Virtual Network Connection Technique to Control Device Remotely and Securely

Ajit Patil
Dept. of Computer Engg
AISSMS COE
Pune, India.

Aishwarya Laturkar
Dept. of Computer Engg
AISSMS COE
Pune, India

Priya Tathawade
Dept. of Computer Engg
AISSMS COE
Pune, India.

Rutuja Takale
Dept. of Computer Engg
AISSMS COE
Pune, India

S. V. Athawale
Dept. of Computer Engg
AISSMS COE
Pune, India.

ABSTRACT
Cloud security has been an important area of research nowadays. The vulnerabilities of cloud system can be easily explored by any attacker and compromise these virtual machines to deploy DDos i.e. Distributed denial of service attack which involves multi step exploitation, compromising virtual machines as zombies, low frequency vulnerability scanning and lastly DDos attack through the zombies. In Infrastructure-as-a-service (Iaas) clouds, detection of zombies is extremely difficult. Hence to prevent virtual machines to be compromised as zombies, a new system was proposed as Network Intrusion detection and Countermeasure Selection, Abbreviated as NICE. It uses countermeasures based on reconfigurable virtual network and uses attack graphs. This paper provides survey of techniques of NICE and countermeasures taken to prevent the machines from being compromised as zombies.

Keywords
Network security, Intrusion detection, Countermeasure selection, Cloud computing, Attack graph, Zombie Detection.

1. INTRODUCTION
The Cloud Security Alliance survey shows that among all security problems, abuse and nefarious use of cloud computing is one of the top security threat, in which attackers can make use of vulnerabilities and system resources in clouds to make attacks. Cloud users usually have the rights to control software installed on their managed VMs, may not work efficiently and can breach the Service Level Agreement. Furthermore, cloud users can install vulnerable software on their VMs, which contributes to loopholes in cloud security. The challenge is to set up an effective vulnerability detection and response system, which will accurately identify attacks and minimize the impact of security breach to cloud users. In a cloud environment infrastructure is shared by millions of users and its abuse and illegal use benefits attackers to use vulnerabilities to deploy attacks more efficiently. Cloud users share computing resources so such attacks are very effective. Resources like interconnected using same switch, file systems and sharing same data storage, even with potential attackers. Virtual Machines have similar setup in the cloud, e.g. VM OS, virtualization techniques, installed vulnerable software etc. invites attackers to compromise multiple VMs.

2. LITERATURE SURVEY
In 2010 H. Takabi, G. Ahn, and J.B. Joshi the challenges of security and privacy in cloud computing environment [1]. Cloud computing has generated important interest in both the academy and industry, but it is yet an evolving paradigm. It aims to integrate the economic utility model with the evolutionary development of existing approaches and computing technologies, which includes distributed applications, services and information infrastructures consisting of pools of networks, computers and storage resources. Some consider a cloud as a new technical revolution, while others see it a natural evolution of technology, economy and culture. Cloud computing reduce costs through optimization and increased operating and economic efficiencies. Cloud computing could enhance collaboration, agility, and scale, thus allowing a global computing model over the Internet infrastructure. This computing paradigm could become a big failure without security and privacy solutions designed for clouds. This article explains the issues of security and privacy challenges in clouds. In this article various approaches to address these challenges and explore the future work needed to provide a reliable cloud computing environment have discussed.

In 2012 B. Joshi, A. Vijayan, and B. Joshi proposed a method for Securing cloud computing environment against DDoS attacks.This paper Cloud Computing is the newly emerged technology of Distributed Computing System. Cloud Computing provide services to its clients into three layers namely, Software as a service, Infrastructure as a service and Platform as a service using web services. It provides service facilities to consumers on demand. Attacker can attack by Saas , Iaas and Paas very easily[2]. Since the resources are gathered at one place in data centers in cloud computing, the DDOS attacks like HTTP & XML in cloud environment are dangerous & can harm all consumers at the same time. These attacks can be detected and resolved by a proposed methodology. The different types of vulnerabilities are detected in proposed system. The SOAP request form the communication between the service provider and client. The SOAP request is send to the cloud through the Service Oriented Trace back Architecture. In this architecture service oriented trace back mark is present, containing proxy within it. To identify the real client the proxy marks the incoming packets with source message identification. Then the SOAP message is travelled via XDetector . The XDetectors used to monitor and filter the DDoS attacks like HTTP and XML DDoS attacks and finally the filtered real client message is transferred to the cloud service provider and the corresponding services are given to the client in secured method. The paper [3] give attention on the detection of the compromised machines called spam zombies that are used for

sending spam messages. A critical economic incentive is there for the controllers of the compromised machines to recruit these machines. Generally many compromised machines are involved in spamming. Networks of compromised machines involved in spamming called as spamming botnets. The aggregate global characteristics of spamming botnets like size of bonnets and the spamming patterns of botnets, formed on the sampled spam messages received at a large email service provider have studied through number of researches.

In 2007 G. Gu, P. Porras, M. Fong, V. Yegneswaran and W. Lee used IDS-driven Dialog Correlation and proposed a method for Detecting Malware Infection . The paper [4] focus on a new kind of network perimeter monitoring strategy, which recognizes the infection and coordination dialog that occurs during a successful malware infection .The two- way communication flows among internal assets and external entities can be tracked by an application called BotHunter . To match a state-based infection sequence model BotHunter developes an evidence trail of data interchange. BotHunter contains correlation engine that is controled by three malware-focused packet sensors, each detecting stages of the malware infection process like exploit usage, inbound scanning, outbound bot, egg downloading, coordination dialog and outbound attack propagation. Dialog trail of inbound intrusion alarms and outbound communication patterns are bound together by the BotHunter correlator that are highly indicative of successful local host infection. When a sequence of evidence is got to match BotHunter's infection dialog model, report is generated to capture relevant events and event sources that were used during the infection process. We consult this strategy as dialog-based correlation, and against it to other intrusion detection and alert correlation methods. Using BotHunter they presented experimental results in both live and virtual testing environments. BotHunter can be used both for operational use and to help stimulate research in understanding the life cycle of malware infections.

In 2012 Z. Duan, P. Chen, F. Sanchez, Y. Dong, M. Stephenson, and J. Barker proposed a method for Detection of Spam Zombies by Monitoring Outgoing Messages[5] In this paper we have seen Compromised machines are one of the key security threats on the Internet; they are often used to dispatch various security attacks such as junk E-Mail and spreading malware, find the theft and DDoS. Given that spamming allows a key economic incentive for attackers to recruit the large number of agreement machines, we are concentrating on the detection of the agreement machines in a network that are involved in the spamming actions, generally known as spam zombies. An effective spam zombie detection system named SPOT is developed by watching outgoing messages of a network. SPOT is developed based on a powerful statistical tool named Sequential Probability Ratio Test, which has limited false negative error and limited false positive rates. The evaluation study based on email trace collected in a very large campus network show that SPOT is an effective and efficient system in automatically detecting agreement machines in a network. Inclusion , the performance of SPOT is compared with two other spam zombie detection algorithms depend on the number and percentage of spam messages originated or forwarded by internal machines, and show that SPOT performs better than these two detection algorithms.

In 2005 X. Ou, A.W. Appel, S. Govindavajhala presented a logic based security analyzer[6] .To determine network's security impact software vulnerabilities, interactions between multiple network elements must be considered.Two important things for analysis of vulnerability. One, the model used in it should able to automatically combine formal specifications of vulnerability from the bug-reporting community. Second, the analysis must be capable to scale to networks of thousands of systems.This can be achieved using MulVAL(end-to-end framework).It is reasoning system performs multihost, multistage analysis of vulnerability. Datalog used as the modeling language by MulVAL for analysis elements (Configuration description and bug specification, privilege model, Operating-system permission and reasoning rules etc.).We can take benefits of existing vulnerability-database and scanning tools by presenting their output in Datalog and giving it to MulVAL .After information is collected, analysis may complete in some seconds.

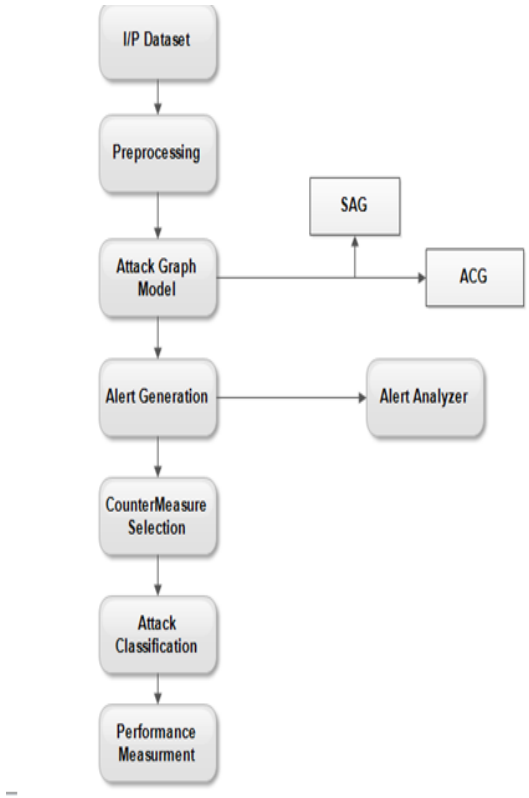

Fig 1: System Workflow

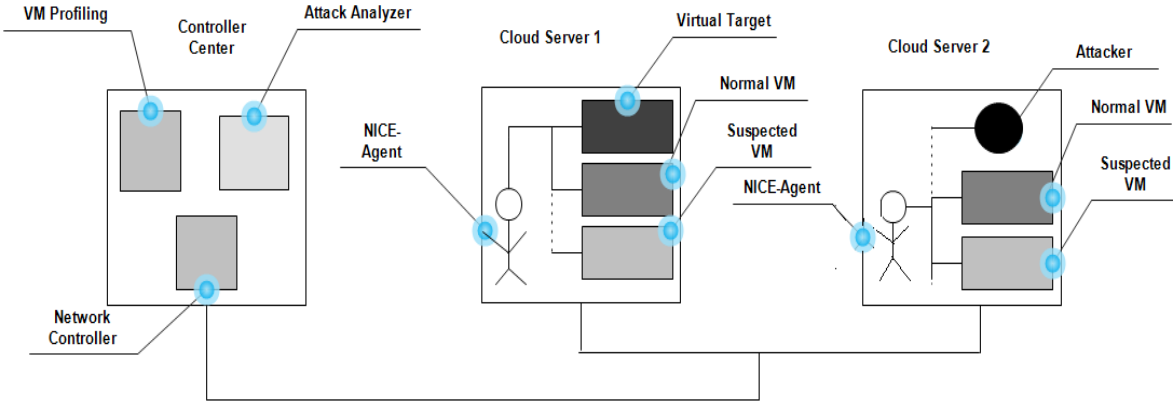

Fig 2: NICE System Architecture

3. PROPOSED SYSTEM

1. Network Intrusion detection and Countermeasure sElection in virtual network systems is proposed to establish a multi stage and in-depth intrusion detection framework.
2. For better analysis of attack detection, NICE incorporates multi stage attack graph analytical procedures into the intrusion detection processes.
3. The design of proposed framework has not been designed intending to improve any of the existing intrusion detection algorithms; indeed, this framework makes use of a programmable and reconfigurable virtual networking approach to detect and counterattack the attempts to compromise user managed VMs, thus preventing Virtual Machines to be compromise as zombies.
4. The framework incorporates a lightweight mirroring-based agent (NICE-A) for network intrusion detection on each cloud server to capture the cloud traffic and analyse it. The NICE-A inspects the virtual system vulnerabilities within a cloud server periodically to establish Scenario Attack Graph (SAGs), and then NICE will decide whether or not to put a VM in network inspection state based on how much severe identified vulnerability towards the collaborative attack goals.
5. NICE constructs a mirroring-based traffic capturing framework to reduce the interference on users' traffic compared to traditional bump-in-the-wire (i.e., proxy-based) IDS/IPS by using software switching techniques.
6. NICE allows the cloud to establish inspection and quarantine modes for suspicious virtual machines according to their current vulnerability state by using scenario attack graph.
7. Based on the cumulative behavior of VMs in the SAG, NICE-A can decide most appropriate countermeasures, for instance Deep packet Inspection or changing the IP Address on the suspected VMs to avoid the attack or to alleviate the effects of the attack. Using this framework, NICE does not need to block traffic flows of the suspected VM in its early attack stage.

3.1 NICE System Architecture

The NICE system is formulated to work in a cloud virtual networking environment. It incorporates a cluster of cloud servers which are interconnected. Latest virtualization solutions are deployed on cloud servers assumed. The classification of the virtual environments can be as Privilege Domains, for example the dom0 of XEN Servers and the host domain of KVM, and Unprivileged Domains, for example VMs. Cloud servers are connected through reconfigurable Privilege Domains. In this work, we refer Open Flow Switches and Open V Switches and their controllers as to the Software Defined Network (SDN). The deployment of security mechanism focuses on assuring a non- intrusive approach to avoid attackers from exploring VMs that can be compromised and use them as a stepping stone for deploying further attacks.

3.2 System Components

In this section, we explain each component of NICE.

3.2.1 Nice-A

The NICE-A is a Network-based Intrusion Detection and Prevention System (NIDS) agent installed in each cloud server of the virtual network. It monitors the traffic going in and out through the bridges that control all the flow of traffic among VMs and in/out from the physical servers. NICE-A will sniff a mirroring port on each virtual bridge in the Open v Switch. Each bridge creates an aloof subnet in the virtual network and connects to all related Virtual Machines. The traffic gathered from the Virtual Machines on the mirrored Software Bridge will be copied like a mirror image to a specific port on a specific bridge using SPAN ERSPAN or RSPAN methods. It's more operative to scan the traffic in cloud server since all traffic in the cloud server pass through it; however our design is not dependent of the VM installed. The NICE architecture aims to minimize the false alarm rate.

We must note that the alert detection quality of NICE-A is depend upon the implementation of NICE-A which uses Snort. Wed on focus on the detection accuracy of Snort in this article. So, the individual alert detection's false alarm rate does not change.

However, the false alarm rate could be minimized through our architecture design. We will see more about this issue in the later section.

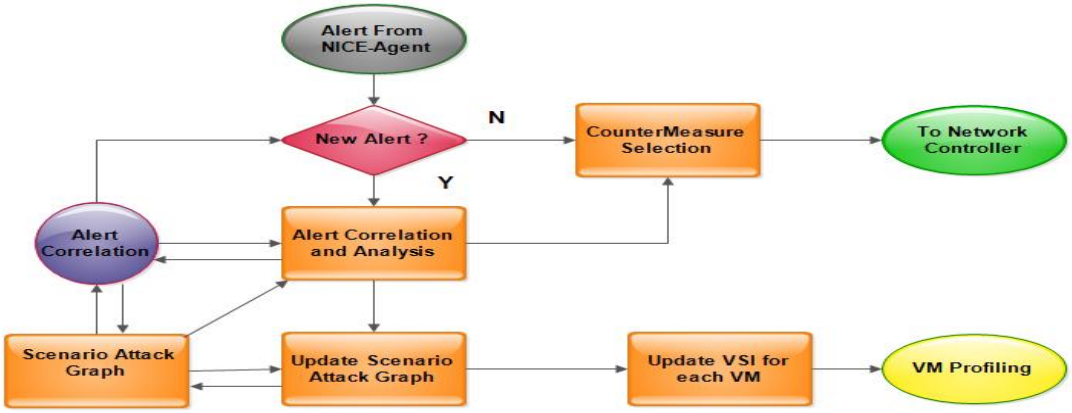

Fig 3: Workflow of Attack Analyzer

3.2.2 VM-Profiling

Virtual machines in the cloud needs to be profiled to get accurate information about their current state, current open ports, running services etc. Connectivity with other Virtual Machines is one of the major factor that counts towards a VM profile and also required is the knowledge of services running on a VM so as to verify the genuiness of alerts relating to that Virtual Machines. An Intruder can use port scanning program to perform an intense exploration of the network to seek for open ports on any Virtual Machine. Therefore information about any open ports on a Virtual Machines and the previous history of opened ports plays an important role in providing information like how vulnerable the Virtual Machine is. All these factors about a virtual machine is maintained in the profile. A database maintains all these Virtual Machine profiles and contain precise information about vulnerabilities, traffic and alert in the VM profiles section. These profiles can be updated as per the new alert generated. Therefore the network programmable.

3.2.3 Attack Analyzer

The Attack Analyzer performs the major function, which consist of attack graph creation and update, alert correlation and countermeasure selection. The construction and use of the scenario Attack (SA) consists of three phase. Firstly information gathering, second attack graph construction, and third is potential exploit path analysis. With this information, attack paths can be modeled with the help of SAG. Each node in the attack graph represents an exploit by the attacker. Each path from an initial node to a goal node represent a successful attack. The Attack Analyzer handles alert correlation and analysis operations. It has two major functions:

1) constructs Alert Correlation Graph (ACG),
2) Gives threat information and appropriate countermeasures to network controller for virtual network reconfiguration .

Algorithm 1 Alert Correlation

Require: alert ac, SAG, ACG

1. if (ac is a new alert) then
2. create node ac in ACG
3. n1 ← vc∈map(ac)
4. for all n2 ∈parent(n1) do
5. create edge (n2.alert, ac)
6. for all Si containing a do
7. if a is the last element in Si then
8. append ac to Si
9. else
10. create path Si+1 = {subset(Si, a), ac}
11. end if
12. end for
13. add ac to n1.alert
14. end for
15. end if
16. return S

3.2.4 Network Controller

It is important component in NICE system and it support programmable networking capability to achieve the network reconfiguration. Each cloud server has software switches that is open v switch and open flow switch, control function for those switches are combined in this controller. Open v switch handles traffic coming and out going from virtual machines and another switch i.e open flow switch handles the communication between cloud servers. Both switches also allows system to set security and filtering rules. Another function of network controller is that it collects the information of current network and this information send to attack analyzer i.e. this is input to attack analyzer then attack analyzer update and construction of attack graph. It is responsible for applying countermeasure which is selected by NICE-A and executed by network controller.

3.3 Countermeasure selection

This algorithm shows that how to select the optimal countermeasure for a given attack. Input to the algorithm is an alert, attack graph G, and a type of countermeasures CM. The algorithm starts by selecting the node vAlert that corresponds to the alert produce by a NICE-A. Before selecting the countermeasure, we count the distance of vAlert to the target node. If the distance is greater than a starting value, we do not perform countermeasure selection but update the ACG to keep track of alerts in the system. For the source node vAlert, all

the reachable nodes (including the source node) are collected into a set T (line 6).

Because the alert is produced only after the attacker has performed the action, we set the probability of vAlert to 1 and calculate the new risk probabilities for all of its child (downstream) nodes in the set T (line 7 & 8).

Now for all t ∈ T the applicable countermeasures in CM are selected and new probabilities are calculated according to the influence of the selected countermeasures (line 13 &14). The change in probability of target node gives the advantage for the applied countermeasure using. In the next double for-loop, we compute the Return of Investment for each advantage of the applied countermeasure based on. The countermeasure which when applied on a node gives the least value of Return of Investment, is regarded as the optimal countermeasure. Finally, SAG and ACG are also updated before finishing the algorithm. The complexity of this Algorithm is O(|V | × |CM|) where |V | is the number of vulnerabilities and |CM| is to represent the number of countermeasures.

Algorithm 2 Countermeasure Selection
Require: Alert,G(E, V), CM

1. Let vAlert= Source node of the Alert
2. if Distance to Target (vAlert) > threshold then
3. Update ACG
4. return
5. end if
6. Let T = Descendant(vAlert) ∪vAlert
7. Set Pr(vAlert) = 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). (7)
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

4. ADVANTAGES

1. It can determines that attack occurred or not with high accuracy
2. It provides direct control over system.
3. It identifies the attack and gives the alert to that virtual machine's and also select countermeasure.
4. It consumes less computational overhead than proxy based intrusion detection solution.
5. It allows system to construct dynamic reconfigurable IDS/IPS.
6. It captures and filters traffic without affecting the user application and network service.

5. CONCLUSION

This paper provides a Review on Network Intrusion detection and Countermeasure selection and the related security concepts. NICE, which is proposed to detect and Prevent, mitigate collaborative attacks in the cloud virtual networking environment. NICE uses the attack graph model to conduct attack analyzer detection and prediction. The proposed solution investigates how to use the dynamically of software switches based solutions to improve the detection accuracy and defeat victim exploitation phases of combining attacks. The system performance evaluation indicates the feasibility of NICE and shows that the proposed optimum solution can significantly reduce the risk of the cloud system from being exploited and misused by internal and external attackers.

6. ACKNOWLEDGMENTS

We feel great pleasure in submitting this Paper on "NICE: Network Intrusion Detection and Countermeasure Selection in Virtual Network System". We wish to express true sense of gratitude towards Prof. M .A .Pund we also wish to thank my teacher Mr. S V Athawale who at very discrete step in preparation of this Paper contributed his valuable guidance and help to solve every problem that arose. Also, most likely we would like to express my sincere gratitude towards my family for always being there when we needed them the most. With all respect and gratitude, we owe my all success to the writers of reference papers that are referred by me in completion of this paper work activity which will be useful in presenting our survey paper.

7. REFERENCES

[1] Cloud Security Alliance, "Top threats tocloudcomputing1.0,"https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf,March2010.

[2] A. Vijayanand B. Joshi," Securing Cloud Computing Environment against DDoS Attacks," Proceedings IEEE International Conference Computer Comm. and Informatics,Jan.2012.

[3] J.B.Joshi, H.Takabiand G. Ahn, " Security and Privacy Challenges in Cloud Computing Environments, "IEEE SecurityandPrivacy,vol.8, no.6, pp.Dec.2010.

[4] Z.Duan, P.Chen ,F.Sanchez, Y.Dong ,M.Stephenson and J.Barker, "Detecting Spam Zombies by Monitoring Outgoing Messages," IEEE Trans. Dependable and Secure Computing,Apr.2012.

[5] G. Gu , P. Porras, V.Yegneswaran, M.Fong , and W.Lee, "Bot Hunter :Detecting Malware Infection through IDS-driven Dialog Correlation," Proc.16thUSENIXSecurity Symp. (SS'07),pp.12:1-12:16,Aug.2007.

[6] X. Ou , S.Govindavajhala, and A.W. Appel, "Mul VAL:A Logic-Based Network Security Analyzer," Proc.14th USENIXSecuritySymp.,pp.113-128,2005.