RESEARCH ARTICLE OPEN ACCESS

Intrusion Detection and Countermeasure in Virtual Network Systems Using NICE Technique

M.S.Haja Moideen,N.Chandru,G.SathishKumar,G.VishnuPrabu, R.Ramachandiran
P.G Student, Department Of Master Computer Application, Sri Manakula Vinayagar Engineering College, Pondicherry-605 107
P.G Student, Department Of Master Computer Application, Sri Manakula Vinayagar Engineering College, Pondicherry-605 107
P.G Student, Department Of Master Computer Application, Sri ManakulaVinayagar Engineering College, Pondicherry-605 107
P.G Student, Department Of Master Computer Application, Sri ManakulaVinayagar Engineering College, Pondicherry-605 107
Assistant Professor, Department Of Master Computer Application, Sri Manakula Vinayagar Engineering College, Pondicherry-605 107

ABSTRACT
The cloud computing has increased in many organizations. It provides many benefits in terms of low cost and accessibility of data. Ensuring the security of cloud computing is a major factor in the cloud computing environment, as users often store sensitive information with cloud storage providers but these providers may be untrusted. In this project we propose anIntrusion Detection and Countermeasure in Virtual Network Systems mechanism called NICE to prevent vulnerable virtual machines from being compromised in the cloud. NICE detects and mitigates collaborative attacks in the cloud virtual networking environment. The system performance evaluation demonstrates the feasibility of NICE and shows that the proposed solution can significantly reduce the risk of the cloud system from being exploited and abused by internal and external attackers.

Key Terms—Network security, cloud computing, intrusion detection

I. Introduction

Wireless Sensor Networks (WSNs) can be defined as a self-configured and infrastructure less wireless networks to monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants and to cooperatively pass their data through the network to a main location or sink where the data can be observed and analyzed. A sink or base station acts like an interface between users and the network. One can retrieve required information from the network by injecting queries and gathering results from the sink. Typically a wireless sensor network contains hundreds of thousands of sensor nodes. The sensor nodes can communicate among themselves using radio signals. A wireless sensor node is equipped with sensing and computing devices, radio transceivers and power components. The individual nodes in a wireless sensor network (WSN) are inherently resource constrained: they have limited processing speed, storage capacity, and communication bandwidth. After the sensor nodes are deployed, they are responsible for self-organizing an appropriate network infrastructure often with multi-hop communication with them.

Then the on board sensors start collecting information of interest. Wireless sensor devices also respond to queries sent from a "control site" to perform specific instructions or provide sensing samples. The working mode of the sensor nodes may be either continuous or event driven.
Global Positioning System (GPS) and local positioning algorithms can be used to obtain location and positioning information. Wireless sensor devices can be equipped with actuators to "act" upon certain conditions. These networks are sometimes more specifically referred as Wireless Sensor and Actuator Networks as described in (Akkaya et al., 2005).Wireless sensor networks (WSNs) enable new applications and require non-conventional paradigms for protocol design due to several constraints. Owing to the requirement for low device complexity together with low energy consumption (i.e. long network lifetime), a proper balance between communication and signal/data processing capabilities must be found. This motivates a huge

effort in research activities, standardization process, and industrial investments on this field since the last decade (Chiara et. al. 2009). At present time, most of the research on WSNs has concentrated on the design of energy- and computationally efficient algorithms and protocols, and the application domain has been restricted to simple data-oriented monitoring and reporting applications (Labrador et. al.2009). The authors in (Chen et al., 2011) propose a Cable Mode Transition (CMT) algorithm, which determines the minimal number of active sensors to maintain K-coverage of a terrain as well as K-connectivity of the network. Specifically, it allocates periods of inactivity forcable sensors without affecting the coverage and connectivity requirements of the network based only on local information. In (Cheng et al., 2011), a delay-aware data collection network structure for wireless sensor networks is proposed. The objective of the proposed network structure is to minimize delays in the data collection processes of wireless sensor networks which extends the lifetime of the network. In (Matin et al., 2011), the authors have considered relay nodes to mitigate the network geometric deficiencies and used Particle Swarm Optimization (PSO) based algorithms to locate the optimal sink location with res pectto those relay nodes to overcome the lifetime challenge. Energy efficient communication has also been addressed in (Paul et al., 2011; Fabbri et al. 2009). In (Paul et al., 2011), the authors proposed a geometrical solution for locating the optimum sink placement for maximizing the network lifetime. Most of the time, the research on wireless sensor networks have considered homogeneous sensor nodes. But nowadays researchers have focused on heterogeneous sensor networks where the sensor nodes are unlike to each other in terms of their energy. In (Han et al., 2010), the authors addresses the problem of deploying relay nodes to provide fault tolerance with higher network connectivity in heterogeneous wireless sensor networks, where sensor nodes possess different transmission radii. New network architectures with heterogeneous devices and the recent advancement in this technology eliminate the current limitations and expand the spectrum of possible applications for WSN sconsiderably and all these are changing very rapidly

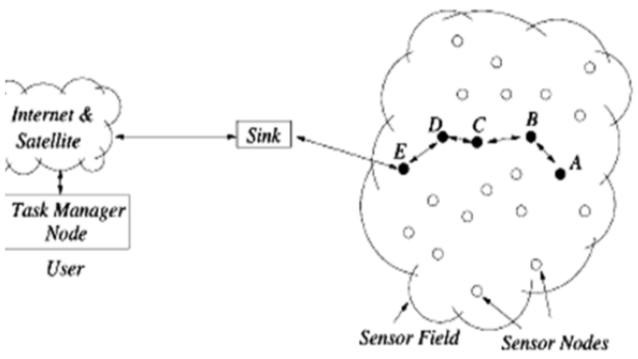

Figure :1.A typical Wireless Sensor Network

II. Design issues of a wireless sensor network

There are a lot of challenges placed by the deployment of sensor networks which are a superset of those found in wireless ad hoc networks. Sensor nodes communicate over wireless, lossy lines with no infrastructure. An additional challenge is related to the limited,usually non-renewable energy supply of the sensor nodes. In order to maximize the life timeof the network, the protocols need to be designed from the beginning with the objective of efficient management of the energy resources (Akyildiz et al., 2002). Wireless Sensor Network Design issues are mentioned in (Akkaya et al., 2005), (Akyildizet al., 2002),(SensorSim; Tossim, Younis et al., 2004), (Pan et al., 2003) and different possible platformsfor simulation and testing of routing protocols for WSNs are discussed in (NS-2, Zeng et al.,1998, SensorSim, Tossiim). Let us now discuss the individual design issues in greater detail. Fault Tolerance: Sensor nodes are vulnerable and frequently deployed in dangerous environment.

Sensor Network Topology: Although WSNs have evolved in many aspects, they continue to be networks with constrained resources in terms of energy, computing power, memory, and communications capabilities. Of these constraints, energy consumption is of paramount importance, which is demonstrated by the large number of algorithms, techniques, and protocols that have been developed to save energy, and thereby extend the lifetime of the network. Topology Maintenance is one of the most important issues researched to reduce energy consumption in wireless sensor networks. Transmission Media: The communication between the nodes is normally implemented using radio communication over the popular ISM bands. However, some sensor networks use optical or infrared communication, with the latter having the advantage of being robust and virtually interference free. Power Consumption: As we have already seen, many of the challenges of sensor networks revolve around the limited power resources. The size of the nodes limits the size of the battery. The software and hardware design needs to carefully consider the issues of efficient energy use.

2.1 Communication structure of a WSN

The sensor nodes are usually scattered in a sensor field as shown in Fig. Each of these scattered sensor nodes has the capabilities to collect data and route data back to the sink and the end users. Data are routed back to the end user by a multi-hop infrastructure-less architecture through the sink as shown in Fig. 1. The sink may communicate with the task manager node via Internet or Satellite.

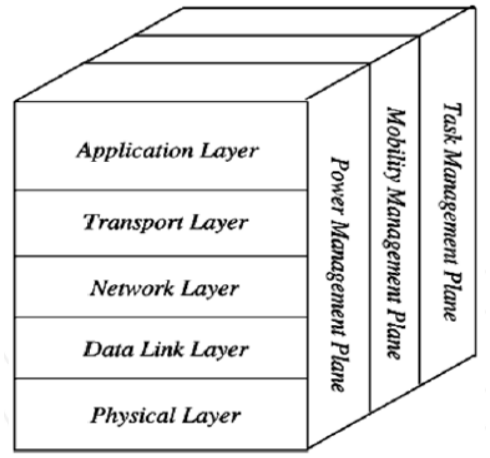

Figure: 2 . Wireless Sensor Network protocol stack

III. Network Intrusion Detection System

In computer security, a Network Intrusion Detection System (NIDS) is an intrusion detection systemthat attempts to discover unauthorized access to a computer networkby analyzing trafficon the network for signs of malicious activity Application protocol-based intrusion detection system

An application protocol-based intrusion detection system (APIDS) is an intrusion detection systemthat focuses its monitoring and analysis on a specific application protocolor protocols in use by the computing system.

3.1 Overview

An APIDS will monitor the dynamic behavior and stateof the protocol and will typically consist of a system or agent that would typically sit between a process, or group of servers, monitoringand analyzing the application protocol between two connected devices.

A typical place for an APIDS would be between a web server and the database management system, monitoring the SQLprotocol specific to the middleware/business logicas it interacts with the database.

3.2 Monitoring Dynamic Behavior

At a basic level an APIDS would look for, and enforce, the correct (legal) use of the protocol. However at a more advanced level the APIDS can learn, be taught or even reduce what is often an infinite protocol set, to an acceptable understanding of the subsetof that application protocol that is used by the application being monitored/protected.

Thus, an APIDS, correctly configured, will allow an application to be "fingerprinted", thus should that application be subverted or changed, so will the fingerprint change.

IV. VIRTUAL NETWORK COMPUTING

In computing, Virtual Network Computing (VNC) is a graphical desktop sharing system that uses the Remote Frame Buffer protocol (RFB) to remotely control another computer. It transmits the keyboard and mouse events from one computer to another, relaying the graphical screen updates back in the other direction, over a network.

VNC is platform-independent – There are clients and servers for many GUI-based operating systems and for Java. Multiple clients may connect to a VNC server at the same time. Popular uses for this technology include remote technical support and accessing files on one's work computer from one's home computer, or vice versa.

VNC by default uses TCP port 5900+N, where N is the display number (usually: 0 for a physical display). Several implementations also start a basic HTTP server on port 5800+N to provide a VNC viewer as a Java applet, allowing easy connection through any Java-enabled web browser. Different port assignments can be used as long as both client and server are configured accordingly.

4.1 Security

By default, RFB is not a secure protocol. While passwords are not sent in plain-text (as in telnet), cracking could prove successful if both the encryption key and encoded password are sniffed from a network. For this reason it is recommended that a password of at least 8 characters be used. On the other hand, there is also an 8-character limit on some versions of VNC; if a password is sent exceeding 8 characters, the excess characters are removed and the truncated string is compared to the password.

UltraVNC supports the use of an open-source encryption plugin which encrypts the entire VNC session including password authentication and data transfer. It also allows authentication to be performed based on NTLM and Active Directory user accounts. However, use of such encryption plugins make it incompatible with other VNC programs.

4.2 Limitations

Unicode is not supported in RFB versions 3.x and lower so it is impossible to transfer clipboard text outside the Latin-1 character set.

The VNC protocol is pixel based. Although this leads to great flexibility (e.g., any type of desktop can be displayed), it is often less efficient than solutions that have a better understanding of the underlying graphic layout like X11 or Windows Remote Desktop Protocol. Those protocols send graphic primitives or high level commands in a simpler form (e.g., "open window"), whereas RFB just sends the raw pixel data.

V. DESIGN OF MODEL OR TECHNIQUE OR ALGORITHM

5.1 Alert Correlation

Algorithm 1 presents VM protection model of NICE consists of a VM profiler, a security indexer, and a state monitor. We specify security index for all the VMs in the network depending upon various factors like connectivity, the number of vulnerabilities present and their impact scores. The impact score of a vulnerability, as defined by the CVSS guide [24], helps to judge the confidentiality, integrity, and availability impact of the vulnerability being exploited. Connectivity metric of a VM is decided by evaluating incoming and outgoing connections.
VM states can be defined 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 unexploited.
3. Exploited. At least one vulnerability has been exploited and the VM is compromised.
4. Zombie. VM is under control of attacker.

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 $S$
```

5.2 COUNTERMEASURE SELECTION

Algorithm 2 presents how to select the optimal counter- measure for a given attack scenario. Input to the algorithm is an alert, attack graph G, and a pool of countermeasures CM. The algorithm starts by selecting the node vAlert that corresponds to the alert generated 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 threshold value, we do not perform countermeasure selection but update the ACG to keep track of alerts in the system (line 3). For the source node vAlert, all the reachable nodes (including thesourcenode)arecollectedintoasetT (line6).Because the alert is generated only after the attacker has performed the action, we set the probability of vAlert to 1 and calculate the new probabilities for all of its child (downstream) nodes in the set T (lines 7 and 8). Now, for all t 2 T the applicable countermeasures in CM are selected and new probabilities are calculated according to the effectiveness of the selected countermeasures (lines 13 and 14). The change in probability of target node gives the benefit for the applied counter- measure using (7). In the next double for-loop, we compute the Return of Investment (ROI) for each benefit of the applied countermeasure based on (8). The countermeasure which when applied on a node gives the least value of ROI, is regarded as the optimal countermeasure. Finally, SAG and ACG are also updated before terminating the algorithm. The complexity of Algorithm 2 is Ϭ(|V| x |CM|), where |V| is the number of vulnerabilities and |CM| represents the number of countermeasures.

Algorithm 2 Countermeasure_Selection

```
Require: $Alert, G(E,V), CM$
1: Let $v_{Alert}$ = Source node of the $Alert$
2: if Distance_to_Target($v_{Alert}$) $> threshold$ ther
3:     Update_ACG
4:     return
5: end if
6: Let $T = Descendant(v_{Alert}) \cup v_{Alert}$
7: Set $Pr(v_{Alert}) = 1$
8: Calculate_Risk_Prob($T$)
9: Let $benefit[|T|, |CM|] = \emptyset$
10: for each $t \in T$ do
11:     for each $cm \in CM$ do
12:         if $cm.condition(t)$ then
13:             $Pr(t) = Pr(t) * (1 - cm.effectivene$
14:             Calculate_Risk_Prob($Descendant(t)$)
15:             $benefit[t, cm] = \Delta Pr(target\_node)$.
16:         end if
17:     end for
18: end for
19: Let $ROI[|T|, |CM|] = \emptyset$
20: for each $t \in T$ do
21:     for each $cm \in CM$ 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($ROI$)
```

VI. Conclusion

In this paper, we presented NICE, which is proposed to detect and mitigate collaborative attacks in the cloud virtual networking environment. 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. The system performance evaluation demonstrates the feasibility of NICE and shows that the proposed solution can significantly reduce the risk of the cloud system from being exploited and abused by internal and external attackers.

NICE only investigates the network IDS approach to counter zombie explorative attacks. 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, as indicated in the paper, we will investigate the scalability of the proposed NICE solution by investigating the decentralized network control and attack analysis model based on current study.

REFERENCES

[1] CoudSercurity Alliance, "Top Threats to Cloud Computing v1.0," https://cloudsecurityalliance.org/topthreats/csathreats. v1.0.pdf, Mar. 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 Comm., vol. 53, no. 4, pp. 50-58, Apr. 2010.
[3] B. Joshi, A. Vijayan, and B. Joshi, "Securing Cloud Computing Environment Against DDoS Attacks," Proc. IEEE Int'l Conf. Computer Comm. 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 and 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. 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. 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. Security and Privacy, pp. 273-284, 2002,
[10] "NuSMV: A New Symbolic Model Checker," http://afrodite.itc. it:1024/nusmv. Aug. 2012.
[11] P. Ammann, D. Wijesekera, and S. Kaushik, "Scalable, graph- based network vulnerability analysis," Proc. 9th ACM Conf. Computer and Comm. Security (CCS '02), pp. 217-224, 2002.
[12] X. Ou, S. Govindavajhala, and A.W. Appel, "MulVAL: A Logic- Based Network Security Analyzer," Proc. 14th USENIX Security Symp., pp. 113-128, 2005.
[13] R. Sadoddin and A. Ghorbani, "Alert Correlation Survey: Frame- work and Techniques," Proc. ACM Int'l Conf.

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 Comm., vol. 29, no. 15, pp. 2917-2933, Sept. 2006.

[15] S. Roschke, F. Cheng, and C. Meinel, "A New Alert Correlation Algorithm Based on Attack Graph," Proc. Fourth Int'l Conf. Computational Intelligence in Security for Information Systems, pp. 58-67, 2011.