Impact Factor (SJIF): 4.542

International Journal of Advance Research in Engineering, Science & Technology

e-ISSN: 2393-9877, p-ISSN: 2394-2444

Volume 4, Issue 4, April-2017

A Highlight of Different Techniques Used in Virtualization to Improve Performance of Hypervisor

Prof. Gopi Bhatt* **Prof. Aniruddh Kurtkoti*** **Prof. Chinmay Joshi*** **Prof. Prerak Thakkar*** **Prof. Siddharth Shah***

*Assistant Professor, Department of Computer Engineering, A. D. Patel Institute of Technology

Abstract

Virtualization and cloud computing are not separate parts. Virtualization is the base whereas cloud computing is the entire concept. So if we consider cloud computing is as whole body then virtualization is the heart of that body. Virtualization in laymen's language is defined as follows: Virtualization basically allows one computer to do the job of multiple computers, by sharing the resources of a single hardware across multiple environments. In this paper we have highlighted techniques used to achieve virtualization of various resources to improve performance of hypervisor. Also some of the advance concepts of virtualization has been denoted by the authors.

I. Introduction

A. Virtualization

Virtualization basically allows one computer to do the job of multiple computers, by sharing the resources of a single hardware across multiple environments.

In technical term Virtualization is a layer mapping its visible interface and resources onto the interface and resources of the underlying layer or system on which it is implemented providing features like Abstraction, Replication and Isolation [1, 10]. Important key points about virtualization are abstraction, replication and isolation. Abstraction refers to the creation of miniature model of actual physical environment that fit in a box known as virtual container. Replication refers to have more than one copy of virtual container with the predefined boundary. Isolation refers to the access of kernel and hardware through virtualization layer.

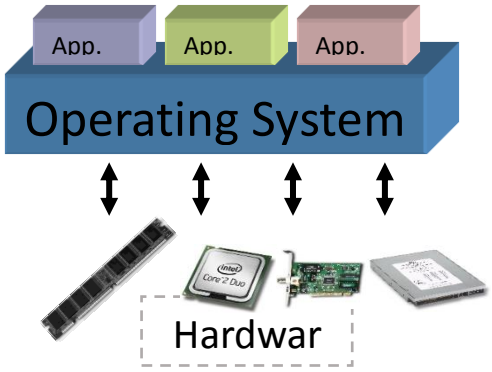

Figure 1: Regular system without virtualization[#]

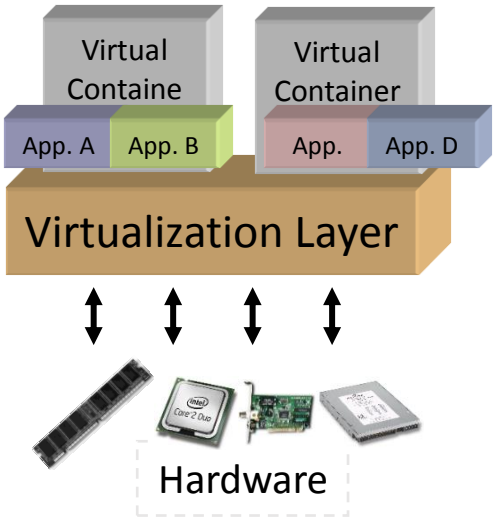

Figure 2: System with Virtualization[#]

There exists three types of virtualizations. Full virtualization, para virtualization and hardware assisted virtualization. This virtualization types are formed on the basis of the execution parameter that are used during virtualization environment. Before understanding virtualization, we must have to understand layered structure of operating system. Older operating system had monolithic structure which had single layer whereas modern operating systems are using layered structure in which it contain multiple layers which are separated based on priority and privileges. In layered structure the innermost ring i.e. ring0 is the most privileged ring which contains kernel module, ring 1 and 2 are having system libraries and APIs and ring 3 has user apps.

- **Full Virtualization[4,7]**:

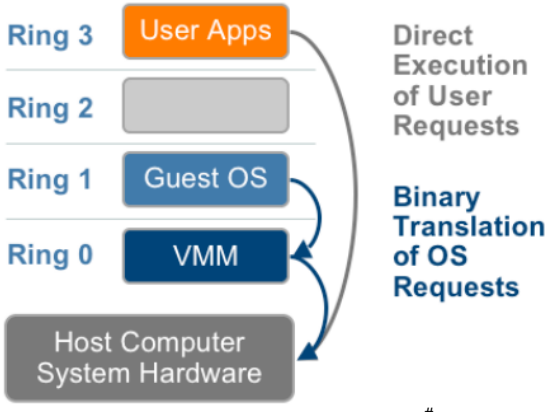

Figure 3: Full Virtualization[#]

In full virtualization, host will have its base operating system, hypervisor resides in ring 0 within kernel module and guest operating system resides in virtual machines in ring1. Now as in traditional operating system, processes at ring1 and 2 are scheduled by scheduler, the guest operating system which resides in ring 1 are considered as normal processes and will be scheduled by scheduler. Any applications running in guest operating system needs to be binary translated to access hypervisor whereas user apps can directly communicate with kernel module.

In full virtualization, as all the device drivers are maintained and handled by base operating system, we can run guest operating systems without modifying it on the hypervisor. Also migration of virtual machine from one host to other would be easier hence provides good portability of guest operating system.

As guest operating systems are considered as processes, they will be treated as normal process by scheduler which hinders the performance and execution of virtual machines are slower than that of actual physical environment. Also because of binary translation performance of full virtualization is degraded.

- **Para-virtualization[4,7]**:

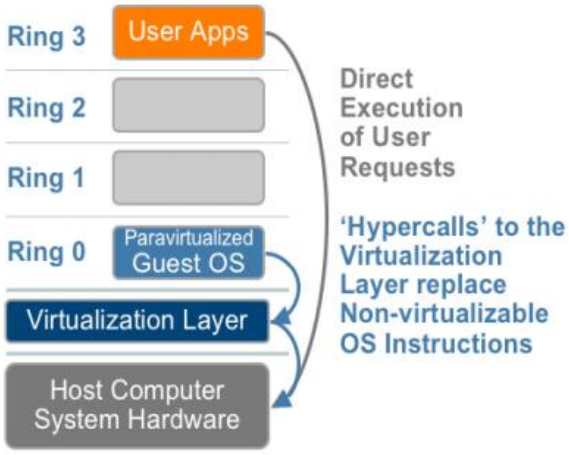

Figure 4: Para-virtualization[#]

In para-virtualization, there is no base operating system, instead, hypervisor or virtual machine monitor is considered as operating system. Hypervisors resides in kernel module within a base operating system and guest operating systems resides in ring 0 which is the most privilege layer.

User applications can send request to kernel module with system calls, similarly, guest operating systems can send request to hypervisor through hyper calls i.e. all privilege instructions from the guest operating systems are converted into hyper calls. Para-virtualized guests runs in most privilege mode hence context switching is faster than that of full virtualization.

As there is no base operating system, all the device drivers must be installed in guest operating systems and hence we need to modify guest operating systems. The advantage of para-virtualization are fast, provides good isolation between guests, higher performance than full virtualization, direct hardware access is possible. Disadvantage of para-virtualization is requirement of modifying guest operating system to execute and communicate with the hypervisor.

- **Hardware Assisted Virtualization[4,7]:**

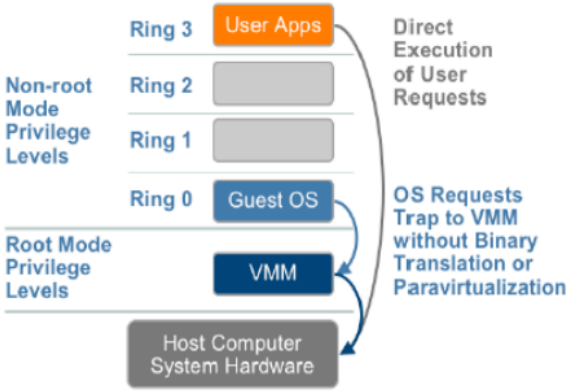

Figure. 5 Hardware Assisted Virtualization[#]

To understand hardware assisted virtualization, we will take an example of threading: if para-virtualization is multithreading then hardware assisted virtualization is the hyper threading. Here, multithreading is software approach whereas hyper threading is hardware approach for implementing threads. Similarly para-virtualization is software based virtualization technique whereas, hardware assisted virtualization is the hardware based virtualization approach.

In this case, hardware are modified in such a way that they can create virtual environment. This provides the flexibility of having assembly instruction like TRAP, which provides faster context switching of guest operating systems. Intel-VT and AMD-V are some examples of hardware assisted virtualization.

Here, as virtualization is provided by hardware itself, no need for binary translation and hyper-calls as in case of full virtualization and para-virtualization accordingly.

B. Hypervisors[1,4,7]:

The heart of virtualization is hypervisor. Two types of hypervisors are there: Type1 and Type2. Type1 hypervisors run directly on hardware whereas type2 hypervisors run as an application on host operating system. VMware ESX, Microsoft Hyper V, Red Hat Linux KVM etc. are type1 hypervisors. Oracle virtual box, VMware workstation etc. are examples of type2 hypervisors.

- **Type1 Hypervisors:**
  Type1 hypervisors are used in para-virtualized environment and hardware assisted -virtualization environment. Type1 hypervisor runs directly on physical hardware. Because there is no intermediary layer between hypervisor and hardware they are also known as bare metal hypervisor.

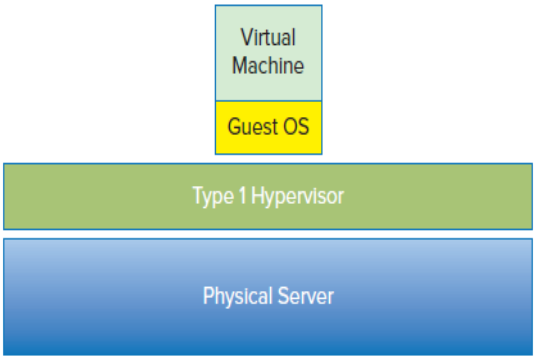

Figure. 6 Type 1 Hypervisor[#]

The advantage of type1 hypervisor as compared to that of type2 hypervisor is isolation. If one virtual machine crashes, the hypervisor and other virtual machines will be unaffected from that. Similarly malicious guest do not harm other virtual machines and hypervisor because of predefined boundaries. Here less overhead is required for guest operating system and hence can accommodate more number of virtual machines on hypervisor. So in financial term, it is better than type2 because it doesn't require cost of base operating system.

- **Type2 Hypervisors:**
  It runs on atop of traditional operating system. Here hypervisor runs as an application on host operating system. All the device drivers are handled by base operating system, so no need to modify guest operating systems.

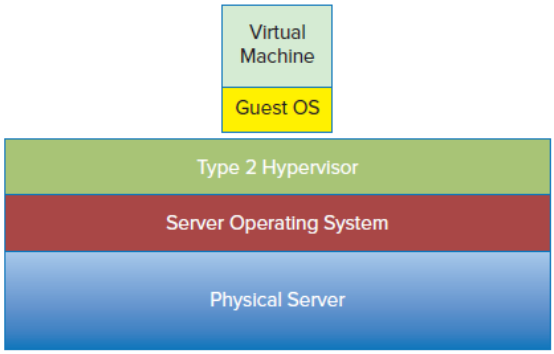

Figure 7: Type 2 Hypervisor[#]

When user applications in guest operating system wants to execute in kernel, first it will send request to hypervisor, which forwards that request to kernel and response follows the reverse path. So execution of guest are slower compared to type1. If the host operating system has issues then that will affect all guest operating systems and hypervisor.

II. Hardware management

A. CPU Virtualization [1,4,10]:

In reality, CPUs are never virtualized, instead time slices are given to each virtual machines (VM). As shown in figure 8, if there are two virtual machines, then if scheduler schedules first virtual machine, its instruction will be executed by CPU and response is given to first VM.

If scheduler schedules second VM the same is the case. It becomes complex if multicore processors are used where each core is known as vCPU. Consider an example of 4 core processor. If two VMs are there and each are provided 2 cores then there is no need of scheduling VM as they can execute parallel on CPU. But if we allocate 3 core to each VM then hypervisor must have to schedule VM.

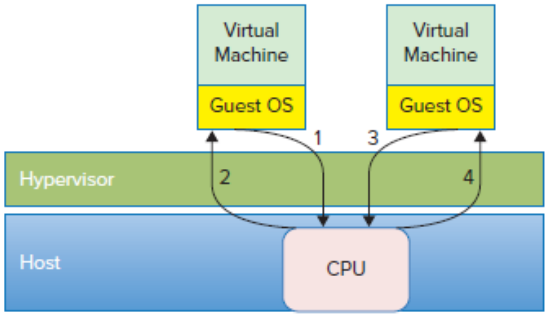

Figure 8: CPU Virtualization[#]

B. Memory Virtualization:

It is the task of hypervisor to provide slice of memory to each VM. As shown in figure 9, first VM is provided 4GB RAM and second VM is provided 2GB of RAM.

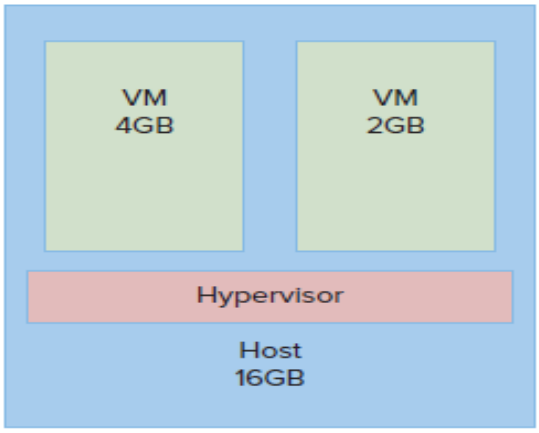

Figure 9: Memory Virtualization#

Moreover, the hypervisor itself reserve some memory to perform operating system related task. VM also reserve extra memory for maintaining data structures like memory management table which contains virtual memory address and related physical address. Here the flexibility is provided to hypervisor that it can increase or decrease memory according to the need of VM. Three techniques are used for memory management.

- Ballooning
- Memory over commitment
- Page sharing

a. Ballooning:

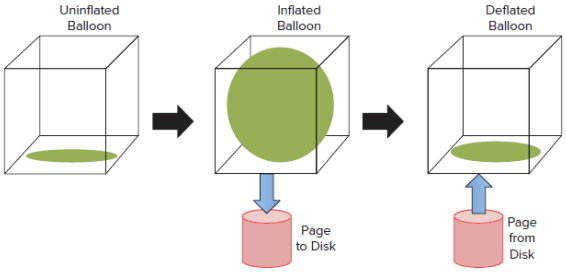

Figure 10: Ballooning Technique#

When there is a memory contention for VM the balloon will be inflated and when it complete its task the balloon will be deflated. Here balloon is a portion of memory. So when there is no space for new pages, some of the pages will be moved from RAM to Disk and vice versa.

b. Memory Over Commitment:

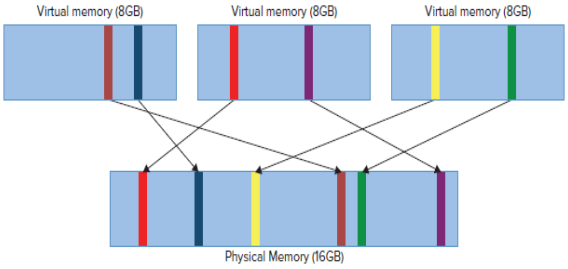

Figure 11: Memory over Commitment Technique#

It is generally seen that VM utilizes only some portion of memory which is allocated to it. For example, if 4GB memory allocated to VM, then it uses only 2GB or 2.5GB of it at any instance of time. Hence hypervisors can do over commitment to provide enough memory to each VM. As shown in fig actual physical memory available is 16GB but the hypervisor can run 3 VM each with 8GM of RAM so here it has done over commitment of 8GB of memory.

The advantage of this scheme is that when one VM is underutilized, the memory can be given to other VMs.

c. Page Sharing:

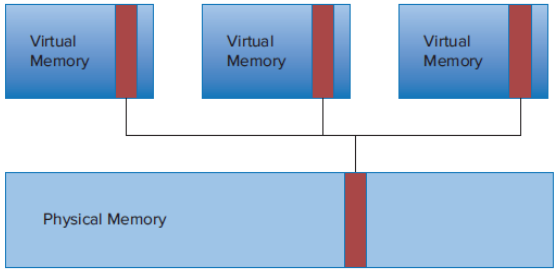

Figure 12: Page Sharing Technique[#]

It is generally seen that, VM running on hypervisors can support the same guest operating systems. For example, if 10 VMs are running on hypervisor then out of those 10, 6 to 7 VM may run one operating system. This operating system needs the same system files. Hence the system files accessed by one will be made available to the other VMs also, which reduces the disk read operations that is known as page sharing.

C. Storage Virtualization [1,4,10]:

There are two types of storages. NAS and non NAS which includes DAS and SAN. In NAS storage is available in network and hence virtual NIC sends a request of block to NIC emulator which forwards the request to actual physical NIC and through TCP IP protocol suit it will fetch the block and provided to VM in reverse path as shown in figure 13.

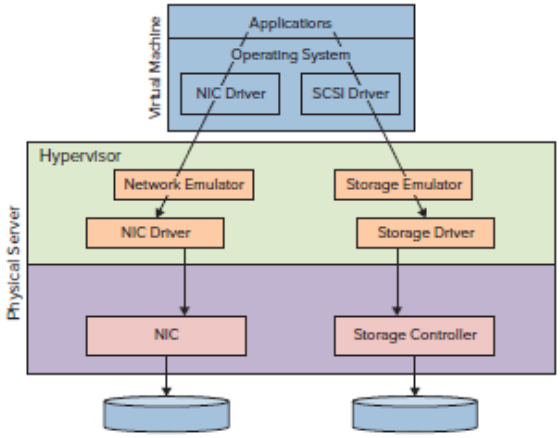

Figure 13: Storage Virtualization[#]

If it used DAS or SAN, request for the block is forwarded to storage controller through virtual SCSI driver and storage emulator. The response will follow the reverse path as shown in figure 13.

D. Network Virtualization [1,4,10]:

When VM wants to communicate with outside world that refers to as external communication, if it wants to communicate with other VM on the same host it is referred to as internal communication.

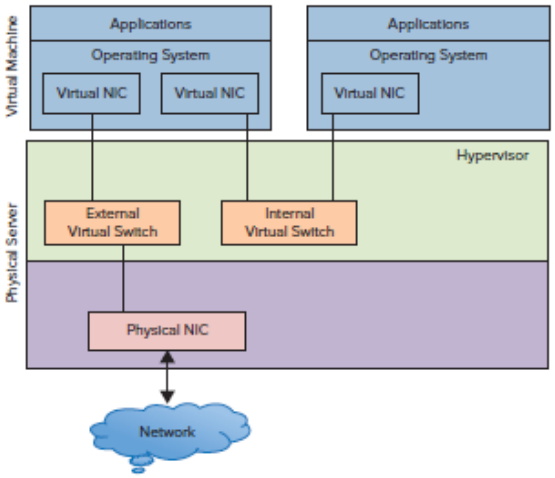

Figure. 14 Network Virtualization[#]

For this, the hypervisor maintains external virtual switch and internal virtual switch. When request is for outside world, external virtual switch forwards the request to actual physical NIC which forwards the request to outside world. When it wants to communicate with other VM, internal virtual switch will forward the request to other VM. Response will follow the reverse path.

III. Conclusion

The concept of virtualization is in use since long time. Different techniques have been implemented since keeping in mind the performance in virtualization. The key focus for performance improvement is on memory rather than on other hardware components like CPU, Network, storage etc. With above mentioned example, we can improve the performance of Hypervisor, just by changing memory management policies.

IV. References

1. Virtualization Essentials By Mathew Portnoy, Wiley Publications 2012
2. Cloud Computing Bible By Barrie Sosinsky, Wiley Publications 2011
3. Cloud Computing: Principals and Paradigms,
   By Rajkuma Buyya, Wiley Publications 2012
4. Virtualization: A Manager's Guide by Dan Kustney, O'Reilly Publications 2013
5. Grids, Clouds and Virtualization, By Massimo Cafaro, Springer Publications 2012
6. Definitive Guide to Xenserver, By David Chisnail, Prentice Hall Publications 2006
7. Understanding Full Virtualization, Para-virtualization and Hardware Assist, White Paper by Vmware
8. Mastering Cloud Computing, By Rajkumar Puyya, Tata McGraw Hill Publications 2013.
9. Citrix Systems Inc., XenServer, http://www.citrix.com/XenServer, 2010.
10. Vmware., Oracle VM, http://www.vmware.com/technology/products/vm, 2010

11. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility, Future Generation Computer Systems, 2009.
12. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, and R. Katz, Above the Clouds: A Berkeley View of Cloud Computing, UC Berkeley Reliable Adaptive Distributed Systems Laboratory White Paper, 2009.
13. B. Sotomayor, R. S. Montero, I. M. Llorente, and I. Foster, Virtual infrastructure management in private and hybrid clouds, IEEE Internet Computing, September/October, 2009.
14. J. Broberg, S. Venugopal, and R. Buyya, Market-oriented Grid and utility computing: The state-of-the-art and future directions, Journal of Grid Computing, 2008.
15. I. Foster, Globus toolkit version 4: Software for service-oriented systems, Journal of Computer Science and Technology,
16. R. Buyya and S. Venugopal, Market oriented computing and global Grids: An introduction, in Market Oriented Grid and Utility Computing, R. Buyya and K. Bubendorfer (eds.), John Wiley & Sons, Hoboken, NJ, 2009
17. P. Barham et al., Xen and the art of virtualization, in Proceedings of 19th ACM Symposium on Operation Systems Principles, New York, 2003
18. M. D. de Assuncao, A. di Costanzo, and R. Buyya, Evaluating the cost benefit of using cloud computing to extend the capacity of clusters, in Proceedings of the 18th ACM International Symposium on High Performance Distributed Computing (HPDC 2009).
19. D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper, Capacity management and demand prediction for next generation data centers, in Proceedings of IEEE International Conference on Web Services, 2007
20. S. Venugopal, J. Broberg, and R. Buyya, OpenPEX: An open provisioning and EXecution system for virtual machines, in Proceedings of the 17th International Conference on Advanced Computing and Communications (ADCOM 2009), Bengaluru,India, 2009.