Detection of Malicious Virtual Machines using the VM Lifecycle in a Cloud Computing Environment

Seulgi Lee, Youngsang Shin, Kyungho Son, Haeryong Park

Korea Internet & Security Agency
{sglee, ysshin, khson, hrpark}@kisa.or.kr

Abstract: It is a new challenge to respond to intelligent hacking attacks that exploit new security vulnerabilities arising due to the characteristics of virtualized infrastructure including the dynamic state change of virtual machines in the cloud computing environment. This makes security management much more complicated. This paper shows that these problems can be effectively handled by tracing the virtual machine lifecycle and presents a way of tracing. This paper proposes a technique of tracing the virtual machine lifecycle using logs and the layer for cloud environment configuration and management. Finally, this paper discusses proactive detection of the malicious virtual machine using the VM Lifecycle.
[Seulgi Lee, Youngsang Shin, Kyungho Son, Haeryong Park. **Detection of Malicious Virtual Machines using the VM Lifecycle in a Cloud Computing Environment**. *Life Sci J* 2014;11(10):462-467] (ISSN:1097-8135). http://www.lifesciencesite.com. 63

Keywords: Cloud; Security; Virtual Machine Lifecycle;

1. Introduction

Cloud computing lets users borrow as much computing resources as they want and pay the corresponding charge. Likewise, users can utilize the physically independent resource using virtualization technology. Nonetheless, the government and enterprises are having second thoughts about adopting cloud computing mainly because of concerns over security breaches such as malicious code infection or data leak. Thus, to spread the cloud environment, security threats must be removed.

The vulnerability in a cloud environment and new security threats inheriting the existing ones can be an imminent problem, and these security threats tend to be more intelligent.

Incidents occurring in a cloud computing environment are increasing gradually. Compared to the existing environment, data is concentrated on a cloud environment. As a result, security should be enhanced because there is potential risk of a large quantity of data being disclosed or damaged if an incident occurs.

Incidents occurring due to careless management or incorrect configuration by the user can be analyzed or detected using the virtual machine lifecycle information. For this, the lifecycle information should be traced and managed. Note that hypervisors making up the virtualization layer do not store and manage this information. Thus, this paper proposes a technique of tracing the virtual machine lifecycle. And, this paper discusses malicious virtual machine detection method using analysis of the virtual machine lifecycle. There are some security threat cases in a cloud computing environment. The possibility of virtualization based vulnerabilities being exposed as "VM Rollback Attack"[1]. Thus, this paper suggests threat model in a cloud environment and confirms overcoming threat model using virtual machine lifecycle. Existing studies suggest the virtual machine life cycle tracing technique. Therefore, this paper proposes a practical use of the virtual machine lifecycle.

2. Threat model in a cloud environment

Security threat model can be configured as described below, and inactive malicious virtual machine can be prevented by tracking the VM lifecycle. Scenarios include cloud environment characteristic security vulnerabilities that can occur in cloud services.

Fig. 1 shows a threat model. The model consists of hypervisor and copied VMs, which originate with the same VM. An attacker attempts to infection the original virtual machine and takes some copies. And an attacker deletes the original virtual machine because the original VM has history of copies. If virtual machine (1-4) occur security events and virtual machine5 is halted, we just detect four security events. Systems can't detect the infected virtual machine 5, which is halted. Therefore, an attacker uses running virtual machine 5 for attack later and copies new VMs from the VM5.

A solution of threat model using the virtual machine lifecycle will be proposed in Chapter 4 to resolve the problem of the detection of hidden malicious virtual machine.

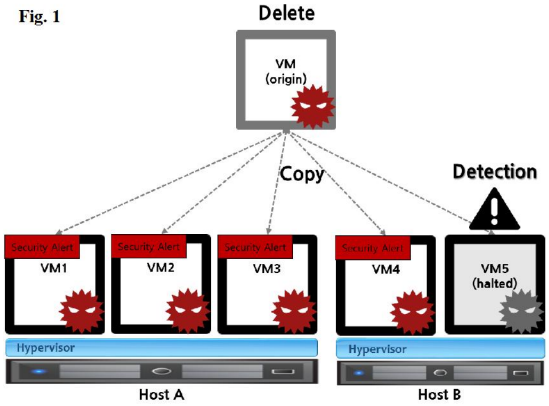

Figure 1. Security Threat Model

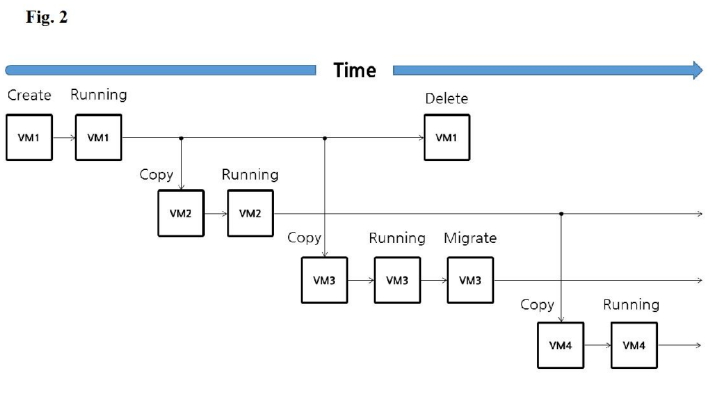

Figure 2. VM Lifecycle example

3. Method of tracing the virtual machine lifecycle

3.1. Definition of the VM Lifecycle

Fig.2 shows a VM Lifecycle example. A virtual machine can be in a state of Created, Running, Down, or Destroyed depending on the command such as Start, Stop, Reboot, and Destroy and its execution result. The series of processes from virtual machine creation to destruction is called "VM Lifecycle [2, 3]."

If we trace the lifecycle of a virtual machine, we can provide the security control function to a cloud environment and important information to detect security threats, which become more intelligent.

Management in a cloud environment is difficult because of the dynamic change of the virtual machine, such as the creation, copying, movement, and deletion of multiple virtual machines. These problems can be solved using the VM Lifecycle. If we trace and save the VM Lifecycle from creation to destruction, we can check the original virtual machine of the examining virtual machine and details of image saving using the snapshot and manage the number of copies and copied location. If we trace the VM Lifecycle, we can allocate resources efficiently and enhance security performance effectively.

3.2. VM Lifecycle tracing by analyzing logs

This chapter discusses a method of tracing the VM Lifecycle by analyzing the “xensource.log” log file created in the open source cloud platform “Citrix XenSever” [4]. The VM Lifecycle tracing with log analysis has an advantage, i.e., no resources are wasted for additional file creation because the log provided by XenServer by default is used.

“xensource.log” has main information which are timestamp, log level, host name, session id, module name, message. Especially log level consist of 4 level(info, debug, warn, error). When the events on virtual machine occur, the next Table 1 is record examples of “xensource.log”. The log of separate event have various characteristic. The events on XenServer is various. So we describe the log of some evenets.

Table 1 shows a chracteristic of events for distinction. Especially, It is so hard for ‘Migrate-Migrated’ pair to distinguish. Because, We have to read two logs which is located separate machines. Firstly, Migrate event occur in machine which has a target virtual machine. “xensource.log” in machine is record ‘VM.pool_migrate’. And the virtual machine is removed in machines, the systems check state of removing VM. The next step is looking for the log in another machine. The log is record ‘VM.pool_migrate’ and complete state. We can trace the VM Lifecycle using “xensource.log”.

Nonetheless, the program of tracing the VM Lifecycle is consist of many modules. And the program is long, complicated. So, stability of the program can be threaten. It also has some shortcomings. For one, virtual machine creation and destruction cannot be known. The format of the

"xensourcee.log" file can also vary depending on XenServer. However, It has merit which is efficiency because of using the log provided Vendor, Citrix. A method of tracing the VM Lifecycle using XAPI will be proposed in Chapter 3.3 to resolve the problem of the existing the VM Lifecycle tracing method using log analysis.

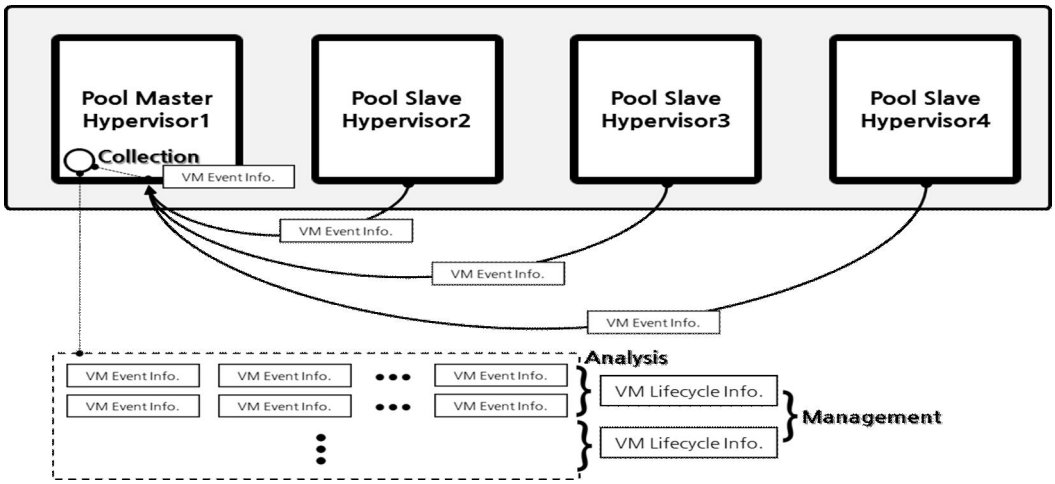

Figure 3. VM Lifecycle Tracing Program Flow

Table 1. "xensource.log" examples on VM event

Event	characteristic
Migrate	VM.pool_migrate VM.remove VM_shutdown VM_check_state
Migrated	VM.pool_migrate VM.pool_migrate_complete VM.import_metadata
Suspend	xenops: VM.suspend
Resume	xenops: VM.resume
Shutdown	xenops: VM.shutdown
Start	xenops: VM.start
Clone	VM.clone

3.3 Lifecycle tracing using XAPI

XAPI is an XML-PRC-based API that provides a function of accessing using HTTPS (443) so that the cloud platform XenServer (sub-project of the open source Xen Project) can be managed[5, 6]. The remote system can also manage XenServer as if issuing a command by accessing the local host. Citrix SDK supports 5 languages: C, C#, Java, PowerShell, and Python [7]. XAPI is a kind of toolstack, a layer for configuring and managing Xen; it is the default toolstack of XenServer and XCP (Xen Cloud Platform)[8]. Unlike other toolstacks, the supported functions are superior, and XAPI management is compatible with the top cloud environment management tools such as OpenStack and CloudStack[9]. And, XAPI is used as a default toolstack(There is a only one toolstack for managing Cloud Datacenter). It is essential element. Therefore, it is important to trace the life cycle using XAPI.

This study tested whether the VM Lifecycle tracing is possible using Citrix XenServer 6.2 SDK and analyzed and arranged the rule.

Fig. 3 shows the operation flow of the VM Lifecycle tracing program. The physical machine running a tracing program accesses the Pool master hypervisor that generates an event, detects the event, and sends the information each time an event occurs. The user can check which virtual machine generated an event and which process was used to execute the event by analyzing the event information.

The test program was configured to save and use the log file as desired by the user when an event occurs, mainly using XenServer SDK's event, VM, and task class. Major items of each class are Timestamp, Class, Operation, name label, and UUID. Unlike the existing log analysis method, life cycle tracing using XenServer SDK has some benefits. In other words, we can configure the format as we want, and we can parse the event information in the memory and send it to the external server that supports integrated security management such as SIEM. Moreover, many events can be analyzed because the parsing process is simple, unlike the existing log analysis method.

Actually, We make the program using python script(reference the example of Citrix XenServer SDK). So we can collect the XAPI events. the middle stage is XAPI event collection result. The result have class, event's action, uuid, name, timestamp, status. The class is various state. But we just use two class which is task and vm. And event's action have three state. the state is add, del, mod. It is important to make groups of two logs(e.g. task add – task del). 'task add - task del' is the start and end of events. In the events, the systems operate a lot of works. Also task pair can be existent in task pair. It is the meaning that is able to record nested.

Table 2. The VM Lifecycle value in database

Column	Description
Host_id	Physical machine(event occurs)
VM_id	Virtual machine(event occurs)
VM_pid	Virtual machine's parent VM
VM_activity	Event class(e.g. start, migrate)
Start_time	Task add time(Event start)
End_time	Task del time(Event finish)
Result	Event status(success or fail)

XAPI Event log Analysis make the VM Lifecycle. And the VM Lifecycle is transfered Cloud ESM. So, We can trace the VMs of whole cloud systems as one VM's lifecycle. One of shortcomings is that current program can't support tracing of lifecycle for using custom template. Custom template is function supported by XenServer. The function means changing of VM status to template library. As a result, the Table 2 shows the value of database.

Figure 4 shows the family tree for virtual machines. Using the VM Lifecycle and basic VM information, we can draw the tree. In cyber crime, we can use the tree for forensics.

Moreover, We can see the color and icon of some virtual machine images. that is the meaning of infected virtual machine. VM2 copies two virtual machine, VM3, VM5. And VM5 makes VM11 and VM7. Because the VMs have relation, we can suspect the VM11 which is halted. So, It is the main idea of scenario on paper.

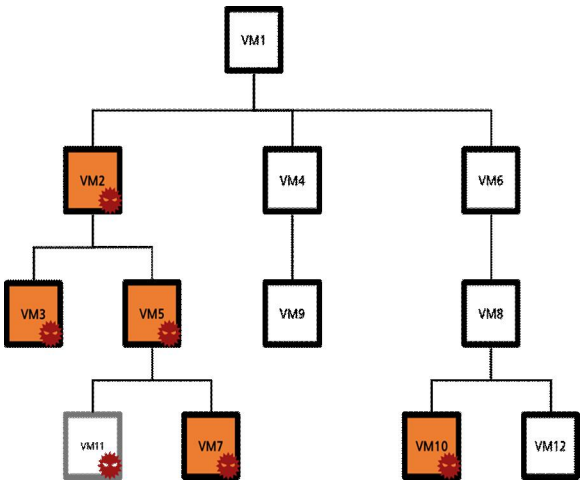

Figure 4. The family tree for virtual machine

4. A solution of the threat model

Table 3 shows the management information of the threat model's VM Lifecycle and VM events that are commonly copied. Simultaneous incidents occurring due to original virtual machine's copy by the user can't be analyzed or detected using the VM Lifecycle because, Systems can't trace the VM Lifecycle. So, we can't detect VM5(halted) which is a infected virtual machine.

Fig. 5 shows the detection method of a malicious virtual machine in threat model's Flow. First, virtual machine 1~4 are occuring individual security alerts. Then the security system analyzes a new clue from the alerts. That is a original VM which is a virtual machine 1~4's parent. The system detects the VM's childs. Because, the child VMs is potential malicious. In this scenario, We suppose that the state of VM5 is halted. But, If we didn't register signatures and can't detect security events on VM5, we apply this case. Analysis with the VM Lifecycle information overcome AV and IPS's false negative case.

In this case, the VM Lifecycle would be the very important clue, the Systems must be detected the VM Lifecycle. The existing cloud platform management systems cannot collect the VM Lifecycle, can't manage history of virtual machine. And, the potential malicious VM couldn't be detected

Table 3. Threat model's VM Lifecycle Info.

VM	VM Lifecycle contents
VM(origin)	Created
VM1	Copied by VM(origin)
VM2	Copied by VM(origin)
VM3	Copied by VM(origin)
VM4	Copied by VM(origin)
VM5	Copied by VM(origin)
VM5	Halted

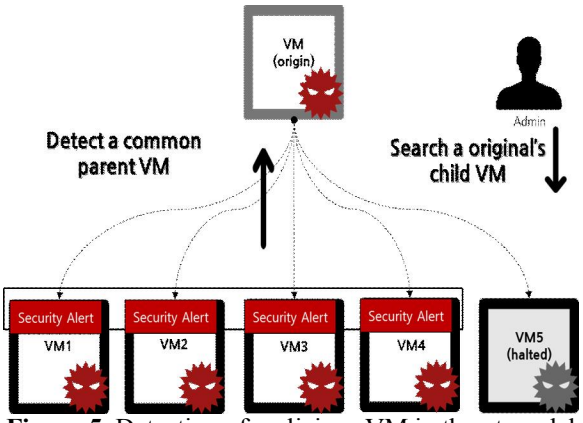

Figure 5. Detection of malicious VM in threat model

5. Utilization of the VM Lifecycle

We discuss the special case of the VM Lifecycle practical use. However outside of that special case, We can utilize the VM Lifecycle information some more. It is the utilization example of cyber crime on cloud computing environment. If cyber crime on cloud occurs, we can form the system environment at the time because of the VM Lifecycle. This is very effective on forensics. Naturally, This scenario is very nice example. But it maybe is possible so for the foreseeable future. We don't have the information which is environment value and memory informataion on cloud systems. It is overcoming using security equipment. Because we support the VM Lifecyle, can check the virtual machine information where the vm is running on physical machine. Consequently, we can know the system environment at the time of cyber crime. For example, private information is leaked from hacking. During tracing origin, we composite environment on cloud systems first. It can be had cloud environment characteristics(i.e. VM Lifecycle). We can trace where is the physical machine and what is the result of malicious virtual machine work. In this paper, we suppose tracing the VM Lifecycle. But using

Network or Host based intrusion prevention systems, we can collect additional security information. So for cope with the such as cyber crime, we adopt the VM Lifecycle for using forensics information. Figure 6 shows the example of reconstruction cloud systems for cyber crime.

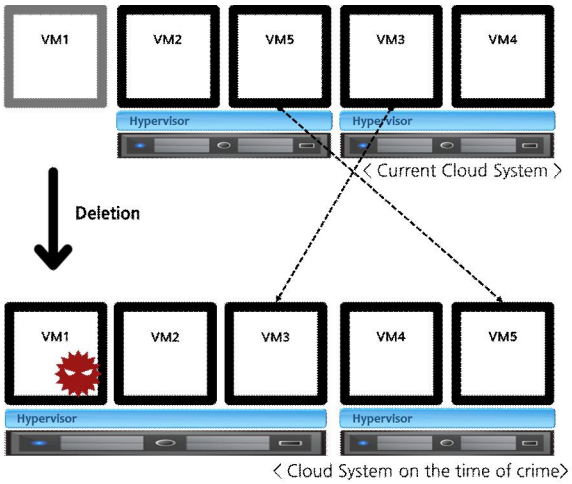

Figure 6. The reconstruction for forensics[10]

We look for security side of cloud systems. But the VM Lifecycle can utilize management side of the systems. When an administrator on cloud systems in company manage VDI(Virtual Desktop Infrastructure), the admin can check a virtual machine history and the VMs creation process and deletion result in whole cloud systems. Because the admin has the information, It is have a merit that utilize resouce management and budget forecast.

So, the effectiveness of managing the VM Lifecycle can be utilized. It is important to analyze with another information such as network connection. In developed program, Supporting custom template is important. But liaison between the cloud based information with the existing information is more than the improvement of program. The relation of them will be provided stable cloud computing environment than before.

6. Conclusion

We have confirmed the necessity of tracing the security threat occurring in a cloud environment and virtual machine lifecycle. And, we propose the practical use of the VM Lifecycle. If we could collect the information, we can reprocess the lifecycle with

few additional information. And then we also researched the method of tracing the VM Lifecycle. Lifecycle tracing using log analysis (existing research method) has shortcomings, i.e., it cannot detect a particular event, and it is significantly affected by format changes. To solve this problem, a method of real-time life cycle tracing was proposed using XAPI. XAPI is the unique toolstack management layer. So we should select it. The characteristic of only measure left make the program using the VM Lifecycle with XAPI versatile. We will consider improving the stability of lifecycle tracing by verifying the connection between the task and virtual machine and checking how the saving information can change if the event fails.

Acknowledgements:

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]

Corresponding Author:
Dr. Youngsang Shin
Korea Internet & Security Agency
Seoul, South Korea
E-mail: ysshin@kisa.or.kr

References

1. Yubin Xia, Yutao Liu, Haibo Chen, Binyu Zang. Defending against VM rollback attack. IEEE/IFIP 42nd International Conference 2012; 1-5.
2. CloudStack. http://cloudstack.apache.org/docs/en-US/Apache_CloudStack/4.2.0/html/Admin_Guide/vm-lifecycle.html.
3. Libvirt. http://wiki.libvirt.org/page/VM_lifecycle
4. Seulgi Lee, Ilahn Cheong, Kyungho Son. A Study on the Logs Analysis for Change tracking of VM State in Cloud Environment. Korea Society of IT Services 2012;236-9.
5. Xen Project. http://wiki.xen.org/wiki/XAPI
6. Wikipedia. http://en.wikipedia.org/wiki/XML-RPC
7. Citrix Systems. http://www.xenserver.org/partners/developing-products-for-xenserver.htm
8. Xen Project. http://wiki.xen.org/wiki/Category:XCP
9. OpenStack. http://www.openstack.org
10. Wikipedia. http://en.wikipedia.org/wiki/Digital_forensic_process