Hypervisor-based Cloud Intrusion Detection System

Jason Nikolai
College of Business and Information Systems
Dakota State University
Madison, SD 57042 USA
janikolai@pluto.dsu.edu

Yong Wang
College of Business and Information Systems
Dakota State University
Madison, SD 57042 USA
yong.wang@dsu.edu

***Abstract*— Shared resources are an essential part of cloud computing. Virtualization and multi-tenancy provide a number of advantages for increasing resource utilization and for providing on demand elasticity. However, these cloud features also raise many security concerns related to cloud computing resources. In this paper, we propose an architecture and approach for leveraging the virtualization technology at the core of cloud computing to perform intrusion detection security using hypervisor performance metrics. Through the use of virtual machine performance metrics gathered from hypervisors, such as packets transmitted/received, block device read/write requests, and CPU utilization, we demonstrate and verify that suspicious activities can be profiled without detailed knowledge of the operating system running within the virtual machines. The proposed hypervisor-based cloud intrusion detection system does not require additional software installed in virtual machines and has many advantages compared to host-based and network based intrusion detection systems which can complement these traditional approaches to intrusion detection.**

Keywords—Cloud Computing, intrusion detection, hypervisor

I. INTRODUCTION

One of the significant challenges in Infrastructure as a Service (IaaS) cloud computing is the lack of ability for cloud users to control security protection in the cloud infrastructure. In a survey of more than 170 businesses, 50 percent of the respondents stated concerns with security issues relating to cloud computing resources [1]. To help address these concerns, controls have been proposed by the Cloud Security Alliance, many of which are process based and are subject to noncompliance and human error. For the controls that are automated and machine based, a gap exists between protecting the cloud users from outsider attacks using perimeter security approaches and attacks from mischievous users who have penetrated the perimeter security controls. These outside attackers become insiders within the cloud environment and can attack other virtual machines within the cloud infrastructure.

One automated security control recommended by the Cloud Security Alliance for cloud computing environments is an intrusion detection system [2]. There are two traditional types of intrusion detection systems: host based and network intrusion detection systems [3]. Host based intrusion detection system are composed of an agent on a host system that examines system calls, logs, file-system modifications, and other host activities to detect intrusions. Network intrusion detection systems monitor network traffic and the content of packets in order to discover malicious traffic.

Both host based and network based intrusion detection systems have advantages and limitations. Network intrusion detection systems attempt to address attacks from outsiders and generally have limited effectiveness against insider attacks [4]. These and other perimeter security controls, such as firewalls, may be less effective in cloud computing environments because of the shared multi-tenancy nature of cloud computing [5]. It is common for multiple cloud users to reside virtually partitioned on a single physical machine which opens up the possibility for attacks over virtual or internal networks [6]. Host based intrusion detection systems can be effective but typically must be monitored and managed by cloud users. This approach can be difficult for cloud users who have several instances in a cloud environment. Furthermore, host based intrusion detection systems can be disabled by a skilled attacker that has breached the instance.

In this paper, we propose a new type of intrusion detection system, **Hypervisor-based Cloud Intrusion Detection System** (HCIDS), to address some of the challenges with traditional intrusion detection systems in cloud environments. HCIDS examines system metrics for cloud instances directly from the hypervisor to seek out potential misuse patterns. Our contributions in this paper include, but are not limited to:

- We propose a hypervisor-based intrusion detection system for cloud environments.
- We demonstrate and verify the effectiveness of the proposed system in a real cloud environment.
- Using developed signatures, we are able to detect 100 percent of two types of denial of service attacks within a cloud environment: denial of service attacks against a cloud instance and a denial of service attacks from a cloud instance against another cloud instance.

This paper is organized as follows. Section II discusses related work. Section III introduces our proposed hypervisor-based cloud intrusion detection system. Section IV demonstrates and verifies the effectiveness of the proposed system. Section V summarizes the paper and future works.

II. RELATED WORK

Our work consists of three components: the use of performance signatures for detecting attacks on a system, detecting anomalies in virtualized environments from outside

of the virtual machine, and an intrusion detection framework for cloud environments.

The use of performance signatures to detect malicious activity and intrusions is proposed in [7] [8]. In [7], Oppenheimer and Martonosi present a model for using performance signature data to detect system security violations. More recently, Avritzer, Cole, and Weyuker demonstrate an approach for detecting attacks on software systems using system performance signatures [8]. In their work, they model the performance characteristics of a system with a reasonable background load to simulate system usage and use CPU, memory, I/O and network usage metrics to detect buffer overflow, stack overflow, SQL injection, denial of service (DoS), and man-in-the-middle attacks. The results from their work show promise for using performance signatures to profile attacks.

An approach for detecting attacks in a virtualized environment outside of the virtual machine instance is to use virtual machine monitor introspection [9]. Garkinkel and Rosenblum present an architecture and prototype using virtual machine introspection to detect attacks in virtual machine instances [10]. In their work, they demonstrate how introspection of the virtual machine can detect rootkits and backdoors, Trojan horses, packet sniffers, and worms by inferring software state based on a priori knowledge of its structure.

Cloud computing intrusion detection is an active research area. To address performance issues with intrusion detection in a cloud computing environment, Dhage, et al propose a distributed intrusion detection system which averts heavy loads on a central intrusion detection server [3]. Their work places multiple mini intrusion detection instances throughout the cloud environment which communicate with a controller. The controller stores pertinent data in cloud logs and uses it for intrusion detection analysis. To enhance the effectiveness and efficiency of network intrusion detection systems, Lin, et al. present a technique for using hypervisors in the cloud to inventory operating systems and services on nodes [11]. Larger security solutions are also suggested such as the Security Audit as a Service architecture for cloud computing environments posited by Doelitzscher [12]. His six-layer security model utilizes modules, including a crypto module, a customer public key infrastructure, an SLA monitoring system, a policy module, a logging module, and an intrusion detection system.

III. HYBERVISOR-BASED CLOUD INTRUSION DETECITON SYSTEM

A. Overview

Hypervisors have access to performance data for the virtual machines that they host. This data provides insight into the activities occurring within a virtual machine without having direct knowledge of the actual operating system, applications or private data residing within the virtual machine. In our proposed Hypervisor-based Cloud Intrusion Detection System (HCIDS), we utilize performance metrics from hypervisors within a cloud environment to detect attack patterns. This approach differs from other performance based intrusion detection systems in that it removes the requirement of having software installed on the host computer, or virtual machine in a virtualized cloud environment. Forcing cloud computing users to install additional software in their instances can be problematic and may be resisted by cloud users. Furthermore, gathering performance metrics directly from the hypervisor and not from the operating system makes our solution operating system agnostic. Using patterns in streaming performance metrics from the hypervisor, we are able to detect and classify abnormal usage.

B. Performance Metrics

The performance metrics used in our work are retrieved from the hypervisors hosting virtual machines within the cloud computing environment. Performance metric data for network data transmitted, network data received, block device read data, block device write data, and CPU utilization is analyzed and is commonly available from all the major virtualization platforms using application programming interfaces (APIs). Our approach retrieves each of these metrics every second. We do not analyze memory utilization because memory allocation is performed once at startup and does not vary with usage, making it irrelevant for detecting attacks.

C. Framework

Our proposed framework consists of three high level components: a controller node, end point nodes, and a notification service. First, the controller node is responsible for near real time analysis of performance data for every virtual machine in the cloud computing environment. Second, the end point nodes gather data on every virtual machine running in the cloud environment from the virtual machine hypervisor and present the data to the controller node. Third,

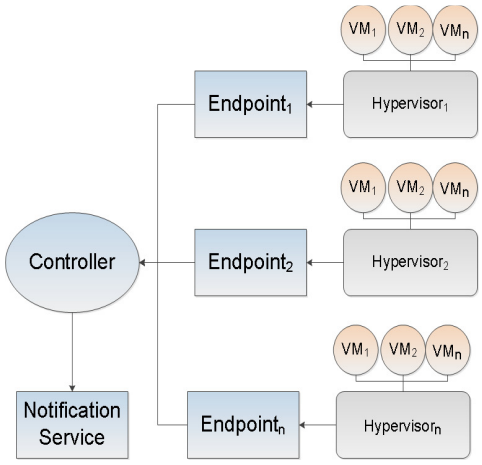

Fig 1. Conceptual Diagram of Proposed System

the notification service provides notification when an attack signature has been identified. Fig 1 illustrates our proposed architecture.

1) Controller Node

The controller node is a service that resides within the cloud environment. Its purpose is to collect and analyze data in near real time from the end point nodes. As data arrives, it is analyzed using a sliding window approach. Windows of performance metrics are analyzed for signatures that suggest suspicious activity.

2) Endpoint Nodes

Endpoint nodes are a conceptual component. They may be agents on each physical system that contains a hypervisor, built within a hypervisor, or API calls to the hypervisor. The purpose of these nodes is to gather and format data from hypervisors and send it to the controller node for analysis. These nodes reside outside of virtual machines and cannot be controlled or manipulated by cloud computing users.

3) Notification Service

The notification service is used to provide alerts that the system has detected a signature of a potential attack. The notification can be a message in a log file, an email, or input into another intrusion detection system.

IV. Experiments and Evaluation

We demonstrate the feasibility of using hypervisor performance metrics to detect attacks on virtual machines in a cloud computing environment using the Eucalyptus infrastructure. Eucalyptus is a private and hybrid cloud solution that is in use by a number of large organizations.

A. Eucalyptus Test Environment

The Eucalyptus Cloud computing system is composed of five main components: cloud controller, Walrus, cluster controller, storage controller, and one or more node controllers [13]. In our test cloud environment, the cloud controller, Walrus server, cluster controller and storage controller all reside on a single physical server. Furthermore, our environment consists of two node controllers which reside on independent physical machines. Table 1 summarizes the hardware configuration of our test environment.

TABLE I. Simulated Cloud Environment

Component	Configuration
Controller	AMD Athlon™ 64x2 Dual Core Processor, 4 GB RAM, Two gigabit Network Interface Cards (NICs)
Endpoint 1	AMD Phonom™ II X4 965 Quad Core Processor, 8 GB RAM, Two gigabit NICs
Endpoint 2	AMD Phenom™ 9150e Quad Core Processor, 6 GB RAM, Two gigabit NICs

B. Simulation Implementation Detail

1) Controller Node

The controller node resides on a machine outside of the Eucalyptus infrastructure and is prototyped using the IBM InfoSphere Streams product. The controller node logic is implemented using IBM Streams Processing Language (SPL) [14] and listens on a UDP socket. Hypervisor performance data is rapidly ingested and analyzed using two sets of sliding windows.

First, ten second sliding windows aggregate data on CPU percent utilization, block device reads, block device writes, network packets received, and network packets transmitted. As metric values enter sliding windows, the mean and maximum values are calculated. Anomalies are defined as values that exceed two times the mean.

Second, a three second sliding window is used to detect attacks. This window populates with anomalies detected from the first sliding window. If an anomaly occurs three times, consecutively, it is labeled as a potential attack pattern and compared to a set of known attack patterns. Unrecognized patterns are ignored.

2) Endpoint Nodes

Two Eucalyptus nodes are used which contain multiple virtual machines. A Python script gathers CPU, block device, and network device metrics using the libvirt API is deployed on each node. This script samples performance metrics every second and sends the data in comma separated value (csv) format to the Controller Node using the UDP protocol for each virtual machine running on the node.

3) Notification Service

The IBM InfoSphere Streams product performs the role of the notification service. The attacks are visualized in a table and written to a file.

C. Simulated Activities

The effectiveness of the system is demonstrated by running denial of service attacks from and against a virtual machine in the cloud environment with and without a simulated valid user workload.

1) Simulated Workload

An Apache web server resides on the cloud instance which serves up a web page that randomly performs different sized reads and writes at intervals of two and five seconds. For each simulated activity, three runs are conducted three times. The first run is performed without a user workload. The second run is conducted with five concurrent users accessing the cloud instance's HTTP server webpage. And, the third run is performed with 10 users concurrently accessing the cloud instance's HTTP server webpage.

2) Simulated Attacks

Two types of denial of service (DoS) attacks are performed to examine the effectiveness of our approach: a HTTP flood attack against a cloud instance and a syn flood attack from the cloud instance against another virtual machine in the cloud environment. First, a denial of service attack against the cloud instance is performed using the tool DoSHTTP [15] from outside the cloud environment using a Windows 7 machine. This attack uses 500 sockets to issue 10,000 requests. Second, hping3 [16] is used to create a syn flood attack from within the virtual machine to attack another virtual machine in the cloud.

3) Attack analysis approach

The primary purpose of our analysis is to determine whether hypervisor performance metrics can be used to detect and classify attacks while minimally flagging normal usage as attacks. We do this by manually observing patterns in performance data when DoS attacks are occurring and creating signatures from these patterns. There are three goals for the attack signatures. The first goal is to reduce or eliminate false positives. A false positive occurs when normal usage is labeled as an attack. An intrusion detection system with a high false positive rate will be ignored or disabled. The second goal is to detect all valid attacks. The more attacks not detected, the less effective the system becomes. And, the third goal is to properly classify the type of an attack. Proper classification is useful for responding to attacks.

With these goals in mind, we perform both DoS attacks three times for approximately 15 minutes under three stress conditions: no users, five concurrent users, and a load of 10 concurrent users. The variability in workload improves the quality of the experimentation by better reflecting a real world cloud application. Furthermore, each run is performed three times to examine the repeatability of results.

We manually observe repeatable anomaly patterns in the hypervisor performance metrics while the attacks occur and create signatures for the attacks. Each signature is composed of five commonly available hypervisor metric variables: Packets Transmitted (Packets TX), Packets Received (Packets RX), Block Device Read Requests (Block Device Read Req), Block Device Write Request (Block Device Write Req), and CPU Utilization (CPU Util.)

A signature is defined by Boolean values for each performance metric. A metric is considered true in a signature if it is repeatedly detected as an anomaly for three consecutive 10 second sliding windows. As previously described, an anomaly is defined as a metric value exceeding twice the mean in a 10 second sliding window. We find that this technique reduces false positives caused by normal system variability. Using this approach, we code patterns for the DoS attacks from and against a cloud instance. The patterns represent the signature for an attack.

The system is applied to normal running conditions without attacks in order to measure false positives. The same three system stress conditions (e.g. no user activity, five concurrent users, and 15 concurrent users) are performed and the results recorded.

D. Results and Observations

The coding of performance metrics reveals repeatable signature patterns for the two DoS attacks. Table II summarizes the attack signatures derived from manual observations during multiple system runs under the three stress conditions.

TABLE II. SIGNATURES

	CPU Util	**Block Device Read Req**	**Block Device Write Req**	**Packets RX**	**Packets TX**
HTTP DoS attack against cloud instance	True	False	False	True	True
Syn Flood from cloud instance	True	False	Any	False	True

To measure the accuracy of signatures, three test runs are performed: DoS attack against the instance, DoS attack from the instance, and no attack for a baseline measurement. Each test run is conducted over a 45 minute period consisting of three 15 minute stress conditions: no user activity, five concurrent users, and 10 concurrent users. Attacks are issued three times during each stress condition, or nine times total per test run. Table III summarizes the results of our findings.

TABLE III. ACCURACY

	False Positives **(number of false positives/number of metric sets analyzed that were not attacks)**	**False Negatives** **(number of attacks not detected/number attacks issued)**	**Misclassifications** **(number of attacks incorrectly classified/number of attacks issued)**
DoS attack against the instance	<1% (8/3043)	0% (0/9)	0% (0/9)
DoS attack from the instance	1.4% (43/3179)	0% (0/9)	0% (0/9)
No attack	0% (0/3091)	NA	NA

A false positive is counted when a set of performance metric data is detected as an attack during normal usage. A false negative is defined as an attack that is not detected. And, misclassifications are attacks that are incorrectly classified as other attacks (e.g., a syn flood attack is classified as a HTTP DoS attack.)

The data from our findings indicate that streaming hypervisor performance metrics can be used to detect denial of service attacks within a cloud environment. Every denial of service attack performed by the instance and against the instance is detected and properly classified. The false positives are mostly detected during the 10 user workload run. We theorize that this workload may emulate a denial of service attack in the environment. Additional investigation is needed to prevent these false positives. Also, it is noteworthy that no false positives are detected when an attack is not applied.

E. Comparison with other approaches

The works in [7, 8] use a host-based intrusion detection approach to run a monitoring agent on a computer to retrieve

performance metrics from the operating system or applications. Our proposed approach does not require any additional software installed in virtual machines. In [9] [10], virtual machine introspection is used. Virtual machine introspection is effective to detect malicious behavior in virtual machines. However, it examines the detailed state of the virtual machine such as memory and register contents and I/O device flags. Cloud users storing confidential data on a cloud instance may have concerns with the snooping of memory on their virtual machines. Further, it also requires knowledge and modification of the underlying operating system. Our approach does not require direct knowledge of the operating system running in virtual machines. We examine the performance metric usage patterns over time. The work in [3] [11] [12] uses distributed intrusion detection system which averts heavy loads on a central intrusion detection server. In our work, an agent is installed on each hypervisor node which communicates with a central decision node.

HCIDS offers at least two advantages over existing intrusion detection techniques. First, monitoring is done outside of the virtual machine and is independent of the operating system or applications running within the virtual machines. This removes the burden of users having to install additional software in their images and cannot be compromised from within the virtual machine instance. Second, insider attacks that potentially would not be detected using perimeter security controls can be detected. If an attacker takes over an instance and then uses that instance to attack other instances in the cloud computing environment, perimeter firewalls and intrusion detection systems generally would not detect this malicious activity.

V. Conclusion

The initial findings from this work indicate that hypervisor performance signatures can be successfully used to detect attacks in a cloud computing environment. This approach offers advantages over other intrusion detection systems alone. First, our framework does not require knowledge of the underlying operating system or applications run on cloud instances. We examine patterns of performance metrics from outside of the instance directly from the hypervisor without placing a burden on the cloud user. Second, the proposed hypervisor-based cloud intrusion detection system can be integrated with and complement existing intrusion detection systems and perimeter defenses to improve the security within cloud environments. When using a defense in depth security strategy, multiple security systems should be considered to detect and thwart attackers.

Our experiments and testing demonstrate the feasibility of using hypervisor metrics for detecting denial of service attacks both against and from a cloud instance. Additional statistical approaches to extract attack patterns and system tuning will be explored next. Further attacks, including enumeration, insider password cracking, and network sniffing will be profiled to test the accuracy of detection and classification systems. Approaches to reduce false positives will be examined. And, a comparison analysis of traditional approaches and our system will be conducted.

References

[1] S. Biggs and S. Vidalis, "Cloud Computing: The Impact on Digital Forensic Investigations," 2010, pp. 1-6.

[2] G. Brunette and R. Mogull, "Security guidance for critical areas of focus in cloud computing v3.0," *Cloud Security Alliance,* 2011.

[3] Dhage*, et al.*, "Intrusion detection system in cloud computing environment," presented at the Proceedings of the International Conference Workshop on Emerging Trends in Technology, Mumbai, Maharashtra, India, 2011.

[4] M. B. Salem*, et al.*, "A survey of insider attack detection research," *Insider Attack and Cyber Security,* pp. 69-90, 2008.

[5] J. Sheridan and C. Cooper, "Whitepaper: Defending the Cloud," 2012.

[6] L. M. Kaufman. Can Public-Cloud Security Meet Its Unique Challenges?, vol. 8, no. 4. IEEE Computer Society, 2010, pp. 55-57.

[7] D. L. Oppenheimer and M. R. Martonosi, "Performance signatures: A mechanism for intrusion detection," in *Proceedings of the 1997 IEEE Information Survivability Workshop*, 1997.

[8] A. Avritzer*, et al.*, "Monitoring for security intrusion using performance signatures," 2010, pp. 93-104.

[9] M. Christodorescu*, et al.*, "Cloud security is not (just) virtualization security: a short paper," in *Proceedings of the 2009 ACM workshop on Cloud computing security*, 2009, pp. 97-102.

[10] T. Garfinkel and M. Rosenblum, "A virtual machine introspection based architecture for intrusion detection," 2003, pp. 253-285.

[11] C. Lin*, et al.*, "Modeling and Analyzing Dynamic Forensics System Based on Intrusion Tolerance," in *Computer and Information Technology, International Conference on*, 2009, pp. 230-235.

[12] F. Doelitzscher*, et al.*, "Designing Cloud Services Adhering to Government Privacy Laws," 2010, pp. 930-935.

[13] Eucalyptus, "Eucalyptus Components," 2013. [Online]. Available: http://www.eucalyptus.com/docs/3.1/ig/euca_components.html

[14] P. Zikopoulos and C. Eaton, *Understanding big data: Analytics for enterprise class hadoop and streaming data*: McGraw-Hill Osborne Media, 2011.

[15] Socketsoft.net*, DoSHTTP*, 2013. [Online]. Available: http://www.socketsoft.net/

[16] S. Sanfilippo, *Hping*, 2013. [Online]. Available: http://www.hping.org/