A Centralized Monitoring Mechanism in Virtualized Environments

Yandong Han*, Zhiyu Hao, Lei Cui & Chonghua Wang

Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

ABSTRACT: Virtualization technology seals the compute and storage resources in the virtual machine to improve resource efficiency, which becomes the core support of cloud computing platform. Meanwhile, monitoring of virtual machines is also increasingly becoming an important security demand of the virtualization platform. Currently, the monitoring system of virtual machine behavior is usually deployed in the management domain on the same physical equipment of monitored virtual machine. From the perspective of system architecture, the distributed monitoring method is not conducive to a cloud platform provider for unified management of computing resources, and the monitoring behavior in the physical host of monitored virtual machine will consume a lot of limited computing and storage resources on the virtual machine server, thus occupying the limited service resources of the virtual machine. Since the observed monitoring overhead of a virtual machine lies mainly in information processing, a kind of centralized VMI mechanism (CVMI) is capable of dividing the monitoring mechanism, which acquires the status information of the virtual machine on the physical host of monitored virtual machine, and then sends information to the centralized security monitoring server, and analyzes the status information, in order to detect that the security events are operated on the centralized security monitoring server clusters. The experimental results show that, CVMI can effectively monitor the behavior of the virtual machine, and reduce the performance loss caused by introduction of performing monitoring in deployment of physical nodes of virtual machine.

Keywords: virtual machine introspection; virtualization; cloud computing

Introduction

The infrastructure as a service (IaaS) is a kind of computing mode of cloud computing, which provides infrastructure services for a variety of Internet applications. As the technology cornerstone of IaaS, virtualization technology has been widely used in a great number of data centers and server clusters. On a virtualization platform, the virtual machine manager allows multiple virtual machine instances to run on a single physical machine, which is also called as virtual machine. Through configuration and management of the virtual machine, the virtualization platform can meet user's dynamic demand for computing resources and storage resources. As an interface of virtual machine access to the hardware, the virtual machine manager controls over CPU and memory and other physical resources, and owns complete and untempered views of all system states of the virtual machine. Such architecture enables the virtual machine manager to thoroughly observe and analyze the client operating system from the outside of virtual machine, and also meets the needs of transparency and separability. For this reason, the virtual machine introspection technology is proposed and widely used in the monitoring system to enhance the system security and reliability.

On the system architecture, the existing research on virtual machine introspection technology pays more attention to the specific security issues, but often overlooks the system architecture for its large-scale deployment on the virtualization platform. For example, CloudSec [1] and CReW, et al [2] use the distributed architecture, information collection and data analysis and other monitoring modules to scatter on each physical host, which are deployed with monitored virtual machine on the same physical host. The advantage of this architecture is that, when it detects
malicious actions, it can timely intervene and control over the virtual machine, but the distributed architecture is not conducive to centralized storage of information, and flexible loading of analytical strategies. Moreover, compute-intensive analysis work will also occupy resources required by normal operation of virtual machine, which is contradictory with the purpose of improving the use efficiency of resources on the cloud computing platform. We observe that the majority of service providers of cloud platform just need to passively observe the virtual machine, without need of active protection, thus inspiring us to divide the whole process monitored, and perform asynchronous operation of tasks with a high computational complexity. To solve this problem, we propose a centralized VMI mechanism, which delivers the information acquisition work of operating state of virtual machine to the monitoring system deployed on the same physical machine, and also transfers the complex analysis work from the physical host of monitored virtual machine, so as to focus on the maintenance of the normal operation of the virtual machine. On a specific server, we focus on analysis of the information of all virtual machines collected, in order to facilitate to flexibly adopt different analysis functions for different virtual machines or different types of information. The centralized architecture is conducive to storage of information and flexible analysis of security. Meanwhile, the centralized architecture can effectively alleviate the pressure of the physical host of monitored virtual machine, so as to focus on the maintenance and operation of the virtual machine.

The remaining parts of this paper are arranged as follows: the first chapter introduces the relevant work of virtual machine introspection; the second chapter introduces design ideas of CVMI, and presents the architecture diagram and specific implementation; the third chapter assesses the experiment, and verifies the availability and performance of CVMI; the last part is the conclusion of this paper and acknowledgement.

2 RELEVANT WORK

In 2003, Garfinkel and Rosenblum [16] proposed a kind of virtual machine monitoring and analysis method from the virtual machine manager, namely virtual machine introspection. Virtual machine introspection technology is to collect information from outside of the virtual machine, in order to assess the operating state of the virtual machine. These states include the using condition of memory, disk and CPU and other performance information, as well as process, files, ports and upper layer of semantic information. The difficulty of virtual machine introspection technology lies in the problem of semantic gap. Since the virtual machine introspection technology directly acquires memory and register and hardware state, and these states are binary information without any semantics, we expect to acquire the process, files, ports and semantic information. The gap between hardware-level binary information observed outside and semantic-level objects in the virtual machine is the semantic gap.

With the rapid development of virtualization products, virtualization technology is widely concerned has been widespread concern in the market, leading to the evolution of the virtual machine introspection technology and tools used for monitoring state and behavior of the virtual machine.

Now, the virtual machine introspection technology is widely used for malicious code analysis [13], intrusion detection [14], honeypot [11], evidence collection [12, 17], file protection [9] and other fields. The reference [8] presents the classification of virtual machine introspection technology from the perspective of application scenarios.

The virtual machine introspection technology is widely used, because it provides a unique access to information. Virtual machine introspection tools are deployed outside of the virtual machine, thus maintaining transparency of internal procedures of the virtual machine. Meanwhile, the introspection tools are also isolated from malicious code in the virtual machine, thus obtaining a better isolation. Figure 1 shows the basic architecture of the virtual machine introspection system. As an interface of virtual machine access to the hardware, the virtual machine manager provides a special acquisition mode of virtual machine state. The virtual machine introspection systems relies on the virtual machine manager to collect the virtual machine state, and analyze the information, and finally provides the analysis results to the upper layer of security module.

In general, the virtual machine introspection can be divided into two phases: information acquisition phase and data analysis phase.

2.1 Information acquisition

Information acquisition phase mainly constructs the upper layer of semantic information or semantic view according to the hardware state of the virtual machine. Semantic view is comprised by a number of semantic information, which is a subset of the virtual machine state.
To divide from the perspective of information acquisition by the user, the existing virtual machine introspection technology can be divided into active and passive information acquisition mode. Active acquisition mode \([7, 13]\) refers to acquire the semantic information inside the virtual machine by the user through scanning or polling methods, which often uses snapshot technology or timing trigger. Passive acquisition mode \([3, 4, 5, 6, 10, 11, 12, 19]\) needs to install event triggers. When there is an event in the virtual machine, it triggers the event triggers, and the monitor collects the relevant information. Two kinds of information acquisition modes have their advantages and disadvantages. Active mode is lack of timeliness, which is suitable for acquisition of static information of the virtual machine in a certain period. Passive mode is suitable for acquisition of dynamic behavior information, with timeliness. Due to the difference of the specific technology of virtual machine introspection, the references \([15]\) and \([18]\) give a more detailed classification and comparative analysis of the existing virtual machine introspection technology from different perspectives.

2.2 Data analysis

Once the semantic view is established, the next step is to analyze the view and provide services for the upper layer of monitoring module. Phof \([14]\) summarizes the virtual machine introspection technology and presents a formalization model of the virtual machine introspection. In this model, the virtual machine introspection uses some classification or clustering algorithms to analyze data, and provide services for the upper layer of the security module.

3 DESIGN AND IMPLEMENTATION

3.1 Architectural diagram

Most of monitors on the cloud computing platform adopt the distributed architecture, as shown in Figure 2. The distributed architecture is to deploy the monitoring program on each set of virtual machine server. The information collection module and monitoring module are deployed together with the virtual machine on the same physical host. The monitoring module will increase computing pressure of the physical host, and compete for computing resources of the physical host with the virtual machine, thus reducing the performance of virtual machine.

In contrary, the centralized architecture shown in Figure 3 removes the monitoring module from the physical host, thus ignoring the pressure of the virtual machine server, and improving the performance of virtual machine. CVMI uses the centralized architectural diagram shown in Figure 3. CVMI transfers the complex analysis work from the physical host of monitored virtual machine to a specific server, and carries out centralized analysis of information of all virtual machines collected, in order to flexibly adopt different analysis functions based on different virtual machines or different types of information. The centralized architecture is conducive to storage of information and flexible analysis of security. Meanwhile, the centralized architecture can effectively alleviate the pressure of the physical host of monitored virtual machine, so as to focus on the maintenance of operation of the virtual machine.
Figure 4 is an architectural diagram of information collection module. CVMI supports active and passive information acquisition modes. The information collection module and active information acquisition mode are deployed in Dom0 user mode, while the subject of passive information acquisition mode is deployed on the virtual machine manager.

3.2.1 Active semantic acquisition mode
The difficulty of active mode lies in resolving the semantic gap. Semantic gap refers to the gap between hardware-level binary information observed outside and semantic-level objects in the virtual machine. To solve the semantic gap, it requires three conditions: (1) capacity to access the memory of virtual machine; (2) rich information of symbol table; (3) functions that can resolve binary information in memory and mapping relationship of symbol table. Libvmi a set of handy API, and provides the capacity to access the memory of virtual machine. For windows system, Microsoft offers complete symbol information. For Linux system, we acquire part of symbol information from System.map file, and compile some simple kernel modules to acquire richer information. We resolve the problems of mapping between binary information in memory and symbol tables through analysis of relationship between each structure in kernel in an artificial way.

For example, Linux maintains all the process information by two-way circular linked list structure, and the address with header of init_task and init_task can be obtained from the System.map, and then any process structure of task_struct can be acquired through traversal of two-way circular linked list structure of the process. task_struct further tracks files_struct, fdtable, file, dentry and other structure, in order to further acquire the relevant files, ports and other information of this process.

3.2.2 Passive mode
The basic process of information collection in passive mode: (1) event trigger; (2) information collection; (3) operation recovery. Event trigger needs to be captured in event processing of our prior concern. When these concerned events occur in the virtual machine, it is able to trigger VMEXIT and be trapped in virtual machine manager, and then perceived by CVMI. After being trapped in the virtual machine manager, CVMI needs to collect some necessary information, such as register values and process number, and so on. After collection of information, CVMI restores operation of the virtual machine. The event trigger may interrupt the operation of some instructions, so CVMI requires software simulation execution of these instructions before delivering the control right to the virtual machine.

In Figure 4, the event sensor is responsible for intercepting the events trapped and collecting information. The event sensor is located in the virtual machine manager, and the information collection module is deployed in Dom0 user state. The communication between the event sensor and information collector mainly uses Xen event channels and shared memory mechanism. The event sensor is a producer of information, while the information collector is a consumer of information.

3.3 Data analysis module

3.3.1 Active semantic acquisition mode
The difficulty of active mode lies in resolving the semantic gap. Semantic gap refers to the gap between hardware-level binary information observed outside and semantic-level objects in the virtual machine. To solve the semantic gap, it requires three conditions: (1) capacity to access the memory of virtual machine; (2) rich information of symbol table; (3) functions that can resolve binary information in memory and mapping relationship of symbol table. Libvmi a set of handy API, and provides the capacity to access the memory of virtual machine. For windows system, Microsoft offers complete symbol information. For Linux system, we acquire part of symbol information from System.map file, and compile some simple kernel modules to acquire richer information. We resolve the problems of mapping between binary information in memory and symbol tables through analysis of relationship between each structure in kernel in an artificial way.

For example, Linux maintains all the process information by two-way circular linked list structure, and the address with header of init_task and init_task can be obtained from the System.map, and then any process structure of task_struct can be acquired through traversal of two-way circular linked list structure of the process. task_struct further tracks files_struct, fdtable, file, dentry and other structure, in order to further acquire the relevant files, ports and other information of this process.

3.2.2 Passive mode
The basic process of information collection in passive mode: (1) event trigger; (2) information collection; (3) operation recovery. Event trigger needs to be captured in event processing of our prior concern. When these concerned events occur in the virtual machine, it is able to trigger VMEXIT and be trapped in virtual machine manager, and then perceived by CVMI. After being trapped in the virtual machine manager, CVMI needs to collect some necessary information, such as register values and process number, and so on. After collection of information, CVMI restores operation of the virtual machine. The event trigger may interrupt the operation of some instructions, so CVMI requires software simulation execution of these instructions before delivering the control right to the virtual machine.

In the centralized architecture used by CVMI, the information collection module and virtual machine are deployed on the same physical machine, which collect information and forwards the information to the dispatcher located on the monitor server. In the architectural diagram of data analysis module shown in Figure 5, the dispatcher is responsible for collecting information sent from each virtual machine, and distributing to different functions or thread processing based on the types of information, and each thread maintains the semantic view on the monitor server. A view is a subset of the virtual machine state, which is comprised by the process files, ports and other information of each virtual machine. According to the semantic information obtained from the view, the monitor is responsible for a number of security monitoring tasks, and also providing services for upper layer of security modules. Upper layer of monitor and security module can be direct access to the view to acquire the required semantic information, without need of polling of the virtual machine.

The view uses a multi-level link list to store processes, files and other information. CVMI uniquely identifies the data according to <IP of physical host, virtual machine number, process number and register EIP>.

4 EXPERIMENTAL ASSESSMENT
The centralized system architecture can effectively alleviate the pressure of the physical host of monitored
virtual machine, so as to focus on the maintenance of operation of the virtual machine. The experiment mainly compares with the usage rate of Dom0 under three conditions and CPU of virtual machine from the layer of virtual machine manager to assess the performance. These three conditions are respectively closed monitor, distributed monitoring architecture and centralized monitoring architecture. In this experiment, the monitoring program mainly collects process and file information in the virtual machine. The virtual machine uses PCMark05 to operate file compression and decompression, encryption and decryption and other CPU-intensive applications. These applications enable the virtual machine to be in a high-load working state during the experiment. Physical machine owns 12G memory, of which CPU is Intel i7 720, with 4 kernels and 8 threads. Virtual machine is installed with windows 7 x64 system, configured with 1G memory and a VCPU. In this experiment, eight virtual machines are simultaneously operating on a set of physical equipment. On Xen platform, xentop tools can observe CPU utilization of the virtual machine from the layer of virtual machine manager. We use xentop tools to collect information at an interval of 1s, and save in the files. Command is xentop -d 1 -b>./xentop.log.

Figure 6 compares with CPU utilization of Dom0 in a certain period of time (81 samples) under three conditions - closed monitor, distributed architecture and centralized monitoring architecture. PCMark05 program operated by eight virtual machines is in a high-load working state. As can be seen from the diagram, under the condition of closed monitor, on the physical host with 4 kernels and 8 threads, CPU utilization of Dom0 is about 150%. The distributed architecture significantly increases the load of Dom0, so as to use excessive CPU computing resources. Due to the complexity of the system, CPU utilization of Dom0 in the centralized architecture is occasionally too high. On the whole, it is significantly lower than CPU utilization of Dom0 in the distributed architecture.

In Figure 7, the left half part represents the average CPU utilization of Dom0 under three conditions - closed monitor, distributed architecture and centralized architecture. Compared to the distributed architecture, Dom0 in the centralized architecture occupies less CPU computing resources. This is because, in the distributed architecture, Dom0 undertakes more monitoring and analysis tools, which requires to occupying more CPU computing resources. In this experiment, the monitoring tools only collect and process the process and file information in the virtual machine, without performing more onerous monitoring tasks. However, the centralized architecture can effectively alleviate the computing pressure of the physical host of monitored virtual machine. Compared to the distributed architecture, the centralized architecture is capable of saving 14.5% of computing resources of the physical host.

On the other hand, due to limited computing resources of the physical equipment, Dom0 and eight virtual machines are in a high-load working state, which require more computing resources. When Dom0 requires more resources to complete the monitoring and analysis work, the computing resources obtained by the virtual machine will be reduced accordingly. In Figure 7, the right half part represents the average CPU utilization of eight virtual machines under three conditions. When the monitor is closed, the average CPU utilization of the virtual machine is 78.23%. When the distributed architecture is used, the average CPU utilization of the virtual machine is 66.9%, which is reduced by 11.33% compared to that in closed monitor. In the centralized architecture, the average CPU utilization of the virtual machines is 71.14%, which is reduced by only 7.09%. More importantly, in the distributed architecture, when the monitoring tool requires to performing more complex analysis work, it requires more computing resources to occupy CPU resources, leading to further decrease of the average CPU utilization of the virtual machine. For the centralized architecture, the information processing and analysis work is performed on a specific server. Therefore, when the monitoring tool requires to performing more complex analysis work, compared
to the distributed architecture, the virtual machine under the centralized architecture can acquire more CPU computing resources.

5 CONCLUSION

This paper proposes a kind of centralized VMI mechanism. Its advantage is that it dismembers the virtual machine introspection into different modules, and enables each module to run asynchronously, so that the pressure of the virtual machine server can be reduced, and the entire virtual machine introspection system can run efficiently. The experimental results show that, compared to the distributed monitoring architecture, the centralized architecture significantly reduces the resource overhead of the virtual machine server, and guarantees the implementation efficiency of the virtual machine, and also offers guidance and help for the implementation and compilation.

ACKNOWLEDGEMENT

We would like to thank the anonymous reviewers for their valuable comments and help in improving this paper. This work is supported by China National Key Technology Support Program (2012BAH46B02).

REFERENCES