Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 5, May 2014, pg.83 – 88

RESEARCH ARTICLE

Host-Based Intrusion Detection and Attack Graph Selection in VNS

B.Manikandan[1], S.Balaji[2]

Department of Information Technology, V.S.B.Engineering College, Karur, Tamilnadu, India
Department of Information Technology, V.S.B.Engineering College, Karur, Tamilnadu, India
Email:manikdoit2124@gmail.com Email**:** hereiambalaji@gmail.com

Abstract— Cloud Computing has several major issues and concerns, such as expectations regulations, performance, trust, and data security issues. DDOS is a multiple hosts attacks made simultaneously in all network. DDoS attacks performed some vulnerable action in early stage such as low-frequency vulnerability scanning, multistep exploitation, and identifying the compromised vulnerable virtual machines as zombies. In cloud environment we can't find the zombies easily in infrastructure structure as a service (IaaS) clouds. This is happen by installing the vulnerable activities in virtual machines. The aim of this research is to examine the major security issues affecting Cloud Systems and the solutions available. And to prevent the countermeasures, zombies and further vulnerable activities we proposed a system called NICE. It is built on attack graph-based analytical models and reconfigurable virtual network-based countermeasures. The implementation of lightweight mirroring-based network intrusion detection agent (NICE-A) on each cloud server is to capture and analyze cloud traffic. The VM enter into inspection state, virtual network reconfigurations can be deployed to the inspecting VM to make the potential attack behaviors prominent.

Keywords - Performance of Systems, Computer Systems Organization, Network-Level Security and Protection, Attack analyzing, Communication Networking

I. INTRODUCTION

Cloud security is responsible to a familiar set of security challenges that manifest differently in the cloud. New technologies and surrounding the data center require a different approach. A set of polices, technologies, and controls designed to protect data and infrastructure from attack and enable regulatory compliance. A recent Cloud Security Alliance (CSA) survey shows that among all security issues, neglect and wicked use of cloud computing is considered as the top security threat [1], in which attackers can develop vulnerabilities in clouds and utilize cloud system resources to deploy attacks. Layered technologies that create a durable security net or grid. Security is more effective when layered at each level of the stack and integrated into a common management framework.

II. RELATED WORK

The area of detect spiteful behavior has been well defined. The work by Duan et al. [2] focuses on the finding of compromised machinery that has been recruited to serve as spam zombies. Their move toward, SPOT, is based on consecutively scanning retiring messages while employ a arithmetical method (SPRT), to quickly determine whether a host has been compromised. BotHunter [3] detects compromised machines based on the fact that a thorough malware syndrome process has a number of distinct stages that allow correlate the intrusion alarms trigger by inbound traffic with resulting outgoing message pattern.

III. NICE MODEL

A. Threat Model

We assume that an attacker can be located either outer or within of the virtual networking system. The attacker's first goal is to abuse susceptible VMs and compromise them as zombies. Our security model focuses on virtual-network-based attack finding and reconfiguration solution to develop the resiliency to zombie investigation. Our work does not involve host-based IDS and does not deal with how to switch encrypted traffic for attack detections. Our upcoming solution can be organized in an Infrastructure-as-a-Service cloud networking model, and we assume that the Cloud Service Provider (CSP) is compassionate.

B. Attack Graph Model

An attack graph is a model tool to demonstrate all possible multistage, multi host attack paths that are crucial to understand threats and then to choose proper countermeasures [8]. In an attack graph, each node represents either precondition or consequence of an exploit. The events are not essentially an active attack because normal protocol interactions can also be used for attack. Assault graph is helpful in identifying latent threats, feasible attack, and identified vulnerabilities in a cloud system.

Definition 1 (SAG). An SAG is a tuple SAG ¼ ðV; EÞ, where

1. V ¼ NC [ND [NR denotes a set of vertices that include three types namely conjunction node NC to represent exploit, disjunction node ND to denote result of exploit, and root node NR for showing initial step of an attack scenario.
2. E ¼ E_{pre} [Epost denotes the set of directed edges. An edge e 2 E_{pre} _ ND _ NC represents that ND must be satisfied to achieve NC. An edge e 2 E_{post} _ NC _ ND means that the consequence shown by ND can be obtained if NC is satisfied.

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 2 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 (VC; VD) in
   SAG using map (a) function,
3. E is a set of directed edges representing correlation between two alerts
4. P is set of paths in ACG.

```
Algorithm 1.Alert_Correlation
Require: Alert $a_{c,}$ SAG, ACG
if ($a_c$ is a new alert) then
    Create node ac in ACG
    $n_1v_c \leftarrow$€ map ($a_c$)
    for all $n_2$€ parent ($n_1$)do
        Create edge ($n_{2,}$alert,$a_c$)
        for all $S_i$ containing a do
            if a is the last element in $s_i$ then
                append $a_c$ to $S_i$
            else
                Create path $S_{i+1}$= {subset ($S_{i,}$ a), $a_c$}
            end if
        end for
        add $a_c$ to $n_{i\ .}$alert
    end for
end if
return S
```

Definition 3 (VM State) Based on the information gathered from the network controller, VM states can be distinct as following:

1. Stable. There does not exist any known vulnerability on the VM.
2. Vulnerable. Presence of one or more vulnerabilities on a VM, which remains idle.
3. Exploited. At least one vulnerability has been exploited and the VM is compromised.
4. Zombie. VM is under control of attacker.

IV. NICE SYSTEM DESIGN

The proposed NICE framework is illustrated in Fig. 1. It shows the NICE skeleton within one cloud server cluster. Major components in this skeleton are distributed and light- weighted NICE-A on each physical cloud server, a network organizer, a VM profiling server, and an attack analyzer. The later three components are located in a centralized control center connected to software switches on each cloud server (i.e., virtual switches built on one or multiple Linux software bridge). NICE-A is a software mediator implement in each cloud server connected to the control center through a dedicated and isolated protected channel, which is divided from the normal data packets using Open Flow tunneling or VLAN approach. The network organizer is answerable for deploying attack countermeasures based on decisions made by the attack analyzer.

A. System Components

The NICE-A is a Network-based Intrusion Detection System (NIDS) agent installed in either Dom0 or DomU in each cloud server. It scans the traffic going during Linux bridges that control all the traffic among VMs and in/out from the physical cloud servers. In our research, sniff is used to implement NICE-A in Dom0. It will sniff a mirror port on each virtual bridge in the Open v Switch (OVS).

B. VM Profilling

Virtual machines in the cloud can be profiled to get precise information about their state, service organization, open ports, and so on. One major factor that counts toward a VM profile is its connectivity with other VMs. Any VM that is associated to more number of machines is more crucial than the one connected to fewer VMs because the effect of compromise of a highly connected VM can cause more damage.

C. Attack Analyzer

The main functions of NICE system are performed by attack analyzer, which includes procedures such as attack graph construction and update, alert correlation, and countermeasure selection. The process of construct and utilize the SAG consists of three phases: Information gathering, attack graph creation, and possible exploit path scrutiny. With this information, attack path can be modeled using SAG. Every node in the attack model represents an exploit by the attacker. Every path from an initial node to a goal node represents a successful attack.

D. Network Controller

The network controller is a key module to support the programmable networking capability to realize the virtual network reconfiguration aspect based on Open-Flow protocol [13]. In NICE, within each cloud server there is a software switch, for example, Open v Switch (OVS), which is used as the border switch for VMs to handle traffic in & out from VMs. The network manager is dependable for collect network information of present Open Flow network and provides input to the attack analyzer to construct attack graphs.

V. MITIGATION AND COUNTERMEASURES

In this level, we present the methods for choosing the solution for a given attack scenario. The countermeasure serves the reason of

1) Defending the target VMs from being compromised; and

2) Making attack behavior stand important so that the attackers' actions can be identified.

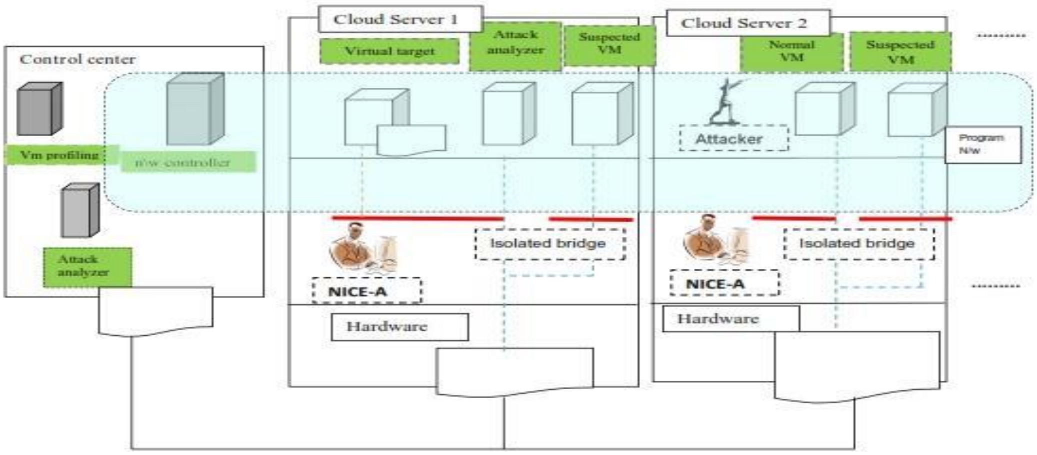


Fig. 1. NICE architecture within one cloud server cluster

A. Mitigation Strategies

Subsection, NICE is able to construct the improvement strategies in response to detected alert. First, we define the term countermeasure pool as follows:
Definition 4

A countermeasure pool CM = (cm1; cm2; cmn) is a set of countermeasures. Where Cost is the unit that describes the fixed cost required to apply the countermeasure in terms of property and operational difficulty, and it is defined in a range from 1 to 5, and high metric means upper cost; Intrusiveness is the unconstructive effect that a countermeasure bring to the examination Level Agreement (SLA) and its value ranges from the least intrusive (1) to the most intrusive (5), and the importance of intrusiveness is 0 if the countermeasure has no impacts on the SLA;Condition is the requirement for the equivalent countermeasure; Efficiency is the percentage of probability changes of the join, for which this countermeasure is applied.

B. Countermeasure Selection

Algorithm 2 presents how to select the best countermeasure for a given attack situation. Input to the algorithm is an alert, attack chart G, and a pool of countermeasures CM. The algorithm starts by select the join v Alert that correspond to the alert generated by a NICE-A. The countermeasure which when functional on a node gives the least charge of ROI, is regarded as the optimal countermeasure. At last, SAG and ACG are also efficient before terminating the algorithm.

Algorithm 2. Countermeasure Selection

```
Require:Alert,G (E, V), CM
Let vAlert= Source node of the Alert
if Distance to _Target (valert)>threshold then
    Update _ACG
    return
end if
Let T =Descendant (valert)UvAlert
Set Pr (valert)=1
Calculate_Risk_Prob (T)
Let benefit [|Tj|, |CM|] =0
for each t € T do
    for each cm € CM do
        if cm.condition (t) then
            Pr (t) =Pr (t)*(1 _ cm, effectiveness)
            Calculate_Risk_Prob (Descendant (t))
            benefit [t, cm] =Pr (target-node)
        end if
    end for
end for
Let ROI [|T|, |CM|]=0
for each t € T do
    for each cm € CM do
        ROI [t, cm] = (Benefit [t,cm])/(cost.cm+intrusiveness.cm)
    end for
end for
UpdateSAG and Update _ACG
return SelectOptimal CM (ROI)
```

VI. PERFORMANCE EVOLUTION

In this segment we present the performance evaluation of NICE. Our evaluation is conducted in two directions: the security performance and the system computing and network reconfiguration transparency due to introduced security mechanism.

6.1 Security Performance Analysis

To demonstrate the security performance of NICE, we created a virtual network testing surroundings consisting of all the presented components of NICE.

Definition 5 (VM SI). VSI for a virtual machine k is defined as V SIk = (Vk + Ek) =2, where

- Vk is vulnerability score for VM k.
- Ek is exploitability gain for VM k.

Apart from calculating the benefit measurements, we also present the estimation based on Return of Investment (ROI) using and represent a comprehensive evaluation.

A.Attack Graph and Alert Correlation

The attack chart can be generated by utilizing network topology and the exposure information, and it is shown in Figure 3. As the attack progresses, the cloud system generates the various alerts that can be related to the nodes in the attack model. Developing a new attack graph requires knowledge of network configuration, running services and their liability information. This information is provided to the attack graph creator as the input.

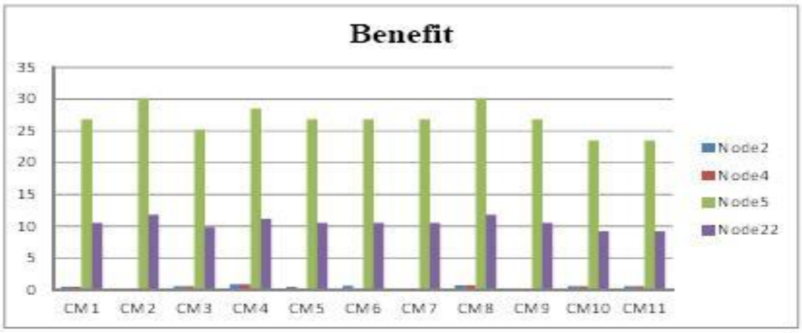
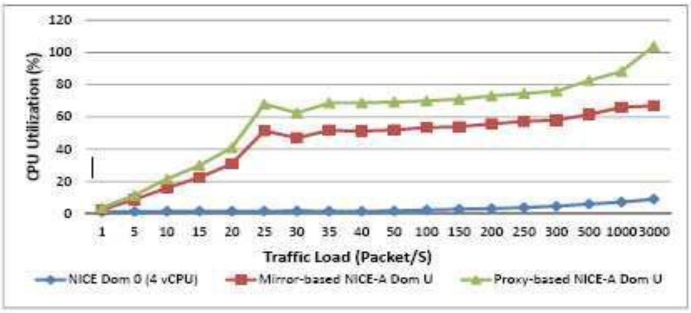


Fig 2.Benefit evaluation chart

B. NICE System Performance

We estimate system performance to provide supervision on how much traffic NICE can hold for one cloud server and use the assessment metric to scale up to a large cloud system. In real cloud system, traffic plan is needed to run NICE but it's not available in the current system and as a result space limitation exists. We will investigate the research involving several cloud clusters in the upcoming.



VII. CONCLUSION AND FUTURE WORK

This paper describes about Network intrusion detection, which is planned to identify and shared attacks in the cloud virtual networking surroundings. It utilizes the attack graph model to perform attack detection and prediction. The planned solution investigate how to use the programmability of software switch based solutions to improve the detection exactness and defeat victim exploitation phases of collaborative attack. NICE shows that the planned solution can appreciably reduce the risk of the cloud system from being demoralized and harmed by internal and external attackers. It only investigates the network IDS approach to counter zombie explorative attacks. In order to progress the detection correctness, host-based IDS solutions are needed to be incorporated and to cover the entire spectrum of IDS in the cloud process. This should be investigating further work. With reference to above explanation we do as follows: explore the scalability of the planned NICE solution by investigate the decentralized network organize and attack examination model based on recent technology.

REFERENCES

[1] Chun-Jen Jung, Pankaj Khatkar, Tianyi Xing, Jeongkeun Lee, Dijiang Huang,NICE-Network Intrusion Detection and Countermeasure Selection in Virtual Network System, IEEE Transactions on Dependable and Secure Computing, Issue Vol 10 No 4 2013.

[2] Joshi, B. Vijayan, A.S. ; Joshi, B.K. Securing cloud computing environment against DDoS attacks Computer Communication and Informatics (ICCCI), 2012

[3] Takabi, H. Joshi, J.B.D. ; Gail-Joon Ahn, Security and Privacy Challenges in Cloud Computing Environments Security & Privacy, IEEE(Volume:8 ,Issue: 6) 2010.

[4] Duan, Z. Peng Chen ; Sanchez, F. ; Yingfei Dong ; Stephenson, M. ; Barker, J.M. Detecting spam zombies by monitoring outgoing messages Dependable and Secure Computing, IEEE Transactions on (Volume:9 , Issue: 2)2012.

[5] Ammann, Paul; Wijesekera, Duminda; Kaushik, Saket Scalable, graph-based network vulnerability analysis 2002.

[6] Sebastian Roschke, Feng Cheng, Christoph Meinel A New Alert Correlationalgorithm Based On Attack Graph,CISIS 2011.

[7] K. Kwon, S. Ahn, and J. Chung, “Network Security Management Using ARP Spoofing,” Proc. Int’l Conf. Computational Science and Its Applications (ICCSA ’04), pp. 142-149, 2004.

[8] 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 Comm., vol. 53, no. 4, pp. 50-58, Apr. 2010.

[9] G.gu p. Porras, V.Yegneswaran,M.Fong,and `W.Lee,”BotHunter:Detecting Malware Infection through IDS-driven Dialog Correlation,”Proc.16th USENIX Security Symp.(SS,07),pp,12:1-12:16,Aug, 2007.

[10] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, “NOX: Towards an Operating System for Networks,” SIGCOMM Computer Comm. Rev., vol. 38, no. 3, pp. 105-110, July 2008.

[11] N. Poolsappasit, R. Dewri, and I. Ray, “Dynamic Security Risk Management Using Bayesian Attack Graphs,” IEEE Trans. Dependable and Secure Computing, vol. 9, no. 1, pp. 61-74, Feb. 2012.

[12] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J.M.Wing, “Automated Generation and analysis of Attack Graphs,” Proc. IEEE Symp. Security and Privacy, pp. 273-284, 2002.

[13] P. Mell, K. Scarfone, and S. Romanosky, “Common Vulnerability Scoring System (CVSS),” http://www.first.org/cvss/cvss-guide. html, May 2010.

ABOUT THE AUTHORS

B.Manikandan received the B.E (CSE) from Indus College of Engineering in 2012 and M.Tech in Information Technology in VSB Engineering College in 2012 under Anna University, Chennai. His Area of Interest includes Network Security, Cloud Computing.

Mr.Balaji Subramani has received the M.Tech (IT) in K.S.Rangasamy College of Technology. He is currently working as an assistant professor in V S B Engineering College. He has total number of 19 publications, 3 papers in International Journals, 7 papers in International conferences and 8 papers in national seminars/conferences and participated in various symposiums and workshops held at different places. His area of interest includes Word sense disambiguation, web mining, Information retrieval and social network analysis.