Developing a Technique to Detect Intrusion in the Network and choosing countermeasure in virtual Network Systems

Monu
Dronacharya College of Engineering

***Abstract*— This paper introduces a new technique of network intrusion detection system (NIDS). First of all we will study about various host based and network based intrusion detection systems and problems of some existing intrusion detection systems. Afterwards we present an intrusion detection system using Alert Correlation Algorithm and Countermeasure selection algorithm to efficiently detect various types of network intrusions.**

I. INTRODUCTION

Generally intrusion is described as any set of actions that attempts to compromise the integrity, confidentiality or availability of computer resources. In order to providing a sense of security in computer and data networks, Dorothy E. Denning anticipated intrusion detection as an approach to defy attacks and misuse. Intrusion detection is done by intrusion detection systems. Today a variety of commercial intrusion detection systems are available. Typically intrusion detection systems are based on the assumption that an intruder will behave differently from the legitimate user and hence can be easily identified. It also assumes that nearly all the unauthorized actions are noticeable.

II. CLASSIFICATION OF INTRUSION DETECTION

Intrusion detection can be classified into two main categories:

1. Host Based Intrusion Detection: HIDSs evaluate information found on a single or multiple host systems, including contents of operating systems, system and application files.
2. Network Based Intrusion Detection: NIDSs evaluate information captured from the network communications, by analyzing the stream of packets which travel across the network.

Here we'll study about the Network Based Intrusion Detection Systems using alert correlation algorithm and countermeasure selection algorithm.

A "network intrusion detection system (**NIDS**)" monitors traffic on a network looking for suspicious activity, which could be an attack or unauthorized activity. For this purpose a large NIDS server can be set up on a backbone network in order to monitor all traffic; or smaller systems can be set up to monitor traffic for a particular server, switch, gateway, or router. NIDS server does not replace primary security such as firewalls, encryption, and other authentication methods.In addition to monitoring incoming and outgoing network traffic, a NIDS server can also scan system files looking for unauthorized activity and to maintain data and file integrity. The NIDS server can also detect changes in the server core components. It also scan server log files and look for suspicious traffic or usage patterns that match a typical network compromise or a remote hacking attempt. It serves a proactive role instead of a protective or reactive function. It can be possibly used in either scanning local firewalls or network servers for potential exploits or for scanning live traffic to see what is actually going on. We can say that the NIDS server is a backup network integrity device.

III. SOME EXISTING NETWORK INTRUSION DETECCTION SYSTEMS

- **Snort:-** It has long been the leader among network intrusion-detection and intrusion-prevention tools, and will most likely continue its reign with continued development from the open source

community and the ongoing support of its corporate parent, Sourcefire Inc. (For many years, Sourcefire has sold a fully featured commercial version of Snort that includes vendor support and immediate updates, while a limited version of the product remains available for free.) Snort has influenced other IDS/IPS vendors in a huge way, either by the way they develop their software or by directly using Snort modules in their offering. Even with Snort's dominance in the market, there are other vendors that offer similar functionality at no cost. Many, if not most, of these intrusion-detection systems (IDS) providers use a combination of engines, some being Snort and other open source software, to create solid, free intrusion-detection services.

- **Security Onion:-** Security Onion is an Ubuntu - based Linux distribution for network monitoring and intrusion detection. The image can be distributed as sensors within the network to monitor multiple VLANs and subnets, and works well in VMware and virtual environments. This configuration can be used as IDS only. It isn't currently supported to be run as an IPS. However, there is the option to run these both as a network and host intrusion-detection deployment, and to utilize services such as Bro IDS and OSSEC to perform the IDS functions of the service. As great as Security Onion is, however, it still needs more assistance with development, which will most likely happen in time.

- **OSSEC:-** OSSEC is an open source host intrusion-detection system (HIDS) that does more than detect intrusions. Like most open source IDS offerings, there are multiple additional modules that can be used with the core functionality of IDS. In addition to network intrusion-detection, the OSSEC client has the ability to perform file integrity monitoring and root kit detection with real-time alerts, all of which are centrally managed with the ability to create different policies, depending on a company's needs. The OSSEC client runs locally on most operating systems, including Linux versions and Windows. It also offers commercial support via Trend Micro's Global Support Team.

- **Open WIPS-NG:-** Open WIPS-NG is a free wireless IDS/IPS that relies on a server, sensors and interfaces. It runs on commodity hardware. Created by the author of Aircrack -NG, this system uses many of the functions and services already built into Aircrack-NG for scanning, detection and intrusion prevention. Open WIPS-NG is modular and allows an administrator to download plug-ins for additional features. The documentation isn't as detailed as some systems', but it allows for companies to perform WIPS on a tight budget.

- **Suricata:-** Out of all the IDS/IPS systems that are currently available, Suricata competes most directly with Snort. This system has an architecture that is similar to Snort's, relies on signatures like Snort, and can even use the VRT Snort rules and the same Emerging Threat rule set that Snort itself uses. Being newer than Snort, Suricata has ways to catch up to in this area. If Snort isn't an option in your organization, this is the closest free tool available to run on an enterprise network.

- **Bro IDS: -** it is similar to Security Onion in that it uses more than IDS rules to determine where attacks are coming from. Bro IDS uses a combination of tools. At one point it used Snort-based signatures converted into Bro signatures. This is no longer the case, and it is now possible to write custom signatures for the Bro IDS. This system is highly documented and has been around for over 15 years.

In computer security, a **Network Intrusion Detection System** (**NIDS**) is an intrusion detection system that attempts to discover unauthorized access to a computer network by analyzing traffic on the network for signs of malicious activity. A recent CSA (Cloud Survey Alliance) survey reports that among all security issues exploitation and despicable use of cloud computing is considered as the main security

threat. In traditional data centers, where system administrators have full control over the host machines, vulnerabilities can be detected and patched by the system administrator in a centralized manner. However, patching known security holes in cloud data centers, where cloud users usually have the privilege to control software installed on their managed VMs, may not work effectively and can violate the Service Level Agreement (SLA).

Furthermore, cloud users can install vulnerable software on their VMs, which essentially contributes to loopholes in cloud security. The challenge is to establish an effective vulnerability/attack detection and response system for accurately identifying attacks and minimizing the impact of security breach to cloud users. In a cloud system where the infrastructure is shared by potentially millions of users, abuse and nefarious use of the shared infrastructure benefits attackers to exploit vulnerabilities of the cloud and use its resource to deploy attacks in more efficient ways. Such attacks are more effective 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. The similar setup for VMs in the cloud, e.g., virtualization techniques, VM OS, installed vulnerable software, networking, etc., attracts attackers to compromise multiple VMs.

In this article, we propose a technique to establish a defense-in-depth intrusion detection framework. For better attack detection, NICE incorporates attack graph analytical procedures into the intrusion detection processes. We must note that the design of NICE does not intend to improve any of the existing intrusion detection algorithms; indeed, NICE employs a reconfigurable virtual networking approach to detect and counter the attempts to compromise VMs, thus preventing zombie VMs.

Cloud users can install vulnerable software on their VMs, which essentially contributes to loopholes in cloud security. The challenge is to establish an effective vulnerability/attack detection and response system for accurately identifying attacks and minimizing the impact of security breach to cloud users. In a cloud system where the infrastructure is shared by potentially millions of users, abuse and nefarious use of the shared infrastructure benefits attackers to exploit vulnerabilities of the cloud and use its resource to deploy attacks in more efficient ways. Such attacks are more effective 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. The similar setup for VMs in the cloud, e.g., virtualization techniques, VM OS, installed vulnerable software, networking, etc., attracts attackers to compromise multiple VMs.

DISADVANTAGES OF EXISTING SYSTEM

1. No detection and prevention framework in a virtual networking environment.
2. Not accuracy in the attack detection from attackers.

IV. PROPOSED SYSTEM

In this article, we propose NICE (Network Intrusion detection and Countermeasure selection in virtual network systems) to establish a defense-in-depth intrusion detection framework. For better attack detection, NICE incorporates attack graph analytical procedures into the intrusion detection processes. We must note that the design of NICE does not intend to improve any of the existing intrusion detection algorithms; indeed, NICE employs a reconfigurable virtual networking approach to detect and counter the attempts to compromise VMs, thus preventing zombie VMs.

ADVANTAGES OF PROPOSED SYSTEM:

The contributions of NICE are presented as follows

- We devise NICE, a new multi-phase distributed network intrusion detection and prevention framework in a virtual networking environment that captures and inspects suspicious cloud traffic without interrupting users' applications and cloud services.
- NICE incorporates a software switching solution to quarantine and inspect suspicious VMs for further investigation and protection. Through programmable network approaches, NICE can improve the attack detection probability and improve the resiliency to VM exploitation attack without interrupting existing normal cloud services.

- NICE employs a novel attack graph approach for attack detection and prevention by correlating attack behavior and also suggests effective countermeasures.
- NICE optimizes the implementation on cloud servers to minimize resource consumption. Our study shows that NICE consumes less computational overhead compared to proxy-based network intrusion detection solutions.

V. SYSTEM ARCHITECTURE:

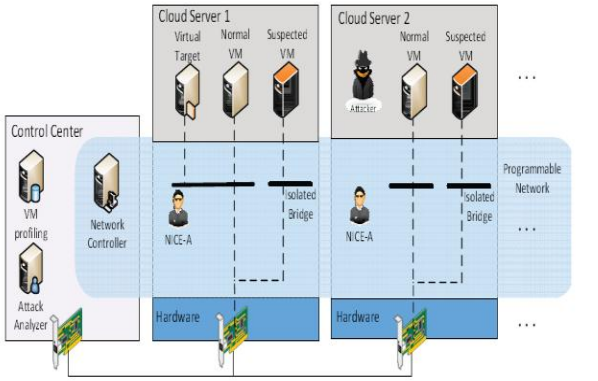

Fig 1: System Architecture within one server cloud cluster

ALGORITHMS USED:

- ✓ Alert Correlation Algorithm
- ✓ Countermeasure Selection Algorithm

1. Alert Correlation Algorithm:

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

Here ACG is the Alert Correlation Graph. It consists of three tuples (A, E, P)

2. Countermeasure Selection Algorithm:

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**
22: ROI [t, cm] =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)

MODULES:

- Nice-A
- VM Profiling
- Attack Analyzer
- Network Controller

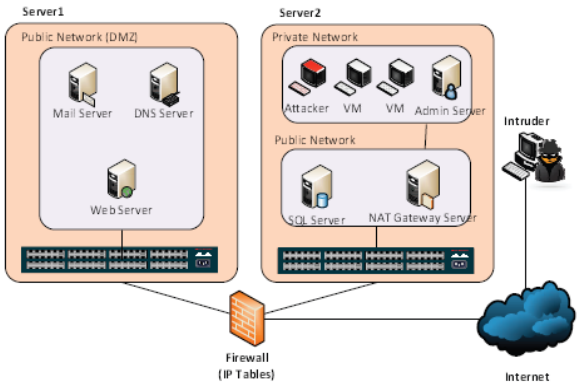

Fig. 2: Virtual network topology for security evaluation

MODULES DESCRIPTION

1. Nice-A:

The NICE-A is a Network-based Intrusion Detection System (NIDS) agent installed in each cloud server. It scans the traffic going through the bridges that control all the traffic among VMs and in/out from the physical cloud servers. It will sniff a mirroring port on each virtual bridge in the Open vSwitch. Each bridge forms an isolated subnet in the virtual network and connects 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 efficient to scan the traffic in cloud server since all traffic in the cloud server needs go through it; however our design is independent to the installed VM. The false alarm rate could be reduced through our architecture design.

2. VM Profiling:

Virtual machines in the cloud can be profiled to get precise 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 knowledge of services running on a VM so as to verify the authenticity of alerts pertaining to that VM. An attacker can use port scanning program to perform an intense examination 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 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 comprehensive information about vulnerabilities, alert and traffic.

3. Attack Analyzer

The major 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 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 handles alert correlation and analysis operations. This component has two major functions:

(1) Constructs Alert Correlation Graph (*ACG*),

(2) Provides threat information and appropriate countermeasures to network controller for virtual network reconfiguration.

NICE attack graph is constructed based on the following information: Cloud system information, Virtual network topology and configuration information, Vulnerability information.

4. Network Controller:

The network controller is a key component to support the programmable networking capability to realize the virtual network reconfiguration. In NICE, we integrated the control functions for both OVS and OFS into the network controller that allows the cloud system to set security/filtering rules in an integrated and comprehensive manner. The network controller is responsible for collecting network information of current Open Flow network and provides input to the attack analyzer to construct 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 applying the countermeasure from attack analyzer. Based on *VM* Security Index and severity of an alert, countermeasures are selected by NICE and executed by the network controller.

VI. CONCLUSION

Here a technique is proposed to detect and mitigate collaborative attacks in the cloud virtual networking environment. The proposed technique is an NIDS. A "network intrusion detection system (**NIDS**)" monitors traffic on a network looking for suspicious activity, which could be an attack or unauthorized activity. For this purpose a large NIDS server can be set up on a backbone network in order to monitor all traffic; or smaller systems can be set up to monitor traffic for a particular server, switch, gateway, or router. NIDS server does not replace primary security such as firewalls, encryption and other authentication methods. Existing intrusion detection systems are based on the assumption that an intruder will behave differently from the legitimate user and hence can be easily identified. It also assumes that nearly all the unauthorized actions are noticeable.

NICE utilizes the attack graph model to conduct attack detection and prediction. The proposed solution investigates how to use the programmability of software switches based solutions to improve the detection accuracy and defeat victim exploitation phases of collaborative attacks. NICE only investigates the network IDS approach to counter zombie explorative attacks.

In order to improve the detection accuracy, host-based IDS solutions are needed to be incorporated and to cover the whole spectrum of IDS in the cloud system. This should be investigated in the future work. Additionally, we can investigate the scalability of the proposed NICE solution by investigating the decentralized network control and attack analysis model based on current study. Intrusion Detection System (IDS) and firewall are widely used to monitor and detect suspicious events in the network. However, the false alarms and the large volume of raw alerts from IDS are two major problems for any IDS implementations. In order to identify the source or target of the intrusion in the network, especially to detect multi-step attack, the alert correction is a must-have tool. After knowing the possible attack scenarios, applying countermeasure is the next import

REFERENCE

[1] Chun-Jen Chung, Student Member, IEEE, Pankaj Khatkar, Student Member, IEEE, Tianyi Xing.

[2] Jeongkeun Lee, Member, IEEE, and Dijiang Huang Senior Member, IEEE- “NICE: Network Intrusion Detection and Countermeasure Selection in Virtual Network Systems”- IEEE TRANSACTIONS ON DEPEDABLE AND SECURE COMPUTING, 2013.

[3] Denning, D. E. and Neumann, P. G. “Requirements and Model for IDES -- a Real-Time Intrusion Detection System”, Tech. report, Computer Science Lab, SRI International, 1985.

[4] K. Scarfone, P. Mell, “Guide to Intrusion Detection and Prevention Systems (IDPS)”. Computer Security Resource Center (National Institute of Standards and Technology).

[5] Ptacek, Thomas H. & Newsham, Timothy N. (January 1998); "Insertion, Evasion, and Denial of Service: Eluding Network Instrusion Detection".

[6] Lunt, Teresa F., "Detecting Intruders in Computer Systems," 1993 Conference on Auditing and Computer Technology, SRI International.

[7] Sebring, Michael M., and Whitehurst, R. Alan., "Expert Systems in Intrusion Detection: A Case Study," The 11th National Computer Security Conference, October, 1988.