International Journal of Science Engineering and Advance Technology, IJSEAT, Vol 2, Issue 12, December - 2014 | ISSN 2321-6905

International Journal of Science Engineering and Advance Technology

A Novel Attack Graph Approach For Attack Detection And Prevention Using Nice

K.Suganya[1], K.T.V Subbarao[2]
1,2 Department of Computer Science And Engineering
1,2 Akula Sree Ramulu institute of Engineering and Technology, prathipadu,Tadepalligudem, A.P, India
[1]ksuganyaprabu@gmail.com [2]ogidi@rediffmail.com

Abstract:
DDoS attacks usually take up early stage actions such as multi-step exploitation, low frequency vulnerability scanning and compromising identified vulnerable virtual machines as zombies and lastly DDoS attacks through the compromised zombies. Within the cloud system particularly the Infrastructure-as-a-Service (IaaS) clouds the detection of zombie study attacks is extremely hard. This is for the reason that cloud users may set up vulnerable applications on their virtual machines. To prevent susceptible virtual machines from being compromised in the cloud we propose a multi-phase distributed vulnerability detection, measurement and countermeasure selection method called NICE which is built on attack graph based on analytical models and reconfigurable virtual network-based countermeasures.

Keywords: Network Security, Cloud Computing, Intrusion Detection, Attack Graph, Zombie Detection.

Introduction:
We recommend NICE (Network Intrusion detection and Countermeasure sElection in virtual network systems) to set up a defence-in-depth interruption detection framework. For better attack detection NICE slots in attack graph procedures into the intrusion discovery procedure. We must note that the plan of NICE does not mean to get better any of the existing intrusion detection algorithms indeed NICE utilizes a reconfigurable practical networking approach to notice and answer the attempts to compromise VMs, thus stopping zombie VMs. In general NICE contains organize a lightweight mirroring-based network intrusion detection agent (NICE-A) on each cloud server to imprison and analyze cloud traffic. A NICE-A episodically scrutinizes the virtual system vulnerabilities within a cloud server to create Scenario Attack Graph (SAGs) and then based on the brutality of identified vulnerability towards the combined attack goals. NICE will make a decision whether or not to put a VM in network inspection state. Once a VM enters inspection state Deep Packet Inspection (DPI) is applied and/or virtual network reconfigurations can be organized to the inspecting VM to make the potential attack behaviours famous.

Related Work:
Many attack graphs based alert correlation techniques have been proposed recently. L. Wang *et al*. planned an in-memory structure called *queue graph* (QG) to draw alerts matching each develop in the attack graph. However the implicit correlations in this design make it hard to use the correlated alerts in the graph for analysis of similar attack scenarios. Roschke *et al*. proposed a customized attack-graph-based correlation algorithm to produce open correlations only by matching alerts to exact exploitation nodes in the attack graph with multiple mapping functions and planned an alert dependencies graph (DG) to group related alerts with manifold correlation criteria. Each path in DG represents a subset of alerts that might be part of an attack scenario. Though their algorithm involved all pairs shortest path searching and sorting in DG which devours substantial computing power.

Existing Method:
In a cloud system where the infrastructure is shared by potentially millions of users, abuse and immoral use of the shared infrastructure benefits attackers to exploit vulnerabilities of the cloud and use its reserve to organize attacks in more capable ways. Such attacks are more expensive in the cloud environment since cloud users usually share computing resources

e.g., being connected through the same switch sharing with the same data storage and file systems even with potential attackers.

Disadvantages:

No detection and prevention framework in a virtual networking environment. It doesn't maintain accuracy in the attack detection from attackers.

Proposed Method:

For better attack detection NICE incorporate attack graph analytical measures into the intrusion detection processes. We must note that the plan of NICE does not intend to progress any of the existing intrusion detection algorithms certainly NICE employs a reconfigurable virtual networking approach to detect and be against the attempts to compromise VMs thus preventing zombie VMs.

Advantages:

NICE is a latest multi-phase distributed network intrusion detection and prevention framework in a virtual networking environment that limits and examines apprehensive cloud traffic without disrupting users' applications and cloud services. NICE can progress the attack detection possibility and expand the resiliency to VM utilization attack without interrupting existing normal cloud services. NICE optimizes the implementation on cloud servers to weaken resource consumption.

System Architecture:

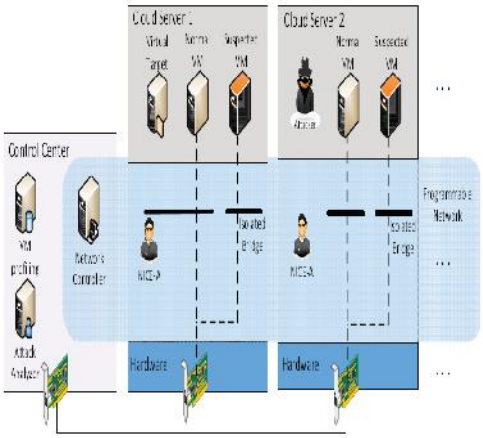


Fig. 1. NICE architecture within one cloud server cluster.

It illustrates the NICE framework within one cloud server cluster. Major components in this framework are distributed and light-weighted NICE-A on each physical cloud server, a network controller, a VM profiling server and an attack analyzer. The latter three components are located in a middle control centre connected to software switches on each cloud server i.e., virtual switches built on one or multiple Linux software bridges. NICEA is a software agent applied in each cloud server connected to the control centre through a dedicated and isolated secure channel which is removed from the normal data packets using Open Flow tunnelling or VLAN approaches. The network controller is accountable for deploying attack countermeasures based on decisions made by the attack analyzer.

Nice-A:

The NICE-A is a Network-based Intrusion Detection System (NIDS) agent installed in each cloud server. It examines the traffic going through the bridges that manage all the traffic among VMs and in/out from the physical cloud servers. It will inhale a mirroring port on each virtual bridge in the Open v Switch. Each bridge forms an inaccessible subnet in the virtual network and joins to all related VMs. The traffic generated from the VMs on the mirrored software bridge will be mirrored to a specific port on a specific bridge using SPAN, RSPAN or ERSPAN methods. It's more competent to scan the traffic in cloud server because all traffic in the cloud server desires go through it. However our design is autonomous to the installed VM. The false alarm rate could be condensed through our architecture design.

Vm Profiling:

Virtual machines in the cloud can be outlined to get accurate information about their state, services running, open ports, etc. One major factor that counts towards a VM profile is its connectivity with other VMs. Also required is the acquaintance of services running on a VM so as to validate the authenticity of alerts pertaining to that VM. An attacker can use port scanning program to carry out a strong assessment of the network to look for open ports on any VM. So information about any open ports on a VM and the history of opened ports plays a important role in influencing how vulnerable the VM is. All these factors mutually will form the VM profile. VM profiles are sustaining in a database and contain all-inclusive information about vulnerabilities, alert and traffic.

Attack Analyzer:

The main functions of NICE system are performed by attack analyzer which contains procedures such as attack graph construction and update, alert correlation and countermeasure selection. The procedure 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. The Attack Analyzer also holds alert correlation and analysis operations. This component has two major functions which construct Alert Correlation Graph (*ACG*), offers threat information and suitable countermeasures to network controller for virtual network reconfiguration. NICE attack graph is created based on the Cloud system information, Virtual network topology and configuration information, Vulnerability information.

Network Controller:

The network controller is a key constituent to hold up the programmable networking competence to comprehend the virtual network reconfiguration. In NICE we integrated the control functions for both OVS and OFS into the network controller that permits the cloud system to set security/filtering rules in an integrated and inclusive manner. The network controller is answerable for collecting network information of current Open Flow network and affords input to the attack analyzer to build attack graphs. In NICE the network control also consults with the attack analyzer for the flow access control by setting up the filtering rules on the corresponding OVS and OFS. Network controller is also responsible for pertaining the countermeasure from attack analyzer. Based on VM Security Index and severity of alert countermeasures are selected by NICE and carried out by the network controller.

Host Based Intrsuion Evalution

1) Log monitor

Monitoring the log file, once the log change, log monitor will send events to the log analyzer immediately.

Generally, we need to monitor three kinds of event logs: application log, security log and system log. We can add three XML nodes in the following configuration file.

The node “localfile” represents the local file when system initialization. The node “location” represents file path in the disk. The node “log_format” represents what type of the log. Log type includes event log, firewall log, SQL log.

When initialize the HIDS, it will automatically load the above log files that need to be monitored. When finished the initialization work, the HIDS will open a demon, and the demon will check every log files to find whether there is changes in the log file. If there really exits a change, then the demon will report to the log analyzer.

2) System resources monitor

Monitoring the use of system resources, and sends the status of the system resources utilization to the system resources analyzer at regular time.

3) Connector

The connector is responsible for receiving messages from log monitor and system resources monitor, and sending these messages to log analyzer and system resources analyzer.

4) Log analyzer

Receiving events from the log monitor, match with the rule base to determine whether there is invasion, if there is invasion occurrence, report to the active response unit.

5) System resources analyzer

Receiving events form the system resources monitor, to calculate whether the abnormal state of current resources use and thus to determine whether the status is invaded, if it find there is invasion, report to the active response unit.

6) Active response unit

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

7) Audit database

Recording the entire process of intrusion detection, and the attack situation, prepare for use when necessary.

Algorithm Used:

Algorithm 2 Countermeasure_Selection

Require: $Alert, G(E,V), CM$

Let v_{Alert} – Source node of the $Alert$
if Distance_to_Target(v_{Alert}) > $threshold$ **then**
  Update ACG
  return
end if
Let $T = Descendant(v_{Alert}) \cup v_{Alert}$
Set $Pr(v_{Alert}) - 1$
Calculate_Risk_Prob(T)
Let $benefit[|T|, |CM|] = \emptyset$
for each $t \in T$ **do**
  for each $cm \in CM$ **do**
    if $cm.condition(t)$ **then**
      $Pr(t) = Pr(t) * (1 \quad cm.effectiveness)$
      Calculate_Risk_Prob($Descendant(t)$)

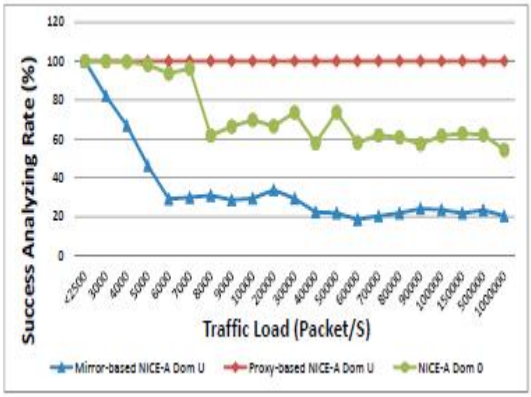
      $benefit[t, cm] = \Delta Pr(target\ \ node).$

    end if
  end for
end for
Let $ROI[|T|, |CM|] = \emptyset$
for each $t \in T$ **do**
  for each $cm \in CM$ **do**

$$ROI[t, cm] = \frac{benefit|t, cm|}{cost.cm + intrusiveness.cm}.$$

  end for
end for
Update SAG and Update ACG
return Select_Optimal_CM(ROI)

Experimental Results:



Result symbolizes the presentation of NICE-A in terms of percentage of productively analyzed packets i.e., the number of the analyzed packets separated by the total number of packets received. The higher this value is more packets this agent can hold. It can be observed from the result that IPS agent displays 100% performance because every packet confines by the IPS is cached in the detection agent buffer. Though 100% success analyzing rate of IPS is at the cost of the analyzing delay. For other two types of agents the detection agent does not hoard the captured packets and thus no delay is introduced. However they all practice packet drop when traffic load is enormous.

Conclusion:

NICE which is planned to notice and alleviate collaborative attacks in the cloud virtual networking environment. The proposed solution examines how to use the programmability of software switches based solutions to improvement the detection accurateness and trounce victim operation phases of collaborative attacks. The system routine measurement expresses the possibility of NICE and proves that the proposed solution can considerably condense the danger of the cloud system from being exploited and abused by internal and external attackers. NICE only looks into the network IDS approach to counter zombie explorative attacks. In order to get better detection accuracy host-based IDS solutions are needed to be included and to cover up the whole spectrum of IDS in the cloud system. NICE uses the attack graph model to behaviour attack detection and prediction.

References:

[1] Coud Sercurity Alliance, "Top threats to cloud computing v1.0," https://cloudsecurityalliance.org/topthreats/csathreats.v1.0.pdf, March 2010.

[2] 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.

[3] 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.

[4] H. Takabi, J. B. Joshi, and G. Ahn, "Security and privacy challenges in cloud computing environments," *IEEE Security & Privacy*, vol. 8, no. 6, pp. 24–31, Dec. 2010.

[5] "Open vSwitch project," http://openvswitch.org, May 2012.

[6] 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*, vol. 9, no. 2, pp. 198–210, Apr. 2012.

[7] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee, "BotHunter: detecting malware infection through IDS-driven dialog correlation," *Proc. of 16th USENIX Security Symp. (SS '07)*, pp. 12:1–12:16, Aug. 2007.

[8] G. Gu, J. Zhang, and W. Lee, "BotSniffer: detecting botnet command and control channels in network traffic," *Proc. of 15th Ann. Network and Distributed Sytem Security Symp. (NDSS '08)*, Feb. 2008.

[9] 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. [10] "NuSMV: A new symbolic model checker," http://afrodite.itc.it:1024/~nusmv. Aug. 2012.

[11] S. H. Ahmadinejad, S. Jalili, and M. Abadi, "A hybrid model for correlating alerts of known and unknown attack scenarios and updating attack graphs," *Computer Networks*, vol. 55, no. 9, pp. 2221–2240, Jun. 2011.

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

[13] 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.

[14] L. Wang, A. Liu, and S. Jajodia, "Using attack graphs for correlating, hypothesizing, and predicting

intrusion alerts," *Computer Communications*, vol. 29, no. 15, pp. 2917–2933, Sep. 2006.
[15] 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.

Authors:

Mrs.K.SUGANYA is a student of ASR Institute of Engineering & Technology, Tadepalligudam. Presently she is pursuing her M.Tech [Computer Science] from this college and she received her MCA from RVS college of Engineering and technology, Dindigul affiliated to Anna University, Trichy in the year 2010. Her area of interest includes Computer Networks and Object oriented Programming languages, all current trends and techniques in Computer Science.

Prof. K.T.V Subbarao, well known Author and teacher received M.Tech (CSE) and working as Principal, Akula SreeRamulu institute of Engineering and Technology, He is an active member of ISTE. He has 12 years of teaching experience in various engineering colleges. To his credit couple of publications both national and international conferences/journals. His area of interest includes cryptography and network security, Distributed databases, Operating systems and other advances in computer Applications.