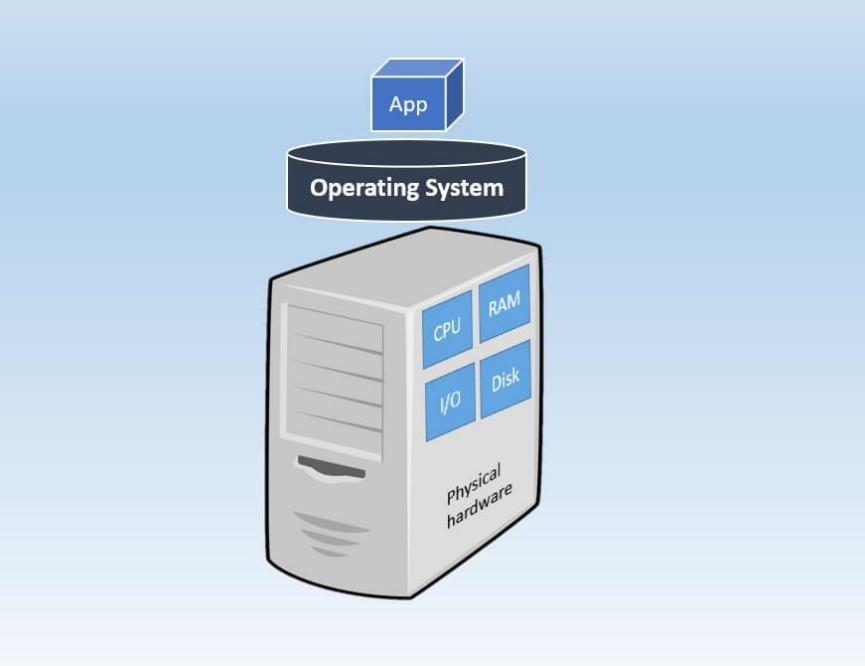
The idea of virtualization originated in the late 1960s and early 1970s during the mainframe days, when IBM was developing robust time-sharing solutions. Time-sharing solutions aimed to increase the efficiency of users and expensive computer resources by sharing the usage of computer resources among a large group of users. This model represented a breakthrough in computer technology: it significantly dropped the cost of providing computing capability and made it possible for organizations, individuals, to use a computer without owning one. Virtualization is driven by similar reasons today for industry standard computing. Servers are so overloaded that most workloads are unable to utilize the capacity in a single server effectively. Virtualization is the best way to improve data center management and resource utilization at the same time. Virtualization techniques are used by Data centers today to abstract physical hardware, create large aggregated pools of logical resources consisting applications, networking, disks, file storage, memory, CPUs and provide computing resources to customers or users in form of consolidated, agile, flexible and scalable virtual machines. Even though the technology has evolved and uses cases have changed the core meaning of virtualization remains the same: to build a computing environment to run multiple independent systems at the same time.

8. Physical Servers vs Virtual Machines

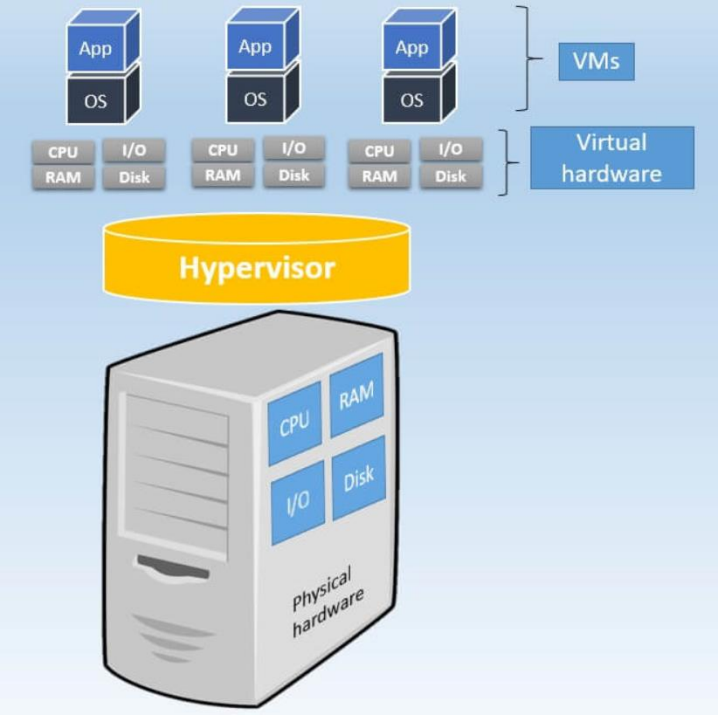
8.1. What is a Physical Server?

A physical server is a single-tenant computer server which means that a specific physical server is owned by a single user. The components and resources of the server are not shared among multiple users. Each server

consists of memory, processor, network connection, hard drive, and single operating system (OS) to run programs, applications, and to control hardware. Software is tightly coupled with underlying hardware in a physical server machine. A physical server is also known as a 'bare-metal server' which is large and heavy due to powerful processing components that it contains.



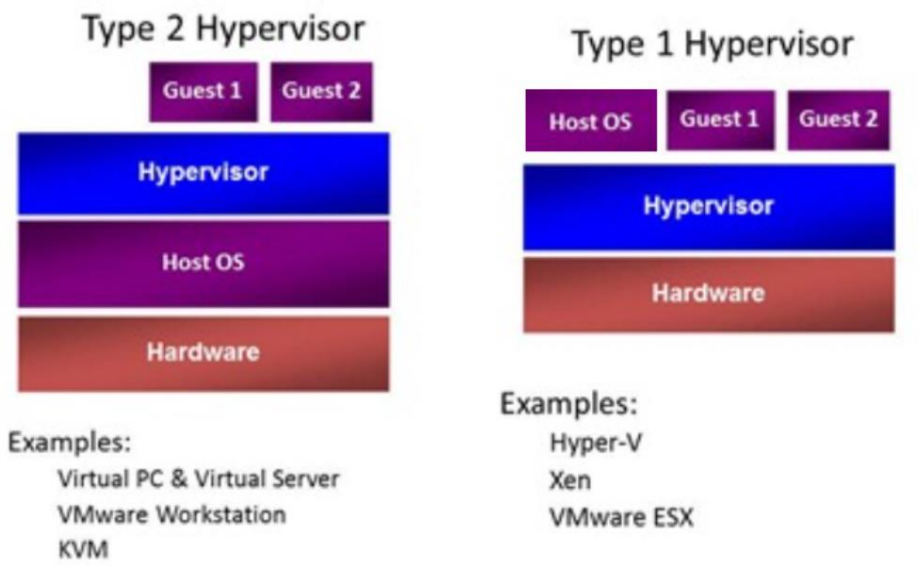
8.2. What is a Virtual Machine?



A virtual machine (VM) is an emulation of actual physical computer. The components of a physical server are virtualized and shared among all VMs running on it. Since, multiple VMs can run on the same physical

hardware a virtual server is also known as a “multi-tenant” server. A hypervisor, also called virtual machine monitor (VMM) manages the complex architecture of a virtual machine. It is a computer software, firmware or hardware that creates and runs one or more guest virtual machines on a single host computer. User can load their OSes and server applications on top of the virtualized hardware. The hypervisor provides guest operating systems with a virtual operating platform and manages their execution which means that a virtualized hardware resources can be shared among multiple instances of a variety of OSes. For example a single physical x86 machine can run all Linux, Windows, and macOS instances. Hypervisor allocates and reallocates the computer resources like CPU, memory, and storage between existing guests or new VMs. Some operating-system level components such as memory manager, process scheduler, i/o stack, device drivers, security manager, network stack and so on are required by hypervisors for their operation. Generally, there are two types of hypervisors.

- Type I
- Type II



Type I hypervisors, also known as ‘bare metal,’ run directly on top of the host’s hardware. Hypervisor emulates the ISA interface exposed by the underlying hardware for management of guest operating systems. They interact directly with the underlying hardware using the ISA interface.

Type II hypervisors, also known as ‘hosted VM’, run as a computer program on an operating system. They need support of operating system to provide virtualization services. They interact with the underlying host OS through ABI and emulates the ISA of virtual hardware for interaction with guest operating systems.

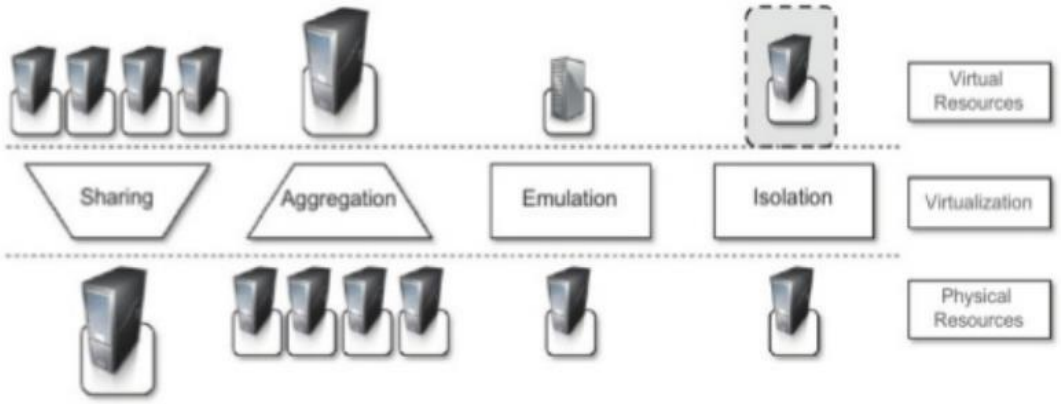
9. Characteristics of Virtualization

9.1. Increased Security

Virtualization opens new possibilities for delivering a controlled, secure execution environment, especially dealing with untrusted code. Guest programs performs their operations against the virtual machine, which then are translated and applied to the host programs. The activities of the guest programs are controlled and filtered by a virtual machine manager which prevents some harmful operations from being performed on the

host machine. Through virtualization techniques the resources exposed by the host can be protected or hidden from the guest thereby increasing the security.

9.2. Managed Execution



Virtualization not only delivers increased security of the execution environment, but also allows implementation of a wider range of features like sharing, aggregation, emulation, and isolation.

- *Sharing*: Virtualization provides the ability to create multiple computing environments within the same host, which leads to full exploitation of the capabilities of a powerful host, which were often underutilized. Virtualized data centers use security feature to reduce number of active servers and power consumption.

- *Aggregation:* In addition to sharing, virtualization also allows grouping of multiple heterogeneous hosts which then are presented to guests as a single unified virtual host. Aggregation is naturally implemented in the middleware layer of the cloud architecture for distributed computing. For example, Cluster management software which harness the physical resources of a homogeneous group of machines and represent them as a single resource.

- *Emulation:* This is one of the key features of virtualization which allows controlling and tuning of the execution environment exposed to guests. A completely different environment with specific characteristics required for a guest program, which are not present in the host machine can be emulated by the virtualized layer, which ultimately is a software program. For example, Old and legacy software that does not meet the requirements of currents systems can be run either by emulating the required hardware architecture or within a specific operating system sandbox, such as the MS-DOS mode in Windows 95/98. Arcade-game emulator is another example that allows us to play arcade games on a normal personal computer. Emulation is very useful in testing environments where guest programs needs to be validated on different platforms or architectures and such wide range of options are not available during development.

- *Isolation*: Through virtualization all guests (operating systems, applications, other entities) executes within a separate environment. They interact with the underlying host resources through abstraction

layer. Isolation ensures increased security as it allows multiple guests to run on the same host without interference with each other. This separation between the host and the guests allows the VMM to control and filter the activities of the guests and prevent the host from harmful operations.

- *Performance Tuning:* With considerable advancement in hardware and software supporting virtualization it is easy to tune the properties of the resources exposed to the guest through virtual environment. This feature provides capability to control the performance of guest and implement quality-of-service (QoS) infrastructure effectively. For example, only a fraction of the host memory or the maximum frequency of the VM processor can be exposed to guest OS using software-implementing hardware virtualization techniques.

Managed execution posses' other advantages also, like easy capturing of the guest program state, persisting and resuming its execution. This, for example, is used in virtual machine migration technique, in which VMM such as Xen Hypervisor controls the operation of guest OS by stopping its execution, moving its virtual image into another machine and resuming the execution in a completely transparent manner. This is an important technique in virtualized data centers used for optimizing efficiency in serving application demands.

9.3. Portability

The application of portability characteristic varies according to the type of virtualization considered.

- In the case of a hardware virtualization solution, the guest can be safely moved and executed on top of several VMs as a packaged virtual image. A virtual image is a general proprietary format which requires specific VMM for execution.
- In the case of programming-level virtualization solution, the binary code representing application components do not need recompilation on any corresponding Virtual machine. For example, JVM, .NET. Since, one version of application can run on multiple platforms without changes, this makes development cycle more flexible and deployment of programs or applications convenient.

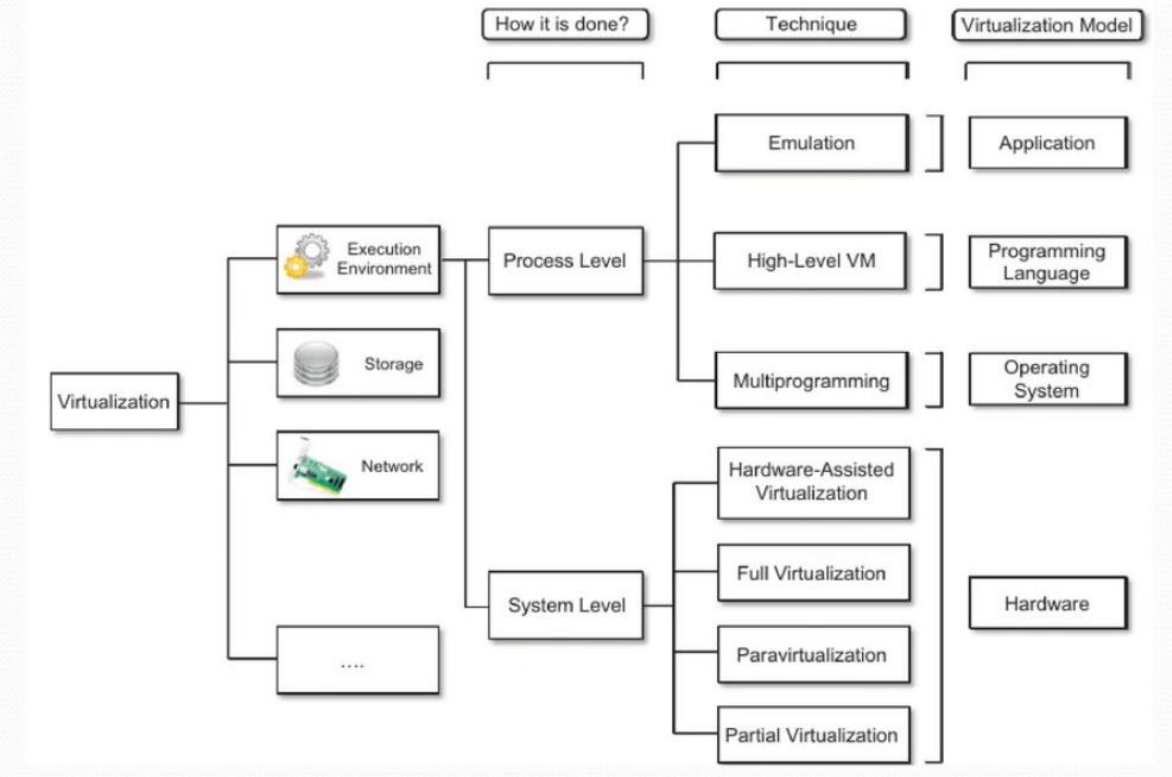
Hence, portability provides availability of a system always with user which is ready to use if the required virtual machine manager is present.

10. Types of Virtualization

Virtualization techniques are mainly used to emulate of execution environments, storage, and networks. Execution virtualization techniques further has two broad categories considering the type of host they require.

- *Process-level techniques:* These techniques have full control of the hardware and are implemented on top of the existing OS.
- *System-level techniques*: These techniques require minimum support from existing OS and are implemented directly on the hardware.

Hardware level, language level, application level, and operating system resources virtualization fall under these two categories which offer guest different type of virtual computation environment.



10.1. Process Level Virtualization

Process-level virtualization techniques allows existing software to adapt to the operating system environment by modifying application instructions at runtime, including resource contention and other dynamic events.

10.1.1. Language Level Virtualization

Programming language-level virtualization provides a uniform execution environment across different platforms. In this technique virtual machine executes the complied byte code and produce machine code for abstract architecture. These Virtual machines consists of simplified underlying hardware instruction set and provides some high-level instructions which maps few features of complied language for them. During runtime, on the fly or jitted5 the byte code can either be complied or interpreted against the underlying hardware instruction set. Virtual machine provided enables the execution of the complied byte code on any OS or platform. This ease the development and deployment process. Moreover, these virtual machines enable managed execution limiting direct access to memory, increased security by filtering the I/O operations, and supports sandboxing of applications. Both Java and .NET are examples of language level virtualization. However, Virtual machine programming languages have trade off with performance. Languages complied against the real architecture performs faster. But this difference is getting negligible with high computer power available on average processors.

10.1.2. Application Level Virtualization

Application level virtualization, also called Emulation, is a virtualization technique that encapsulates computer programs from the host operating system on which they are executed. With this technique, applications are not installed in the local machine but run as though they were. During runtime, the application appears

directly interfacing with the original OS and resources managed by it, however it can be sandboxed and isolated to various degrees.
Application level virtualization generally includes emulation of file systems, libraries, and operating system component. Also, following emulation strategies can be used to execute program binaries complied for different hardware architectures.

- *Interpretation*: In this technique emulator interprets every single source instruction for executing native ISA instructions. Since, each instruction is emulated interpretation has more overhead and poor performance but low startup cost.
- *Binary translation*: In this technique equivalent functions are used to convert every single source instruction to native instructions. Also, instructions are cached and reused after translation of a block of instructions. Since, translated instruction blocks are directly executed, binary translation has better performance but high initial cost.

CrossOver is an example of application level virtualization, which enables execution of Windows applications directly on the Mac OS X operating system.

10.1.3. Operating System Level Virtualization

Operating system level virtualization is a virtualization paradigm which involves altering of an OS to provide separated execution environments for applications, that are managed concurrently on a single computer. Unlike hardware virtualization, this technique involves no VMM or hypervisor. Single OS kernel allows creation of multiple isolated user space instances. These instances include containers (Docker), Zones (Solaris), virtual private servers (OpenVZ), partitions, virtual environments (VEs), virtual kernel, or jails which appears to be real computers for programs running in them. A program which runs on an ordinary OS system can access all resources (connected devices, files and folders, network shares, CPU power, hardware) of that computer. But programs which run inside container can access resources assigned to that container. It is responsibility of the OS kernel to share the system resources among the instances for reducing the impact of instances on each other. In addition to isolation, the kernel also provides resource-management features to limit impact of one container's activities on other. There is no need for emulation and applications use OS system calls directly, thereby limiting the overhead. Operating systems which supports OS virtualization are general-purpose and timeshared, they have the capability to provide stronger namespace and resource isolation. OS virtualization can be considered as advance implementation of the standard chroot mechanism on Unix-like OS, in which the apparent root folder for the current running process and its children can be changed. Server consolidation is an example of OS virtualization implementation where multiple application servers share the same technology: operating system, application server framework, and other components.

10.2. System Level Virtualization

System-level virtualization, also called Hardware-level virtualization is a virtualization technique which emulates underlying host machine resources and creates an abstract execution environment for guest OS. In this virtualization model, the physical computer hardware or machine is represented as host, the emulation as virtual machine and the hypervisor as virtual machine manager (refer section 8.2). The hypervisor which enables the abstraction of the underlying physical hardware is a software program or a combination of software and hardware. While process virtual machines provides ABI to VMs, system level virtualization expose ISA to VMs. Depending upon the behavior of underlying hardware expected, hardware-level virtualization includes several different techniques.

10.2.1. Hardware-assisted virtualization

Hardware-assisted virtualization is the use of computer's hardware to provide architectural support for creation of virtual machine manager, which allows guest OS to run in complete isolation. This technique was originated in IBM System/370. Intel VT, AMD V introduced extensions to the x86-64-bit architecture are examples of hardware-assisted virtualization. These extensions aim to reduce performance cost experienced by emulation of x86 with hypervisors which was significantly costlier during software emulation of x86 hardware.

10.2.2. Full virtualization

Full virtualization is the technique in which virtual machine manager provides a complete emulation of the entire underlying host hardware. This allows guest operating system to run directly on the top of virtual machine without any modification, as it were to run on the raw hardware. Full virtualization leads to complete isolation, increased security, emulation of different architectures, sharing of same platform among different systems. Privileged instructions change the state of the resources exposed by the host therefore to overcome the challenge of interception of privileged instructions, VMM must contain them. Full virtualization would be achieved if a virtual environment is provided for all the instructions. This leads to performance and technical implementation overhead. However, through hardware-assisted virtualization, with a combination of hardware and software, restricting potential harmful instructions to be executed directly on the host full virtualization is implemented efficiently.

10.2.3. Paravirtualization

Paravirtualization techniques provides a software interface to the virtual machines which is slightly modified from the underlying hardware–software interface. To achieve this the guest operating system is recompiled, installed inside a virtual machine, and run over the hypervisor program operating on the host OS. To take advantage of paravirtualization the guest operating system must be explicitly ported for the para-API, this is the reason that paravirtualization was mostly implemented in opensource environment. Xen used this technique to provide solutions for Linux-based OS ported on Xen hypervisors for operation. The operating systems which cannot be modified and ported can make use of ad hoc device drivers to implement paravirtualization. The drivers remap the execution of critical instructions to the para-API provided by the hypervisor. For example, Xen paravirtualization-aware device drivers (Xen Windows GPLPV) provide solutions to run Windows based virtual guest on Xen hypervisor. The main intent of this virtualization technique is to enable the execution of performance-critical operations directly on the host, which are significantly more difficult to run in a virtualized environment than in non-virtualized one. Special well defined 'hooks' are provided by paravirtualization which allows guest and host to request and acknowledge these performance critical tasks. This relocation of critical tasks execution from virtual to host domain makes VMM light and simple.

10.2.4. Partial virtualization

Partial virtualization, as name defines emulates underlying hardware partially. In this scenario the entire operating system cannot run in virtual machine or in complete isolation, however some or many applications can. It is easier to execute than full virtualization. Partial virtualization is preferred more in cases when multiple users share same computer resources. For example, address space virtualization in time-sharing systems is key form of partial virtualization, where multiple applications and users share the same hardware resources (disk, processor, and network) running concurrently in independent address space.

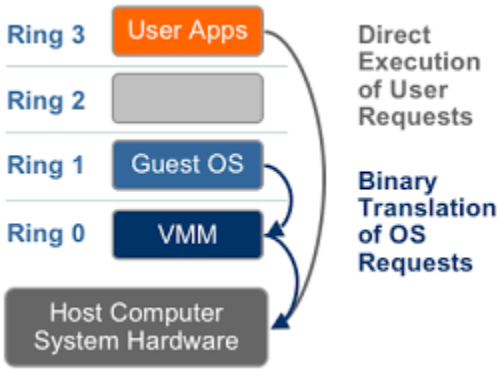
11. Virtual Machine Monitor (VMM)

According to classic definition of VMM stated by Popek and Goldberg'74, A virtual machine is an efficient, isolated duplicate of the real machine. A virtual machine monitor has three essential characteristics. First, the VMM provides an environment essentially identical to the original machine for execution of programs, second, programs running in this environment show only minor decrease in speed, and last, the VMM is in complete control of system resources. Therefore, providing illusion of multiple machine, retaining control of physical machine are few implementation goals of a VMM, which leads to virtualization of subsystems like processor, memory, I/O devices.

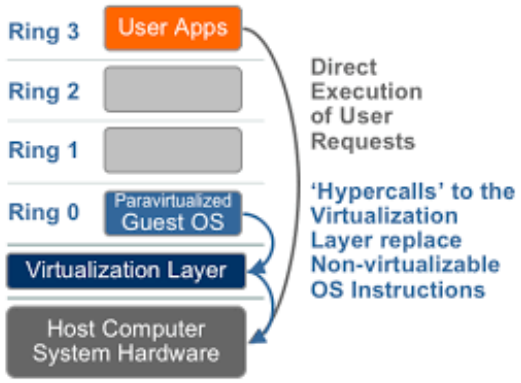
11.1. Processor Virtualization

To manage access to computer resources the x86 architecture provides a hierarchy of privileges, called ring of security (Ring 0, Ring 1, Ring 2 and Ring 3); Ring 0 represents most privileged instruction and Ring 3 least privileged. OS must run in Ring 0 as it requires direct access to hardware and memory and all user-level applications run in Ring 3. The virtualized layer (VMM) placed in x86 architecture virtualization is expected to operate in most privileged Ring 0 mode for creation and management of VMs. However, the inability to trap and emulate all sensitive and privileged instructions requests from Guest OS at runtime made x86 architecture virtualization difficult. To handle this issue and to virtualize the CPU on the x86 architecture, three techniques were proposed.

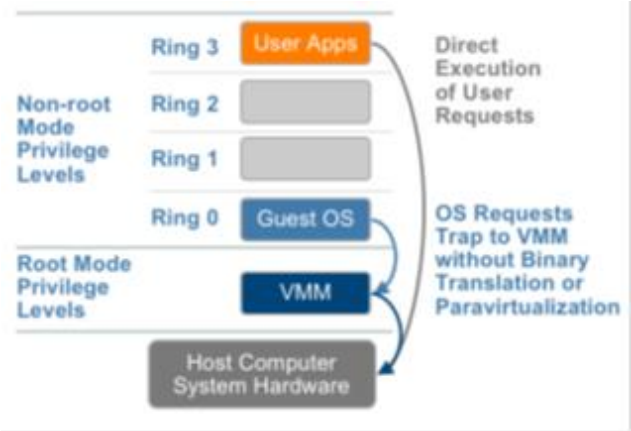
- *Full virtualization using binary translation and direct execution*: In this technique VMM provides complete CPU emulation to handle and execute privileged instructions of unmodified guest OS kernels. The sensitive OS calls are trapped using binary translations.



- *OS assisted virtualization or paravirtualization:* The kernel of guest OS gets modified in this technique to operate on hypervisor. All privileged operations which must run in Ring 0 are replaced with hyper calls (calls to hypervisor). The hypervisor then performs the instruction, or task on behalf of the guest kernel.



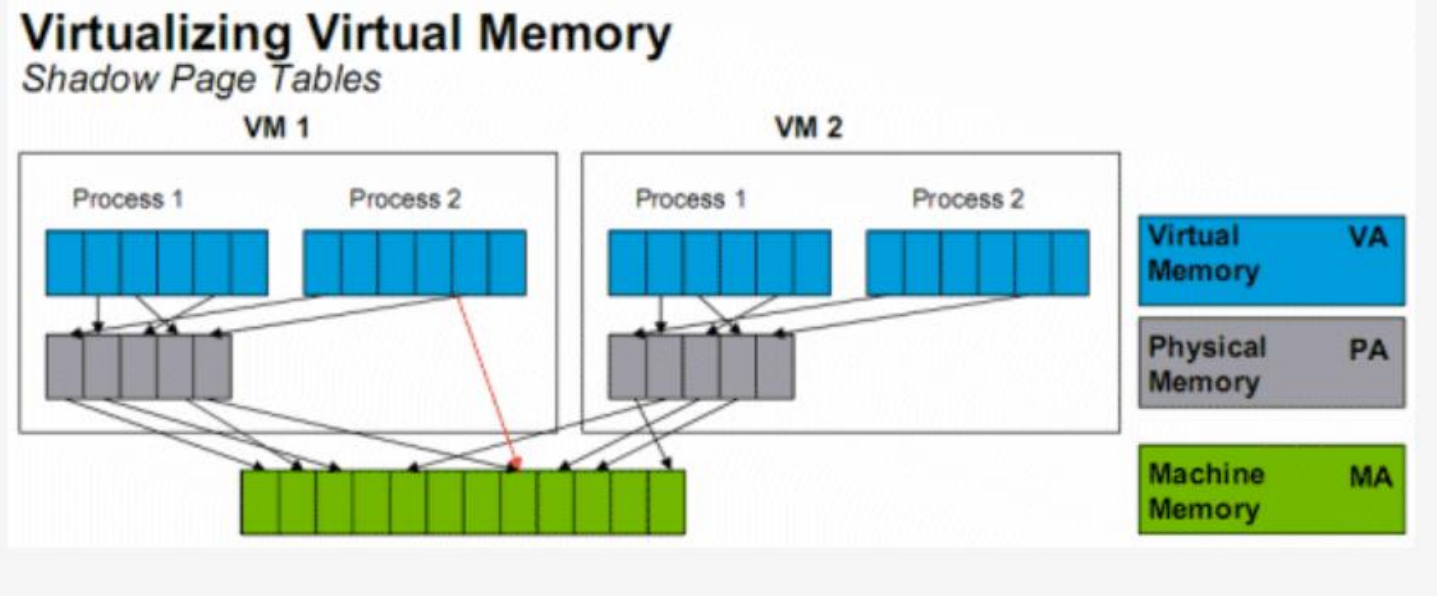
- *Hardware assisted virtualization:* This technique allows the VMM to run in a new root mode below ring 0, leaving Ring 0 available for unmodified guest OS. Since, the sensitive and privileged calls are automatically trapped to hypervisor, its removes need for binary translation or paravirtualization.



Processor virtualization also includes multiplexing of CPU. A hypervisor multiplex full operating system which leads to representation of multiple copies of the same processor among VMs. To manage then execution of instructions of multiple VMs on this shared CPU, CPU schedulers comes into picture. Priority based scheduling, Proportional-share scheduling (weighted fair scheduling), task aware VM scheduling are few algorithms used by VMM for scheduling utilization of resources among several VMs. Linux CFS, Xen (BVT, SEDF, Credit) are examples of some schedulers.

11.2. Memory Virtualization

Memory Virtualization is a virtualization technique which allows sharing of physical system memory with several VMs dynamically. This technique is like traditional OS virtual memory management technique which use memory management unit (MMU) and a translation lookaside buffer (TLB) for management of page tables, which used to map virtual page numbers with physical page numbers (physical memory frames). In virtualization to support each guest OS an emulated MMU instance is required. The guest OS handles the mapping of guest virtual memory to the guest physical memory addresses and the VMM performs mapping of guest physical mapping with actual(host) machine(physical) memory using shadow page tables.



Like traditional OS, to reduce time taken to perform two level of translation, the VMM also uses TLB hardware to map the virtual memory directly to the host machine memory. The shadow tables are updated every time guest OS changes its virtual memory to its physical memory mapping. MMU virtualization posses some overhead on all virtualization techniques which can be reduced by solutions offered by second generation hardware assisted virtualization.

11.3. Device and I/O Virtualization

Device and I/O virtualization refers to management and routing of I/O requests between virtual devices and the shared physical hardware. It can be achieved in multiple ways:

- *Direct I/O assignment:* This technique allows a guest to directly DMA to/from host memory. For example, Intel VT-x allows guest drivers to directly write in register I/O device (DMA descriptor) and Intel VT-d allows actual host devices to directly access the memory space of a VM. Though this technique increases the throughout as it bypasses the need of emulation in VMM, but it is limited in scalability and flexibility as physical devices are coupled to VM tightly.
- *Software based I/O virtualization and management:* This technique provides more features and flexibility. For example, with help of networking (virtual NICs and switches) virtual networks are created between VMs, multiple physical NICs can be represented as a single resource, better fault tolerance can be achieved by migrating VMs on different systems maintaining their MAC addresses. With software-based virtualization, VMM emulates the host hardware and provide each VM with a standardized set of virtual devices. This leads to virtual machine standardization and portability across platforms.

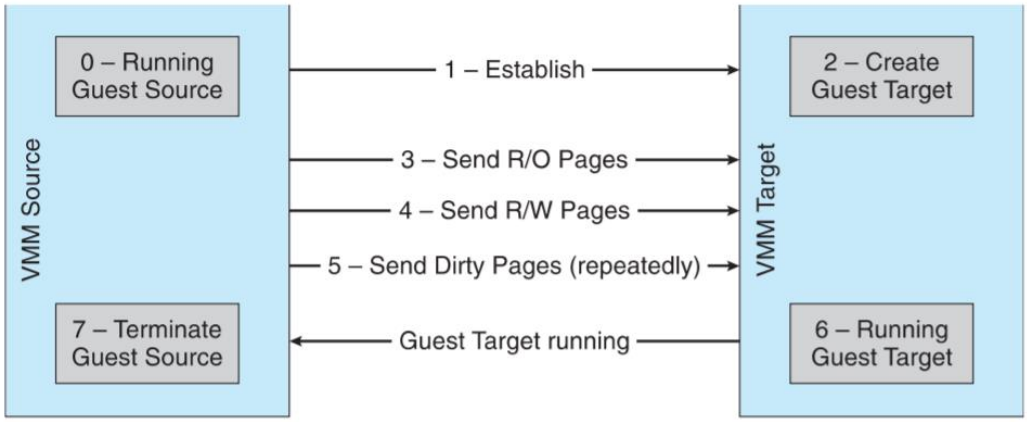
12. VMM Storage Management

VMM provides both boot disk and general data access to the VMs. Since, it manages several guest virtual machines concurrently standard disk portioning is not enough. Type 1 hypervisors store guest root disk and VM configuration information as disk image within the file system provided by VMM, whereas type 2 hypervisor store this information as files in file system provided by the host OS. To create new guest the required files are duplicated. To move guest the respective files are transferred from one file system to another. VMM provides physical to virtual mapping by converting native disk blocks to VMM format and also

handles virtual to physical mapping by virtual format to native or disk format. VMM also provide access to network attached storage, disk images, disk partitions and so on.

13. Live Migration

VMM features provides the ability to move running virtual machine(guest) or application from one physical machine to another without disconnecting the client or application. Memory, storage and network connectivity of original guest machine is transferred to the destination. Live migration is very useful feature of VMM which provides ease of resource management, maintenance downtime and increases overall fault tolerance. Live migration works according to the following steps:



1. Connection between the source VMM and target VMM is established
2. A new guest (VCPU) is created by the target VMM
3. All read-only guest memory pages are sent to target by source VMM
4. All read-write pages are sent to the target by source VMM, marking them clean
5. Step 4 is repeated by source as while transferring some pages were probably modified by guest and were dirty then.
6. The guest is freeze by source VMM when cycle of step 4 and 5 becomes very short. The final state of VCPU, final dirty pages, and other details of guest are sent by source in this step, and the target VMM is informed by the source to start running the guest.
7. On successful execution of guest on target VMM acknowledgement is sent to source which then terminates guest on it.

14. Dark Side of Virtualization

Virtualization has few downsides also as mentioned below:

- *Performance Degradation:*

Guests systems experience increased latencies because of the presence of abstraction layer between the guest. Incase of bare metal where the hypervisor emulates the entire underlying hardware, performance degradation can be caused due to overhead of virtual processors maintenance, privileged instructions (trap and simulate privileged instructions) support, console functions and so on. Virtual machine manager execution and scheduling with other applications, binary translation and interpretation, execution, and filtering of managed applications at the runtime environment, access to memory and other physical resources can lead to

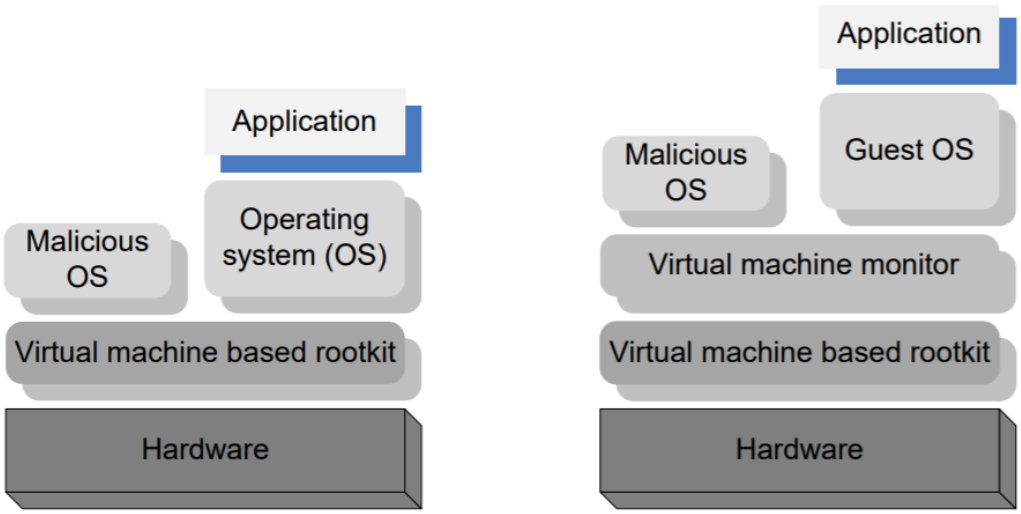
performance degradation. However, some virtualization techniques like paravirtualization, containerization improves the performance of guest program by reducing most of its execution to the host unaltered.

- *Inefficiency and degraded user experience:*

The abstraction layer introduced by virtualization management software can lead to suboptimal utilization of the host. For example, device drivers: only a subset of the features presented in host can be mapped by the default graphic card provided by the virtual machine causing degraded user experience.

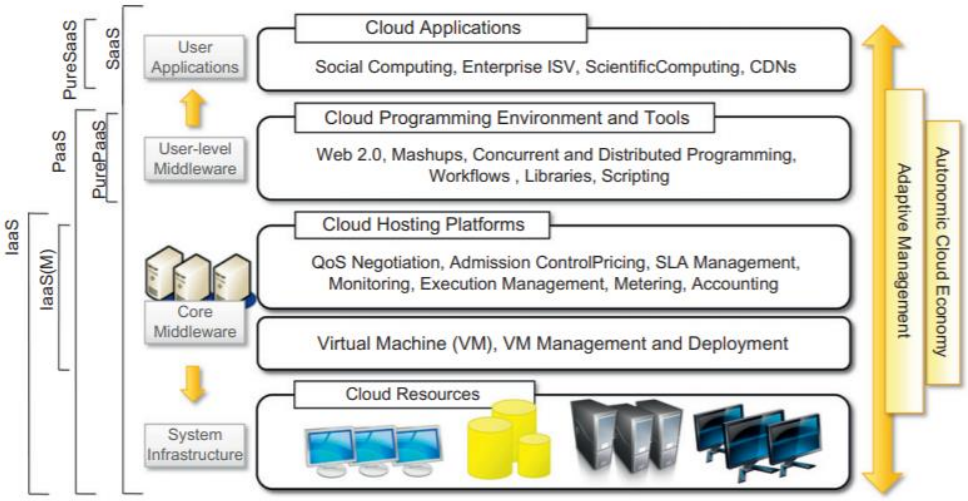
- *Security holes:*

The capability provided by virtualization to emulate a host in a completely transparent manner, opens door to a new and unexpected form of phishing. A rouge VMM, a VMBR (a Virtual-Machine Based Rootkit) is inserted between the underlying host and OS, this Rootkit – malware possess a privileged access to a system. The VMBR enables a malicious OS to preload itself before the guest OS. The malicious OS then act as a thin VMM, it controls and manipulates the guest OS to extract sensitive information. The protection of VMBR malware makes the malicious OS invisible to the guest OS and the software running beneath it. BluePill and SubVirt are examples of such malware.



15. Cloud Computing and Virtualization

All types of services provided by cloud relies on a distributed infrastructure which is either owned by the provider or rented from a third party. A datacenter, a collection of clusters, or a collection of distributed desktop PCs, workstations, and servers composing a heterogeneous distributed system can form a cloud infrastructure layer. Different layers (a virtual machine manager, a development platform, or a specific application middleware) are then stack on top of this layer. Virtualization decouples the infrastructure layer of cloud architecture from higher layers which leads to the evolution of the physical hardware resources on their own without disturbing the software stacks running on top. The cloud computing architecture can be represented as a layered architecture which is composed of the following layers:



- *System Infrastructure*, also called physical Infrastructure builds the bottom of the stack. Hardware Virtualization is implemented in this layer which includes VM, virtual networking and virtualized storage. Hardware level virtualization makes use of Paravirtualization, Partial virtualization. Virtual machines migrations techniques to improve utilization of resources, load balancing of processing nodes, isolation of applications, tolerating the faults in virtual machines, to increase the portability of nodes and to rise the efficiency of the physical server. Hypervisors used manage the pool of resources and represent them as a collection of VMs. VM technology with storage and network virtualization makes the physical infrastructure completely virtualized and controlled. Moreover, techniques like programming level virtualization creates a portable execution environment for applications to be run and controlled. This also implies that in the cloud environment applications are developed using specific technology generally Java, .NET, or Python.

- *Core Middleware*, also known as software management layer, is responsible for managing underlying virtualized infrastructure. It contains a scheduler which handles the allocation and execution of virtual machine instances by interacting with other components like VM Pool manager, VM repository, reservation, monitoring, QoS/SLA management, accounting billing components for taking decisions according to defined scheduling policies. The IaaS services provided by the cloud includes combination of cloud hosting platforms and resources, which is supported until this layer generally. Some IaaS solutions include both the management and infrastructure layer. However, some includes only the management layer which is often integrated with other IaaS solutions.

- *User-Level Middleware* offers developers a platform including Web-based interfaces, command-line tools and frameworks for concurrent and distributed programming. Developers use these APIs to develop their applications specifically for the cloud. These advanced web interfaces allow integration of platform solutions into the software management layer. Some examples of such solutions are OpenNebula, OpenStack. Cloud applications which do not need to run in the virtualized environment for the reason of performance overhead use the frameworks provided in this layer. Since, the service offered is a development platform, it is also called the Platform-as-a-service (PaaS). A pure PaaS solution provides only the services included in this layer. However, general PaaS solutions include infrastructure as well.

- *User Applications* The top layer known as User Applications contains services delivered at the application level. In most cases these are Web-based applications that rely on the cloud (underlying components and technologies) to provide service to end users. The horsepower of the cloud provided by IaaS and PaaS solutions allows independent software vendors to deliver their application services over the Internet.

16. Improvements to Cloud Computing and Virtualization

As discussed in the above sections' hardware virtualization possess performance overhead. The overhead mainly derives from the following aspects:

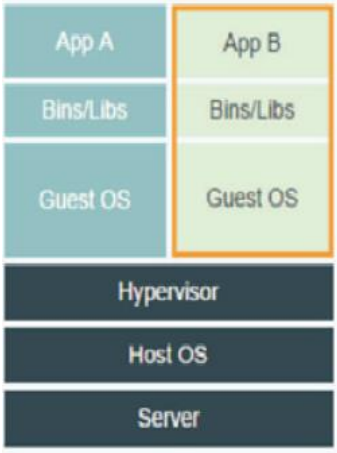
- The overhead of running different applications on the backend cloud servers.
- The overhead of transferring the information data from back-end to users.

To overcome the above shortcomings following improvements can be made to cloud OS:

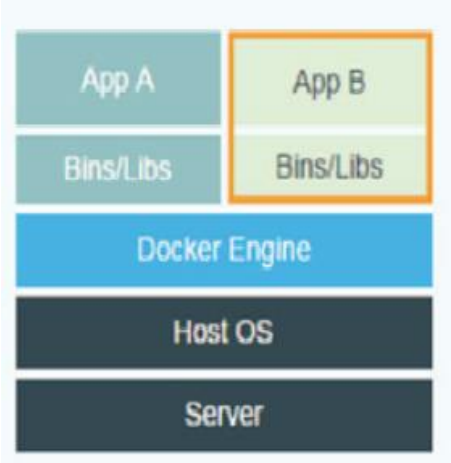
- Using Docker and X11 protocol
- Edge Computing

16.1. Docker and X11 protocol

Virtual machines which is widely used in cloud computing to do the application deployment on cloud server. In the virtual machine architecture applications deployed on Guest OS are based on Host OS and Hypervisor. Although the architecture provides different Guest OS, it must manage and maintain the whole two-level OS environment. This is not efficient as it will lead to performance attenuation. Drawbacks which cannot adapt to user's requirements of concurrency, ease of management as well as high responsiveness is overcome by using Docker with X11 in cloud OS. To do the application deployment and physical resource management a new technology Docker is proposed.



Architecture of Virtual Machine in cloud OS



Architecture of Docker in cloud OS

Docker is an open source application level container engine, which provides additional layer of abstraction and automation of operating system level virtualization on Windows and Linux. As Docker does not need to deploy the entire virtual machine with the Guest OS, it is much more lightweight than traditional virtual machine. With Docker, able to run each component in a separate container with its own dependencies and its own libraries all on the Host OS but within separate environments or containers. Only need to build the docker configuration once. Containerizing applications will completely isolate the environments and the applications can then have their own processes, services, network interfaces. As all containers run on the same OS kernel, all software's will depend only on that which can reduce the resource consumption. In traditional cloud operating system to do the remote access and control, Virtual Network Console (VNC) is used in cloud OS that can transmit the full graphical windows interface to another computer through network. But VNC need to transmit every frame of software graphic interface will cost a large network bandwidth. To reduce the network overhead and improve the quality of remote-control access X11 protocol is used. X11 protocol translates the graphic command instead of each pixel of graphic interface causing less network overhead.

16.2. Edge Computing

With the development of new technologies more and more smart devices are connected to the Internet. For a huge amount of data or the devices access to the traffic are all directly routed to the cloud data center, causing it to be burdened and bringing along with it many problems such as network congestion. Edge computing is a distributed computing paradigm that supports the establishment of large-scale, distributed architecture allowing a certain

percentage of data to be within a certain scope for it to be processed on the edge. Only a small number of necessary data and access traffic will be routed to the cloud data center. Capabilities of Edge Computing includes:

- *Low network latency:* Reduction in spatial distance brings about shortened propagation latency. It significantly reduces the latency caused by various routing/forwarding network devices which are processing different scenarios on complex networks, reducing the overall latency.
- *To support large bandwidth scenarios:* By processing large traffic on the edge, edge computing can effectively avoid network congestion related problems, and greatly reduce costs.
- *dealing with highly concurrent access:* It uses a distributed architecture to mitigate the load on the cloud data center.

17. Conclusion

Cloud Computing and virtualization is a large umbrella under which a variety of tools, technologies and concepts gets implemented. The root of virtualization is to provide illusion/emulation of a specific environment, to provide an abstraction layer. All the virtualization techniques play a fundamental role in building cloud infrastructure and providing services on demand. It is the driving force which makes software delivery and operation simple leading to rapid adoption of cloud solutions. Cloud computing solutions considerably reduces the capital costs of IT assets and transform them into operational costs thereby increasing the profits of enterprises. Although the technology is rapidly adopted there is constant need to research and improvement in areas like management, security, energy efficiency, social and organizational issues.

18. References

[1] Chen ZN, Chen K, Jiang JL et al. Evolution of cloud operating system: From technology to ecosystem. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 32(2): 224–241 Mar. 2017. DOI 10.1007/s11390-017-1717-z.

[2] Mastering Cloud Computing: Foundations and Applications Programming Book by Christian Vecchiola, Rajkumar Buyya, and S. Thamarai Selvi

[3] Zhou F F, Ma R H, Li J et al. Optimizations for high performance network virtualization. Journal of Computer Science and Technology, 2016, 31(1): 107-116.

[4] https://docplayer.net/7926360-Iaas-cloud-architectures-virtualized-datacenters-to-federated-cloud-infrastructures.html

[5] System Performance Evaluation of Para Virtualization, Container Virtualization, and Full Virtualization Using Xen, OpenVZ, and XenServer Anish Babu S. ; Hareesh M.J. ; John Paul Martin ; Sijo Cherian ; Yedhu Sastri Year:2014 | Conference Paper | Publisher: IEEE

[6] Overview of virtualization in cloud computing Nancy Jain ; Sakshi Choudhary 2016 Symposium on Colossal Data Analysis and Networking (CDAN)Year: 2016 | Conference Paper | Publisher: IEEE

[7] Zhen Du, ZhuQing Xu, Fang Dong, Dian Shen: A Novel Solution of Cloud Operating System based on X11 and Docker, 978-1-5386-1072-5/17 2017 IEEE DOI 10.1109/CBD.2017.13

[8] Extending the Boundaries of the Cloud with Edge Computing: https://www.alibabacloud.com/blog/extending-the-boundaries-of-thecloud-with-edge-computing 594214.

[9] Edge Computing: https://en.wikipedia.org/wiki/Edge computing