Design of a Hypervisor-based Rootkit Detection Method for Virtualized Systems in Cloud Computing Environments

Tongwook Hwang
Information Security Group
Korea Internet & Security Agency
Seoul, Korea
E-mail: twhwang@kisa.or.kr

Youngsang Shin
Information Security Group
Korea Internet & Security Agency
Seoul, Korea
E-mail: ysshin@kisa.or.kr

Kyungho Son
Information Security Group
Korea Internet & Security Agency
Seoul, Korea
E-mail: khson@kisa.or.kr

Haeryong Park
Information Security Group
Korea Internet & Security Agency
Seoul, Korea
E-mail: hrpark@kisa.or.kr

***Abstract*—Cloud computing is becoming increasingly popular. Many companies utilize cloud computing services to minimize IT infrastructure costs. The popularity of cloud computing has attracted the interest of cyber criminals. As the result, virtualized environments are a valid and attractive target for APT attacks. Since the key components in APT attacks are rootkit malware that provides stealth, detecting rootkits is an effective measure for protecting against APT attacks. Traditional rootkit detection algorithms are based on non-virtualized environments, where a detection agent tries to identify incoherency in OS system calls to detect rootkits. However, applying these algorithms to cloud computing environments entails installing a copy of the detection agent in every virtual machine, resulting in inefficient storage use and performance degradation. We propose a hypervisor-based, out-of-the-box rootkit detection system that takes cloud computing environments into consideration. The method utilizes vIPS platform to gain many beneficial traits including hypervisor-independency, agentless virtual security appliance structure, and usability. Therefore the method provides effective protection against rootkits in cloud computing environments.**

Keywords - Cloud security; Virtualization security; Rootkit detection; Hypervisor; Rootkit

I. INTRODUCTION

Cloud computing is making a revolutionary change in IT infrastructures [28]. By sharing IT infrastructures and paying only for the amount of resource used, cloud users can greatly reduce costs for IT infrastructure such as building and maintaining server farms. However, the great prevail of cloud computing comes with its own shadows. Now cloud computing environments are rapidly becoming the target of various malicious activities including much feared APT(Advanced Persistent Threat) attacks.

APT attacks deliberately target a single entity for a prolonged time. It entails gaining stealthy foothold inside the target network, carefully gathering intelligence, and widening influence only when it is sure not to be exposed. This kind of stealthy operation is usually performed by rootkits.

Rootkits are a kind of malware that focuses on providing stealthy control over the victim. Rootkits come in various flavors, attacking different portions inside the system, even including a kernel code space and a bootloader. Once successfully installed, rootkits render themselves and any other payload invisible to the OS and applications, and focus on covering up for the payload that usually provides remote control (ex. backdoors) or passive intelligence collection (ex. keyloggers). The intelligence gathered and the control exercised enable rootkits' owner to prolongably fish for critical information and timely take an action based on the information. Therefore, detecting rootkits is critical and effective for preventing any security breaches from resulting in prolonged damage that may lead to APT attacks.

Methods detecting rootkits can be categorized into two types: in-the-box and out-of-the-box approaches. In-the-box approach is the traditional method where detection agents are placed inside the target machine and look for any artifacts or incoherency that a rootkit may generate. Out-of-the-box approach takes the opposite path and attempts to detect the rootkits from outside of the target machine. To this end, it virtualizes the target system in question and utilizes VM introspection techniques to hunt up rootkits.

Each method has advantages and disadvantages. However, in cloud computing where machines in question are already placed in a virtualized environment, out-of-the-box approach has the upper hand, because out-of-the-box rootkit detection method can fully take an advantage of virtualization.

In this paper, we present an out-of-the-box rootkit detection system that is optimized for virtualized systems in cloud computing environments. We first discuss our principles for designing the rootkit detection system and show the design for the virtualized environments. To fulfill the design principles better, we design the rootkit detection system as an extension on top of the vIPS platform [26],

which is a hypervisor-independent virtualized host/network IPS platform designed to operate as a virtual security appliance. The proposed rootkit detection system can check for rootkits inside virtual machines without suffering drawbacks such as manipulation from rootkits or A/V storms.

This paper is organized as follow. First, we provide a background for cloud computing and virtualization in Section 2. In Section 3, we explain the limitation of the traditional security protection tools in virtualized environments. Next, we discuss our principles to design our out-of-the-box based rootkit detection system and present its design in Section 4. After we provide related works in Section 5, we conclude this paper in Section 6.

II. BACKGROUND

A. *Cloud Computing*

Cloud computing is a new methodology that revolves around dynamically lending and borrowing IT resources. The principal idea is to allow users to rent IT resources such as computation power, storage, and application when they need the specific resource. The users need to pay only for the amount used for the time being. The customers spend less money, yet the providers still profit because they can now assign and manage the resource much more efficiently.

The enabling technology for making cloud computing feasible is virtualization. Virtualizing various components of a computer including CPU, RAM, storage and network, allows the providers to merge the total resources as a pool and rent them out as demanded. The dynamic nature of virtualization leads to reduction of total cost of ownership for both customers and providers.

B. *Virtualization*

Virtualization technology allows a single physical IT resource to be dynamically split and used as multiple logical IT resources. One of the good examples is the server virtualization, which allows a single server machine to be split into multiple VM(virtual machines) that can be dynamically adjusted as needed.

The hypervisor is located between VMs and hardware and plays a major coordinating role which is to assign the shared resources including CPU, memory, and network to VMs and to schedule their usage. Thus, the hypervisor enables VMs to execute its own guest operating systems as an independent host. It also supports isolation between the VMs, preventing virtual machines from accessing each other.

There are two types of hypervisors. Type 1 hypervisor is also called as bare-metal hypervisor since it works right on a bare hardware. Type 2 hypervisor is called hosted type hypervisor because it is operated on a host operating system which is an operating system on a bare hardware.

Type 1 hypervisor directly controls the hardware and provides a high performance virtualization. Thus, Type 1 is adopted mainly in enterprise environments. The well-known examples of Type 1 hypervisor are open source Xen and KVM, and proprietary VMware ESXi and MS Hyper-V. For personal use, Type 2 hypervisor are generally used because it may be easily installed and managed. However, it shows a lower performance than Type 1 since it runs on a host operating system and does not directly control the hardware. The well-known examples of Type 2 hypervisor are VMware Workstation, VMware Fusion, Parallels Desktop, Oracle VirtualBox, and QEMU. The most prominent difference of virtualized and non-virtualized IT resources including server, storage, and network is the cost-efficient provision and management of the resource. For example, in the traditional IT environment, it takes at least a few weeks or months to purchase and deploy a physical server. However, in the virtualized environment, it takes just a few minutes to create and set up the virtual machine for use. When the virtual machine is paused or shut-downed, it can be stored as an image file. Thus, it can be restarted as restored from the saved VM (Virtual Machine) image file. The virtual machine can also be migrated to another virtualized system when its host system has technical problems or is overloaded. The live migration is called vMotion or XenMotion by VMware and Citrix respectively.

C. *Rootkit*

Rootkits are a kind of malware that focuses on stealth and control. Rootkits hide inside infected systems, somehow manipulating OS system call results in Figure 1.

Rootkits are a kind of malware that focuses on providing stealthy control over the victim. Rootkits come in various flavors, attacking different positions inside the system, even including the kernel code space and the bootloader. Once installed, rootkits render themselves and any other payload invisible to the OS and applications, and focuses on providing cover for the payload that usually provides remote control (ex. backdoors) or passive intelligence collection (ex. keyloggers).Therefore, rootkits can easily become the foothold for the next attack. These properties make rootkits the tool of choice for APT attacks.

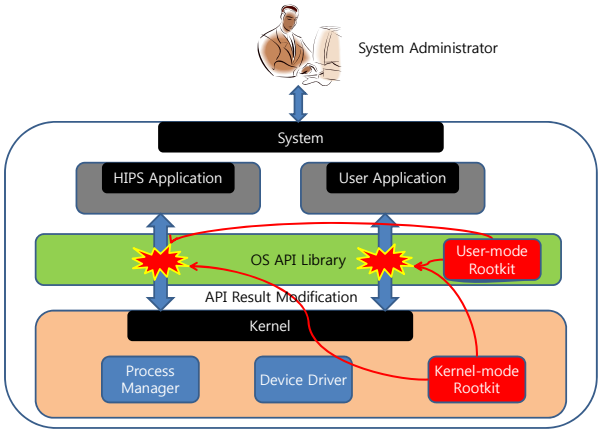

Figure 1. The mechanism of rookits

III. COMPARISON OF ROOTKIT DETECTION METHODS IN VIRTUALIZED SYSTEM ENVIRONMENT

In this section, we discuss two types of rootkit detection methods including in-the-box rootkit detection methods(the

classical antivirus techniques) and out-of-the-box rootkit detection methods. We discuss how the two types compare in cloud computing environments and why.

A. *In-the-box Rootkit Detection : Traditional Method*

In-the-box rootkit detection methods are widely used by conventional anti-virus/-malware programs. These methods require an agent to be installed inside the system that is to be checked for rootkits. The agents scan the system from the inside, identifying known rootkit signatures or incoherencies caused by rootkits.

Even when these methods are applied to cloud computing environments, they do not take the virtualized status of the systems into account. Instead they treat virtual machines as real computers and installs an agent into each virtual machine inside the system. Each copy of the agent independently manages its own version of virus/malware signature.

This form of rootkit detection may have the following problems in cloud computing environments. First, it may cause A/V storm. In enterprise environments, host IDS/IPS software is installed in each host and set up to perform a virus scan at specific time, e.g. 12 a.m. Thus, all of the anti-virus/-malware agents perform the scan at the same time. Usually, this is fine. But in cloud computing environments, they generate an extremely high workload in the virtualized system since all of them reads from storages shared by virtual machines and perform signature matching with those files for virus/malware, causing disk/network bandwidth saturation. This situation is called A/V storm. A/V storm greatly reduces the performance of services provided by virtual machines in the virtualized system, in which the virtual machines under anti-virus scanning are co-located.

Second, the virus/malware signature should be maintained independently for each copy of the agent installed. Each of the signatures in virtual machines needs to be maintained as the same version to keep the same security level of virtual machines. This requirement makes the security management of virtual machines complicated since virtual machines may be dynamically created, paused, restarted, or moved into another virtualized system.

Third, because the agent works inside the target machine, if the machine is really infected with rootkits, the rootkits may detect the agents and attempt to fool it. As rootkits often manipulate the OS kernel in order to provide the stealth power required, it is relatively easy for the rootkit to hide from an agent process once the presence and identity is known. This leads to incorrect status analysis of the system, approving the rootkit-infected system as a clean one.

B. *Out-Of-The-Box Rootkit Detection*

Out-of-the-box rootkit detection methods use a different approach. As part of the installation, the methods migrate the target system to a virtual machine by installing a Type 1 hypervisor. Then they utilize VM introspection technology to gather a clean view of the system from the hypervisor. This is why the methods are called out-of-the-box: they work outside of target system, the “box”.

In a cloud computing environment, out-of-the-box approaches synergizes with the cloud computing environments in many ways. This is because out-of-the-box approaches and cloud computing both utilizes virtualized systems as the basis of operation, providing a common ground.

First, the need to install a hypervisor and migrate physical hosts to virtual machines is alleviated. This is because in cloud computing environments, the systems in question are already virtualized. This reduces the total operation overhead, because virtualization overheads are already accounted for by cloud computing. Also it leads to decrease in setup costs.

Second, it is extremely difficult for a rootkit inside the target virtual machine to detect and fool out-of-the-box rootkit detection methods. The rootkit detection methods do not install anything inside the target virtual machine nor rely on the OS system call of the target machine. Even if the rootkit keeps track of the virtualized state of the target machine in order to detect the installation of out-of-the-box rootkit detection methods, the tracking yields no information in cloud computing environments, because the infected system was virtualized from the beginning. Therefore it is much easier to check for rootkits without worrying about malicious interferences.

Third, as the rootkit detection workload is centralized at one point in each server, it becomes easier to avoid A/V storms. As the tests are all carried out by a single entity per virtualized system, it becomes feasible to micro-schedule checking for each virtualized machine. This level of control is not possible for the in-the-box rootkit detection methods, because in that case there is no option but to give each agent a predetermined fixed schedule, rendering workload-based modifications to the schedules infeasible.

IV. DESIGN OF ROOTKIT DETECTION SYSTEM

A. *Design Requirements of Rootkit Detection System*

In this section, we discuss the key requirements for designing our rootkit detection system. These requirements specify the constraints that any rootkit detection system must hold to provide optimized performance in cloud computing environments.

1) *Agentless Virtual Security Appliances*

The rootkit detection system needs to be implemented as an agentless security virtual appliance. This structure should be able to access the internal states of VMs through a hypervisor API call or similar libraries, while staying out of the virtual machines in question. This observation and analysis of the internal states and events of VMs including the contents of virtual CPU, memory, and disk is called VM introspection. Even though VM introspection is the outside observation, it can build an almost same semantic view of system states and events as a semantic view obtained inside VM. Thus, VM introspection is critical to support tamper-resistant, high-fidelity out-of-the-box VM monitoring, resulting in the basis of intrusion detection/prevention.

Recent malware is getting gradually more stealthy and elusive. They are trying to detect and compromise even anti-

malware software located in the compromised system as well as to hide their own presence from intrusion detection in the system. Many out-of-the-box based approaches have been recently proposed. They place their detection facility in an independent VM without locating an agent inside VM monitored. Thus, they enable the detection facility to be isolated from the monitored VM, making it hard for malware to sense and subvert the detection facility itself.

VM introspection is a crucial technique for agentless virtual security appliance. VMware has introduced VMsafe API [23] and vShield Endpoint Security API [24], which allow third-party security vendors to leverage VM introspection-based approach to better monitor, protect, and control guest VMs for their ESXi hypervisor [25]. For Xen hypervisor and KVM, LibVMI has been presented in [12]. However, it has only limited capabilities as VM introspection tool. To ensure that the result of VM introspection is correct, methods securing a hypervisor need to be supported such as kernel integrity check to prevent from subverting VM introspection such as work in [1].

2) Hypervisor Independece

There are various virtualization platforms available based on different hypervisors, including Xen, VMware ESXi, KVM, and MS Hyper-V. Although VMware ESXi is the current market leader, Citrix Xen, KVM, and Microsoft Hyper-V have increasingly grown their market share [18]. Xen is popular in large cloud service providers such as Amazon. KVM have been increasingly gaining its popularity after RedHat Enterprise Linux began to adopt KVM into its product line. Furthermore, MS Hyper-V is seeing increase in use after the release of MS Windows Server 2012. Therefore, locking into a single hypervisor based platform would significantly hamper the applicability of the rootkit detection system. The ability to run on different hypervisor types becomes crucial in developing a practical rootkit detection system that can be implemented and utilized in any cloud computing environment.

3) Performance

Performance is a key issue in cloud computing environment, compared to traditional non-virtualized environments. In traditional environments, there always exists some slack in resources even in high load scenarios, and therefore avoiding major performance hogging was not difficult. However, most resources are supposed to be fully utilized by being shared among VMs in virtualized computing environments. This means that scheduling a rootkit detection for each of VMs requires taking into consideration of all VMs inside the virtualized system. This may place additional burden on a rootkit detection system.

4) Useablility

Rootkits are a serious and immediate threat to any systems. But the threat is even greater in cloud computing environments, because one compromised VM has a potential to infect all VMs in the same virtual network of the same virtualized system.

To this end, it is imperative to take measures as soon as possible once any rootkit is detected. However, uninstalling rootkits from VM in question is very hard because removing rootkit itself does not ensure the normal operation of VM infected. Naïve deletion of rootkits can even cause the crash of VM due to the corruption of critical system structures.

Therefore, a rootkit detection system should be able to effectively cooperate with SIEM(Security Information and Event Management) system to take immediate emergency measures such as isolating VM in question.

B. Rootkit Detection VSA Architecture

In this section, we present the architecture of our rootkit detection system in virtualized environments for cloud computing. The overview of the architecture is presented in Figure 2. The VSA(Virtual Security Appliance) consists of vIPS platform, Detection framework, Management framework, and miscellaneous modules. vIPS platform is a virtual host/network IPS platform that provides hypervisor independent API for virtual machine introspection [26]. Detection framework is the workhorse of the VSA. It functions as a rootkit detection engine and a signature database . Management framework takes care of communicating with vIPS platform. It mainly takes care of policy management and intrusion notification. Miscellaneous modules provide utility functions required by the other modules.

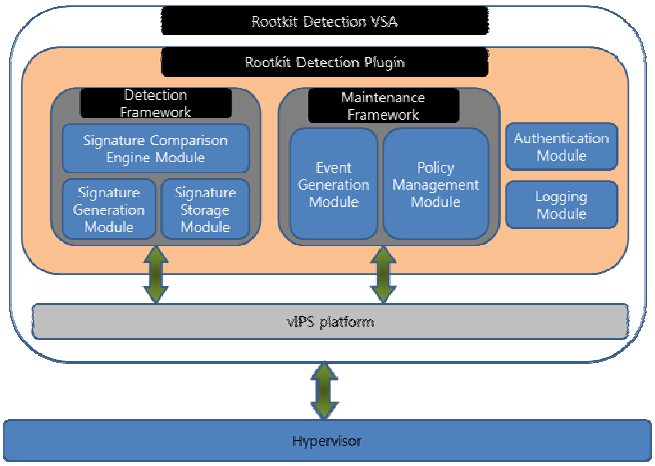

Figure 2. Architecture of Rootkit Detection VSA

1) vIPS platform

We presented vIPS platform in [26], which is a hypervisor-independent virtual host/network IPS platform to aid developing flexible and effective VSA. It provides virtualization platform-neutral API for accessing VM information, including VM introspection. Also, vIPS platform provides easy integration with SIEMs, improving usability in cloud computing environments where there will be many VMs shared across many virtualization systems to take care of. vIPS platform allows additional IDS/IPS functionalities to be added on as a plugin. We designed the rootkit detection system as a plugin service on vIPS platform.

2) Detection framework

Detection framework is the main workhorse of the rootkit detection system. It consists of three modules: signature comparison engine module, signature storage module, and signature generation module.

Signature comparison engine module with rootkit signatures in the signature storage module managed by Management framework.

Signature storage module stores rootkit-related signatures as a database, allowing fast lookups. Signatures can include rootkit signatures that identify the existence of rootkits, and clean signatures that ensure the integrity of a specific structure inside VM, such a files or one of the memory sections.

Signature generation module utilizes the VM introspection module to look for comparison targets for rootkit detection rules, and either pass the data on to signature comparison engine module or register the new signature as cross-compared.

3) Management framework

Management framework is the administrative part of the rootkit detection system. It consists of two modules: policy management module and event generation module.

Policy management module receives and applies the policy settings configured by external management software such as SIEM via vIPS framework

Event generation module notifies the vIPS framework of the detection result, and generates security events or alerts if an intrusion has been detected.

4) Miscellaneous modules

Miscellaneous modules include logging module and authentication module. They act as utility functions for the security VSA.

V. RELATED WORKS

A few works have previously discussed VMI-based out-of-the-box approach for rootkit detection. [5,10,17,27,29]. Garfinkel and Rosenblum present a VMI-based IDS with a signature-based detection engine for VMware Workstation, combining callback functions for predefined malicious events with polling for other malicious changes inside the virtual machine [5]. Payne et al. presents a VM introspection library called XenAccess [17], which is now called LibVMI [12]. Now LibVMI supports Xen and KVM hypervisors, but its accesses are limited to CPU registers and memory of virtual machines.

Most current commercial VMI-based intrusion detection tools focuses on VMWare products. Prime examples are Juniper Networks' vGW [9] and Trend-Micro's Deep Security [21]. None of the aforementioned except LibVMI includes architecture to support any hypervisor other than VMware.

VI. CONCLUSION

In this paper, we propose an effective rootkit detection method for detecting rootkits inside VMs in cloud environments. It builds upon vIPS platform, an effective virtualized system security protection platform, to create a hypervisor independent, agentless virtualized security appliance.

We are currently in the process of implementing the rootkit detection plugin service for vIPS platform. As a future work, we plan to evaluate of our implementation in terms of performance and rootkit detection accuracy.

ACKNOWLEDGEMENT

This work was supported by the IT R&D program of MOTIE/KEIT. [10041872, Development of Virtual Network Intrusion Prevention Techniques: Analysis, Detection, and Prevention of Hacking in Virtualized Environments for Cloud Computing]

REFERENCES

[1] S. Bahram, X. Jiang, Z. Wang, J. L. Mike Grace, D. Srinivasan, J. Rhee, and D. Xu. DKSM: Subverting virtual machine introspection for fun and profit. In IEEE International Symposium on Reliable Distributed Systems (SRDS), 2010.

[2] F. Bellard. QEMU. http://wiki.qemu.org.

[3] Citrix. Xen. http://www.xen.org.

[4] Forrester Research. The evolution of cloud computing markets, 2010.

[5] T. Garfinkel and M. Rosenblum. A virtual machine introspection based architecture for intrusion detection. In Network and Distributed System Security Symposium (NDSS), 2003.

[6] Gartner. Cloud computing: Key initiative overview, 2010.

[7] R. P. Goldberg. Architectural Principles for Virtual Computer Systems, pages 22-26. Harvard University, 1973.

[8] X. Jiang, X. Wang, and D. Xu. Stealthy malware detection through VMM-based "out-of-the-box" semantic view reconstruction. In ACM Conference on Computer and Communications Security (CCS), 2007.

[9] Juniper Networks, Inc. vGW Series Virtual Gateway. http://www.juniper.net/us/en/products-services/security/vgw-series/.

[10] T. Kittel. Design and implementation of a virtual machine introspection based intrusion detection system. Master's thesis, Technische Universitat, Munchen.

[11] KVM. Kernel based virtual machine (KVM). http://linux-kvm.org.

[12] LibVMI project. vmitools. |http://code.google.com/p/vmitools/.

[13] P. Mell and T. Grance. The NIST definition of cloud computing, 2009.

[14] Microsoft Corp. Microsoft server and cloud platform. http://www.microsoft.com/en-us/server-cloud/windows-server/hyper-v.aspx.

[15] Oracle Corp. Virtualbox. http://www.virtualbox.org.

[16] Paralles. Parallels desktop. http://www.parallels.com.

[17] B. D. Payne, M. D. P. de A. Carbone, and W. Lee. Secure and flexible monitoring of virtual machines. In Annual Computer Security Appliances Conference (ACSAC), 2003.

[18] E. Shein. Microsoft, Citrix and KVM continue to erode VMware's virtualization domination. http://www.networkcomputing.com/virtualization/microsoft-citrix-and-kvm-continue-to-ero/232601849.

[19] Sourcefire. Snort. http://www.snort.org.

[20] Symantec. State of cloud survey: Global findings. http://www.symantec.com/about/news/resources/press_kits/detail.jsp?pkid=stateofcloud2011, 2011.

[21] Trend Micro, Inc. Deep Security. http://www.trsendmicro.com/us/enterprise/cloud-solutions/deep-security/.

[22] VMware. vCloud networking and security. http://www.vmware.com/products/datacenter-virtualization/vcloud-network-security/.

[23] VMware. VMsafe. http://www.vmware.com/go/vmsafe.

[24] VMware. vShield endpoint. http://www.vmware.com/products/vshield/overview.html.

[25] VMware, Inc. VMware. http://www.vmware.com.

[26] Y.Shin, M.Yoon, K.Son. Design of a Versatile Hypervisor-based Platform for Virtual Network-Host Intrusion Prevention. In ICIPT, 2013.

[27] X. Jiang, X. Wang, and D. Xu. Stealthy Malware Detection and Monitoring Through VMM-Based Out of the Box Semantic View Reconstruction. ACM Transactions on Information and System Security, Vol. V, No. N, June 2008, Pages 1-27.

[28] Cisco. Cisco Global Cloud Index: Forecast and Methodology, 2012–2017. http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns1175/Cloud_Index_White_Paper.html.

[29] Chris Rogers Virtual Disk Integrity in Real Time, Xen Project Developer Summit