Journal of Science and Technology
ISSN: 2456-5660 Volume 06, Issue 01, Jan-February 2021

www.jst.org.in *DOI: https://doi.org/10.46243/jst.2020.v6.i1.pp01-08*

Virtualization Effects on The Security

Dinesh Kumar Agarwal [1], Dr Vikas Kumar [2]
[1](SRM Institute of Science and Technology, Department of Information Technology, Modinagar)
[2](Subharti university, Department of Mathematics, Meerut)
[1]Corresponding Author: dk75mnr@gmail.com

To Cite this Article
Dinesh Kumar Agarwal and Dr Vikas Kumar , "Virtualization Effects on The Security", Journal of Science and Technology, Vol. 06, Issue 01, Jan-February 2021, pp01-08

Article Info
Received: 28-07-2020 Revised: 27-09-2020 Accepted: 12-10-2020 Published: 27-10-2020

Abstract: *Virtualization is a technology which provides the efficient and effective usage of computing system .the computing system is not only a single personnel system but also a system consisting of multiprocessors systems. To use the system efficiently & effectively. We will be using virtualization technology .now a days business world is very competitive .customers demanding for new products this technology will provide opportunity to produce new products with the same resources with the existing infra structure. Within this technology one important issue is the "security". It has changed the classical definition of security. The classical definition suggest that try to detect the security violations at the first place and then take the action but with the virtualization we can create the separate environment for security related application that would not effect the trusted information with the system .The concept has change the scenario of security related solutions by inviting the malicious from the communication system & executing them with the separate virtualization machine environment that would not affect the trusted information with the same system*

Keywords*: Visualization, Multiprocessors, Security, Application, System*

I. Introduction

Madnick and Donovan first proposed the use of virtual machines to increase security in multiprogrammed operating systems. They recognized that the complexity of such systems increased the probability that an exploit could be found that allowed a malicious user to disturb the operating system. Such an exploit would allow the user to directly affect another user's data, by bypassing the operating system's built-in protection. They proposed running a VM per user, each with a private copy of the operating system. This increases security because now a malicious user would have to exploit vulnerability in both the operating system and the VMM in order to violate protection. Assuming independent failures in both the operating system and the VMM, this probability is vastly smaller that that of the OS alone.The IBM KVM/370 [33] and the VAX VMM Security Kernel [40] utilized virtual machines in a similar manner. Each was designed to support military-level security, which mandated separate physical machines for work on different security classifications. These projects demonstrated that equivalent security could be gained using VMs. Despite these benefits, interest in VMs for security.

The use of virtual machines with the introduction of the Disco project revived the notion of using virtual machines for strong isolation. Their goal was to support the multiplexing of diverse Internet services on the same physical machine. In this setting, strong isolation is desired, services potentially require different operating environments, and no sharing of data is required between services.

VMs is the perfect solution under such conditions.VMs is able to scale the number of simultaneous VMs on a single machine to the hundreds through the extensive use of Para-virtualization, modifying the virtual architecture (which requires changes to guest operating systems) to allow for increased scalability.

II. What is Virtualization

When computer systems were first invented they were mammoth systems that were large & expensive to operate. Due to their size, expense & demand for their usage, Computer systems quickly involved to become time sharing systems so that multiple users could use them simultaneously. As computers became more prevalent however, it became apparent that simply time sharing a single computer was not ideal. For example:

misuse of the system could easily bring the computer system to a halt for all users. For organizations that could afford it , they simply purchased multiple computer systems to solve these problems. Having multiple Computer systems proved beneficial for the following reasons:

Isolation: - In many situations it is beneficial to have certain activities running on separate system. For example an application may be known to contain bugs to interfere with other applications on the same system. Placing the application on a separate system guarantees it will not effect the other applications.

Performance: - Placing an application on its on system allows it to have exclusive access to the system's resources & thus have better performance than if it have to share the system with other applications. Most organizations at the time were not capable to purchase multiple computer systems. It was also recognized that purchasing multiple computer systems was often wasteful, as it is hard to keep them busy all the time. However, having multiple computers obviously had it's benefits, so taking cast & waste into consideration **IBM** in 1960's began developing the first virtual machines that allowed one Computer to be shared as if it were several.

*Virtual Machines***:-**IBM defined the virtual machine as a fully protected & isolated software abstraction with the looks of a computer system's hardware (fig 3.1). IBM designed their virtual machine systems with the goal that applications, even operating systems, run in the virtual machine would behave exactly as they would on the original hardware. Now, the term encompasses a large range of abstractions for example J.V.M. That does not watch an existing real machine

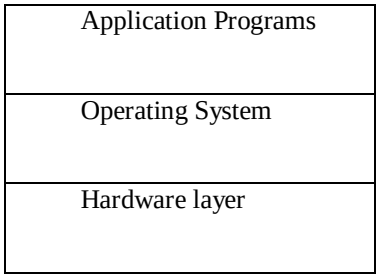

Figure 2.1 Modern Computer System Structure

2.1 Levels of virtualization

A modern computer system is Composed of layers beginning with hardware & including layers of an operating system & application Programs running on top of the operating system .

Virtualization software abstracts virtual machines by interposing a layer at various places in the system. According to Position of virtualization layer in the computer system we are having following types of virtualization

2.1.1 Hardware – Level Virtualization

Here the virtualization layer sits right on top of the hardware exporting the virtual machine abstraction. Because the virtual machine looks like the hardware, all the software written for it will run in the virtual machine.

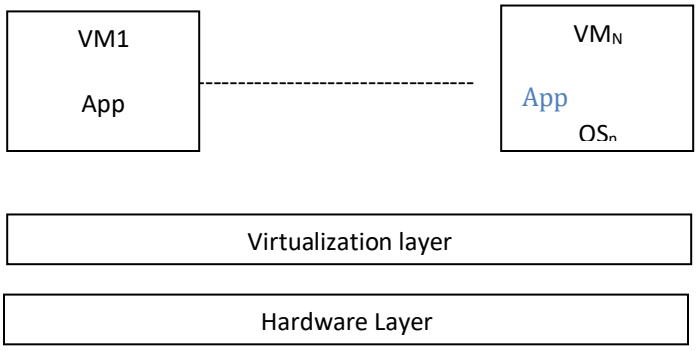

Figure 2.2 Hardware Level Virtualization

Operating System-Level Virtualization
In this case the virtualization layer sits between the operating system & the application programs that run on the operating system. The virtual machine runs applications, or sets of applications, that are written for the particular operating system being virtualized.

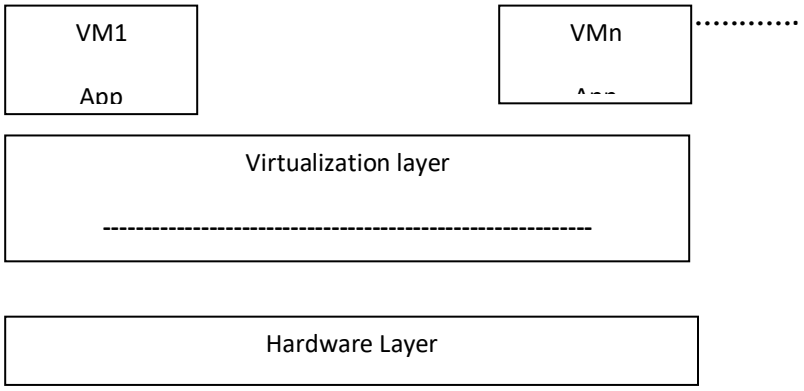

Figure 2.3 Operating System Level Virtualization

2.1.3 High-Level Language-Virtualization: In high level language virtualization, the virtualization layer sits as an application program on top of an operating system. The layer exports an abstraction of the virtual machine that can run programs written & compiled to the particular abstract machine definition. Any program written in the high level language & complied for this virtual machine will run in it.

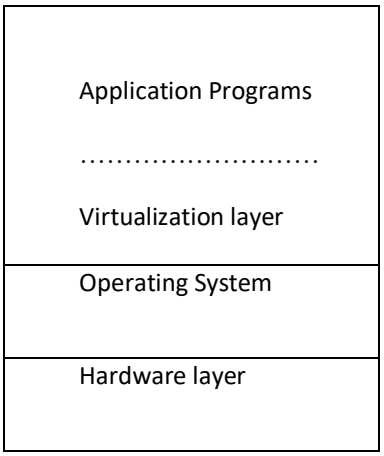

Figure 2.4 High-Level languages Level Virtualization

2.2. General Advantages of Virtual Machines:
There are different types of Virtual Machines providing different -2 types of characteristics. But all of them share the following advantages.

Software Compatibility: The Virtual machine provides a compatible abstraction so that all software written for it will run on it. For example, a hardware level virtual machine will run all the software, **o**perating systems & applications written for the hardware.

Isolation: The virtual machine abstraction isolates the software running in the virtual machine from other virtual machines & real machine. This isolation provides that bugs or hackers can be contained within the to virtual machine & can not adversely effect other parts of the system.

Encapsulation: The software layer exporting the virtual machine abstraction is an example of level of indirection this layer can be used to manipulate & control the execution of the software in the virtual machine. It can also use this indirection to enhance the software or to provide a better execution environment.

Performance: Adding a software layer to a system adds overhead, which can adversely affect the performance of the software running in the virtual machine. But the benefits of the virtual systems outweigh any overhead that they introduce.

2.3. Virtual Machine Monitor Benefits:

Virtual Machine Monitors normally allow a system manger to configure the environment in which a VM will run. VM Configurations Can be different from real machine. For example, a real Machine may have 32 MB of Memory, but a Virtual Machine may have only 8 MB. They allow concurrent execution of different operating systems on the same hardware. They allow users to isolate untrusted applications. For example, a program downloaded from internet could be tested in a VM. If the program contained a virus the virus would be isolated to that VM. When upgrading an operating system or migrating to a different operating system, it is common that some applications do not work on the new operating system. VMM allows both the new & old operating system on the same hardware. VMM allows multiple copies of an operating system running on a scalable computer.

It also allows these operating systems to share resources each other.

III. Virtualization Impact on The Security

3.1 Virtual Machine & Security

Security is an important factor If the programs of Independent and possibly malicious users are to coexist on the same computer system. A combined virtual machine monitor/operating system (VMM/OS) approach to Information system isolation provides substantially better software security than a conventional multi programming operating system approach. This added protection is derived from redundant security using Independent mechanisms that are Inherent In the design of most VMM/OS systems.

During the past decade the technique of multiprogramming (the concurrent execution of several Independent programs on the same computer system) has been developed to take full Advantage of medium- and large-scale computer systems (cost economics, flexibility, easy of operation, hardware reliability and redundancy, etc.). Unfortunately, in transferring physically isolated information systems

Fig 3.la to physically shared information systems Fig 4.lb, we must cope with the problems of operating system compatibility, reliability, and Security. the Virtual Machine approach provides effective solutions to these problems.

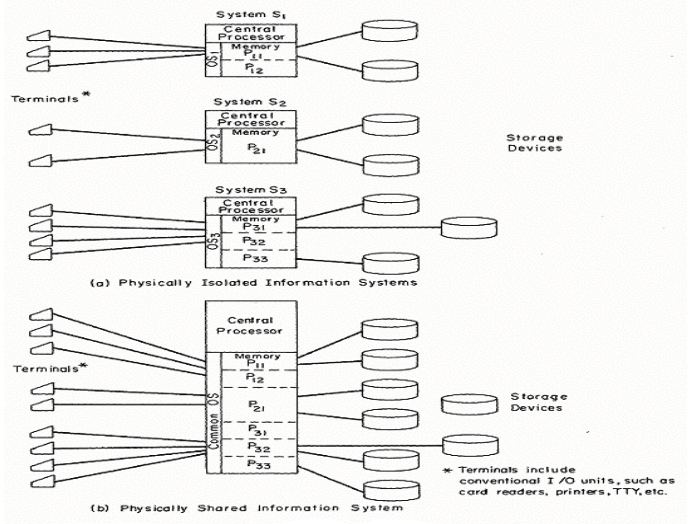

Figure 3.1 Isolated & Share Information System

3.2 Virtual Machine Approach for Isolation and Compatibility:

Virtual machine may be defined as a replica of a real computer and real computer System simulated by a combination of a Virtual Machine Monitor (VMM) software program and appropriate hardware support. Thus, a VMM can make one computer system function as a multiple physically isolated systems Figure 3.2. A VMM do this by controlling the multiplexing of the physical hardware resources such that the telephone company multiplexes communications enabling separate and, hopefully, Isolated conversations over the same wires.

VMM is different from a conventional operating system. A VMM restricts Itself to the task of multiplexing and allocating the physical hardware, this presents an Interface that appears Identical to a "bare machine". In fact, it is necessary to load a conventional operating system Into each virtual machine in order to accomplish useful work. This latter fact provides the basis for the solution to the operating system compatibility problem. Each virtual machine is controlled by a separate, and If necessary different, operating system.thls solution need the extra additional overhead In information system operation, but this overhead can be kept rather low. Depending upon the precise economics and benefits of a large-scale system. VMM approach is often preferable to the operation of the multiple physically isolated real| systems.

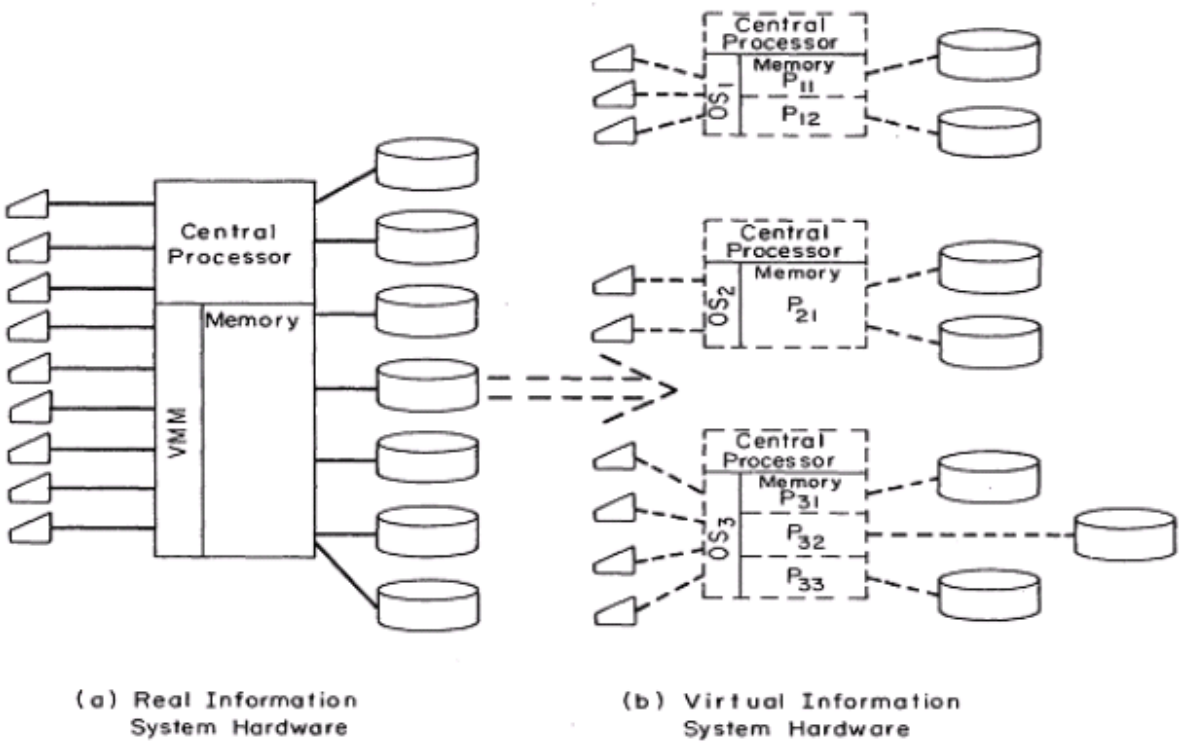

Figure 3.2 Real and Virtual Information Systems

3.3 Security & Reliability in a Virtual Machine Environment:

In the above virtual machine approach solves the OS compatibility problems by allowing different operating systems to run and coexist on the same computer at the same time. In the View of security and reliability in a virtual machine environment. We can show that the virtual| machine approach results in a system that Is much less susceptible to such failures than a Conventional multiprogramming operating system. The problems of software reliability and security are quite similar. A reliability failure is any action of a user's program that causes the system to cease correct operation (e.g., "stops" or "crashes"), a security failure Is a form of reliability failure that allows one user's program to access or destroy the data or programs of another Isolated user or gain control of the entire computer system.

3.3.1 Contemporary Operating System Environment

Most contemporary operating systems, In conjunction with appropriate hardware support, provide mechanisms to prevent reliability and security failures. In this project I am only concerned about Isolation security (no user is allowed access to any other user information). The problem of generalized controlled access (a user Is allowed restrictive access to another user's Information) Is much more difficult but, such facility Is not needed for this environment Illustrated In Figure I. Under "Ideal" circumstances, most current operating systems can provide Isolation security. 0S/360, for example, uses the System/360's lock and key protection to insulate users from each other and from the operating system. The supervisor/problem state modes further prevent users from "gaining control" of the system. Thus, it should be possible to isolate users.

Figure3.3 (a) illustrates the coexistence of multiple programs on the same Information system. Such a system is susceptible to security violations if a single hardware or software failure were to. Occur. Typical modern operating systems consist of thousands, possibly millions, of Instructions. The user programs Interface with the operating system through hundreds of parameters (e . g . , supervisor calls, program Interrupts , I / 0 requests and interrupts,) . At the current time there is no known way to systematically validate the correct functioning of the operating system for all possible parameters. In fact, most systems tend to be highly vulnerable to Invalid parameters. The operating system, running with protection disabled and assuming that the address parameter corresponds to a user's data areas transfers the return data to that location. If the address provided actually corresponds to locations with in the time the operating system, the system can be made to destroy or disable It self. Host system attempt to detect this kind of error but there are many other techniques. Figure 4.3 see some of the factors contributing to the problem. In order to provide sufficient functionality to be effectively for a large and heterogeneous collection of programs, the operating system must be quite comprehensive and, thus, more error. In *general, a* single logical error in the operating system software can invalidate the entire security mechanism. Furthermore, as depicted In Figure 4.3(a), there Is no more protection between the programs of differing user groups or the operating system than there is between the application programs of a single user group.

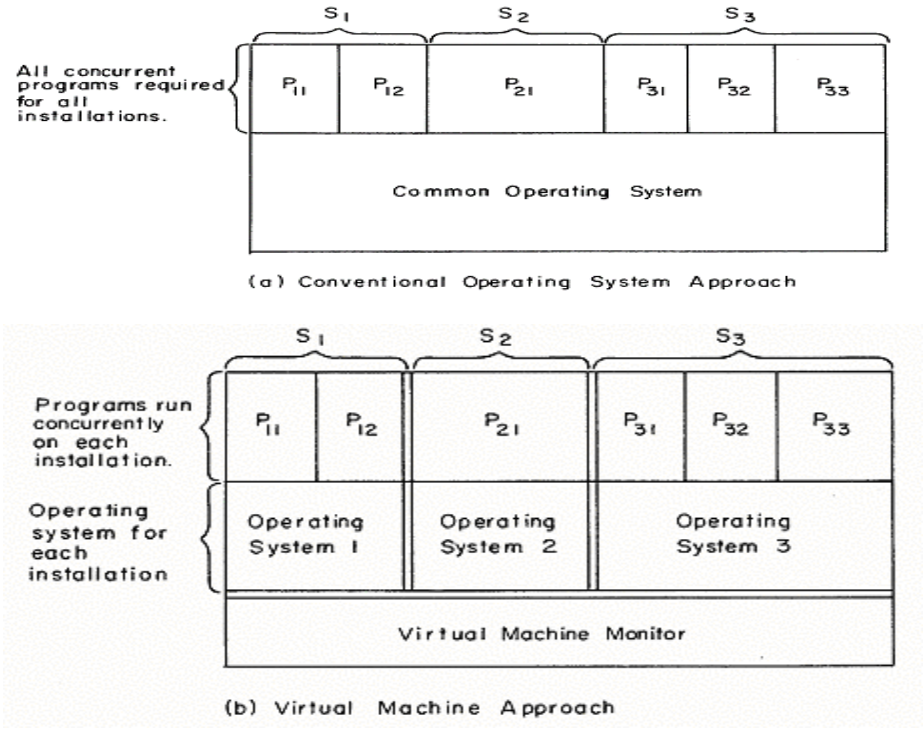

Figure 3.3 Comparison of Conventional OS and VMM / OS Approach

3.3.2 Virtual Machine Environment

Figure 3.3(b) shows the virtual machine approach to a physically shared system. This arrangement has numerous security advantages. If we define Ps(P) to be the probability that a given run of program P will cause a security violation to occur, the following conditions would be expected to hold:

$$Ps(PlOS(n)) < Ps(PlOS(m)) \quad \text{for } n<m$$

OS (I) refers to a conventional operating system multiprogramming at level I the probability of system failure tends to Increase with the load on the operating system. In particular, a mono programming system, 0S (1), tends to be much simpler and reliable than a comprehensive

Multiprogramming system. Furthermore, the m-degree multiprogramming system often requires intricate alterations to support the special needs of the m users, especially if m is large. These problems have been experienced in most large-scale multiprogramming systems. These problems remove in the VM systems each virtual machine may run a separate each operating system may be simpler and less a single comprehensive all-encompassing operating system

$$Ps(OSIVMM(k)) < Ps(PlOS(m)) \quad \text{for } k<m$$

VMM (1) means a virtual machine monitor, virtual machines. The operating system, OS, on a particular virtual machine has the same relationship to the VMM (k) as a user's

Program, P, has to a conventional multiprogramming operating system, OS(m). Using the same rationale as In A above, the smaller the degree of multiprogramming (I.e., k<m), the smaller the probability of a security violation. Furthermore, since virtual machine monitors tend to be shorter, simpler, and easier to debug than conventional multiprogramming operating systems,

Even when k=m, the VMM Is less error-prone. For example, since the VMM Is defined by the hardware specifications of the real machine, the field engineer's hardware diagnostic software can be used to checkout the correctness of the VMM .

IV. Redundant Security Mechanisms

If the Individual operating systems, OS, and the virtual machine monitor, used identical security mechanisms and algorithms, then any user action that resulted In penetration of one could also penetrate the other. That Is, first take control of the OS and then, using the same technique, take control of the VMM. This Is 1oglcally analogous to placing one safe Inside another safe - but having the same combination on both safes. To remove this danger, the OS and VMM must have redundant security based upon Independent mechanisms.

In a VMM/OS environment using VM/370 and 0S/360 as example systems. Let us consider main memory security first. 0S/360 uses the System/360-370 lock and key hardware to isolate one user's memory area from invalid access by another user ' s program.VM/370, on the other hand, the Dynamic Address Translation (DAT) hardware to provide a separate virtual memory for each virtual machine - Independent of the locks and keys. Thus, a malicious user would have to overwhelm both the lock and key and the DAT mechanisms to violate the Isolation security of another coexisting program on another virtual machine.

Abbreviations Used:

VMM- Virtual Machine Monitor
VM- Virtual Machine
T (L) – Translator corresponding to language
VMware-Virtual Machine Software
I-Interpreter, O.S. - Operating system, H.W-Hardware
Ps(PIOS(n)) – Probability of security failure When n program’s are executing in the OS
Ps(OSIVMM(k)) – Probability of security failure When VMM runs k instance of the OS

References

[1] G. Sun, Z. Chen, H. Yu, X. Du, and H. Yu “Online parallelized service function chain orchestration in data center networks,” IEEE Access, vol. 7, pp. 100147–100161, 2019
[2] L. Cui, F. P. Tso, S. Guo, W. Jia, K. Wei, and W. Zhao, “Enabling heterogeneous network function chaining,” IEEE Trans. Parallel Distrib. Syst., vol. 30, no. 4, pp. 842–854, Apr. 2019
[3] S. Bera, S. Misra, and A. V. Vasilakos, “Software-defined networking for Internet of Things: A survey,” IEEE Internet Things J., vol. 4, no. 6, pp. 1994–2008, Dec. 2017
[4] V. Network and I. Planning, “SDN-NFV reference architecture,” no. February, pp. 1–220, 2016
[5] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, “Network function virtualization: state-of-the-art and research challenges,” IEEE Commun. Surv. Tutorials, vol. 18, no. 1, pp. 236–262, 2016

[6] F. Reynaud, F. X. Aguessy, O. Bettan, M. Bouet, and V. Conan, "Attacks against network functions virtualization and software-defined networking: state-of-the-art," IEEE NETSOFT 2016 - 2016 IEEE NetSoft Conf. Work. Software-Defined Infrastruct. Networks, Clouds, IoT Serv., pp. 471–476, 2016
[7] Y. Li, M. I. N. Chen, and S. Member, "Software-defined network function virtualization : a survey," vol. 3, 2015.
[8] Ghansah, B. and S. Wu. Distributed Information Retrieval: Developments and Strategies. in International Journal of Engineering Research in Africa. 2015. Trans Tech Publ.
[9] F. Wang, et al. The implementation of virtualization technology in EAST data system, Fusion Engineering and Design, vol. 84, p.766–769, (2014).
[10] Nie, A study on the application cost of Server Virtualization, in 2013 Ninth International Conference on Computational Intelligence and Security, (2013).
[11] S. Perez, ―Mobile cloud computing: $9.5 billion by 2014‖, http://exoplanet.eu/catalog.php, 2010.
[12] White Paper, ―Mobile Cloud Computing Solution Brief,‖ AEPONA, November 2010