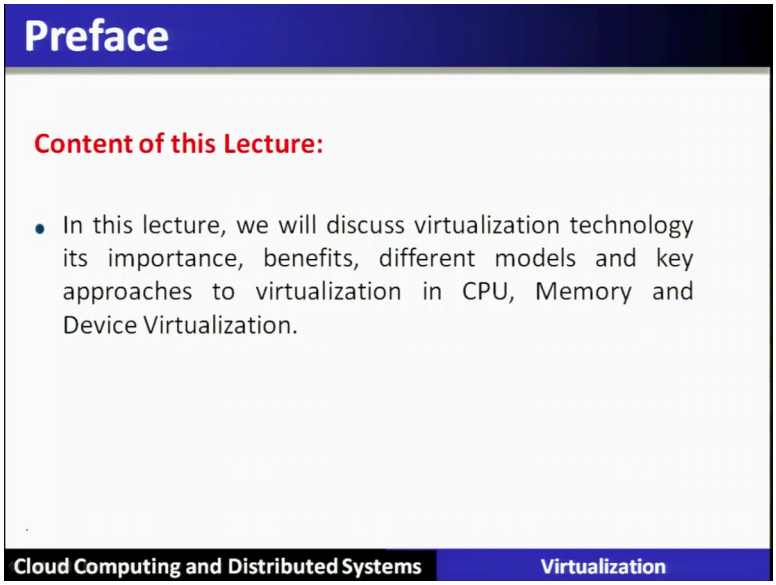
Cloud Computing and Distributed Systems
Dr. Rajiv Misra
Department of Computer Science and Engineering
Indian Institute of Technology, Patna

Lecture - 02
Virtualization

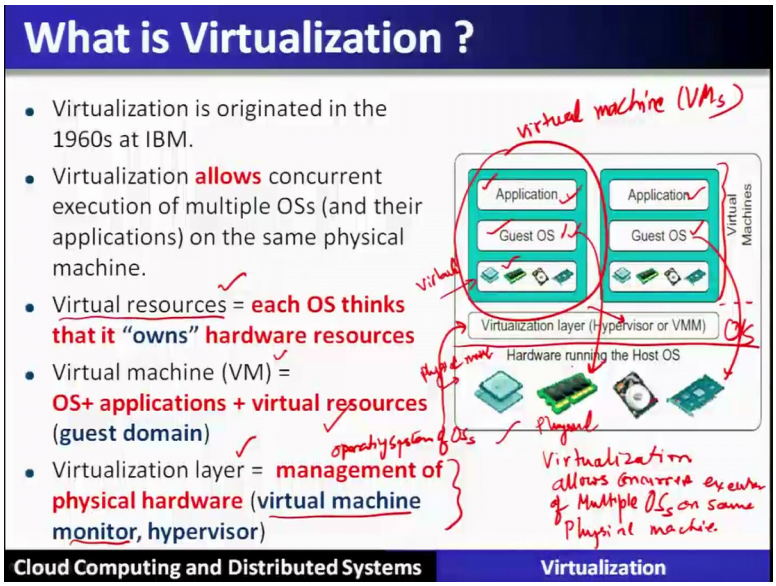
Virtualization; preface content of this lecture.

(Refer Slide Time: 00:17)



We will discuss virtualization technology, it is importance, benefits, different models used for virtualization. And the key approaches to the virtualization with respect to the CPU, memory and device virtualization is a part of this lecture. What is virtualization?

(Refer Slide Time: 00:45)



Virtualization is originated in the year late that is 1960's at IBM. So, it is not a new concept. So, virtualization allows concurrent execution of multiple operating systems, and their applications on the same physical machine.

So, in this example you can see here, that there is one piece of hardware that is CPU memory disk and devices. This particular piece of hardware, if it is being used by more than one application and it is corresponding operating systems, then this particular way of sharing the common hardware across multiple such machines which are the virtual machines. So, this will allow the concurrent execution of multiple operating systems and also their applications simultaneously over a same piece of hardware. So, this is possible using the virtualization.

So, virtualization allows the concurrent execution of multiple operating systems on the same physical machine. So, this is a physical machine and now it has multiple virtual machines; that means, operating system it is application and it is virtual resources together is called a virtual machine. So, what becomes the virtual resource? So, here each operating each operating system thinks that it owns the entire hardware resources. So, these hardware resources, this operating system which is called a guest operating system, thinks that it owns these hardware resources.

Similarly, the other guest operating system, and there may be many more such virtual machines residing and concurrently executing simultaneously. They think that this

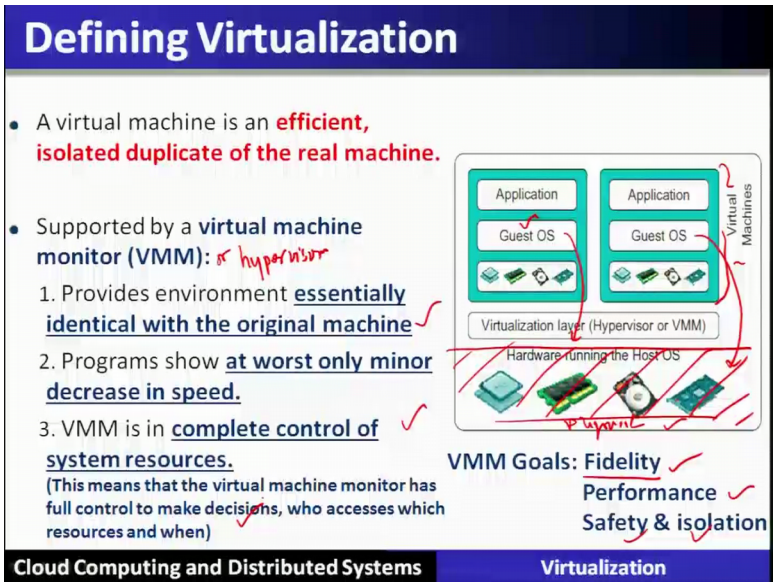
hardware they own, and if they think this way then it is called the virtual resources. So, the physical resources are being shared that is why they are called virtual resources.

Now, the concept of a virtual machine is that set of operating system which is called a guest operating system. And the application which runs on that corresponding operating system and the virtual resources together is called the virtual machine. So, these particular several operating systems will now be accessing the same piece of hardware. So, that means, this guest operating system will be either they have own this entire resources or some part of the resources which are available as a part of the physical machine.

So, there is a requirement which is called the virtualization layer or you can also think of an operating system of several operating systems. So, operating system is required which will basically provide the physical resources to be shared across multiple operating systems. And this particular operating system of operating systems is called basically the virtualization layer. So, virtualization layer is also called as hypervisor or virtual machine monitor. So, the virtualization layer or hypervisor or virtual machine monitor is nothing but an operating system of operating systems. So, operating systems of the virtual machine requires to be managed by or requires to be supported by this virtualization layer that is the hypervisor.

Similarly this operating system or which is called a virtualization layer or a hypervisor will use these hardware's or will manage the physical hardware resources and will provide the virtualized view to the several operating systems which are called a guest operating systems. Defining the virtualization, a virtual machine is an efficient isolation of the duplicate of the real machine.

(Refer Slide Time: 06:11)



So, here you can see that this particular virtual machine will duplicate the real physical machine. So, this is to be done in an isolated in an efficient manner. So, virtual machine is an efficient isolated duplicate of the real machine, which is supported by the virtual machine monitor or the hypervisor.

This particular virtual machine monitor or the hypervisor will provide environment that is essentially identical with the original machine. So, all the resources which are available as part of the physical machine will; that means, this guest operating system will have an illusion as if they basically are owning these physical resources. And that particular illusion is being supported by the virtual machine monitor or and hypervisor.

In other sense, this particular illusion is to be done or is to be achieved in a real sense. In the sense it is a copy of all the physical resources will be available as a part of the virtual resources to the guest operating systems. Hence the one of the important goals of virtual machine monitor is fidelity. The second goal of virtual machine monitor is that the programs show at most only the minor decrease in the speed.

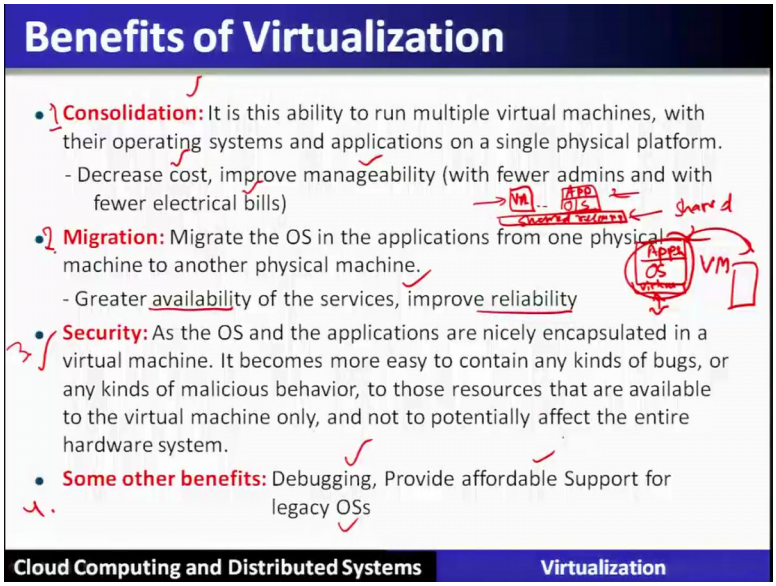
So, for example, these resources are being shared across several guest operating systems. So, not all resources are available to these guest operating systems which are available as far as the physical machine is concerned physical resources, but some part of the physical resources is available.

Hence, this particular way the programs will have only a small decrease in the speed. Because the full resources are not available, but other full resources are shared across the resources. Hence the performance also is one of the requirements that there will be an efficient or a good performance in doing so; that is, in sharing the resources across several operating systems.

Now, the third one which is supported is basically that virtual machine monitor is in the complete control of the physical resources; that means, if it is not done in that manner then, these operating guest operating system may interfere. The use of these physical resources with the other guest operating systems, hence a complete isolation and safety is also the third important requirement of the virtual machine monitor or the hypervisor.

So, basically the virtual machine monitor will take the full control and then makes the decision of a controlled access of these resources to or across this guest operating systems, to ensure the safety and also ensures the isolation across different virtual machines which are running over the same physical machine.

(Refer Slide Time: 10:05)



Benefits of the virtualization: the first benefit is the consolidation. We will understand what you mean by the consolidation. So, consolidation is the ability to run multiple virtual machines with their operating systems and applications on single physical platforms. So, again I am giving you a illustration that this is a VM virtual machine 1 2

and so on several virtual machine which comprises of an OS and application. They will run on shared resources. This particular way is called a consolidation.

In the sense, that this physical resource will be shared across multiple virtual machines or a many virtual machines. This is called a consolidation, because it will decrease the cost why because the same set of hardware resources cannot be fully utilized by one physical set of operating system and applications. If there are several such virtual machines running; obviously, the overall cost will decrease, and also will improve the manageability of all these resources and a fewer other electricity bills and other maintenance cost. Hence the consolidation is one of the key factors to utilize the hardware's or the physical resources. And this is done by the virtualization and this process is called consolidation.

Now second important benefit of virtualization is the migration. So, that means, migrate the operating system in the application from one physical machine to another physical machine. Now given that the application and it is corresponding operating system they are bundled together with the virtual resources, and this is called virtual machine. Now this will be detached from the physical machine this complete package can be migrated or may be copied to some other place whenever there is a scope of improvement or efficiency or availability or improving the reliability.

There are various regions which requires the mobility to migrate the entire virtual machine from one physical machine to another physical machine; thereby, increasing the availability and also will improve the reliability and the performance of the system. So, the maintenance here becomes quite automatic and manageable with the migration in place. So, this is another benefit of the virtualization.

Third benefit is called security. So, as the operating system and applications are nicely encapsulated in a virtual machine that we have seen in the migration also. Therefore, it becomes an easy to contain any kind of bugs or any kind of malicious behavior to those resources that are available to the virtual machine only, and not potentially affect the entire hardware system.

Therefore, it ensures the security if multiple virtual machines are basically accessing the same or sharing the same set of hardware in this particular scenario. So, whatever

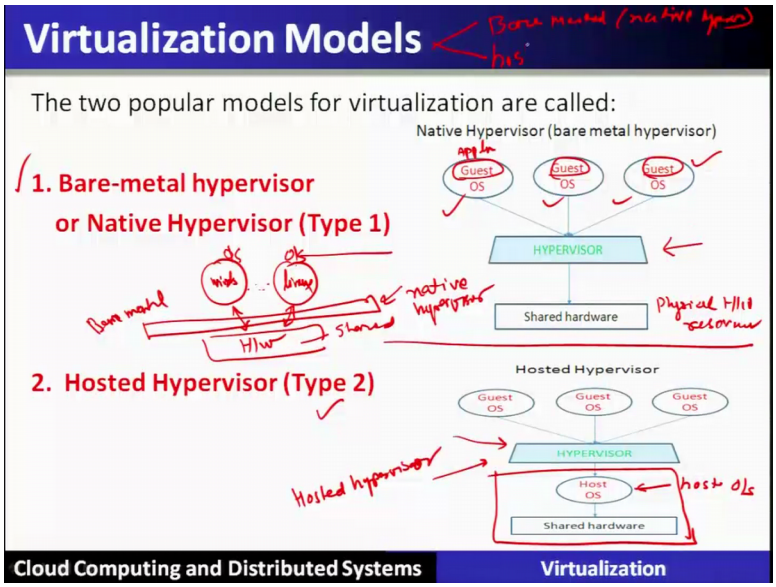
problem is happening at one virtual machine will not suffer to the other virtual machine that is called security and isolation.

Now both are some other benefits for example, whenever a debugging is required to be done into a one virtual machine. It will not affect the other virtual machines. Hence this particular debugging or malfunctioning at one virtual machine will be isolated or will not disturb the other virtual machines. And the process and the efficiency of the machine or usage will go on.

Similarly, it will also provide the affordable support for legacy operating systems. What do you mean by this is that, the legacy operating systems to support in a new hardware a new hardware some more provisions are being provided by the architects of this particular machines.

Now, there is no longer required of this kind of things. Why because in the virtualization whatever is the legacy operating system and applications they are detached from the physical machine. Therefore, it will provide an affordable support for the legacy operating system in this manner this is another benefit of virtualization.

(Refer Slide Time: 15:23)



Now, let us see different models which are basically most prominent for virtualization. There may be there are various other models available, but we are only going to discuss most important or a prominent model for virtualization. The first one is called the bare

metal hypervisor, or it is also called the native hypervisor and it is called as a type one hypervisor.

See in this particular example, over the hardware in the hypervisor sits just above the hardware or a shared physical machine or a physical hardware resources to support various different operating system and their corresponding applications. Then it is called the bare metal hypervisor.

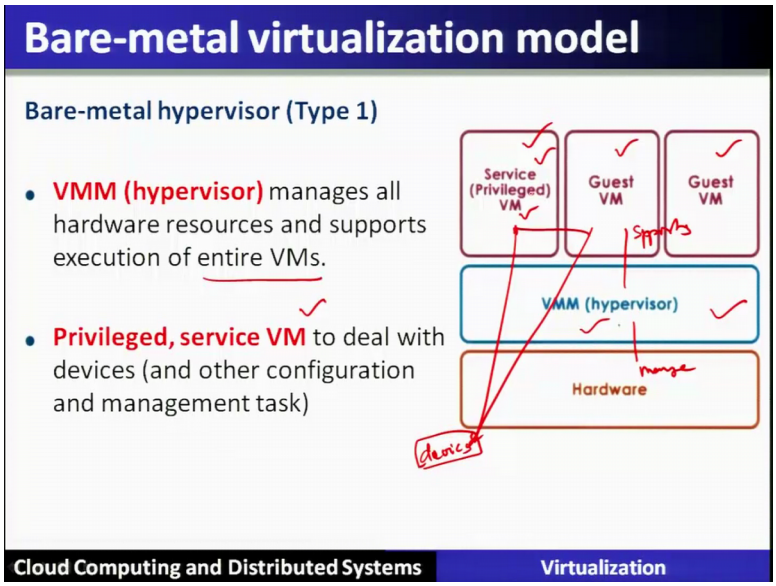
So, in another way that you can think of for example, there is a Microsoft windows, this is Linux. There are different operating systems now running over the same piece of hardware, which was earlier not possible, either the windows was running or the Linux was running in this particular moved now all different operating systems OS can run simultaneously over a same piece of hardware; that means, the hardware is shared across different operating system running. That is being possible with the help of the virtualization layer, which is called bare metal hypervisor or it is in need the hypervisor.

Why because, it will basically sit just above the physical hardware and that is why it is called in native hypervisor. The different operating systems which will be using the hardware using with the support of bare metal hypervisor they are called the guest operating systems.

Now, second type of virtualization model is called a hosted hypervisor or a type 2 hypervisor in hosted hypervisor shared hardware continues to use it is own operating system, and that is called a host operating system. Now to support several other operating systems by this particular operating system to share the hardware it requires another layer which is called virtualization layer which is called a hosted hypervisor.

So now, the physical machine comprises it is hardware and the operating system which will be shared across multiple operating system with the support of the hypervisor. So, hypervisor sits just above the host operating system and gives and supports the guest operating system. If that is the model, then it is called type 2 hypervisor. It is also called as hosted hypervisor. So, these 2 models we have seen; that is, the bare metal or a native hypervisor, the other one is called a hosted hypervisor.

(Refer Slide Time: 19:25)

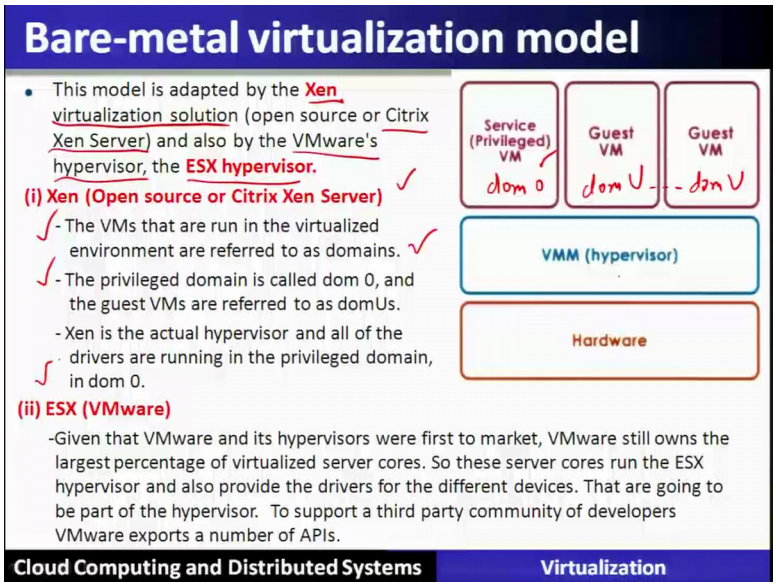


Let us see some details inside this particular virtualization model which is called as a bare metal virtualization model. So, in a bare metal hypervisor, this particular hypervisor that is virtual machine monitor sits just above the hardware. So, this is called a bare metal hardware why because, this hypervisor directly sits over the hardware and manages the hardware resources. This will also support for a several virtual machines with the help of a privileged virtual machine.

So, virtual machine monitor the hypervisor manages the hardware resources, this will manage hardware resources. And also supports the execution of several or entire virtual machines. Now there is a concept of privileged service virtual machine to deal with the devices and other configuration and the management task.

For example, the devices to share the devices across the guest virtual machines; so, virtual machines would like to access the device is directly. This is done through the service or the privileged virtual machine. This will support the axes of devices to the other guest virtual machines. Hence, the virtual machine monitor that is the bare metal hypervisor with the help of service privileged virtual machine continues to support the use of the configurations and management task related to the devices.

(Refer Slide Time: 21:24)



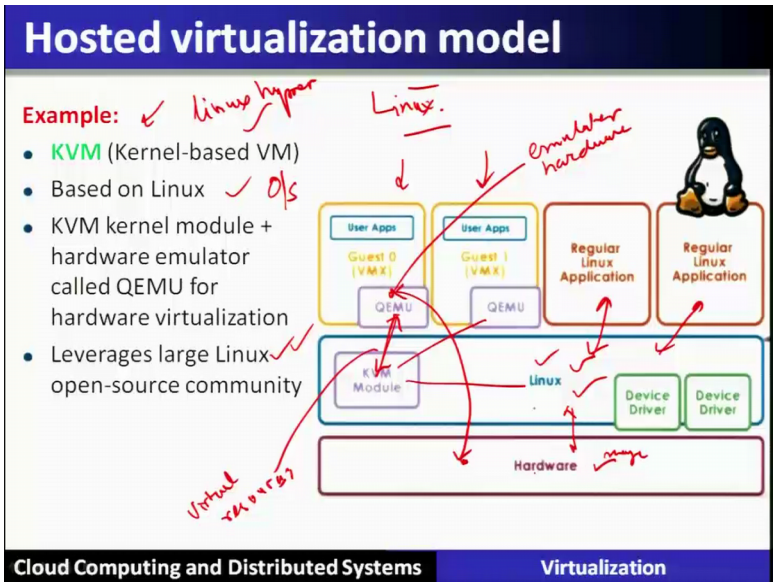
Now we can see that the bare metal virtualization model was adopted here in Xen virtualization solution, which is an open source or it is also available with Citrix Xen Server. This bare metal virtualization model is also used by the VM wares hypervisor which is called ESX hypervisor.

So, let us see these two industry standard or these two hypervisors which are used in the industry which basically are adopting the bare metal virtualization model. The first one is we are going to discuss is Xen.

So, the virtual machines that are run in virtualized environment, they are termed as the domains. The privileged domain is called domain 0, and the guest VMs are referred to as domain u. This is the terminologies which are used in Xen hypervisor. So, then is the actual hypervisor now all the drivers are running in the privileged domain. That is in the domain 0 that we have earlier explained in the previous slide of bare metal virtualization model.

Now another way of using the bare metal virtualization is seen here in VMware ESX hypervisor. Now given that VMware and it is hypervisors were first to market, we are very still owns the largest percentage of virtualized server course. So, the server course run ESX hypervisor run also provide the drivers to the different devices. They are going to be part of hypervisor to support the third party community of the developers. So, VMware exports a number of APIs.

(Refer Slide Time: 23:39)



Now, let us see hosted hypervisor is rated or type 2 hypervisor. So, in this model at the lowest level there is a full-fledged operator host operating system that manages all the hardware resources. So, this operating system which is called a hosted operating systems sits just above the hardware, and it is task is to manage the hardware resources.

Now, the host operating systems will integrate a virtual machine monitor. That is responsible for providing the virtual machine with their virtual platform interface. And for managing all of the context switching schedule. So, let us see the example over here.

Now, in the hosted hypervisor examples, hypervisor example the native applications; native applications; that means, who are using the same operating system can directly be supported the hosted hype using the host operating system in the hosted hypervisor. Now in the hosted hypervisor, we will use the host operating system or the host operating system will integrate virtual machine monitor to support the different virtual machines.

So, the different virtual machine will have their own guest operating systems, different guest operating systems are being supported by the virtual machine monitor which in turn will basically use the host operating system. And this host operating system is having the responsibility of managing the entire hardware. So, the hardware is shared through the host operating system via this hosted hypervisor across all the VMs or across all the guest operating systems. So, this particular way is supported and if it is then it is called the hosted hypervisor.

The good part of hosted hypervisor is that this host operating system. It is internal functioning will be leveraged to basically support this virtual machine monitor or the hosted hypervisor; so that it will continue to support several virtual machines. And at the same point of time it will also continue to use the legacy applications which will try to use the physical machine and several applications will be sharing the hosted operating system.

Similarly, the host operating system will have the support for the devices which are again being available being shared across all the guest operating system and the native operating system through hosted hypervisor. So, this model is called hosted virtualization model which continues to use it is host operating system. And so, that it will also support the virtual machine monitor. And this is called hosted virtualization or hosted hypervisor, which intern supports several virtual machines.

The example of a hosted virtualization model is seen in a kernel based, in a Linux based hypervisor, which is called KVM that is called kernel based virtual machine. It is based on the Linux operating system. So, Linux operating system you know that it sits over the hardware, and continues to support it is applications. And we will manage the hardware resources, also the device drivers.

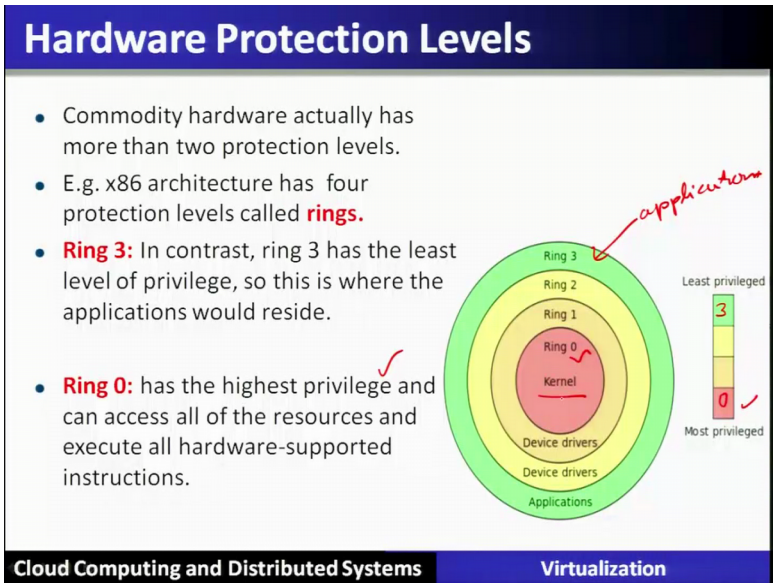
Now, whenever there is a virtual machines on the Linux platform. Then this Linux supports or integrates the KVM module that is the kernel virtual machine the kernel based VM. So, this particular module which is being supported by the Linux operating system intern which will support with the help of QEMU. Similarly, it is supported by QEMU. So, this QEMU is emulator, this is the hardware emulator.

So, QEMU is a hardware emulator; that means, this hardware which is available this will be reflected or will be made available to the guest operating system or to the virtual machines. And this particular QEMU with the help of KVM module will provide the virtual resources.

So, QEMU is other hardware emulator which will give a complete picture of the hardware resources which are available and this particular KVM module with the QEMU are going to support several VMs. This is also going to be important, why because this way that is hosted virtualization using the Linux operating system is going to be very useful why because Linux is an open source community.

So, whatever it is development taking place in the Linux will continue to be basically used up, here in this way of supporting this virtual machines in the Linux operating system.

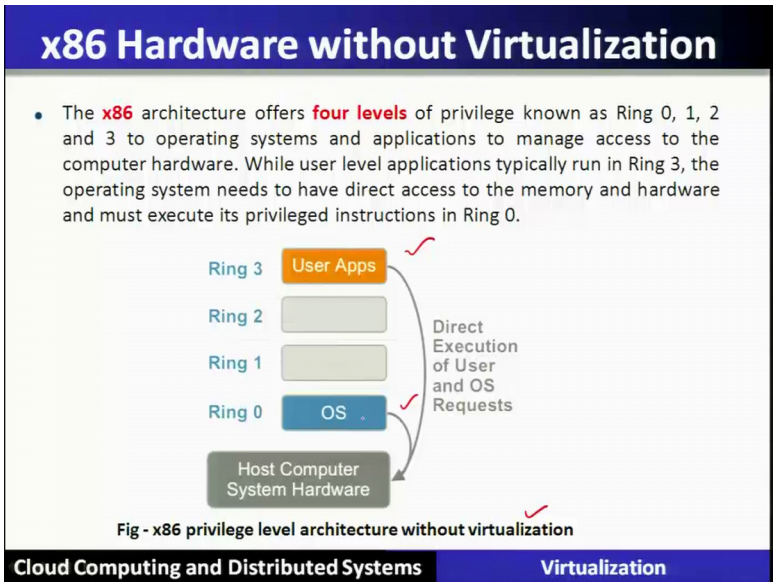
(Refer Slide Time: 29:56)



Now, let us see the hardware, production levels which are being used here in the hypervisors or in the virtual machine monitors. Now you know there are different commodity hardware's and these commodity hardware's has more than 2 production levels. For example, x86 architecture has 4 different production levels which are called as a ring, where as you see that kernel resides in the ring 0.

So, that means, kernel 0 has the highest privilege. So, level 0 has the highest privilege where as the level 3 has the least privilege. So obviously, the least privilege; that means, the applications will basically be dealt at the level 3 or a ring 3 privilege privileges whereas, the highest privileges are where the kernel or a code of the operating systems.

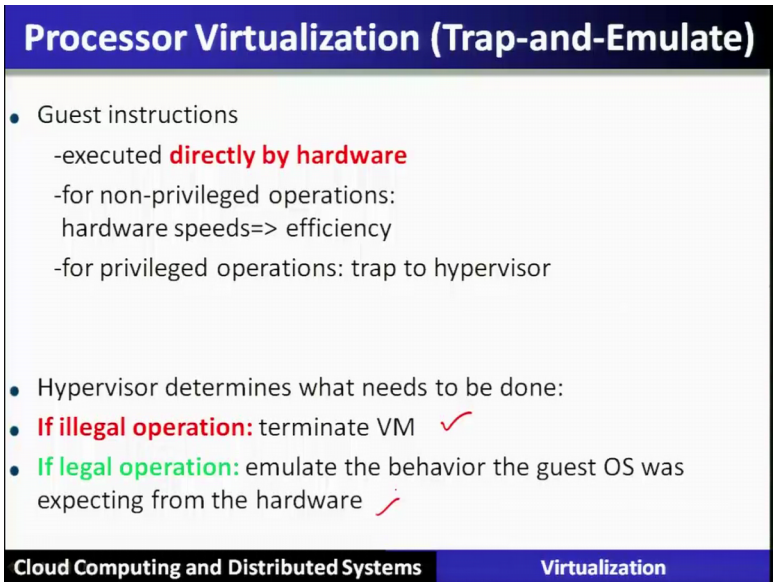
(Refer Slide Time: 31:03)



Now we can see here the diagram or the illustration of the x86 privilege level architecture. Without virtualization, here you can see that the user application are at ring 3 privilege level, where as the highest privilege level that is ring 0 will be assigned to the operating system kernel.

So, operating system is basically directly managing the hardware resources and supports all the applications. So, any user if non privileged instructions directly can be accessed through the hardware, but the privileged instructions are done through the operating systems, here in the normal scenario without any virtualization.

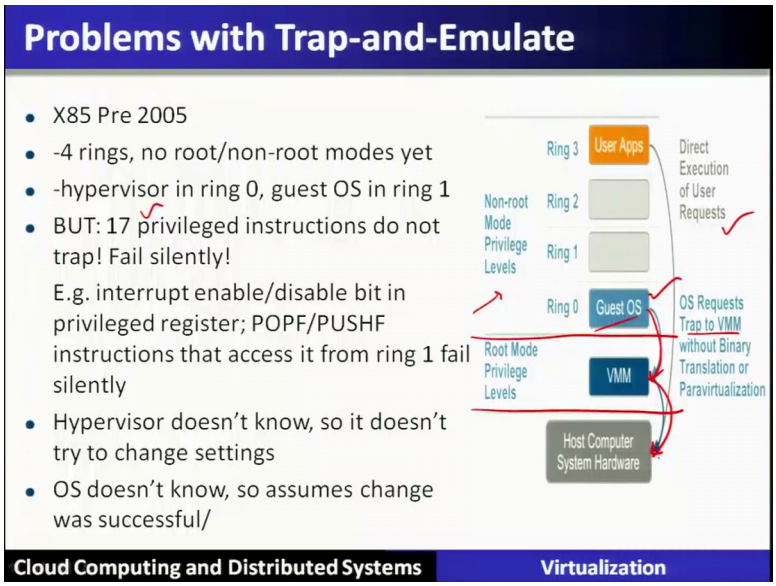
(Refer Slide Time: 31:52)



Now as far as the processor also supports the virtualization in the form of a trap and emulate instructions or constructs. So, the guest instruction that is the guest operating systems executed directly by the hardware; now for the non-privileged operations to ensure the efficiency and it can run at the hardware speeds. Now for the privileged operations, this will give a trap to the hypervisor; that means, the guest operating system will generate a trap to the hypervisor, and this is like a system call and hypervisor inter will call on behalf of the guest operating system to execute the privileged instructions.

So, hypervisor will determine what is to be done. If whenever there is a trap the hypervisor will ensure whether it is illegal operation, then it will terminate the virtual machine. If it is a legal operation, then it will emulate the behavior the of the guest operating system which is expecting to be executed at the hardware speed from the hardware. So, without any much loss of efficiency with the help of with the support of hypervisor all the privilege instructions are also executed at the guest operating system level.

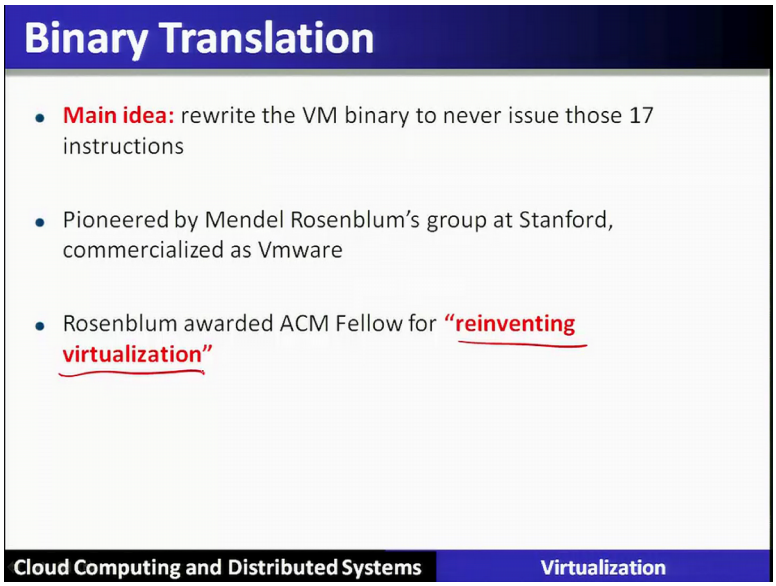
(Refer Slide Time: 33:16)



Let us see the scenario to support this particular ring is basically to support this trap at emulate instructions. So, all this particular ring 0 1 2; they are basically categorized as non-root mode privilege levels. So, user can directly execute the hardware in the normal situation; so, that the application can get the hardware speed or efficiency. Whenever the guest of whenever there is a privileged instructions generated by the guest operating system, this will trap to the virtual machine monitor which works at the root level or a root mode privileged level.

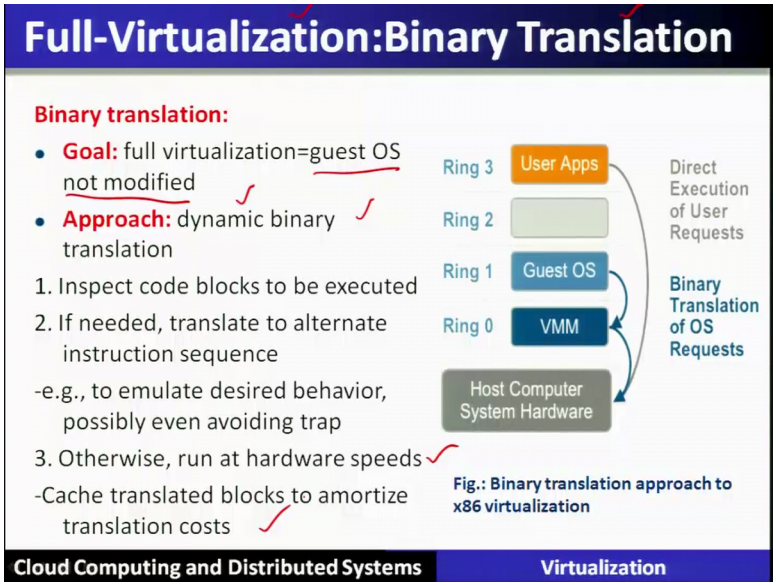
Now this particular trap intern will generate a hypervisor called to the hardware and this will be executed. So, here we have seen that there are some 17 different privileged instructions with this particular guest operating system requires to be executed with the help of the hypervisors. So, there are different mechanisms to support these 17 different privileged instructions different models of the hypervisor that we will see in the next slide.

(Refer Slide Time: 34:47)



So, this particular concept of binary translation is to rewrite this virtual machine binary to never issue those 17 different trap or instructions. This particular idea of binary translation was pioneered by Mendel Rosenblum of his Stanford which was commercialized in VMware software. And for this work which is which is renamed as reinventing virtualization has conferred an award to Rosenblum as ACM fellow.

(Refer Slide Time: 35:28)



Now, let us see the binary translation how it works which has revolutionized the VMware software and which is having a good a market in case of in this virtualization or

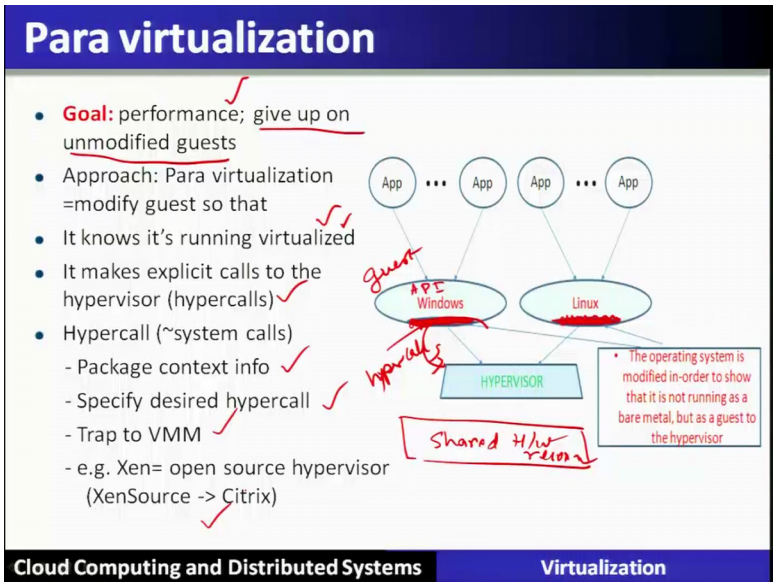
in a cloud computing scenario. So, binary translation here in the goal is that the guest operating system is not modified. If it is, then it is called a full virtualization, and this full virtualization is done through the binary translation.

So, let us see the approach that when such privileged instructions are generated it cannot be known beforehand. It will be done or it will be detected at dynamically during the runtime. So, dynamically at the dynamic time these binary translation has to be done; that means, those privileged instructions has to be identified by the hypervisor in full virtualization and it has to execute it or it has to process it. If it is not able to execute then it will feel silently. Now for that this binary translation will do, let us see the steps or the approach of battery translation for such privilege instructions in case of full virtualization where the guest operating system is not modified.

So, first it will inspect the blocks to be executed. If needed, then it will translate to an alternate instruction sequence. For example, to emulate the desired behavior possibly even avoiding the trap. So, it has to know what instruction what privilege instruction will come in the future just by inspecting the block code. Now otherwise it will run at the hardware speed.

So, the cash translated blocks are there; that means, instead of trapping one instruction, all the instruction all such instructions are inspected in the blocks and they will be given at one go that is the batches are being given to run at the hardware speed. That is all done in the binary translation to make the efficient use of it and also. The cache will translate the blocks to amortized the translation cost for several instructions if they are batch. So, isn't so that particular translation cost will be reduced in this manner.

(Refer Slide Time: 38:12)



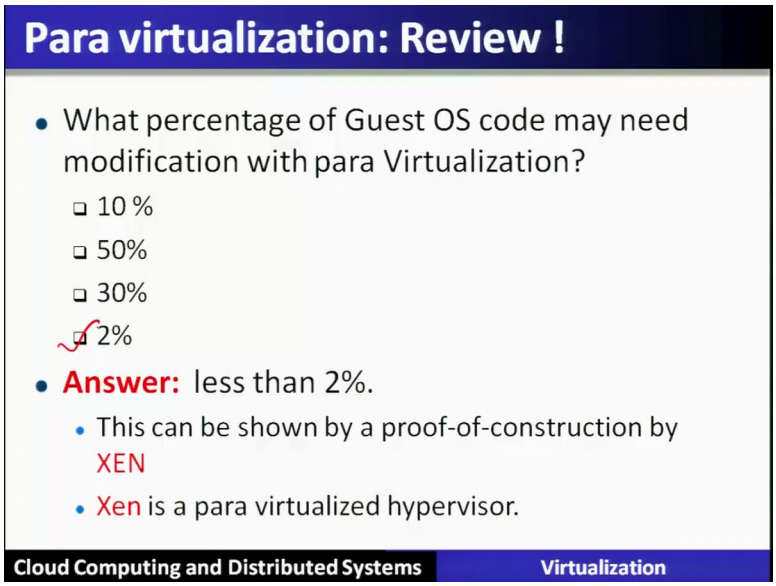
So, in contrast to the full virtualization, there is another method of virtualization which is called para virtualization. Let us see the goal of Para virtualization. Goal is basically the performance; it will give up on unmodified guest. So, the guest is no longer and modify. So, guest operating systems are to be modified to basically get the performance of this particular privileged instruction translation and execution at a higher efficiency.

So, the approach of a Para virtualization is to modify the guest operating system for example, this is the guest operating system. Only a little portion of the guest operating system is modified, but as far as the APIs are concerned which are supporting different application they are not modified, then some internal part is modified which is interfacing with the hypervisor. So, that means, only the privileged instructions which are required to be directly access to the hardware is being modified.

Now in this particular approach which is called a Para virtualization, where the guest operating system is a little modified so that all the applications which are running in this particular model knows that it is running the virtualized in a virtualized scenario. So, this means that it makes an explicit call to the hypervisor, and this is called a hypervisor call.

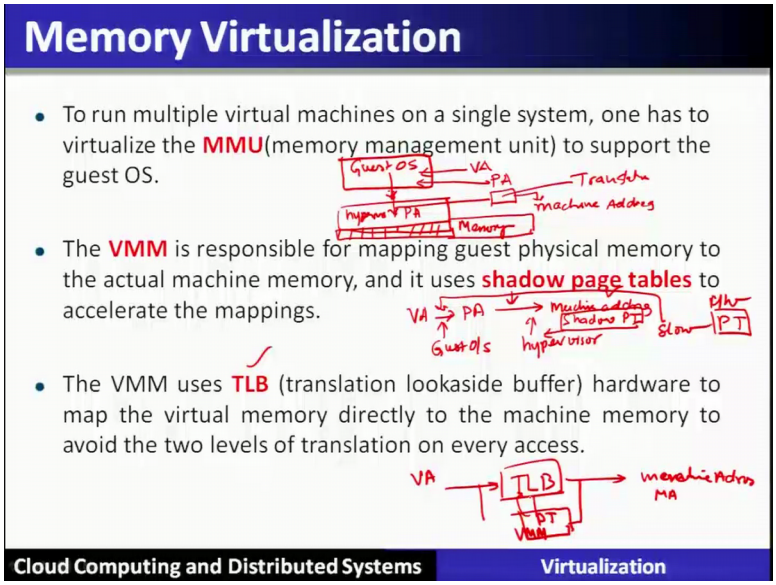
So, it will using this modified it will make a hyper call. So, hyper calls are like system calls, and they are having the packaged context information is specify the desired hyper calls, and then it will trap to the virtual machine monitor that is the hypervisor. Example of this particular Para virtualization is done in the Xen supported by the Citrix.

(Refer Slide Time: 40:29)



Now, let us see how much in para virtualization. What is the percentage of guest operating system code that needs to be modified in the para virtualization. It is more than it is 50 percent, means some of the code is of the guest operating system is to modified or it is less than 50 percent that is 30 percent code is to modified, or even it is lesser than 10 percent, or it is less than 2 percent. So, only less than 2 percent code is modified in para virtualization to it. So, it is doable, it is and this particular proof is given in Xen para virtualized hypervisor.

(Refer Slide Time: 41:12)



Memory virtualization to run multiple virtual machines on a single machine one has to virtualize, the memory management unit to support the guest operating systems. The virtual machine monitor is responsible for mapping the guest physical memory to the actual machine memory. And it uses the shadow page tables to accelerate the mapping. Virtual machine monitor uses translation look aside buffer hardware to map the virtual memory directly to the physical to the machine memory to avoid the 2 level of translation on every access.

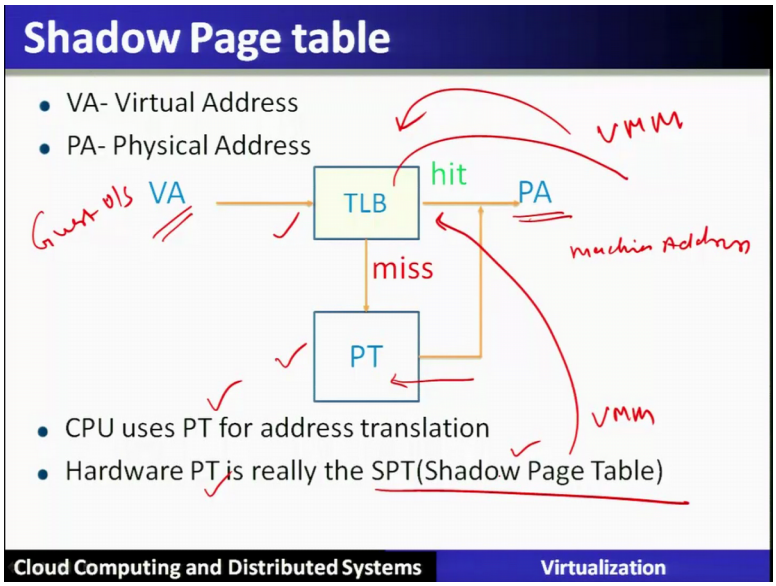
So, let us see through examples that whenever the guest operating system, q a memory address and this particular memory address will be taken by the hypervisor. So, the guest operating system will have the virtual address, and he has it is view of the physical address. This physical address when it is given to the hypervisor, because hypervisor is actually managing the hardware or the memory let us say.

So, this particular physical address is to be translated into the memory into the machine address. So, this particular there are 2 level so, virtual address is to be translated into the physical address this is done by the guest operating system. This particular physical address will be translated to the machine address, and this is to be done with by the hypervisor. So, just see that there are 2 in direction. Now here will be added over, here hence this memory operation becomes slow.

Now, let us see how this efficiency aspect is being covered here in the memory virtualization. So, the virtual machine monitor will use a concept which is called the shadow page table. So now, there are 2 page tables; one is called shadow page table which will be maintained by the hypervisor and there will be a physical page table which will be maintained by the hardware. If both are same then basically the efficiency will be achieved at the hypervisor level itself. So, virtual machine monitor uses the translation look aside buffer hardware to map the virtual memory directly to the machine memory to avoid 2 levels of translation on every access.

So, using TLB the virtual address is directly translated into the to the machine address with the help of TLB by the hypervisor. Now it may be possible that translation look aside buffer which is a cache will not have the information about this particular virtual address then it will have a miss. Then it will go into a physical or a page table, and then it will be resolved by the page table and in that process it will be updated the translation.

(Refer Slide Time: 45:46)

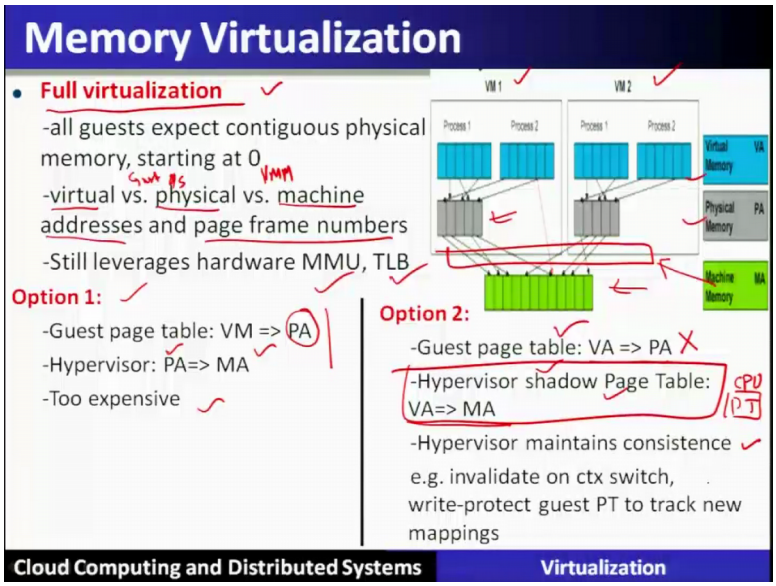


Let us see this particular method of using the shadow page table. So, this particular shadow page table will be maintained by this one, will be maintained by the virtual machine monitor. So, here whenever there is a virtual address given by the guest operating system; so, this particular hardware is a part of virtual machine monitor.

So, it will directly give the machine address. So, it will have a look into the translation look aside buffer that is a cache for that corresponding virtual address. And if it is there then it will directly generate the machine address. So, this particular TLB or the; so, basically otherwise it will a access to the page table.

CPU uses the page table for the address translation and the hardware page table which is there with the CPU is really the shadow page table which will be maintained by the virtual machine monitor. If both are same, then basically there will be a direct hit. Otherwise, if there is a miss then from the physical translation that information is basically stored in the shadow page table so that the future access to back virtual addresses can be fasten up. So, in this way the memory virtualizations achieves and efficiency as far as.

(Refer Slide Time: 47:33)



Now let us see the memory virtualization in the case of a full virtualization. Full virtualization means that the guest operating system is not modified; that means, the virtual addresses are directly given virtual addresses which are converted by the guest operating systems physical address and that is given to the hypervisor. So, there are 3 different levels as I told you the virtual address, physical address this will be translated in the guest operating system versus this physical address versus machine address. This will be translated by the hypervisor.

So, basically in full virtualization the hypervisor has to leverage the hardware knowledge of machine or a memory management unit and translation look aside buffer as I told you earlier. So, there are 2 options option one says that the guest operating system will have it is festival which will translate into a physical address. But this physical address is different than the machine address.

So, but that information is having at the hypervisor level in the full virtualization. So, this physical address will be translated into the machine address, but as we have seen that this particular 2 level translation is too expensive and the memory accesses become slower.

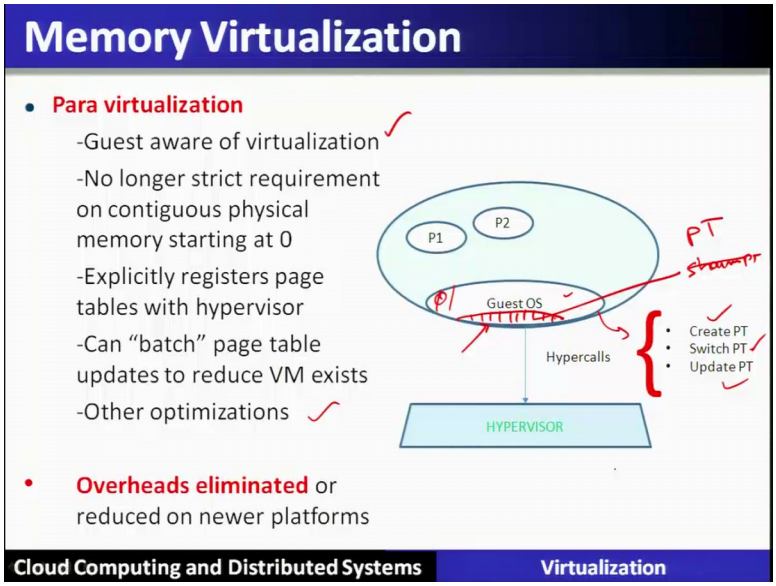
So, this particular hardware example, we can see over here this is the virtual machine one virtual machine 2. They have their virtual addresses which are generated by the process. And they can translate into the physical memory virtual memory physical memory.

When this physical memory or physical addresses are given, then basically the hypervisor comes in between and it will translate into the machine address. The hardware which in the previous slide we have seen that it will used to accelerate this.

The other option is that the guest page table that is when the guest is doing from virtual address to the physical address translation. That is done through the guest page table is it really required. So, hypervisor will now use it is shadow page table which will directly translate the virtual address into the machine address. Hence, this particular guest page table can be avoided this particular translation can be avoided directly with this particular hypervisor.

So, hyper for this the hypervisor maintains a consistence, consistency between the shadow page table and the page table which is there with the CPU at the hardware. So, both are we have seen the previous example, how they are going to be updated so that the consistency between shadow page table and the page table is maintained; so that the virtual address can directly be translated into the machine address in most of the cases.

(Refer Slide Time: 50:52)



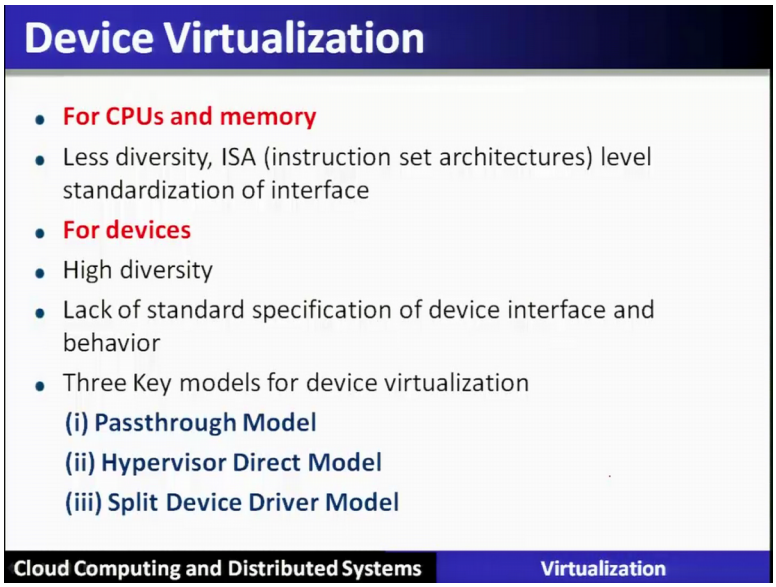
Now, in case of Para virtualization, let us see the memory virtualization; how it will takes place? Now in Para virtualization the guest is aware of the that it is running in the virtualized scenario; that means, the guest operating system will be modified a little bit that these in tables which hypervisor was earlier doing in full virtualization will be available at the guest operating system. So, guest operating system will have something

called shadow page table. So, directly it can be translated with the help of hypervisor calls.

So now the thing is here explicitly it will register the page table with the hypervisor. So, the page table will be registered with the hypervisor. Now the hypervisor can batch the page table updates to reduce virtual memory exist. And this will also have the other optimizations by way of batching it together.

Now as far as to maintain the page table at the guest operating system level, it will issue the hyper calls to create the page table and to switch the page table and to update the page table. So, all these hyper calls will continue to maintain the consistent page table at all the guest operating system levels. Now here this particular overhead will be reduced and this will make a more efficient way to access the memory using para virtualization.

(Refer Slide Time: 52:36)



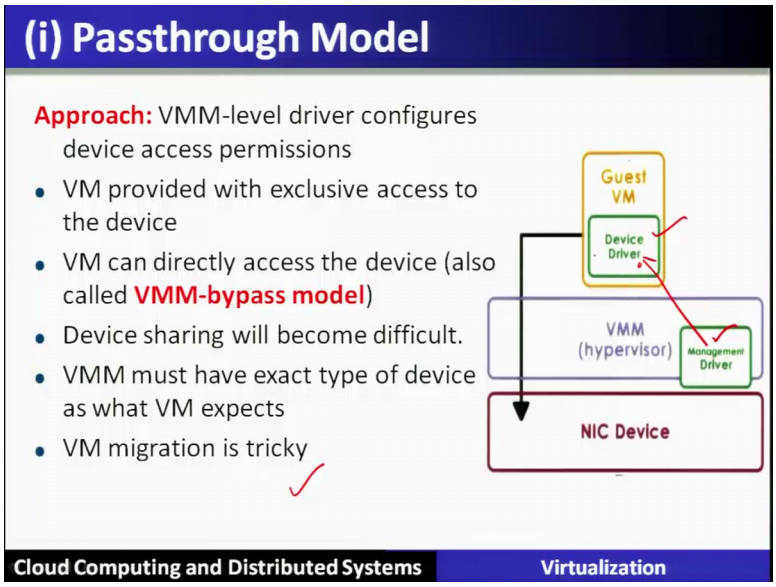
Now, we are going to see the device virtualization. After looking up the CPU and memory virtualization, which is mostly standardized as far as the interfaces are concerned and instruction set architecture given the instruction set architecture these are all standardized and they are less diverse also. So, CPU and memory virtualization has proper algorithm and proper shape and different models basically supports that.

Now, for the devices are having high diversity, and also all the devices lacks in the standard specification interfaces and the behavior. So, there is no common standard

which basically will follow for all the devices. So, different devices different types of devices will have it is own standards. And also there are different diversity as I told you different type of devices we will have different behaviors. So, as far as device virtualization is concerned is not so state forward; like, we have seen using the techniques for the CPU and the memory.

So, we are going to see the device the 3 different key models for the virtualization of devices. They are the pass through model hypervisor direct model split device driver model. Let us see one by one all 3 different type of models used in the device virtualization.

(Refer Slide Time: 54:11)

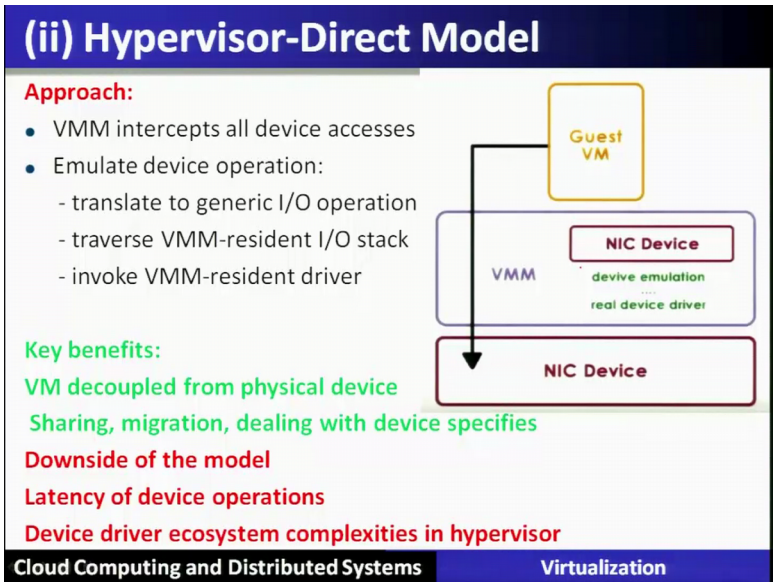


Now, first model is called pass through model. Here the approaches that virtual machine monitor level configures the devices and device access permissions to be given to the guest virtual machines to access the devices. So, virtual machines are provided with the exclusive access to the devices with the support of virtual machine monitor, which basically will have it is management routine which will basically supports this particular direct access which will bypass the virtual machine hypervisor. So that is the guest operating systems, the guest virtual machines can directly access the devices and in an efficient manner.

The device sharing is very difficult in this particular scenario. So, virtual machine monitor must have exact type of device as what the virtual machine expects. Now here

the virtual machine migration is tricky by because, the drivers of guest virtual machines or the guest operating system directly are accessing or tied up with the devices. So, the migration is a more difficult or a tricky. So, in the next model we will see how this problem is going to be solved.

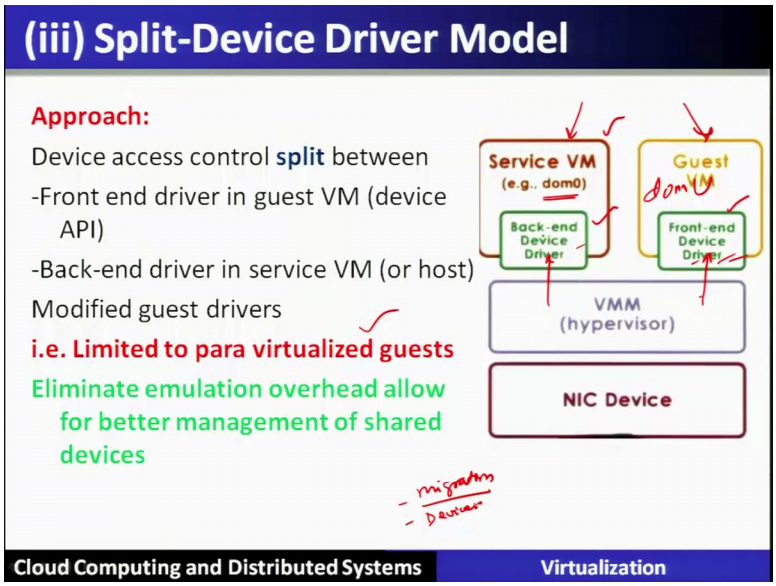
(Refer Slide Time: 55:55)



Hypervisor direct model approach virtual machine monitor intercepts all device accesses. Emulate the device operations in the sense it will translate to the generic I/O operations, and traverse the virtual machine monitor resident I/O stack. Invoke virtual machine monitor resident driver, and it will allow the access; that means, the devices are sometimes or it is virtualized. So, the guest operating system is supporting or using supported by the proper way of generic device operations through the virtual machine monitor.

The key benefits here are that virtual machine is decoupled from the physical device, unlike in the previous method which we have seen. And this will support the sharing migration are dealing with the device is specifies now downside of the model is that the devices are accessed through the virtual machine monitor. Hence it will give it will add to the latency for the devices for the device operations in the virtual machine accesses to the device. So, device driver ecosystem is the adding the complexity in the hypervisor level.

(Refer Slide Time: 57:30)

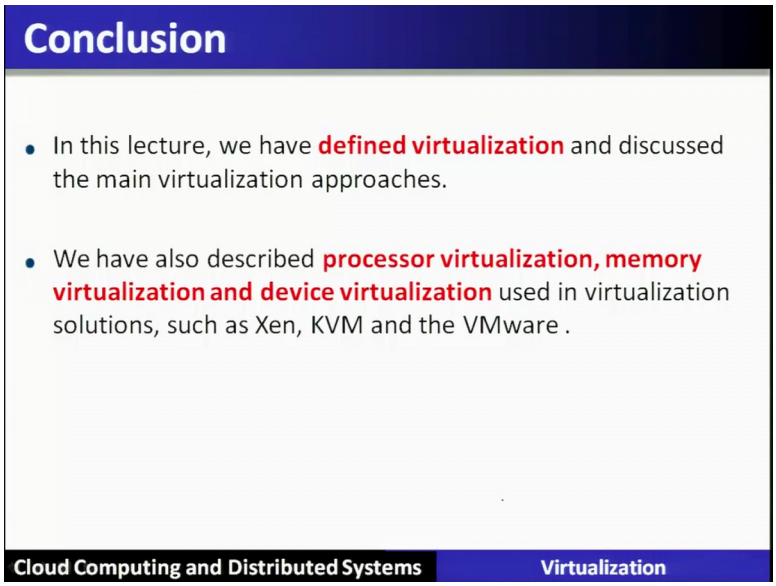


Now, the next method is called is split device driver method. Here the device access control is a split between the front end driver in the guest VM and back end driver in the service VM or the host let us see through this particular example. This device access control is split into 2 parts. The first one is called front end device driver in the guest VM that is through the device API. The other one is called back end device driver in the service VM. So, service VM as you know that it is called domain 0 and the guest VM or the guest operating system or the VMs is called domain U.

Now, here this particular split device driver model is limited to the para virtualized guest operating system, why because this is going to be para virtualized. Now here it will eliminate the emulation overhead and it will allow for better management of the shared devices, because hypervisor will support the guest operating system through the front end device drivers. And also the service VM that is the privileged VM which will provide the back end device driver together they will basically allow to access the device driver directly.

So, with this is split device driver model, all the important aspects that is the migration is possible here in this case, and also it will support using para virtualized method. The device accesses or being given to the virtual machines, so, that; they may access it efficiently.

(Refer Slide Time: 59:38)



So, conclusion in this lecture we have defined virtualization and discuss the main virtualized virtualization approaches. We have also described the processor virtualization, memory virtualization and device virtualization, used for virtualized solutions. And also we have shown some of the solutions such as Xen KVM and VMware virtualization methods or hypervisors.

Thank you.