International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161

Available at http://inpressco.com/category/ijcet

Research Article

# Exposition of Solutions to Hypervisor Vulnerabilities

Bhavini Kanoongo[Á*], Puja Jagani[Á] , Parth Mehta[Á] and Lakshmi Kurup[Á]

[Á]Computer Dept., Mumbai University, D.J.Sanghvi COE, Vile Parle (W), Mumbai, India

Accepted 10 Sept 2014, Available online 01 Oct 2014, **Vol.4, No.5 (Oct 2014)**

***Abstract***

*With the breakthroughs in VLSI technology and accumulative throughput of every device of the servers, there are huge amounts of computing power that could not be exploited completely or in an optimized way. By employing adequate utilization of hardware resources, virtualization technique has solved this problem. Virtualization is nothing but to abstract computer resources. It generates a virtual environment by separating user and applications from the distinct hardware features used by them to execute their task. The goal of developing a virtual environment is to enhance resource usage by aggregating miscellaneous and autonomous resources. Though the popularity of virtual machines is rising and they are being extensively adopted, the concept is however not new. A layer bounded by OS and the concealed hardware, is the omnipotent hypervisor which is the fundamental element of virtualization. It supervises the physical platform and accesses its entire resource set which consists of the memory allocated to the guest virtual machines (VMs). Not only as a medium of isolation and inspection, but also from the security point of view hypervisors, are extremely prominent. Using various architectures such as the HyperWall, NoHype and Coordinated IDS/IPS approach, we review the various solutions to tackle the threats associated with Hypervisor-secure virtualization.*

***Keywords:*** *Cloud Computing, Hardware Security, Hypervisor Security, Integrity, Virtualization.*

## 1. Introduction

In today's IT space a technology that has a colossal impact is Virtualization. It's a procedure wherein a physical computer is divided into many partly or fully isolated machines, often called as virtual machines (VM) or guests. Several virtual machines can run on a host machine with each having its own OS and applications. In reality the VMs share the existing hardware of the host computer; however for each process such a technique creates an illusion of running on a separate independent physical computer. The key here is to have an isolation that is robust enough so that vulnerabilities in one virtual computer do not alter the VM or the underlying host machine.

Further, a hypervisor is software which allows different operating systems to utilize the hardware of the physical machine. They lie between the OS of the host computer and the virtual environment. Red Hat defines hypervisor as a software layer which isolates the operating system from the hardware thereby allowing numerous operating systems to use the same physical hardware. It runs on the host computer permitting virtual machines to run on the same as well. Xen.org calls hypervisor as a substitute term for virtual machine manager, since it stands for 'beyond supervisor', that is in-charge of managing multiple 'supervisor' kernels.

Hypervisors are also an indispensable risk factor unique to virtual environments. If they are compromised or improperly configured, each VM present on that hypervisor is at a potential risk. The hypervisor lends a singular point of access into the virtual environment and is therefore possibly also a singular point of failure. Improperly configured hypervisors could lead to a single point of jeopardy for the safety of all hosted components. A compromised hypervisor could supersede controls and acquire direct access to the virtual environment, regardless of how securely the respective virtual machines or components are configured. Along with affording a probable entry point to the VMs running on it, the hypervisor itself generates a new attack surface which does not prevail in the physical world and which may be susceptible to direct attacks.

Loopholes in hypervisor isolation technology, security consolidation and patching as well as access controls could be recognized and exploited, permitting attackers to acquire access to individual VMs. Moreover, a secure hypervisor can also be exploited unless it is accurately configured. This is especially the case due to hypervisor's default out-of-the-box configuration which is often not very secure. It is essential that access to the hypervisor be restrained depending on least privilege and need to know, and that independent audit of all activities be imposed. Hypervisors are not all identical, and it is exceptionally crucial to select a solution that supports the necessary security functions required for each environment. Section 2 discusses about the various architectures for hypervisor secure virtualization.

## 2. Review of literature

*Corresponding author: **Bhavini Kanoongo: Lakshmi Kurup** is working as Assiatant Professor

### *2.1 Hyperwall Architecture*

In this architecture, protection is provided to the guest VMs from invasive attacks by a detrimental hypervisor. We are interested in the increasingly popular IaaS (infrastructure-as-a-service) cloud computing model where the infrastructure provider, such as the Amazon EC2 service (Amazon), maintains physical servers and leases VMs to the customers. (Jakub Szefer *et al*, 2012) While the customers give their individual guest operating system (OS) as well as applications to be run in the leased VMs, the framework provider presents the hardware and software of virtualization.

HyperWall's key aspect is the Confidentiality and Integrity Protection (CIP) tables (Jakub Szefer *et al*, 2012) that are only attainable by hard-ware. Based on the customer's specifications, all or portions of the VM's memory are protected by the CIP tables. The memory ranges that need protection from accessibility of hypervisor or DMA (Direct Memory Access) are mentioned by the customer. A hardware disciplined preservation of memory resources is provided by the table. In the HyperWall, hypervisor is provided full access to control the platform, to begin, pause or halt VMs as well as alter memory assignment. This architecture also supplies signed hash calculations to verify that the customer's original VM image and described protections were actually started at VM launch so as to cross-check the behavior of hypervisor. Additionally, during the VM's lifetime, proof of trust can be created to verify that the described VM protections have not been breached, or to highlight the number of attempts made at violating the protections have taken place. After the termination of a VM, its guarded memory is zeroed out through the hardware so that leakage of data and code can be prevented. The hypervisor along with the DMA attain full access of the VM's memory.

There is few hardware alterations needed to implement HyperWall. This architecture reuses the existing property of micro-processor system.

There are new HyperWall registers to accommodate protection details about the presently executing VM. The registers for every processor core are amended by hardware and they can be saved as well as restored when the hypervisor switches VMs. The instructions are modified as well along with the addition of two new instructions. The additions include a hardware block which generates random numbers, a cryptographic engine, a state machine, etc.

Though there are modifications done only in the micro-processor and the hypervisor is considered untrusted, the hypervisor which is responsible for the platform will have to interact with the architecture. The hypervisor has to perform functions apart from the usual tasks. It not only needs to save as well as restore the registers at the time when the VMs are disrupted and resumed, but it also has to control the vmlaunch, vmterminate and vmresume instructions in order to start, terminate or resume respectively. The attestation calculation from hardware registers need to be read and modifications in the procedure to update memory mapping at VM runtime. The software memory manager found in the guest OSes should never map the sensitive code or data such that it is located in the pages accessible to the hypervisor. (Jakub Szefer *et al*, 2012) If the buffers amid VM and devices that the hypervisor has access are only pages, then the mapping of data or code to the pages will never be employed by the manager as the pages are fixed to be made use of by the device drivers of those devices. If only a tiny subset of pages is protected then to ensure that the sensitive data is never mapped onto unprotected pages, guest OS modifications are needed.

The unprivileged sign bytes along with the tmg instructions are added. The data at the desired address is signed by the sign bytes. Apart from the data, the signature consists of the values of VID and NC register as well as the current privilege level. The SKcpu key is used to sign with, once all the data is accumulated. The privilege level determines who invoked the instruction, whether it was hypervisor, OS, or user software. For a user or OS invocation, the VID is used to specify the VM that called it. For hypervisor invocation, its value is 0. To assure freshness, NC is used. E.g. The VM is requested by the customer to sign data and use it. When the PKvm is created by a VM, the sign byte is used so that the customer can authenticate that the PKvm actually came from that VM. Random number generator is used by the tmg and random bytes are returned to the guest.

Hyper-privileged instructions are used to start, terminate and resume a VM. The HyperWall protections are enabled when the launch of a new VM is signaled by the vmlaunch signal to the hardware. The VM's protected memory is zeroed out and their respective entries are cleared in the CIP tables to reclaim memory, when the vmterminate instruction is used. When interrupted, the HyperWall registers that describe that VM are saved. The vmresume instruction is used to signal that the VM can be resumed and the hardware checks the integrity of the new restored HyperWall registers.

The cryptographic routines which have been included as an additional hardware in the microprocessor are implemented as dedicated circuits, microcode or software routines in on-chip ROM (Jakub Szefer *et al*, 2012).

### *2.2 NoHype Architecture*

In this architecture, the main focus is on protecting the hypervisor from the attacks by the guest. A malignant VM can cause a VM exit can occur either due to injection of a vicious code or create a bug in hypervisor. This malicious VM could result in violation of confidentiality as well as integrity which may cause the VMs to crash or degrade the speed of the hypervisor. This causes the denial-of-service (Jakub Szefer *et al*, 2011) attack breaching availability. This architecture removes the need for communication amid VMs and hypervisor, which results in prevention of such attacks.

We assume that the sources of information such as the cloud infrastructure are safe. To ensure the prevention of hardware attacks, efficient physical security measures are employed by the use of surveillance cameras as well as by limiting the access to the physical facilities. Although the security of guest OSes is not vital, there is a requirement

for the cloud provider to make compatible OS kernels as needed to start a VM. The customer is responsible for the protection of the software that runs in the VMs. However, the security and definiteness of the software which manages cloud is not considered. The interface used by the cloud customers to request, terminate and manage the VMs, are also presented by the cloud management software. It requires dedicated servers for interactions with the NoHype servers. In this architecture, we assume that it is secure.

The key feature of this architecture is that it capitalizes on the exclusive use model of shared and hosted cloud to remove hypervisor attack surface. A new method of eliminating the attack surface is used instead of defending attack vectors that have been employed to attack hypervisors. This helps us preserve the semantics of virtualization machinery. The VM can be initiated with changeable parameters, halt VMs as well as run many VMs on same physical server. This architecture is also realizable on commodity hardware due to its design. The four main ideas of this architecture are as follows:

- Pre-allocating memory and cores (Jakub Szefer *et al*, 2011): The processor memory and cores are pre-allocated so dynamic management of resources need not be done by the hypervisor. The reason this is possible is because specifications of the exact resources required before the creation of the VM are given by the customer.
- Using only virtualized I/O devices (Jakub Szefer *et al*, 2011): To eliminate the need for virtualization software to imitate I/O devices, these devices are dedicated to the guest VM. In this architecture, the devices are virtualized by themselves. The only requirements are storage, graphic cards and network connection (NIC). Hence there are a finite devices which need virtualization support.
- Short-circuiting the system discovery process (Jakub Szefer *et al*, 2011): This architecture lets the guest OS perform its usual initialization procedure to decrease the modifications of guest OS. However, slight modifications are performed on the configuration system of cache system for later use. To overcome the drawbacks of commodity hardware, it is sustained by temporary hypervisor. For instance, to determine the availability of devices, the guest OS carries out many reads to PCI configuration space to find out the devices present.
- Avoiding indirection (Jakub Szefer *et al*, 2011): The hypervisors need to carry out indirections which map virtual view to the real hardware be-cause they show a brief view of device which is not a 1:1 mapping of virtual to the real hardware. As the guest virtual device is brought in direct contact with the hidden hardware, the need for hypervisors to carry them out are removed. The guest VM accesses real processor ID as well as averts the necessity for indirection.

## 3. Server Defense Approach

Since the hypervisor does not have services that stop remote protocols, nearly all hypervisor vulnerabilities cannot be remotely exploited. These vulnerabilities usually get exploited from malware that constitutes a virtual machine (VM). Thus, one of the leading methods for protection against attacks to hypervisor vulnerabilities is by prohibiting malware from getting installed into the virtual machine in the first place. Accompanying the dynamic makeup of virtualized environments are challenges for intrusion detection/prevention systems (IDS/IPS). It is also burdensome to attain and maintain consistent security.

One of the important principles in the approach to virtualization security is ‘defense in depth’ that is a primary security requirement to identify ‘de-perimeterization’ in the IT infrastructure of an organization. Because of the incapacity of appliance-based security to manage attacks between VMs on the same system, especially after virtualization very apparently stressed upon the challenges of de-perimeterization, there is a pressing need for top-end security. There is clearly a need for mechanisms to be set up directly on the physical server i.e. providing protection as close to the asset as possible.

### *3.1 Approaches in virtualization security*

Prior to extensive availability of security solu-tions purpose developed for VMware VMsafe APIs, two basic avenues are usually present with security software to safeguard virtual machines—one is to employ a virtual security device inside the virtualized computing space, to guide the traffic flow between one or more guest VMs and a virtual switch (vSwitch) [see Fig 1]

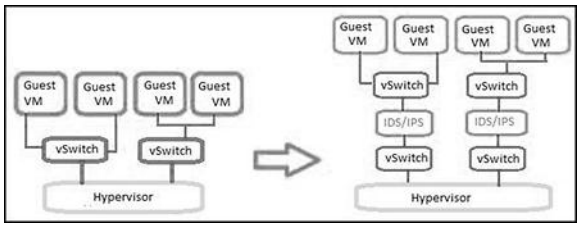


**Fig 1:** Virtual security appliance approach to IDS/IPS approach

The same IDS/IPS service, in the other approach, can be set up on each virtual machine. [See Fig 2]

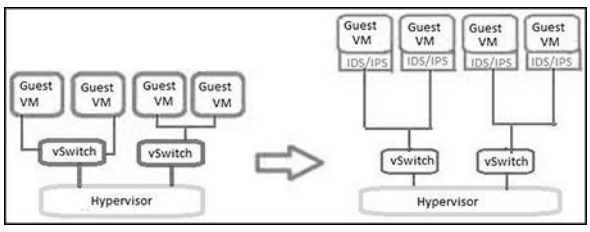


**Fig 2:** VM centric IDS/IPS

The VM-centric method unlike the virtual-security appliance method avoids the shortcomings of inter-VM traffic and mobility as well as deficiency of visibility. There also exists a challenge in front of VM-centric architecture to deploy an IDS/IPS security device on every

virtual machine. This however gets diminished by adopting mechanisms like templates-as stated by VMware. Nevertheless, due to the dynamic nature of virtualized environments, virtual machines can still be exposed to the production environment bereft of a security agent.

*3.2 Security Watch Dogs*

The VMware VMsafe program empowers you to set up committed security VMs that allow privileged access to hypervisor APIs. This makes it viable to develop an exclusive security control called security watchdog VM. It provides an advanced means of carrying out security controls within a virtual environment [see Fig 3].

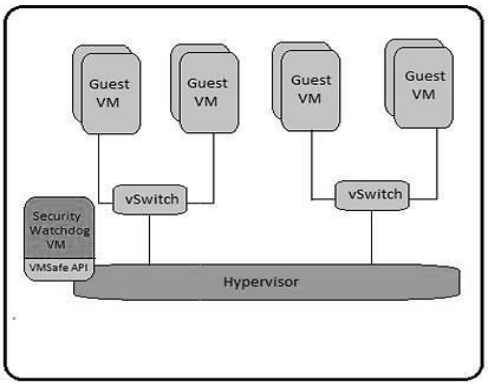


**Fig 3**: Security Watchdog VM

*3.3 Coordinated Security Approach*

The coordinated approach comprises of a VM-centric client that can be arranged as distinctive virtual devices.

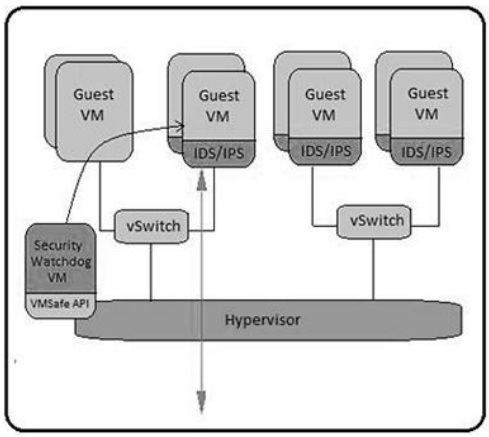


**Fig 4:** Coordinated IDS/IPS Approach

It can also be a security watchdog deployed by VM in order to protect many virtual machines. This approach takes care that the critical IT assets such as the VMs can be preserved by deployment of software on the assets themselves. Simultaneously, the security watchdog VM protects the non-critical assets. [see Fig 4].

There are six main focal points of this coordinated approach. Intrusion detection and prevention coordination, Virtualization management integration, Enterprise management, Comprehensive IDS/IPS functionality, Multiple virtualization architectures and Software licensing models (Trend Micro, 2009).

## Conclusions

With the advent in technology, all the enterprises want top class security which is not only scalable and manageable, but also reliable and capable of staying one step ahead of the new threats. In this paper we have discussed the three different architectures to tackle the threats of hypervisor-secure virtualization in detail. As the speed of threats increases, so do risks and costs.

Though a virtualized IT framework faces many of the same threats faced by the physical server surroundings, investment in architectures providing security is necessary. With the enhancements in security, protection of virtualized IT resources increases. Adopting one of the above approaches with security software will enable optimized protection, spontaneous solution deployment and also ensures baseline for security of all virtual machines. This eliminates the bottlenecks as well as need for redundant controls. We aim to expand the security of virtualization to cover all the mission-critical systems.

## References

Amazon Elastic Compute Cloud, *Amazon EC2*, http://aws.amazon.com/ec2/.

Jakub Szefer, Ruby B.Lee (2012), Architectural Support for Hypervisor-Secure Virtualization, *ASPLOS XVII Proceedings of the Seventeenth International Conference on Architectural Support for Programming Languages and Operating Systems, New York, NY, USA*, pp 437-450.

Jakub Szefer, Eric Keller, Ruby B. Lee, Jennifer Rexford (2011), Eliminating the Hypervisor Attack Surface for a More Secure Cloud, *CCS'11 the ACM Conference on Computer and Communications Security Chicago, IL, USA*, pp 401-412

Trend Micro (2009), Meeting the Challenges of Virtualization Security, *Server Defense for Virtual Machines, A Trend Micro White Paper*.