Faculty Research & Publications   Beacom College of Computer and Cyber Sciences

12-2016

A System for Detecting Malicious Insider Data Theft in IaaS Cloud Environments

Jason Nikolai
Dakota State University

Yong Wang
Dakota State University

A System for Detecting Malicious Insider Data Theft in IaaS Cloud Environments

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

Yong Wang
College of Computing
Dakota State University
Madison, SD 57042 USA
Email: yong.wang@dsu.edu

Abstract—The Cloud Security Alliance lists data theft and insider attacks as critical threats to cloud security. Our work puts forth an approach using a train, monitor, detect pattern which leverages a stateful rule based k-nearest neighbors anomaly detection technique and system state data to detect inside attacker data theft on Infrastructure as a Service (IaaS) nodes. We posit, instantiate, and demonstrate our approach using the Eucalyptus cloud computing infrastructure where we observe a 100 percent detection rate for abnormal login events and data copies to outside systems.

I. INTRODUCTION

On August 25, 2006, Amazon EC2, one of the leading Infrastructure as a Service (IaaS) cloud offerings, went into beta [1]. Since then, cloud computing has become a big business. The largest technology companies in the world now provide cloud computing offerings and solutions [2][3][4]. However, cloud computing is not without challenges. According to the Cloud Security Alliance [5], data theft and insider attacks are two of the nine critical threats facing cloud security.

Insider attacks fall into three categories: malicious, accidental, and non-malicious [6]. Malicious insiders conduct activities such as ip theft, information technology sabotage, fraud, and espionage, with intent of doing harm or personal gain. Accidental insider attacks occur when unintentional misuse of systems is performed by a user without the intent of harm. And, non-malicious insider attacks are intentional attacks where the user attempts to perform self-benefiting activities but without malicious intent.

Technical controls exist for reducing the risk of insider attacks, including intrusion detection systems, security information and event management, data loss prevention, access control systems, and honey-tokens. In addition, non-technical controls are used and consist of psychology prediction models, education and awareness, as well as information security policies [7]. Although controls exist, not all insider attacks can be detected. Furthermore, several approaches for addressing insider attacks are reactive and not predictive in nature. Techniques for preventing such attacks are needed [8].

Although no single approach can prevent all insider threats, a multi-faceted proactive technique can be used to reduce the risk of damage caused by inside attackers [8]. Several types of attacks exist, including unauthorized extraction of data, data tampering, asset destruction, illegal downloading, eavesdropping, spoofing, social engineering, resource misuse, and installing of malicious software [9].

Our work puts forth a new security control for detecting one type of insider attack, unauthorized extraction of data, or data theft. The importance of reducing the risk of data theft gained recent international attention when the National Security Agency contractor, Edward Snowden, downloaded and disseminated classified documents about intelligence programs [7]. A system to detect and possibly thwart such actions has significant potential to contribute to a successful defense in depth [10] security strategy and reduce the damage of data theft from inside attackers.

We posit a system profiling approach for detecting abnormal login activity and data transfers from IaaS cloud computing nodes hosting tenant virtual machines. This approach aims to address the problem of rogue administrators as described in [11] who attempt to steal data from nodes as discussed in [12]. Our approach uses k-nearest neighbors anomaly detection to detect abnormal variations in bytes sent over the network and number of active users on the cloud node. Furthermore, we examine system state data consisting of open files and network connections to improve detection and provide forensic data for investigation.

Unauthorized extraction attacks are especially important to address in IaaS cloud environments to prevent theft of tenant virtual machine data. Although encryption may reduce risk of insiders having the ability to use stolen data, encrypted virtual machine images and data store files may be copied from nodes and attacked offline.

In our system, agents are installed on cloud nodes hosting virtual machines and data. The system is trained under normal cloud operating conditions. Then, the system monitors for anomalies in transmitted network data and active user logins using a k-nearest neighbors anomaly detection. This data is used to detect anomalies that exceed normal operating thresholds established during training.

Our results suggest that using k-nearest neighbors anomaly detection to monitor node network transmissions and number of active users along with system state information can be used to detect 100 percent of abnormal login activity and data copies to outside systems by users. Furthermore, we observe a zero false positive detection rate when anomalies in active user counts and bytes transmitted are detected along with supporting system state data.

The remainder of this paper is organized as follows. Section II summarizes related work. Section III explains our system

and approach for detecting insider attacker data theft. Section IV describes the instantiation of our system. Section V details the evaluation and results of our proposed system. And, Section VI concludes the paper and suggests future works.

II. RELATED WORK

A number of works have been posited to reduce the threat of insider attacks. Stolfo, Salm, and Keromytis posit an approach to mitigate attacks using fog computing where they detect abnormal usage patterns and present potential attackers with misinformation [13]. Claycomb and Nicoll discuss the threat of rogue administrators and suggest process based approaches to deal with the threat. Furthermore, they discuss future research topics which include predictive models [14]. Colombe and Stephens suggest an approach to visualize intrusion detection system data to detect insider attacks [15]. Babu and Bhanu research an approach for using key stroke dynamics to detect insiders [16]. A more related and interesting technique for detecting insider attacks is the use of machine learning and rule based detection posited by Khorshed and Wasimi [17]. They suggest that rule based learning can be used to detect insider attacks in a cloud environment and that continuous cloud monitoring is an important part of cloud security. Sriram, Patel, and Lakshmanan posit a hybrid protocol using selective encryption and data cleaning along with user profiling and decoy technology to address the problem of inside attackers. One aspect of their work related to our approach is the use of a neural network that examines volume of data downloaded, nature of the operations, division of the task, ip address, and log files. [18]. In previous work, we put forth a system for detecting attacks from and against virtual machines in an IaaS cloud environment using anomalies in performance metrics obtained from the hypervisor [19], and a streaming cloud in-trusion monitoring and classication system (SCIMCS) to assist cloud providers with multiple security systems by filtering noisy alert messages and classifying previously recognized attacks [20].

To the best of our knowledge, applying the approach described in this paper is novel. Although insider attack detection and prevention is an active research area, we are unable to find existent works specifically targeting the problem of insider data theft using anomaly detection, system metrics, and system state information. Furthermore, we demonstrate our work through instantiation and experimentation with promising results.

III. INSIDER DATA THEFT DETECTION SYSTEM

In this section, we describe our approach for detecting data theft. Such a system could be applied to an overall defense in depth approach and reduce the risk of insider data theft from rogue administrators.

A. Overview

An IaaS cloud environment typically consists of virtual machines hosted on physical nodes which run node controller agents. A single node may host virtual machine instances of different tenants. Each tenant may run various applications and workloads. In addition, virtual machines are dynamic and may be created and destroyed by tenant requests at anytime resulting in an ever changing environment.

Our work posits an approach to detect data theft within an IaaS cloud environment. More specifically, we seek to detect rogue administrators following the flow in Fig. 1.
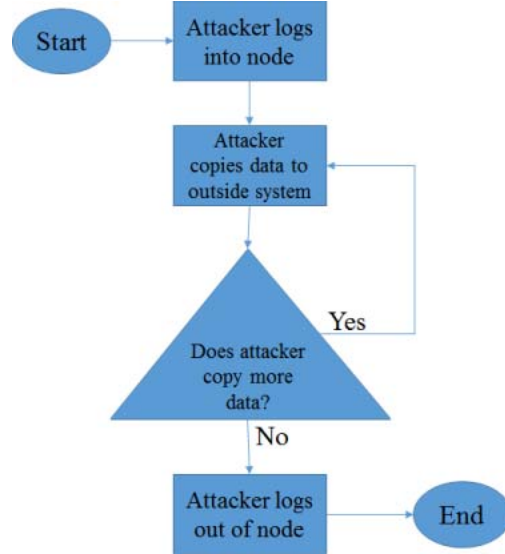

Fig. 1. Insider Data Theft Flow Chart

Most production deployment policies restrict administrator login to systems. Our approach supports these controls by logging unusual login events. In addition, knowledgeable attackers are aware of forensic countermeasures. Hence, all of our detection and analysis must be performed on near real time data and persisted on a remote system.

The specific pattern that our system detects is shown in Fig. 1 and consists of three steps: attacker logs in node, attacker copies data to outside system, and attacker logs out from node. Each event is considered an attack anomaly. In our approach, we examine the anomaly value for the number of active users and amount of data sent from the node. This approach allows the system to adapt to various environments and adjust to normal fluctuations that can occur in the environment. Furthermore, we examine system state forensic data for open connections and open files. In order for data theft to occur, an external connection must exist and data files are open for reading.

B. Approach

Our approach for detecting insider data theft uses a three step technique illustrated in Fig. 2. First, the system is trained. Then, the system is put into monitoring mode. And, finally, a state-based rules approach is used to detect signatures of insider data theft.

1) Train: A goal of training mode is to not burden security operations with excessive tuning in order to achieve accurate results. A system with a complicated training requirement

Fig. 2. Insider Data Theft Defection Approach

lacks scalability. To achieve this goal, the system is placed into training mode while normal IaaS cloud activities occur. Our assumption is that attacks are not occurring during this period. Restricted access and additional manual monitoring may be applied during this period to reduce the likelihood of an attack.

Virtual machines are created and terminated. Tenants run various workloads. Data is sent from an agent on each cloud node hosting virtual machines to our system. During this time, we examine two system metrics for detecting insider attacks: number of active users on nodes and number of tcp bytes transmitted to the network from these nodes. Maximum k-nearest neighbors anomaly scores derived using the IBM^{TM} InfoSphere Streams anomaly detection operator [21] are calculated to be used later in monitoring mode.

Values arrive separately for each metric every second and are stored in memory. The first 20 values create the reference pattern. A current pattern of 10 values is compared to a subsequence of the reference pattern calculating an anomaly score. A total score is computed from the subsequence comparison scores. As each value arrives, a score is computed, and the window slides to the left. As anomalous events occur, the score increases. This total score is the anomaly value used by our system.

The pattern sizes of 20 and 10 are derived through empirical analysis with a goal of accuracy and performance in mind. Tuning these values is beyond the scope of this work and is considered as future work. In addition, although number of active users and bytes sent over the network are the two metrics used for insider data theft detection, we collect metric anomaly data on user space, cpu usage, virtual memory, network connections, input/output read bytes, input/output write bytes, network bytes received, network bytes sent, number of users logged into node, and number of processes. In future work, we plan to investigate machine learning techniques with the goal of deriving more complex insider attack signatures.

2) Monitor: The system is placed into monitor mode with no attack assumptions. Similar to training mode, system metric data is sent from agents on cloud nodes to our system. Virtual machines are created, terminated, and tenant workloads run. Anomaly scores are calculated as in training mode. However, instead of calculating maximum anomalous scores for each system metric, the calculated values are compared to the maximum values derived during the training period. Values that do not exceed the maximum scores are filtered from the system and ignored.

The plot in Fig. 3 illustrates sample anomaly values for network transmission over seconds. Fig. 4 shows the anomaly values compared to the trained reference pattern.
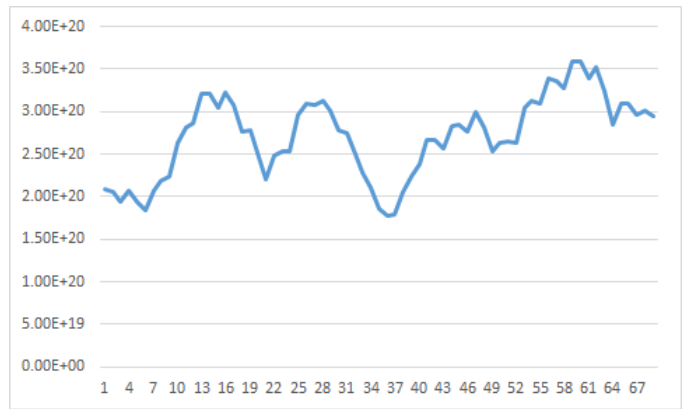

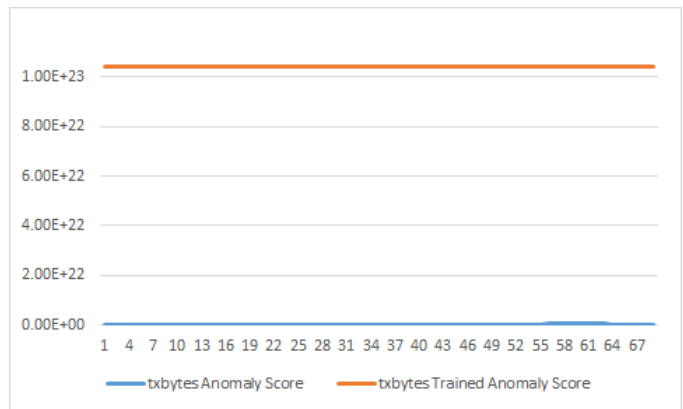Fig. 3. Network txbytes Anomaly Scores under Normal Conditions

Fig. 4. Network txbytes Anomaly Scores under Normal Conditions with Trained Max

From Fig. 4, one can observe that the trained maximum anomaly score is greater than the current tcp bytes transmitted anomaly score. Hence, anomalous activity is not detected.

3) Detect: Detection of insider data theft involves three events: a login anomaly (E1), a data transfer anomaly (E2), and forensic evidence (E3) as shown in Fig. 5. When all three events are present (A3), we observe a 100 percent detection rate with a zero percent false positive rate. In this case, a login anomaly (E1) is detected followed by an abnormal data transfer (E2). And, forensic evidence (E3) is detected for both a network connection to the node and open files being copied. The forensic evidence is collected by the agent and is analyzed after E1 and E2 anomaly events occur.

During experimentation, we examine condition A1 and A2 in isolation. We reproduce a false positive for condition A2 by performing denial of service attacks between virtual machines hosted on nodes in the environment. In the case where E1 is present, a false positive is generated. To simulate A1, we turn off forensic evidence detection to examine the false positive and negative rates. We find that using both E1 and E2 as a vehicle to detect insider attacks is mostly successful. However, instances such as Denial of Service attacks or massive data transfers from virtual machines hosted on nodes can result in
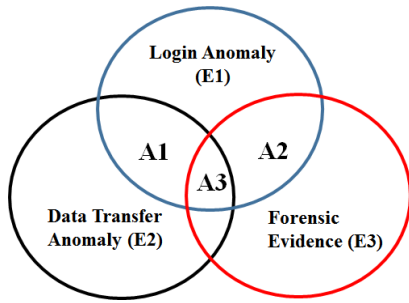
Fig. 5. Detection of Insider Data Theft Venn Diagram

false positives and false negatives.

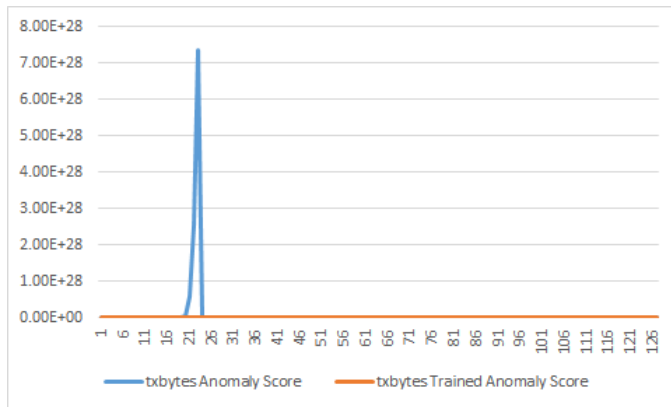Fig. 6 illustrates one sample from our experimentation for event E2.

Fig. 6. Network txbytes Anomaly

Fig. 7 shows a sample from our experimentation for event E1.
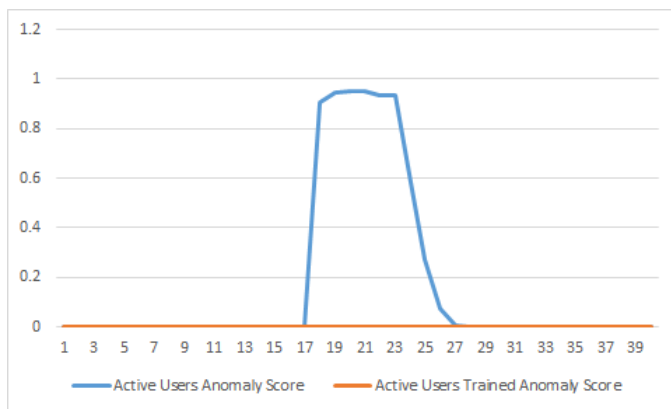

Fig. 7. Active User Anomaly

Both Fig. 6 and Fig. 7 illustrate significant anomaly scores above the trained maximum for network transmitted bytes and active user logins. The detection of all three conditions is required to eliminate false positives.

IV. System Instantiation

We demonstrate our system in an IaaS cloud environment running the Eucalyptus cloud infrastructure shown in Fig. 8.
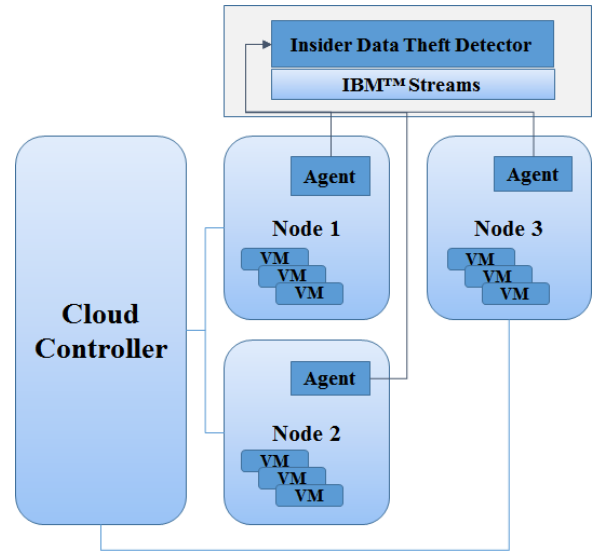

Fig. 8. Cloud Environment

A. Cloud Environment

Fig. 8 provides a conceptual view of the experimentation environment. The Eucalyptus cloud framework is used because of its similarities to the popular Amazon cloud infrastructure. Furthermore, IBM™ InfoSphere Streams provides a scalable infrastructure with built-in analytical functions. Both technologies are available free of charge for research purposes.

The physical environment consists of five multi-core systems connected over Gigabit Ethernet on a private network. The Cloud Controller contains the management components for the Eucalyptus cloud infrastructure [22]. The Nodes contain Eucalyptus node controllers and our agent written in Python to gather system metrics. And, Insider Data Theft Detector runs IBM™ InfoSphere Streams and the implementation of our system.

B. Insider Data Theft Detection System Implementation

Our system implementation consists of two components, the agents that gather system metrics from cloud nodes and the detector which analyzes the data.

1) Agent: The Agent component is written in Python and runs on every node hosting virtual machines. It uses the psutil package along with calls to netstat to gather, format, and send data to the detector using a UDP socket connection. The output includes system metrics as described earlier and open connections as well as file state data.

2) Detector: The Detector is written in two programming languages: Python and Streams Processing Language (SPL). The Python script has two modes of operations, training and monitoring. Similarly, two SPL programs are used for training and monitoring.

In training mode, the Python script ingests metric data from agents and sends it to the SPL program. The anomaly detector operator is used to calculate an anomaly score and maximum training data is persisted in a JSON formatted file.

In monitoring mode, the Python script loads the json file into memory and enriches agent data with maximum anomaly values established during training. The SPL program ingests the data from the Python script and calculates the anomaly score similar to training mode. However, instead of saving scores to a file, anomaly values are compared with the maximum values established during training. Values that exceed the maximum anomaly value for a given metric are passed to an alert script written in Python. System state data is persisted to a file as it arrives from each agent and acts as a forensic trail.

The alert Python script listens for abnormal login events and data transfer events. When an event is detected, the forensic data associated with the event is retrieved from the data file. If forensic data related to open connections and files is retrieved from the data file for the event, an alert is logged indicating that a data theft attack occurred. Login events are always logged.

V. EVALUATION AND RESULTS

This section details training the system, data theft attacks executed, attack messages reported, and summarizes our results.

A. Training the System

Training of the system occurred over a one hour period of time. To test the effectiveness of our approach, we refrain from applying tenant virtual machine workloads during the training period. Instead, we train the system by launching and terminating several different virtual machines. We create up to 12 medium and small virtual machines with Centos 6, Centos 7 and Ubuntu precise images. After the virtual machines become accessible, they are terminated. This approach creates a baseline of activity for the Eucalyptus environment without attempting to predict the workload of users. Furthermore, this meets to the goal for a practical and simplistic training approach.

B. Normal Operating Conditions

Under normal operating conditions, tenant virtual machines are randomly created and terminated placing the cloud environment into various states consisting of starting, stopping, and running virtual machines under load and idle state. Furthermore, at times, tenant virtual machines place excessive network traffic load on 50 percent of the virtual machines transferring data to and from nodes. Load is placed on the virtual machines using system updates and web data transfers.

C. Data Theft Attacks

The goal of our experimentation is to demonstrate the effectiveness for using our system to detect data theft of tenant data on IaaS cloud nodes and virtual machines. We use copies of actual virtual machine data approximately five gigabytes in size. In addition, we test the system with smaller data theft events, including the theft of data files 500 megabytes and 100 megabytes in size. Various data sizes provide supporting evidence for the effectiveness of the system.

D. Attack Messages Reported

We conduct 48 attacks during our experimentation. An attack is considered an unapproved login or data transfer event. Of these attacks, 48 are detected and reported. A sample of the system output is shown below in the following format: [node reporting], [date reported], [time reported], [alert message], [forensic data]. The forensic data is reduced because of space constraints.

..."node1.cloud.res","2016-03-12","14:09:42","[INSIDER] [Node: node1.cloud.res] [Attack Detected: Abnormal user login activity detected] ", "@sconn(...laddr=(192.168.1.98 22) raddr=(192.168.1.110 52925)..."

"node1.cloud.res","2016-03-12","14:13:49","[INSIDER] [Node: node1.cloud.res] [Attack Detected: Abnormally large data transfer detected] ","...popenfile(path= /root/theft1..."

The forensic data reveals the ip address of the attacker and the file being copied.

E. Results

The goal of our work is to detect insider data theft in IaaS cloud environments. Our approach uses three events, login anomalies, data transfer anomalies, and forensic data to detect attacks. During experimentation, we perform attacks using all three events for attack detection and observe a 100 percent detection rate in under 60 seconds with zero false positives for 48 attacks contained in 233,829 data sets sent by node agents. The results are shown in Table I.

We also examine each event in isolation and find flaws in the use of single events:

1) Login Events (E1) Only: Examining login events in isolation using our approach detects 100 percent of anomalous user login and logout activity. However, this cannot be used to detect insider data theft.

2) Data Transfer Anomaly Events (E2) Only: Data transfer anomalies can be solely used to detect data theft events. However, we observe an unacceptably high 22.6 percent false positive rate under extreme operating conditions. These extremes occur during excessive starting and stopping of all virtual machines in the cloud environment and under heavy cloud tenant data transfer workload. Furthermore, when performing denial of service attacks between tenant nodes, we observe a 100 percent false positive rate using these events in isolation.

3) Forensic Data (E3) Only: System state data for open connections and open files also can be used to detect both abnormal login activity and data theft attacks. However, this

TABLE I
EXPERIMENTATION RESULTS

IaaS State	VM Work-load	Type of Attack	Number of Attacks	Percent Detected
No Virtual Machines Running	None	Login	3	100%
No Virtual Machines Running	None	Five GB Data Theft	3	100%
10 Virtual Machines Starting	None	Login	3	100%
10 Virtual Machines Starting	None	Five GB Data Theft	3	100%
10 Virtual Machines Running	None	Login	3	100%
10 Virtual Machines Running	None	Five GB Data Theft	3	100%
10 Virtual Machines Stopping	None	Login	3	100%
10 Virtual Machines Stopping	None	Five GB Data Theft	3	100%
10 Virtual Machines Running	Five VM workload	Login	3	100%
10 Virtual Machines Running	Five VM workload	Five GB Data Theft	3	100%
10 Virtual Machines Stopping	Five VM workload	Login	3	100%
10 Virtual Machines Stopping	Five VM workload	Five GB Data Theft	3	100%
10 Virtual Machines Running	Five VM workload	Login	3	100%
10 Virtual Machines Running	Five VM workload	500 MB Data Theft	3	100%
10 Virtual Machines Running	Five VM workload	Login	3	100%
10 Virtual Machines Running	Five VM workload	100 MB Data Theft	3	100%

approach is unreliable. During our experimentation, we observe normal connection activity between the node controller and the cloud controller. While rules could be created to filter out known connections, the complexity of creating rules and filters would complicate the system and not meet our requirement for usability and ease of use.

VI. CONCLUSION

We put forth a train, monitor, detect pattern for detecting insider data theft attacks. Our system profiling approach utilizes a combination of system metric anomalies and system state data. More specifically, we use a k-nearest neighbors anomaly detection algorithm to score the number of active users on nodes and bytes sent over the network. Excessive scores compared to scores calculated during training indicate an attack event. In addition, system state data on open connections and files is collected and analyzed. Our experimentation suggests that the combination of login events, data transfer events and system state events results in a 100 percent detection rate for insider data theft attacks with a zero percent false positive rate.

To expand on our work, three areas should be explored. First, scalability of the approach needs to be tested in a large IaaS cloud environment. Second, various anomaly detection approaches should be explored. And, third, leveraging machine learning techniques to find rules may reveal combinations of system metrics for better detection of insider attacks.

REFERENCES

[1] J. Barr, "Amazon EC2 Beta," 2006. [Online]. Available: https://aws.amazon.com/blogs/aws/amazon_ec2_beta/
[2] Google, "Google Cloud Platform," 2015. [Online]. Available: https://cloud.google.com/
[3] IBM, "IBM Cloud," 2015. [Online]. Available: http://www.softlayer.com/
[4] Microsoft, "Microsoft Cloud," 2015. [Online]. Available: http://www.microsoft.com/enterprise/
[5] Cloud Security Alliance, "The Notorious Nine Cloud Computing Top Threats in 2013," Tech. Rep. [Online]. Available: https://downloads.cloudsecurityalliance.org, year = 2013
[6] C. Willis-Ford, "Education & Awareness: Manage the Insider Threat," in *Fissea Working Group*, 2015. [Online]. Available: http://csrc.nist.gov/organizations/fissea/2015-conference/presentations/march-24/fissea-2015-willis-ford.pdf
[7] N. Elmrabit, S.-H. Yang, and L. Yang, "Insider threats in information security categories and approaches," pp. 1–6, 2015.
[8] M. Maxim, "defending against insider threats to reduce your IT risk," *Security and Compliance, Jan*, 2011.
[9] M. B. Salem, S. Hershkop, and S. J. Stolfo, "A survey of insider attack detection research," in *Insider Attack and Cyber Security*. Springer, 2008, pp. 69–90.
[10] SANS, "Defense in Depth," 2001. [Online]. Available: https://www.sans.org/reading-room/whitepapers/basics/defense-in-depth-525
[11] R. C. William, "Insider Threats to Cloud Computing: Directions for New Research Challenges," N. Alex, Ed., vol. 0, 2012, pp. 387–394. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/COMPSAC.2012.113
[12] A. J. Duncan, S. Creese, and M. Goldsmith, "Insider Attacks in Cloud Computing," pp. 857–862, 2012.
[13] S. J. Stolfo, M. B. Salem, and A. D. Keromytis, "Fog Computing: Mitigating Insider Data Theft Attacks in the Cloud," pp. 125–128, 2012.
[14] W. R. Claycomb and A. Nicoll, "Insider Threats to Cloud Computing: Directions for New Research Challenges," pp. 387–394, 2012.
[15] J. B. Colombe and G. Stephens, "Statistical profiling and visualization for detection of malicious insider attacks on computer networks," in *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*. ACM, 2004, pp. 138–142.
[16] M. B. Bondada and S. M. S. Bhanu, "Analyzing User Behavior Using Keystroke Dynamics to Protect Cloud from Malicious Insiders," pp. 1–8, 2014.
[17] M. T. Khorshed, A. Ali, and S. A. Wasimi, "Monitoring Insiders Activities in Cloud Computing Using Rule Based Learning," pp. 757–764, 2011.
[18] M. Sriram, V. Patel, D. Harishma, and N. Lakshmanan, "A Hybrid Protocol to Secure the Cloud from Insider Threats," pp. 1–5, 2014.
[19] J. Nikolai and Y. Wang, "Hypervisor-based cloud intrusion detection system," *2014 International Conference on Computing, Networking and Communications (ICNC)*, pp. 989–993, 2014. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6785472
[20] J. Nikolai and Y. Wang, "A streaming intrusion monitoring and classification system for iaas cloud," in *2016 IEEE 9th International Conference on Cloud Computing*, June 2016.
[21] J. Cancilla, "Anomaly Detection in Streams," 2015. [Online]. Available: https://developer.ibm.com/streamsdev/docs/anomaly-detection-in-streams/
[22] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "The eucalyptus open-source cloud-computing system." IEEE, 2009, pp. 124–131.