A Framework for Improved Intrusion Detection and Countermeasure Selection in Cloud Systems

P.Ramya
Student, Department of Computer Science
UCEK, Kakinada
Andhra Pradesh, India
ramya.pasapuleti@gmail.com

Mrs.Suneetha Eluri
Assistant Professor, Department of Computer Science
UCEK, Kakinada
Andhra Pradesh, India
suneethaeluri83@gmail.com

***Abstract*—** With the recent rise of cloud computing, virtualization has become an increasingly important technology. Virtual machines (VM) are rapidly replacing physical machine infrastructures for their abilities to emulate hardware environments and share resources. Virtual machines are vulnerable to theft and denial of service attacks. Cloud users can install vulnerable software on their VMs, which essentially leads to violation in cloud security. Attackers can explore vulnerabilities of a cloud system and compromise virtual machines to deploy further large-scale Distributed Denial-of-Service (DDoS). This paper proposes a framework to detect and mitigate attacks in the cloud environment. Host-based IDS solutions are incorporated in Network Intrusion Detection and Countermeasure Selection in virtual Network Systems (NICE) framework to cover the whole spectrum of IDS in the cloud system. The proposed solution improves the detection accuracy.

Keywords*-Intrusion detection system; virtual machines; Distributed Denial-of –Service; Attack graph*

*****

I. INTRODUCTION

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network. Since cloud computing is distributed in nature, supports multi-user and multi-domain platform, it is more prone to security threats.

A. Cloud Computing

Cloud computing refers to applications and services offered over the Internet. These services are offered from data centers all over the world, which collectively are referred to as the "cloud." The main enabling technology for cloud computing is virtualization. Servers in the cloud can be physical machines or virtual machines. Advanced clouds typically include other computing resources such as storage area networks (SANs), network equipment, firewall and other security devices.

Cloud computing offers three different service models:

1) Infrastructure as a Service (IaaS)
2) Platform as a Service (PaaS)
3) Software as a Service (SaaS)

These cloud services may be offered in a public, private or hybrid network.

Cloud computing exhibits the following key characteristics:

- Agility improves with users' ability to re-provision technological infrastructure resources.
- Application programming interface (API) accessibility to software that enables machines to interact with cloud software in the same way that a traditional user interface facilitates interaction between humans and computers.
- Device and location independence enable users to access systems using a web browser regardless of their location or what device they use.
- Virtualization technology allows sharing of servers and storage devices and increased utilization. Applications can be easily migrated from one physical server to another.
- Reliability improves with the use of multiple redundant sites, which makes well-designed cloud computing suitable for business continuity and disaster recovery.
- Maintenance of cloud computing applications is easier, because they do not need to be installed on each user's computer and can be accessed from different places.

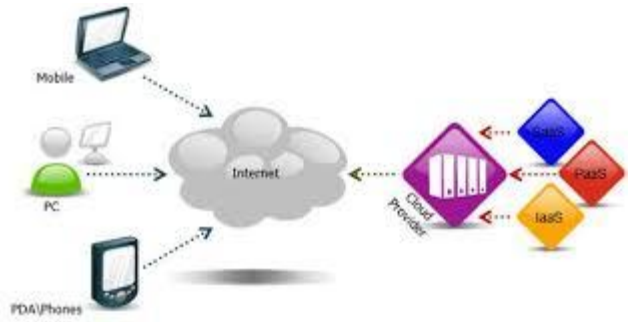

Figure 1. Cloud Computing

B. Intrusion Detection System

Intrusion detection (ID) is a type of security management system for computers and networks. An Intrusion detection system gathers and analyzes information from various areas within a computer or a network to identify possible security violations, which include both intrusions (attacks from outside the organization) and misuse (attacks from within the organization). It uses vulnerability assessment, which is a technology developed to assess the security of a computer system or network. There are two general types of intrusion detection systems: a host-based Intrusion Detection System (HIDS) and a network-based Intrusion Detection System (NIDS).

 ISSN: 2321-8169
1812 – 1815

Network-based IDS can be a dedicated hardware appliance, or an application running on a computer attached to the network. It monitors traffic in a network or coming through an entry-point such as an Internet connection. It can monitor traffic only in its local network segment, unless it employs sensors. In switched and routed networks, a sensor is required in each segment in which network traffic is to be monitored. When a sensor detects intrusion, it will report it to a central management console.

Host-based IDS is usually a software application installed on a system and monitors activity only on that local system. It communicates directly with the operating system and has no knowledge of low-level network traffic. Most host-based intrusion detection systems rely on information from audit and system log files to detect intrusions. They can also monitor system files and system resources, and incoming application data.

II. RELATED WORK

A. Attack Graph Model

An attack graph is a modeling tool to demonstrate all possible multi-host, multi-stage attack paths. System administrator analyze attack graph to understand threats and then to decide appropriate security measures. In an attack graph, each node represents either precondition or consequence of an exploit. Attack graph identifies potential threats, possible attacks and known vulnerabilities in a cloud system. Since the attack graph provides details of all known vulnerabilities in the system and the connectivity information, we get a whole picture of current security situation of the system. Using attack graph we can predict the possible threats and attacks by correlating detected events. If an event is recognized as a potential attack, specific countermeasures can be applied to mitigate its impact or take appropriate actions to prevent it from contaminating the cloud system.

Attack graph is constructed based on the cloud system information, Virtual network topology and configuration information, Vulnerability information.

Definition 1 (Scenario Attack Graph). *A Scenario Attack Graph is a tuple SAG= (V, E), where*

1) $V = N_C \cup N_D \cup N_R$ denotes a set of vertices that include three types namely conjunction node N_C to represent exploit, disjunction node N_D to denote result of exploit, and root node N_R for showing initial step of an attack scenario.

2) $E = E_{pre} \cup E_{post}$ denotes the set of directed edges. An edge $e \in E_{pre} \subseteq N_D \times N_C$ represents that N_D must be satisfied to achieve N_C. An edge $e \in E_{post} \subseteq N_C \times N_D$ means that the consequence shown by N_D can be obtained if N_C is satisfied.

Node $v_c \in N_C$ is defined as a three tuple (Hosts, vul, alert) representing a set of IP addresses, vulnerability information and alerts. N_D works like a logical OR operation and contains details of the results of actions. N_R represents the root node of the scenario attack graph.

Alert correlation is used to identify the source and target of the intrusion in a multi-step attack. We define Alert Correlation Graph (ACG) to map alerts in ACG to their respective nodes in SAG.

Definition 2 (Alert Correlation Graph). *An ACG is a three tuple ACG = (A, E, P), where*

1) A is a set of aggregated alerts. An alert $a \in A$ is a data structure (src, dst, cls, ts) representing source IP address, destination IP address, type of the alert, and timestamp of the alert respectively.

2) Each alert a maps to a pair of vertices (v_c, v_d) in SAG using map(a) function, i.e., $map(a) : a' \rightarrow \{(v_c, v_d) | (a.src \in v_c.Hosts) \wedge (a.dst \in v_d.Hosts) \wedge (a.cls = v_c.vul)\}$.

3) E is a set of directed edges representing correlation between two alerts (a, a') if criteria below satisfied:

i. $(a.ts < a'.ts) \wedge (a'.ts - a.ts < threshold)$

ii. $\exists (v_d, v_c) \in E_{pre} : (a.dst \in v_d.Hosts \wedge a'.src \in v_c.Hosts)$

4) P is set of paths in ACG. A path $S_i \subset P$ is a set of related alerts in chronological order.

Algorithm 1 returns a set of attack paths *S* in Attack Correlation Graph.

Algorithm 1 Alert_Correlation

Require: alert a_c, SAG, ACG
1: **if** (a_c is a new alert) **then**
2: create node a_c in ACG
3: $n_1 \leftarrow v_c \in map(a_c)$
4: **for all** $n_2 \in parent(n_1)$ **do**
5: create edge (n_2.alert, a_c)
6: **for all** S_i containing a **do**
7: **if** a is the last element in S_i **then**
8: append a_c to S_i
9: **else**
10: create path S_{i+1} = {subset(S_i, a), a_c}
11: **end if**
12: **end for**
13: add a_c to n_1.alert
14: **end for**
15: **end if**
16: **return** S

III. EXISTING SYSTEM

Network intrusion detection and countermeasure selection in virtual network systems (NICE) establishes a defense-in-depth intrusion detection framework. It incorporates attack graph analytical procedures into the intrusion detection processes for better attack detection.

A. NICE System Overview

Major components in NICE framework are distributed and light-weighted network intrusion detection agent (NICE-A) on each cloud server, a network controller, a VM profiling server, and an attack analyzer.

1) NICE-A:

Network intrusion detection agent (NICE-A) is deployed on each cloud server to capture and analyze cloud traffic. It scans the virtual system vulnerabilities within a cloud server. When suspicious or anomalous traffic is detected intrusion detection alerts are sent by NICE agent to Attack analyzer.

2) VM Profiling:

Virtual machines in the cloud can be profiled to get information about their state, services running, open ports, etc. VM profile contains details about its connectivity with other

VMs. Knowledge of services running on a VM is required to verify the authenticity of alerts pertaining to that VM. An attacker can use port scanning program to examine the network to look for open ports on any VM. Information about open ports on a VM and the history of opened ports plays a significant role in determining how vulnerable the VM is. All these factors combined will form the VM profile. VM profiles are maintained in a database and contain information about vulnerabilities, alert and traffic.

3) Attack Analyzer:

The major functions of attack analyzer includes attack graph construction and update, alert correlation and countermeasure selection. The process of constructing and utilizing the Scenario Attack Graph (SAG) consists of three phases: information gathering, attack graph construction, and potential exploit path analysis. With this information, attack paths can be modeled using SAG.

After receiving an alert from NICE-A, alert analyzer matches the alert in the ACG. If the alert already exists in the graph, the attack analyzer performs countermeasure selection procedure and then notifies network controller to deploy the countermeasure actions.

4) Network Controller:

Network controller is responsible for collecting network information and provides input to the attack analyzer to construct attack graphs. Network controller is also responsible for applying the countermeasure. Based on optimal return of investment and risk probability of the node countermeasures are selected by attack analyzer and executed by network controller.

B. Countermeasure Selection

Algorithm 2 presents how to select the optimal countermeasure for a given attack scenario. Input to the algorithm is an alert, attack graph G, and a pool of countermeasures CM.

Definition 3 (Countermeasure Pool). *A countermeasure pool CM = {cm1, cm2, . . . , cmn} is a set of countermeasures. Each cm ∈ CM is a tuple cm = (cost, intrusiveness, condition, effectiveness), where*

1) cost is the unit that describes the expenses required to apply the countermeasure in terms of resources and operational complexity, and it is defined in a range from 1 to 5, and higher metric means higher cost;

2) intrusiveness is the negative effect that a countermeasure brings to the Service Level Agreement (SLA) and its value ranges from the least intrusive (1) to the most intrusive (5), and the value of intrusiveness is 0 if the countermeasure has no impacts on the SLA;

3) condition is the requirement for the corresponding countermeasure;

4) effectiveness is the percentage of probability changes of the node, for which this countermeasure is applied.

Algorithm 2 Countermeasure_Selection

Require: Alert, G(E, V), CM
1: Let v_{Alert} = Source node of the Alert
2: **if** Distance_to_Target(v_{Alert}) > threshold **then**
3: Update_ACG
4: **return**
5: **end if**
6: Let T = Descendant(v_{Alert}) ∪ v_{Alert}
7: Set Pr(v_{Alert}) = 1
8: Calculate_Risk_Prob(T)
9: Let benefit[|T|, |CM|] = ∅
10: **for** each t ∈ T **do**
11: **for** each cm ∈ CM **do**
12: **if** cm.condition(t) **then**
13: Pr(t) = Pr(t) ∗ (1 − cm.effectiveness)
14: Calculate_Risk_Prob(Descendant(t))
15: benefit[t, cm] = ΔPr(target_node)
16: **end if**
17: **end for**
18: **end for**
19: Let ROI[|T|, |CM|] = ∅
20: **for** each t ∈ T **do**
21: **for** each cm ∈ CM **do**
22: $$ROI[t, cm] = \frac{benefit[t, cm]}{cost.cm + intrusiveness.cm}$$
23: **end for**
24: **end for**
25: Update_SAG and Update_ACG
26: **return** Select_Optimal_CM(ROI)

IV. PROPOSED SYSTEM

In the proposed system Host-based Intrusion Detection system is integrated in NICE to improve attack detection. Host-based Intrusion Detection system analyzes the traffic to and from the specific computer on which the intrusion detection software is installed. The software makes use of log files or auditing agents of the system in the form of sources of data. A host-based system monitors as well as analyzes a systems internals along with the network packets. The result of the scan performed by the host-based intrusion detection system are logged into a secure database and compared with the knowledge base to detect any malicious activity. A host-based system analyzes logs and consists of information regarding the status of your system. Log analysis-based HIDS includes: collection of log file data, pre-decoding of log file, decoding of log file, analysis of log file and report events.

A. System components

1) Log Monitor:

Log monitor monitors the log files. It monitors three kinds of event logs: application log, security log and system log. It will check every log file to find whether there is a change in the log file. If there is a change, then log monitor will report to the log analyzer.

2) Log analyzer:

Log analyzer receives events from the log monitor and matches them with the rule base to determine whether there is invasion. If there is invasion occurrence, it reports to the response unit.

3) Response unit:

Receives events from the log analyzer and decides what kind of operation to perform. Usually, the normal operations include notifying users, auditing, disconnecting from network and so on.

B. *Advantages*

- HIDS can provide another layer of security by detecting attacks missed by other security tools in the architecture.
- HIDS can determine whether an attack occurred or not with greater accuracy and fewer false positives.
- Direct control over system entities like memory, registry, system files etc

V. CONCLUSION

In this paper a multiphase detection system is proposed to detect various attacks in the cloud. In proposed solution Host-based Intrusion Detection System is incorporated in the NICE framework. It covers the whole spectrum of Intrusion Detection System and improves the detection accuracy. It can reduce the Distributed Denial-of-Service attacks in the cloud virtual system environment. It can reduce the risk of the cloud system from being exploited and abused by internal and external attackers.

REFERENCES

[1] Chun-Jen Chung, Pankaj Khatkar, Tianyi Xing, Jeongkeun Lee, and Dijiang Huang, "Network Intrusion Detection and Countermeasure Selection in Virtual Network Systems", IEEE Trans. Dependable and Secure Computing, vol. 10, no. 4, July/August 2013

[2] Cloud Security Alliance, "Top threats to cloud computing v1.0," https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf, March 2010.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," ACM Commun., vol. 53, no. 4, pp. 50–58, Apr. 2010.

[4] B. Joshi, A. Vijayan, and B. Joshi, "Securing cloud computing environment against DDoS attacks," IEEE Int'l Conf. Computer Communication and Informatics (ICCCI '12), Jan. 2012.

[5] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. M. Wing, "Automated generation and analysis of attack graphs," Proc. IEEE Symp. on Security and Privacy, 2002, pp. 273–284.

[6] X. Ou, S. Govindavajhala, and A. W. Appel, "MulVAL: a logic-based network security analyzer," Proc. of 14th USENIX Security Symp., pp. 113–128. 2005.

[7] R. Sadoddin and A. Ghorbani, "Alert correlation survey: framework and techniques," Proc. ACM Int'l Conf. on Privacy, Security and Trust: Bridge the Gap Between PST Technologies and Business Services (PST '06), pp. 37:1–37:10. 2006.

[8] S. Roschke, F. Cheng, and C. Meinel, "A new alert correlation algorithm based on attack graph," Computational Intelligence in Security for Information Systems, LNCS, vol. 6694, pp. 58–67.Springer, 2011.

[9] A. Roy, D. S. Kim, and K. Trivedi, "Scalable optimal countermeasure selection using implicit enumeration on attack countermeasure trees," Proc. IEEE Int'l Conf. on Dependable Systems Networks (DSN '12), Jun. 2012.

[10] X. Ou, W. F. Boyer, and M. A. McQueen, "A scalable approach to attack graph generation," Proc. of the 13th ACM conf. on Computer and communications security (CCS '06), pp. 336–345. 2006.

[11] Mitre Corporation, "Common vulnerabilities and exposures, CVE," http://cve.mitre.org/. 2012.

[12] O. Database, "Open source vulnerability database (OVSDB)," http://osvdb.org/. 2012

[13] NIST, "National vulnerability database, NVD," http://nvd.nist.gov. 2012