# Unit-II
# Virtualization Types – Server, Storage, Network, and Application

**Abhirup khanna**
**Department of Virtualization**

# Virtualization Types – Server, Storage, Network and Application

- 2.1 Server Virtualization
- 2.2 Types of Server Virtualization
- 2.3 Storage Virtualization
- 2.4 Types of Storage Virtualization
- 2.5 Network Virtualization
- 2.6 VPN and VLAN
- 2.7 Application Virtualization
- 2.8 Challenges, Solutions, and Benefits of application virtualization.
- 2.9 Understanding Desktop Virtualization

# Types of Server Virtualization…

- The main criteria for classification is:
- Extent of emulation required
- and
- The interface between the hypervisor and virtual machines.

# Types….

- **Full Virtualization**
- **Emulation**
- **Simulation**
- **Hardware Assisted Virtualization (HAV)**
- **Para Virtualization**

# Full Virtualization

- Refers to a set of techniques to ***fully simulate*** the underlying physical server environment.
- The simulation is so effective that an **operating system** designed to run on a physical server ***can be run unmodified on a fully virtualized virtual machine***.

# Fully Virtualized Virtual Machine (FVVM)

- A fully virtualized virtual machine (FVVM) is completely isolated from other virtual machines on the same physical host and is unaware of their existence.
- Any operation executed on a virtual machine (triggered by VM-resident OS or applications) does not have any external impact on other FVVM running on the same physical server.
- The hypervisor traps and effectively **translates any machine instructions for IO operations or any other instructions that require direct or privileged access to the underlying physical hardware.**
- The requests from one VM are not allowed to alter the state of another VM.

# Emulation

- Emulation refers to a set of techniques to ***virtualize a completely distinct hardware*** than the underlying physical hardware.
- Emulation could be achieved in hardware or software.
- Emulation refers to the use of **firmware and hardware** to emulate an altogether different hardware.
- It emulates different architecture (in software code) on a totally different physical hardware.

# Emulation…….How?

- Emulation is only possible through binary instruction translation.
- The machine instructions of the emulated processor are translated to native machine instructions before dispatching to physical hardware.
- For example:
- *Qemu-PPC can emulate the PowerPC architecture environment on an x86 physical machine.*
- *Qemu-ARM can emulate the ARM architecture environment on an x86 physical machine.*

# Other Examples……

- ***8051 microcontroller*** *emulators emulate 8051 hardware on a* ***x86 physical machine****.*
- ***Rosetta*** *is Apple's emulator for* ***PowerPC*** *architecture.*
- ***OVPsim Emulator*** *emulates varied processor architectures including MIPS architecture on a physical x86 server.*

# Dip in Performance…

- Binary instruction translation, causes a **performance hit**.
- Usually hardware being emulated is a legacy processor or an embedded processor.
- These processors, in their physical form run at very low clock frequencies that are only a fraction of that of the host processor.
- Hence, if an old hardware/embedded system is being emulated on a faster host, the performance is typically more than the emulated hardware.
- On the other hand, if the emulated hardware compares to the host hardware in terms of original performance, the emulation performance would be at best reasonable or suffer from bottlenecks for the purpose involved.

# Emulation……..

- Emulation is used for the **purpose of running old legacy applications in their native OS environments** while ***<u>making use of the cutting edge high performance hardware available today</u>***.
- This eliminates the effort to modify or port these applications to current OS/Hardware technologies.
- Another reason to use emulation is to **test/pre-debug embedded systems** or new hardware before going in for actual production.

# Simulation

- Simulation refers to techniques that create an exact virtual footprint of the underlying hardware for the purposes of virtualization.
- The main reason for **virtualization-by-simulation** is isolation and ***consolidation of OS and applications***.
- ***OS and Application consolidation is to reduce the number of OS and application instances respectively.***
- Reduction of ***SW licensing costs.***
- Reduces ***less number of OS instances*** that have to be managed

(including ***number of patches, service packs and security fixes that have to be applied).***

- Multiple VMs run on a single physical hardware that closely matches the VM logical characteristics.

# Simulation

- Simulation requires **limited binary instruction translation**, since the VM is simulating a similar processor and most of the less privileged instructions can be simply passed to the underlying hardware processor for execution.
- Only instructions that are **highly privileged** (or might affect the state of other virtual machines) are controlled by the hypervisor and executed on a sharing basis.
- This relieves the hypervisor from binary instruction translation, thus resulting in comparatively higher performance than emulation.
- ***When we refer to Full Virtualization in the industry, we generally refer to simulation.*** [Very thin line of Difference]

# Hardware-Assisted-Virtualization (HAV)

- Hardware Assisted Virtualization (HAV) use features provided by the hardware to improve performance of the simulated virtual machines.
- ***The approach uses features provided by the host physical processor to improve efficiency of virtualization thereby improving the performance of the virtual machines.***
- Intel and AMD implemented the hardware features required for HAV in their latest processors.

# Intel and AMD…….

- The hardware features were named **Intel VT-x** and **AMD-V** respectively.
- It is a kind of Full-Virtualization, where VMs run in isolation without altering the state of another VM.
- A pre-requisite to achieve this, is to ***trap specific instructions that could affect the state of other VMs.***
- Earlier x86 processors lacked the capability to generate a trap on these instructions and they have to be simulated in software.
- This resulted in **performance hit** to the VMs.

# More……..

- After the introduction of Intel VT-x and AMD-V the hypervisor could rely on the hardware to generate the trap, ***thereby cutting down on the time*** required to virtualize and simulate the traps in software.
- This improved efficiency and performance of the virtual machines.

# More……..

- A common drawback of this technique was that ***not all*** hardware supported HAV.
- Hence, the widespread deployment of this was ***limited*** in presence of legacy or old hardware.
- But today, almost all new processors supported HAV, hence the hypervisors could make use of it at a larger scale.

# Para Virtualization….

- Para Virtualization requires specialized hypervisor support in the form of an ***application programming interface*** (API).
- API is used by virtual machines to request services from the hypervisor.
- This specialized support from the hypervisor is known as the ***Hyper-call*** or ***Para-API***.
- The virtual machine operations could be run in the ***virtual context (simulation)*** or the ***Hyper-call context***.

# Performance in Hyper-Call….

- The performance of operations in the ***virtual context is slower*** and suffers from performance degradation.
- It would be ***better*** if these could be run in the hypervisor context ***through the API calls.***
- These results in significant improvement in performance compared to a fully virtualized machine.

# How to bring about Para-Virtualization

- The guest operating systems hosted in a virtual machine have to be modified or adapted to call these APIs instead of the regular machine instructions.
- Commercially available operating systems with closed-source are generally not amenable to such a setup.
- ***Linux and FreeBSD*** provided required modifications as part of their kernel tree to run on Para-virtualized setups.
- The most common example of a Para-virtualized machine is a **Xen-VM** running in Para-virtualized mode.
- A specialized version of the Linux kernel is required for installation and the boot up of the Linux OS on Xen-VM.

# Considerations for Server Virtualization

- **SPOF (single point of failure) in Server Virtualization**
- Server virtualization – reasons for preferring – increased utilization, better manageability, flexibility, easy sizing etc.
- However, consolidation using virtualization also raises concerns on the availability of the infrastructure.
- Since, multiple physical servers are converted and consolidated to virtual machines to run on a single physical server, this raises concerns on the reliability of the single underlying physical hardware.
- The underlying physical server becomes a single point of failure (SPOF) for all the virtual machines hosted on it.

# SPOF…….

- This increases the risk multifold as compared to traditional physical infrastructure wherein, there was usually one application running on a physical server.
- The risk was low since the failure of the server would impact only one application.
- With an increase in the system configuration of the physical servers, it is now possible to place hundreds of VMs on a single physical machine.
- Solutions need to be found to mitigate concerns on the failure of the underlying physical server.

# SPOF……..

- The most popular approach is to have a **redundant physical server** of almost same capacity on standby, to adopt the VM in case the primary server fails.
- Virtual machines can also be run on both the physical servers in a mutual takeover configuration.
- This configuration makes good use of the standby server by keeping it utilized, while providing availability to the other physical server.
- In case of failure, one server can take-over the VMs from the other server.
- There are multiple solutions available in the market to provide the ***highly available infrastructure***.
- Exs: IBM PowerHA, HP Serviceguard, VMware-HA etc.

# Patch Management

- Patch Management is an important consideration when moving to a virtualized server infrastructure.
- Virtualization allows an administrator to take snapshot of a running virtual machine into a file.
- This snapshot could be used to provision the machine at a later date.
- There’s **no need to repeat the configuration** on the VM as the configuration could be cloned from a VM snapshot.
- Incremental patch updates may be done on the active or inactive VM and the same could be saved as an updated snapshot.

# Patch Management…….

- Virtualization relieves the administrator of the burden to repeat the configuration and patch updates by providing a ***simple VM save-restore*** model.
- The snapshot or VM image library/store saves administration time involved in keeping the servers up-to-date or provisioning new servers.
- The servers could be provisioned in a matter of minutes as compared to 6-8 weeks in traditional infrastructures.
- While Server Virtualization optimizes and saves significant cost on most of the system administration tasks, it may also lead to **proliferation of unaccounted-for VM images and mismanaged VM libraries or store.**

# Contd……….

- Proper management of the VM-store and processes are required to manage and maintain the provisioning, up-dation and recycling of the VM libraries.
- **Virtualization management tools and VM lifecycle management tools** in the market address these concerns and provide a streamlined workflow to manage the virtualized infrastructure.

# Common VM tools available commercially..

- Parallels Virtual Automation
- Microsoft System Center Virtual Machine Manager
- VMM – Virtual Machine Manager for managing Xen and KVM Virtual machines

# Some Issues.....Migration of existing IT infrastructure

- With an increase in adoption of Virtualization, more and more customers would want to move their applications to virtual infrastructure.
- An important part of this process is **virtualizing the physical servers** involved in running the application.
- To achieve this, an exact replica or a virtual machine (with similar OS, configuration, patch updates) has to be created to host the application.
- This is achieved through a popular mechanism supplied as part of any virtualization solution in the marketplace, called the **P2V** (Physical to Virtual).

# P2V…………….

- Using P2V, the configuration of the physical server is transformed to an **equivalent virtual image** configuration.
- The virtual image can then be booted up and the application hosted on this.
- With large datacenters, the P2V process can be **automated** and finished early with limited downtime and impact to end-users.

# Licensing.........

- Traditional Licensing for large applications was tied to the number of physical CPUs/cores running the application.
- **When these applications are moved to a virtual platform, it becomes possible to run multiple copies of the same application in multiple VMs but still pay for a license limited by the number of physical CPUs on the machine**.
- This was realized by the application companies as a major loss in revenue.
- The licensing terms have changed ever since and now for most applications, the license is tied to the number of virtual CPUs running the application.

# Licensing..........

- Licensing is an important consideration for virtualizing the server infrastructure.
- Important license terms and conditions have to be checked on an application to application basis to estimate the cost involved.
- The software license cost generally overshadows the hardware costs in the long run.
- Hence, the cost involved to license has to be properly worked out before planning to move.
- Increased adoption of virtualization in the competitive market has resulted in favorable license terms for companies going in for applications designed for virtualized infrastructures.

# Desktop Virtualization

- Desktop virtualization refers to the technique of creating an abstraction of **desktop clients** or **end-user computing** equipment.
- This is not very different from virtualizing a physical server.
- The process involves creating a logical abstraction or a virtual image of the desktop and placing it on a centralized physical server.
- The end-user has an option to access their virtual desktops hosted on the centralized server **using remote desktop client (RDP, VNC (Virtual Network Computing etc.).**

# Desktop Virtualization…

- This technique allows for consolidation of several desktops in the company to a single centralized physical server.
- The end-user client equipment runs a **light-weight client with minimal resources.**
- Thin clients are about 10-20 times cheaper than desktops.
- Depending on the solution and context, desktop virtualization can mean **hosting critical desktop** applications or placing the entire desktop environment of end-user desktops on the centralized server.
- The end-users can then access this applications and data from the centralized server without ever installing the application on their local desktops.

# Desktop Virtualization.........How?

- Desktop virtualization is achieved using a set of hardware-software technologies and systems commonly termed as **VDI (Virtual Desktop Infrastructure)**.
- The idea behind VDI is to **create a virtual instance of each desktop and store it on a centralized server**.
- The applications hosted on the virtual instance can be triggered or run from any thin device ( a thin client or a smart mobile device ) thus enabling the user to execute the applications from remote location without requiring physical access to the type of hardware required to run the application.

# Main Requisite for Desktop virtualization…

- A **common requirement** for enabling desktop virtualization using VDI is the **availability of reasonable network bandwidth between the end-user client and the centralized server.**
- ***Main Advantage:***
- The desktop virtual instances located on the central server can be easily powered down when not in use by the end-user.
- This kind of monitoring is generally not easily possible in case of physical desktops which are left running even when not in use.

# Benefits of Desktop Virtualization

- Desktop virtualization offers the **same benefits** as the Server virtualization from the cost, consolidation and utilization perspective.

# (1.)Easy Management of Desktop Instances…

- Since the desktops are now centralized and located on a remote server, it is easy to keep the desktops upto-date using advanced virtual machine monitoring and management software.
- Easy snapshots enable easier backup of desktop environments.
- These snapshots can also be used to spawn new desktop instances for new users without going for a long installation and configuration cycles.
- Easy to power-up or power-down desktops based on need and usage trends.
- This allow for a **granular resource management** on the centralized server.
- The freed up capacity can be used for other useful purposes or for accommodating new set of desktop users

# Contd.....

- **(2.)Ease of application access**
  Hosted desktop instances can be accessed anywhere anytime from an end-user device like a smart phone or a thin client. This improves the mobility of the workforce.
- **(3.)Sharing of performance-intensive applications**

Access to performance intensive applications which were traditionally hosted on high-end machines were constrained by their location and the number of simultaneous users.

- With the centralization of these desktop machines, the applications can be hosted on a central server, and can be accessed over a network anywhere anytime by any number of users.

# Contd….

- **(4.) Improved desktop availability**
- The centralized server can be configured and monitored for high availability.
- This ensures that the desktop instances hosted on the server will have limited or no downtimes.
- Individual downtimes of desktops are also avoided.
- Also, the workflow involved in getting a desktop serviced is reduced.
- **(5.) Application deployment and Licensing**
- Run-time optimization on usage of licenses can be done as it is easy to track the desktops that are currently powered off.
- Multiple simultaneous users of an application can be easily estimated and controlled to arrive at better licensing terms from different vendors.
- **Instead of going for a desktop based license for an application, a server based license for the application turns out to be lot more economical**.

# Constraints in desktop virtualization

- □ **Network bandwidth**: An important resource to enable desktop virtualization using VDI is the availability of a reasonably fast network.
- The network infrastructure has to be sized based on the number of simultaneous clients accessing the central desktop server.
- Any bottleneck in the network will have an impact on large number of users.
- This limits the widespread adoption of VDI especially when the end-users are geographically dispersed over many locations.
- □ **Security of the network :** Security of the network connecting the end-users with the desktop server is important as any kind of SPI (sensitive personal and business information) moving over the network can be easily compromised if required security infrastructure is not in place.
- □ **Graphic Intensive Applications :** Certain applications like graphic animations, movie editing, 3d graphics, games require quick response times when it comes to display and input response. These applications are not recommended for VDI.
- □ Application requiring direct access to peripherals or embedded debuggers /programmers used in the hardware design sector that require direct access to host connected serial, usb ports are not well tested on VDI.

# Types of Desktop Virtualization

**Centralized Server Method :**

In a centralized method, the desktop virtual machines are kept on a central server.

The end-user equipment could be a thin-client capable of presenting a remote display to the end user, very similar to a RDP protocol client.

The hosted virtual desktops may be provisioned on-demand, be maintained in current configuration for longer runs, or be saved and archived in a desktop-repository on the server.

One major requirement for this kind of model is a good network connection.

With increasing number of users and the geographic dispersion of users, this model may not scale-out so well.

# Contd………

- **Shared Load Method :**

In this method, it is possible for the end-user client to take a copy of the hosted virtual desktop and run it locally.

This enables the users to have an option when the network connection is not up to the mark or there is a need to do heavy graphic processing on the desktop requiring **fast response with no packet loss**.

The **checked-out desktop instances may be checked back** in to the server using an incremental synchronization method on a periodic basis.

This method brings down the dependency on the network and facilitates running applications with special response requirements.

Remote execution of the desktop instances is also supported.

One of the major drawbacks of this method is when the instances are run locally; the end-user equipment must be capable enough to run the load.

# Contd…

- **Client Hosted Method :** In this method, the virtual instances reside and run on the end-user equipment with no exceptions.
- Access to the server is only required to manage the desktop instances and keep the associated configuration data on the server.
- End-user equipment with capable hardware is a pre-requisite for this kind of virtualization.

# Anatomy of Server Virtualization

- Understanding the internals of a standard Server Virtualization solution:
- Eg:  Open-source virtualization product **(Xen Server Virtualization**).
- Xen Server allows multiple guest OSs to run on virtual machines managed by an underlying layer called the **Hypervisor**.
- Xen Server provides **bare-metal server virtualization**.
- This basically means that the hypervisor resides over the hardware.
- Hence, it is a **Type-I Hypervisor.**
- The hypervisor has complete control on the four major hardware resources in the system – CPU, Memory, Network and Storage.
- The hypervisor multiplexes these resources among the virtual machines.

# There are three major layers in a Xen Server:

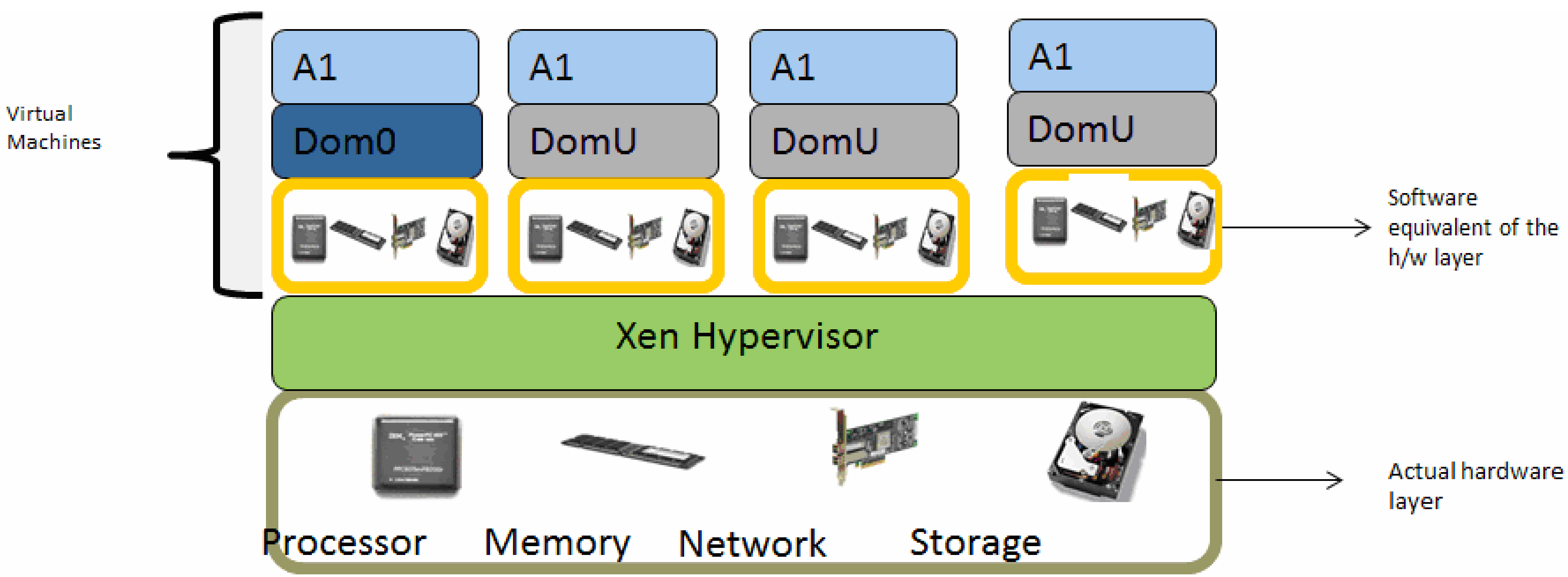
- 1. **Hardware Layer** : The hardware architectures supported by Xen are x86, x86_64, ia64 and ARM.
- Xen will generally make use of the virtualization capabilities provided by the hardware.
- While running on x86 and variants, it typically uses either **Intel-VTx** or **AMD-V** virtualization features.
- Recently, there are additional features supported in hardware called the **SR-IOV** for IO Virtualization.

# Anatomy contd…

- **2. Hypervisor** : The Xen hypervisor provides the required virtual platform for defining and running the virtual machines.
- The hypervisor supports two kinds of virtualization – **Para and Full Virtualization.**
- In case of Para Virtualization, the Xen hypervisor provides the Hyper-call APIs which the guest machines use to request services from the hypervisor.
- **Para-Virtualization is not a pure-virtualization; rather it is implemented as a set of APIs**.
- The virtual machine OS hosted on Para-Virtualized VM requires modification to use the Hyper-call APIs.
- Only OSes with explicit support for Para-Virtualization can be run in this mode.
- Full Virtualization provides full emulation of the underlying platform with help from the hardware.
- No OS modification is necessary to run on a fully virtualized Xen machine.

# Contd……

- 3. **Domains Layer :** A differentiating factor of any virtualization solution is the platform and the manageability tools it provides to the system administrator.
- Xen Hypervisor boots into a **control domain (Dom0), a default OS, typically a version of Linux.**
- This control domain provides the administrator with a platform to **create, manage, store and delete and track virtual machines.**
- The control domain has complete access to the hardware on the physical machine.
- The control domain can be seen as the first virtual machine on the physical system.
- However, starting with the second virtual machine, the virtual machines are termed as **DomU**.
- The **DomU** are the actual user defined virtual machines.
- Using the control domain, the administrator can tailor make the configuration of DomU.
- A DomU could be a para-virtualized machine or a fully virtualized machine.


Virtual
Machines
A1
A1
A1
A1
Dom0
DomU
DomU
DomU
Software
equivalent of the
h/w layer
Xen Hypervisor
Processor
Memory
Network
Storage
Actual hardware
layer