Trusted Infrastructure Design for Secure Virtualization in Cloud Computing: A Review

Shiv Kumar Tiwari[1], Subhrendu G. Neogi[1], Ashish Mishra[2], Saurabh Singh[3], Hameed Khan[3*], Sunil Kispotta[3], Chhayansh Purohit[3]

[1]*Amity University, Gwalior, M.P. India*

[2]*Gyan GangaInstitute of Science & Technology Jabalpur M.P. India*

[3]*Jabalpur Engineering College, Jabalpur, M.P. India*

Abstract: *Virtualization is a fast-growing technology that benefits computer systems, including resource efficiency, mobile software, and reliability. Virtualization may also increase security by allowing distinct operating environments for different applications with varying security requirements. A small trusted computing base (TCB) is especially desirable for safety-critical applications since it reduces the attack surface that might jeopardize the entire system's security. The TCB program combines hardware and virtual machine monitor (VMM) from the design of shared vision and the whole operating system (OS), which provides device drivers and VM machine control capabilities. Due to its high code base and high risk, it is unacceptable to trust this management system in many systems. Consider the "Operating Computer as a Service" scenario, in which remote customers use a virtual machine (VM) on a remote computer platform to launch a guest operating system and applications. A computer service without OS management on a distant platform would be preferable for many customers. In this article, we are providing a secure running environment with an untrusted OS on a virtual computer platform is addressed. It offers safe virtualization architecture with a certain runtime, network interface, and secondary storage for guest virtual machines. In an untrusted management context, the suggested architecture considerably decreases the TCB of safety-critical guest VMs, resulting in better safety. To show how secure remote computing services might be utilized, we constructed a prototype of the solution using the Xen virtualization platform. We assess the suggested architecture's performance penalties and find a minimal sentence.*

Keywords: *Cloud Computing, Virtualization, Virtual Machine, Secure environment, Quality of Service, and Reliability.*

1. Introduction

A virtualization is a new approach that combines virtual computer platform applications into many different logical resources. The virtual machine is the name given to each virtual machine (VM). The virtualization environment allows VMs to build, integrate, store, read, modify, share, migrate, and retrieve [1], reducing administrative costs and simplifying system management. Easier control, in contrast, might lead to security issues. All VMs can be quickly copied and changed if the management environment is hacked. Various possible attacks with hardware access to standard computer programs, such as a cold-boot attack [2], can then be performed on a virtual computer software platform. Due to the higher degree of privileges of the management system, management assaults (for example, by exploiting their flaws) might easily defeat the security protection in guest VMs.

There are two main approaches to virtualization, as shown in Fig. 1. In I-type virtualization projects, the virtual machine monitor (VMM) sits directly above the hardware and captures all communications between VM and hardware. Manage VM stays on top of VMM and is responsible for managing another VM host and hardware connection. The Xen System [3] is a well-known example of this type of visual structure. VMM is a virtualization hosting system that includes VMware Player [4] (OS). The host OS manages guest computers and provides I / O drivers.

VMM's operating environment consists of VMM and Type I VMs, Type II VMMs, and the host OS. Both approaches, however, create security concerns, such as "How can VM trust its malevolent or susceptible environment?" We address this question by presenting two particular application scenarios.

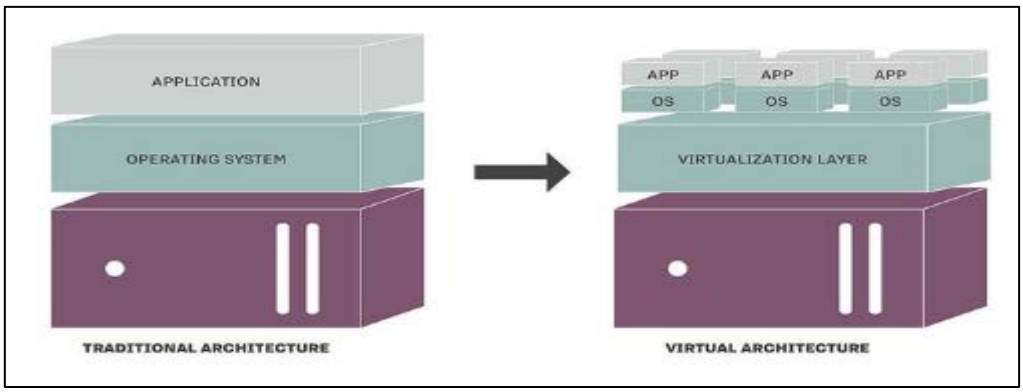

Figure 1 Types of virtualization structures.

In recent years, computers and cloud computing have grown in popularity. Virtualization technology is used by companies such as Amazon.com's Elastic Computing Cloud [5] to provide customers with low-cost computer capabilities. A VM image that includes programs,

libraries, data, and related configuration parameters is generated and used in the service provider's data center. Customers who do not wish to pay for their own computer systems overheads but simply for their existing computer resources will find, this an appealing option. The challenge is to determine how much you can trust the VM's operating environment and ensure that your data is securely protected. The ubiquitous computer team has promoted the idea of keeping the user's work environment on a portable computer so that any computer available [6]. Term virtualization, concept, and programs and data may be stored on a portable OS image storage device. Instead of bringing a computer with him, the user can load his virtual machine into the virtualization environment of his colleagues or a third-party computer firm. In this circumstance, how can the user protect the privacy of the VM data if he wishes to execute it on an untrusted PC?

In general, we initially define the system's trustworthy computing base (TCB) to secure the software system's trustworthiness. Then we look at the TCB's integrity to see if we can trust it. Type I virtualization designs include virtual machine hardware, virtual machine management, and management. The hardware, VMM, and host OS make up the VM TCB in the Type II virtualization architecture. With virtualization, while the hardware is always in TCB and VMM has minimal codebases, it is easy to ensure a complete operating system. VM OS or application administrators cannot be trusted because VMM and operating systems are usually managed by the same organization, such as cloud computing service in our threat model. The reasons we trust VMM but not the OS are:

1. VMM and OS source code sizes are very different.
2. The OS is less reliable due to known and unknown errors and many potentially dangerous programs.
3. The OS management interface is not very well maintained.

In this work, we focus on creating Type I virtualization and use Xen as a demonstration platform [5]. We demonstrate how to remove the management system from VM TCB to ensure that the VM environment is confidential and complete even if the management system is not sustainable [7].

2. Review Literatures

The paradox of the relationship between virtualization and security [8] naturally divides investigations into two groups: security enhancement and protection itself.On the other hand, virtualization can be used to increase security. In many research studies, virtualization has been used to conduct experiments from a secure domain to a target domain [9 - 12]. Lares [10] is a system that actively monitors the visual, safe, and visually impaired system. The

VMM Reconstruction Strategy aims to close the semantic barrier created by VM introspection [9]. Methods based on virtualization in kernel integrity have been suggested by Petron and Hicks [11] and Riley et al. [12], while VMWall [13] is a firewall-based firewall. Terra [14] is a new design that allows multiple VMs with different security levels to operate independently, which reduces the chance of interference. Overshadow [15] and SP3 [16] rely on other processes for unstable guest OS in VMM (also known as a hypervisor). Similar mechanisms are used by both the approach and Overshadow to adjust the visibility of memory pages for different parties. Here's how to tell the difference:

1) Overshadow protects application data from unreliable operating systems while protecting visitor domains from administrative domain corruption.

2) During operation, Overshadow focuses on memory page protection while focusing on environmental protection. The proposed strategy protects domain creation, domain retention, domain recovery, domain closure, live migration, memory pages, and virtual CPU (vCPU) status.

3) The suggested vital management technique tackles a more complex three-case scenario (VMM, DomU, or remote user) than Overshadow, which employs a single key for all applications and has a separate untrustworthy but powerful entity (management OS).

The security of virtualization is a major issue. Security problems such as scalability, transience, software life cycles, diversity, mobility, identity, and data life are underlined in virtual environments. Visible rootkits (VMBRs) [17] detect security issues through the virtualization layer. SHype [18] is a security assurance project that requires operators of a virtual resource area to follow a comprehensive system access policy.

As mentioned in [19], an in-depth study of virtualization is based on a single principle: fragmentation between visitor systems is much better than differentiation between processes provided by conventional operating systems. In theory, because the hypervisor layer is thinner than the standard operating system, it can be easily tested and has a high level of risk. However, in VM management, the complete OS is often part of the TCB virtualization system, which has a significant impact based on Small VMM and substantial partitions. Our approach seeks to address the existing TCB VM visitor security challenges in practical terms. We propose a secure virtualization structure that separates the operating system from a virtual machine (VM) TCB and shows how the OS can be used in Xen's virtualization system.

The root protection of the Terra [14] project, "Even the stadium administrator does not jeopardize important privacy," may lead you to believe that the goal of releasing the TCB Management OS has already been achieved. Management OS is a simple command interface

to build. All administrative functions, including creating, saving, restoring, and closing the VM, are applied to VMM, a reliable Terra threat model. Given the advanced capabilities that require access to the memory of VM guests, we believe they should be left in the OS to keep VMM as small as possible, in the same way, that Xen architecture does. Our design is different from Terra's because TCB does not contain a visual control interface (as does Terra) and the most challenging management functions [20]. It proposes a strategy with the same goal as ours; it includes the entire OS kernel and the new DomB within TCB.

Potential threats to VM in an uncontrolled administration, followed by a real-life example of effective attacks using a management system following the rapid launch of Xen properties. These attacks are easy to carry out and difficult to protect against current virtualization programs. This attack encourages our research on the safe operation of VM under unreliable OS management.

3. Xen Architecture

Xen shows the Architecture type for virtualization. The Xen Hypervisor stays between the operating system and the hardware. The hypervisor, OS kernel, and user applications are three levels of software in the Xen virtualization system. Although security considerations require VM separation, there are times when an inter-VM connection is needed. The x86 building has four fortunes, ranging from ring 0 to 3 (Shallow right level). Ring 0 is used by the OS kernel, while traditional user programs use the third ring. The hypervisor uses a single VM cable, and the Xen architecture ring 0. With standard configurations, Xen uses hypercalls to communicate from the VM kernel to the hypervisor layer, in much the same way as system calls go from 3 to 0. The diversity of policies and approaches is an essential part of the Xen system design. Although the Xen hypervisor uses methods, the process is recorded on a virtual machine. Inter-VM is a shared memory system that can be built using a grant table or external map.

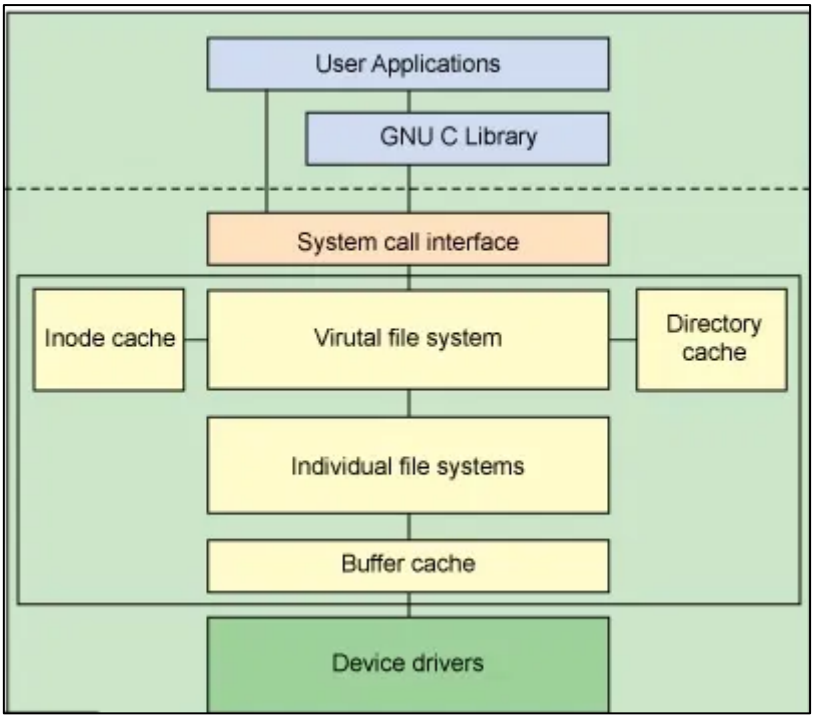

Figure 2 Split device driver model.

The grant table item is generated on each memory page the VM wants to share with another VM. The object specifies which domain is used and which memory rights are granted. If a user from another section requests access to this memory page, the hyperviewer checks the grant table to ensure that sharing is enabled.

The external map is a feature of the OS management system (Dome0 in Xen Architecture) that allows you to directly map the memory pages from other domains to their address area. Dom0 is the only one that can complete this map. This mapping method is used for various administrative tasks, including domain creation, storage, and retrieval. In addition to the Xen layer architecture hypervisor, Dom0 has excellent access to all VMs. For example, creating an external map, the hypercall Dom0 can only do. In a proposed way, Dom0 is considered unreliable, and Dom0's role is defined and damage assessed if Dom0 is malicious.

Hardware management is an essential function of Dom0. Typically, device drivers are available on Dom0. Figure 2 shows the model of the network card driver of the network. In the empty DomU domain, the package is extended from the app to the front-end driver. The package is assigned to Dom0, containing the background driver and the actual device driver using pre-shared memory methods. Dom0 finally sends the data packet to the hardware layer of the actual device. Dom0 also does VM management functions in Xen architecture. Virtual machines (VMs) can be built, copied, stored, read, modified, shared, transported, and restored

to Dom0. However, on a trusted Dom0, these services should be closely monitored to maintain the integrity and confidentiality of the VM of the visitor, which is our primary goal.

3.1 Security threats to untrusted DomU

In the proposed system with Dom0, the hypervisor is not in TCB, which means that the vulnerable Dom0 cannot interrupt the hypervisor. Memory protection requirements: Dom0 will not have access to hypervisor working time memory. Any device that can access direct memory (DMA) can access any part of the virtual memory without the added protection of computer hardware. Dom0 has access to the Xen memory address space on all device drivers and may set DMA. To protect against DMA attacks, AMD hardware should be protected from IOMMU [21] or Intel VT-d [22] input/output memory controls, which must contain a code that controls hypervisors. Researchers have shown using a hypervisor to introduce new memory attacks, even when using hardware protection [23]. On the other hand, these attacks are aided by hardware errors that need to be addressed but do not affect our approach.

1. Dom0 does not compromise the image of the hypervisor kernel due to image protection limitations.
2. Dom0 has hard disk storage for Xen-hypervisor kernel image.
3. Dom0 can change and restart this image to load the hypervisor in question. The strategy is to start the hypervisor by measuring and testing it using reliable computer technology [24]. Instead of Dom0, the hypervisor should use the Trusted Platform Module (TPM).

3.2 Example of concrete offensive

Encryption is an excellent tool for keeping information confidential. Data confidentiality can be guaranteed if we have enough encryption keys and keep them secure. We encrypted the plain text to a virtual machine and stored the cipher text on the hard drive in the most specific case. Therefore, we do not care if the hard disk is lost or stolen. The keys are stored in the system memory at all times. We ignore memory protection when the system is operating normally. Cold-boot attacks allow attackers to retrieve their data from their machine, which can extract memory chip [2]. However, like a virtual computer program, the memory device is stored in an image file with a simple command "save domain" in Dom0. Apart from access to the portable machine, this file can be used to retrieve data encryption keys.

DomU customers use our Linux dm-crypt solution tools to create the encrypted disk. When DomU is running, the Dom0 controller keeps the VM in a memory image file. DomU encryption keys must be stored in this file. Although the exact location of the key is unknown, known methods can be used to reduce the chances. We analyzed 128MB image files using the process described in [2] and identified all encryption keys on the main desktop

in less than 1 second. Once the cryptographic keys have expired, DomU may decrypt the entire cipher texttext to the encrypted disk. Cold boot attacks include physical access to the memory chip before chip content is compromised. This attack is effective but challenging to perform due to physical access requirements and time limits. However, any Dom0 opponent can successfully launch an attack in visual mode. As long as the Dom0 administrator is not malicious, malicious software with root access, malware hackers, or someone with a DomU image file can easily break into DomU and extract all private data from the encrypted disk. After all, although there have not been many successful attacks on the hypervisor layer yet, there have been many attacks on operating systems.

He can use remote users to verify VMM using reliable computer techniques [24], and he does not need to rely on an administrator or operating system for this task. The proposed structure provides a secure network interface, a particularly busy time, and secure storage, including remote users' safe VM operating environment. We propose a threat model, assess security requirements, discuss the proposed virtualization design of architecture, and finally provide relevant information.

4. Threat to the model

The proposed approach will look at a client that creates a secure VM in a remote computing cloud computing platform. We see that reliable computer systems monitor and ensure the integrity of the young hypervisor [24]. VM Dom0 management is a complete operating system managed by the administrator.

Security threats can come from many companies an attacker may use an OS error to break Dom0 and gain root control and other special access rights. Cloud computing customers are criminals. A few problems have been detected that allow a domain that does not have a part (DomU) to control Dom0 [23]. As a result, a customer using DomU can manage Dom0 and separate from a separate VM client in a cloud computing environment. In the cloud computing system, there are invaders. While hardware can verify the hypervisor layer, Dom0 is managed by a system administrator. A negligent or malicious administrator may disclose or alter important client information.Hackled Dom0 can handle network I / O and secondary storage without access to the hypervisor memory address area. Attacks on computer systems and side channels go unnoticed. Attacks on computer systems and a separate channel require physical access to computers, challenging to accomplish in a cloud computing environment.

Security Requirements Analysis

We can summarize why a client does not trust Dom0 using the threat model described above: The vulnerability operating system window (the time between threat detection and release of updates by security providers).Do0 Driver driver security problems often arise due to a sloppy or malicious system administrator's lack of security. Our job is to maintain the privacy and integrity of the virtual machine (VM), which is essential for maintaining security when dealing with uncontrolled VM management. DomU should contain the following features: Achieving secure VM remote operation in an unreliable environment, Dom0: Secure network connection between client and server.

Secure interaction between client and server is required for access control, installation instructions, and returned results. Unlike the cloud computing environment, TCB needs a keyboard and virtual connectors to convert its VM into a virtual mobile computer user interface. Any explicit text transmitted from DomU to the outside world is displayed under the Xen design in Dom0, which may be harmful. TLS (Security Layer Security) [25] is a method that ensures the confidentiality and integrity of communications. Even if TLS is deactivated in this document's safe operating time, Dom0 can still delete cryptographic TLS keys from the RAM or CPU register. A safe working environment is essential. This ensures confidentiality and integrity by providing a vcpu-protected climate and secure storage. Dom0 should not access sensitive information or VM RAM, which is necessary for security in vCPU registration. Dom0, on the other hand, is important in the management of these resources. This emphatic research and key contribution keep domains without understanding their content is Dom0's way of keeping domains. Sensitive data should be stored in a second storage location, such as a hard drive provided using a remote computer. Another option is maintaining a network file system (NFS) [26], allowing VM to access network data. In Dom0, however, all hard disk drivers and network devices are not trusted. The security and integrity of sensitive data stored on the VM of the visitor are essential.

The secure network interface is sufficient for secure storage if the client uses NFS for secondary storage. Data confidentiality may be compromised if the hard disk is used for secondary storage as Dom0 has access to it. If information is not encrypted, Dom0 may undermine data integrity. As a result, a disk encryption system such as dm-crypt on Linux or BitLocker on Windows is required. The problem is that the OS must be encrypted before it can be started for the key to be accessed before the user interface can request a password. In the context of virtualization, the hypervisor may be authorized to authenticate pre-launch authentication.

The most important of the three factors described above is a safe working environment. As mentioned earlier, a secure network interface and backup copy are now available. However, solutions to protect the VCPU guest VM VM and RAM for VM administrators have not been thoroughly evaluated. However, all methods depend on a secure operating environment for network security and maintenance: all cryptographic algorithms and security agreements are executed in real-time. The keys used and the issued code cannot be protected without a certain operating period. While AES encryption is used for secure storage, the attack indicated that the keys might be retrieved in a secure operating time zone. Therefore, the main focus is only on constructing and operating a safe workplace.

5. Conclusion

This study suggested building a secure VM operating system on a reliable operating system. Secure network connectivity, secure second storage, and specific operating conditions are all part of the system. In the Xen virtualization program, we have created a secure working environment. The proposed method protects DomU from an unmanaged Dom0 administrative domain. However, Dom0 can still perform common domain management tasks such as domain building, maintenance, and restoration. According to performance tests, overhead is closely related to domain construction, savings, and renewable energy activities performed once or for a relatively short time during the DomU life cycle. At a slower rate of 1.06 percent, DomU performance is almost equal.

Using the proposed secure virtualization framework, we have analyzed that a reliable VM computer environment can be built with strict security and minimal operational penalties, even under an unreliable administrative system.

REFERENCES

[1] T. Garfinkel and M. Rosenblum, “When Virtual Is Harder Than Real: Security Challenges in Virtual Machine Based Computing Environments,” Proc. Conf. Hot Topics in Operating Systems, pp. 20-25, June 2005.

[2] J. Halderman, S. Schoen, N. Heninger, W. Clarkson, W. Paul, J. Calandrino, A. Feldman, J. Appelbaum, E. Felten, and E. Foundation, “Lest We Remember: Cold Boot Attacks on Encryption Keys,” Proc. Usenix Security Symp., pp. 45-60, July 2008.

[3] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, “Xen and the Art of Virtualization,” Proc. ACM Symp. Operating Systems Principles, no. 5, pp. 164-177, Oct. 2003.

[4] VMware Player, http://www.vmware.com/products/player,2012.

[5] EC2, http://www.redhat.com/f/pdf/rhel/EC2_Ref_Arch_ V1.pdf, 2012.

[6] R. Caceres, C. Carter, C. Narayanaswami, and M.T. Raghunath, “Reincarnating PCs with Portable SoulPads,” Proc. ACM MobiSys, pp. 65-78, 2005.

[7] C. Li, A. Raghunathan, and N.K. Jha, “Secure Virtual Machine Execution under an Untrusted Management OS,” Proc. Int’l Conf. Cloud Computing, pp. 172-180, July 2010.

[8] M. Price and A. Partners, “The Paradox of Security in Virtual Environments,” Computer, vol. 41, no. 11, pp. 22-28, Nov. 2008.

[9] X. Jiang, X. Wang, and D. Xu, “Stealthy Malware Detection through VMM-Based ‘Out-of-the-Box’ Semantic View Reconstruc- tion,” Proc. ACM Conf. Computer and Comm. Security, pp. 128-138, Oct. 2007.

[10] B. Payne, M. Carbone, M. Sharif, and W. Lee, “Lares: An Architecture for Secure Active Monitoring Using Virtualization,” Proc. IEEE Symp. Security and Privacy, pp. 233-247, May 2008.

[11] N.L. Petroni Jr. and M. Hicks, “Automated Detection of Persistent Kernel Control-Flow attacks,” Proc. ACM Conf. Computer and Comm. Security, pp. 109-115, Oct. 2007.

[12] R. Riley, X. Jiang, and D. Xu, “Guest-Transparent Prevention of Kernel Rootkits with VMM-Based Memory Shadowing,” Proc. Int’l Symp. Recent Advances in Intrusion Detection, pp. 1-20, Sept. 2008.

[13] A. Srivastava and J. Giffin, “Tamper-Resistant, Application-Aware Blocking of Malicious Network Connections,” Proc. Int’l Symp. Recent Advances in Intrusion Detection, pp. 39-58, Sept. 2008.

[14] T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh, “Terra: A Virtual Machine-Based Platform for Trusted Comput- ing,” Proc. ACM Symp. Operating Systems Principles, pp. 193-206, Oct. 2003.

[15] X. Chen, T. Garfinkel, E.C. Lewis, P. Subrahmanyam, C.A. Waldspurger, D. Boneh, J. Dwoskin, and D.R. Ports, "Over- Shadow: A Virtualization-Based Approach to Retrofitting Protec- tion in Commodity Operating Systems," Proc. Int'l Conf. Architectural Support for Programming Languages and Operating Systems, pp. 2-13, Mar. 2008.

[16] J. Yang and K.G. Shin, "Using Hypervisor to Provide Data Secrecy for User Applications on a Per-Page Basis," Proc. ACM Int'l Conf. Virtual Execution Environments, pp. 71-80, Mar. 2008.

[17] S. King and P. Chen, "SubvVirt: Implementing Malware with Virtual Machines," Proc. IEEE Symp. Security and Privacy, pp. 314-327, May 2006.

[18] R. Sailer, E. Valdez, T. Jaeger, R. Perez, L. van Doorn, J. Griffin, and S. Berger, "sHype: Secure Hypervisor Approach to Trusted

Virtualized Systems," IBM Research Report RC23511, 2005.

[19] P. Karger and D. Safford, "I/O for Virtual Machine Monitors: Security and Performance Issues," IEEE Security and Privacy, vol. 6, no. 5, pp. 16-23, Sept./Oct. 2008.

[20] D. Murray, G. Milos, and S. Hand, "Improving Xen Security through Disaggregation," Proc. ACM Int'l Conf. Virtual Execution Environments, pp. 151-160, Mar. 2008.

[21] M. Ben-Yehuda, J. Mason, O. Krieger, J. Xenidis, L. van Doorn, A. Mallick, J. Nakajima, and E. Wahlig, "Utilizing IOMMUs for Virtualization in Linux and Xen," Proc. Ottawa Linux Symp., 2006.

[22] Intel VT-D, http://www.intel.com/technology/virtualization/technology.htm, 2012.

[23] Xen Owning Trilogy, http://www.invisiblethingslab.com/itl/ Resources.html, 2012.

[24] Trusted Platform Module (TPM) Specifications, https:// www.trustedcomputinggroup.org/specs/TPM/, 2012.

[25] The Transport Layer Security (TLS) Protocol Version 1.2, http:// tools.ietf.org/html/rfc5246, 2012.

[26] NFS, http://tools.ietf.org/html/rfc3530, 2012.

[27] D. Chisnall, The Definitive Guide to the Xen Hypervisor. Prentice Hall, 2008.

[28] C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, “Live Migration of Virtual Machines,” Proc. Symp. Networked Systems Design and Implementation, pp. 273- 286, May 2005.

[29] SK Tiwari, DS Rajput, S Sharma, SG Neogi, A Mishra,”Cloud Virtual Image Security forMedical Data Processing”,Mathematical Modeling and Soft Computing in Epidemiology, 317-345,2020.

[30] Sk Tiwari, SG. Neogi, A Mishra “ Design and Implementation of Secure System for Virtual Machine Image in Cloud Computing”. Design Engineering, 15044-15054,2021 http://www.thedesignengineering.com/index.php/DE/article/view/4787