HTTP SERVICE BASED NETWORK INTRUSION DETECTION SYSTEM IN CLOUD COMPUTING

Abstract

Recently, the usages of Cloud Computing are increasing rapidly and gained tremendous success over the internet. Therefore, security is the major challenge in Cloud computing and one of the major issues is to protect the Cloud resources and the services against network intrusions. So Network Intrusion Detection System (NIDS) are installed in the Cloud networks to detect the intrusions in the system. In this paper we proposed an NIDS based on Naïve Bayes Classifier to be implemented in Cloud. The main aim of the NIDS is to improve the performance by preparing the training dataset which can detect the malicious connections that exploit the Cloud HTTP services. In the training phase, the Naïve Bayes Classifiers select the important Network traffic that can be used to detect the attacks. In the testing and execution phases the proposed NIDS using the Naïve Bayes Classifier classifies the services based on the selected features into normal or attacks. The proposed IDS carried out on NSL-KDD’99 dataset and results in high detection with low false alarm as compared with other similar IDS.

KEYWORDS

Cloud Computing; Network based Intrusion Detection System; Naïve Bayes Classifier

I. INTRODUCTION

These days, most of the services are processed through a computer networks over the internet. The tremendous rises in the computer networks and huge accessibility of internet have gained a lot of positive aspects. However, the attackers may access the confidential information or network resources by exploiting the internet. To avoid anomaly intrusions in the Cloud network some security tools are used such as firewalls, antivirus software and Intrusion Detection System (IDS). The traditional attacks such as IP spooling, Address Resolution Protocol (ARP), port scanning, Denial of Service (DoS), Distributed Denial of Service (DDoS) etc can be detected using firewall. But the firewall can only detect the network intrusions at the boundary of the network, inside attacks cannot be captured by it. Therefore, to handle all types of intrusions firewall is not a efficient technique.

NIDS is another technique to detect the intrusions in Cloud Computing. It acts as a alert system and raise alarm when any unauthorized users wants to penetrate through the system. The accuracy of the NIDS depends on three parameters, the first parameter is the detection technique (signature based or anomaly based), the second parameter is the installation location (front end or back end) and last parameter is the configuration technique (distributed or centralized).

In this paper, we propose a NIDS based on Naïve Bayes Classifier to detect the intrusions and also the positioning of NIDS in the Cloud networks has been recommended. Bayesian classifier implements the statistical method to classify the network events as normal or as attacks. In the training phase, the algorithm classifies the dataset into normal and attack
classes by computing the conditional probabilities for each of the different classes. In the testing and execution phase, classifier classifies the unknown network traffic based on the probability value of different classes. Our main objective is to decrease the impact of intrusions and ensuring higher detection rate with reduce false alarm rate.

Rest of the paper is organized as follows: the section II presents the related work and theoretical background, section III presents the detailed description of the proposed NIDS, implementation and results of the proposed NIDS is given in section IV and finally conclusion and future works are described in section 5 with the references at the end.

II. RELATED WORK AND THEORETICAL BACKGROUND

A. Problem Statement

The objective is to design an efficient NIDS that can detect the intrusions in Cloud network with reduced false positives and false negatives.

B. Related Work

In [3] the IDS is proposed based on anomaly detection approach, it implements the combination of K-means, K-Nearest Neighbor classifier and Naïve Bayes Classifier. It uses entropy based algorithm for feature selection and classifies the attacks into different categories like DOS, U2R, R2L, and probe. The performance of the IDS is good but the computational time may increase.

In [4] a score based multi-cycle detection algorithm is proposed based on Shiryaev-Roberts procedure. This procedure is computationally inexpensive and it can be easily implemented in real time IDS. The proposed IDS minimize the detection rate, however, it increases the false alarm rates. Therefore, it implements an additional filtering technique to increase the detection accuracy, which it may increase the processing delay.

A tree based IDS classification algorithms are proposed in [5]. Implementing the Random Tree model it could achieve 97.47% of detection accuracy and its false rate is about 2.5%.

In [6] the IDS based on the combination of Snort and Bayes theorem has been proposed to be implemented in Cloud Computing. In this, Snort checks the captured packets using signature based detection and Bayesian Classifier identifies the intrusion packets and normal packets. The IDS achieves the detection rate of about 96% and 1.5% false positive rate.

In [7] a virtual machine compatible IDS have been proposed that consists of two main components: IDS management unit and IDS sensor. IDS management unit includes event gatherer, database management system, analysis component and remote controller. IDS sensor capture the malicious events and transmits the events to event gatherer and stores it in event database. Analysis component analyze the stored events as per design. IDS-VMs are controlled by the IDS Remote Controller which can communicate with other IDS-VMs and IDS sensors. Here sensor can be treated as a NIDS and it is configured by IDS remote controller. The function of IDS-VM is to controls, monitors and configures the VM. This technique prevents the VMs from being compromised but requires multiple instances of IDS.

C. Theoretical Background

1) Bayesian Classifier:

It operates on a strong independence assumption [8]. This means the effect of an attribute value on a given class does not depend on other attributes. The classifier follows the statistical approach which can predict the probability of a network request belongs to a normal or attack class.

Let X is a data tuple. H is the hypothesis that X belongs to a particular class C. P(H | X)
Is the posterior probability of $H$ conditioned on $X$. Using Bayesian theorem, the probability $P(H \mid X)$ of a hypothesis $H$ on a given data tuple $X$ can be defined by Eq. (1)

$$P(H \mid X) = \frac{P(X \mid H) P(H)}{P(X)}$$

(1)

Let $D$ is a training set of tuples and their related class labels. Each tuple is represented by a vector $X=(x_1, x_2, \ldots, x_n)$ and each tuple contains $n$ attributes $A_1, A_2, \ldots, A_n$. Assume there are $m$ number of classes $C_1, C_2, \ldots, C_m$. Classifier will predict that tuple $X$ belongs to class $C_i$, iff

$$P(C_i \mid X) > P(C_j \mid X) \quad \text{for} \quad 1 \leq j \leq m, j \neq i$$

The maximal $P(C_i \mid X)$ can be derived from Eq. (2). Since $P(X)$ is constant for all classes, we need to maximize $p(X \mid C_i) P(C_i)$ which is defined in Eq. (3)

$$P(C_i \mid X) = \frac{P(X \mid C_i) P(C_i)}{P(X)}$$

(2)

$$P(C_i \mid X) = \frac{P(X \mid C_i) P(C_i)}{P(X)}$$

(3)

Thus, Bayesian classifier predicts the request packet is normal or attack based on the previously stored packets.

2) Snort:
It is an open source IDS implements signature based intrusion detection technique. It is widely used, easily configurable and can run on multiple platforms like GNU/Linux, Windows, etc. It captures the network data packets and compares their contents with the predefined known attack patterns for any correlation.

2) Information gain for feature selection:

Information Gain is used to rank the features individually based on the class labels. Let $S$ be a training data set consisting of $m$ classes. Suppose the data set contains $S_i$ samples belongs to class $I$, then the information needed to classify the given sample is defined in Eq. (4)

$$\text{Info}(S_1, S_2, \ldots, S_m) = - \sum_{i=1}^{m} \frac{S_i}{S} \log(S_i / S)$$

(4)

Suppose feature $F$ with values $\{f_1, f_2, \ldots, f_v\}$ divide the data set into subsets $\{S_1, S_2, \ldots, S_v\}$ and let $S_j$ contains $S_{ij}$ samples of class $i$. Entropy of features $F$ is defined in Eq. (5)

$$E(F) = \sum_{i=1}^{m} \frac{S_{ij}}{S} \text{Info}(S_{1j}, S_{2j}, \ldots, S_{ij})$$

(5)

Information Gain for feature $F$ can be computed using Eq. (6)

$$\text{Gain}(F) = \text{Info}(S_1, S_2, \ldots, S_m) - E(F)$$

(6)

III. PROPOSED NIDS BASED ON BAYESIAN CLASSIFIER IN CLOUD COMPUTING

A. Integration of NIDS in Cloud
As shown in fig. 1 Cloud Computing consist of two ends i.e. Front end and back end. The end users can access the Cloud offered services through front end. The front end is connected between the external network and internal network. Processing server executes the users request and allows accessing the VM instances. Internal networks give the design structure of the VM instances. For e.g., each VM has two network IPs named public IP and private IP [9]. Each VM can communicate with each other through private network. The mapping of public IP of VM to private IP of VM can be done through Network Address Translation (NAT).
As shown in Fig. 1 NIDS module can be installed at different positions in the Cloud. When the module is positioned at the front end of the Cloud, it can only detect intrusions at the external network but unable to detect internal intrusions. When the module is positioned on the processing server, it can detect intrusions both at the internal and external networks of the Cloud. However, the efficiency of the NIDS may decrease since a large number of packets pass through the server. When the NIDS is integrated on each VM, it can detect intrusions on each VM. Such configuration requires multiple instances of NIDS, which makes it very difficult to manage the NIDS since the VMs are dynamically migrated, provisioned or de-provisioned.

B. Proposed Framework of NIDS

As shown in Fig. 2, the NIDS module consists of three main components: packet preprocessing, analyzer, and storage system.

Preprocessing module converts the captured packets to a specific format by removing redundant information that has a very low impact on detection. The analyzer decides whether the captured data packet is normal or an intrusion by using signature-based and anomaly detection techniques. The analyzer consists of three components: Snort, Bayesian Classifier, and Alert Log. Alert Log system logs intrusion packets and sends an alert message to other NIDS. Other NIDS store these alert messages in their storage device. There are two types of storage devices: Knowledge base and Behavior base. Knowledge base stores the rules or predefined known attack patterns, and this device is used by Snort for intrusion detection. Behavior base stores both the normal and intrusion packets and is used by the Bayesian Classifier to detect the intrusions.

NIDS module implements two types of detection techniques: Signature-based detection and Anomaly-based detection. Signature-based detection compares the captured data packets with the rules stored in Knowledge base to detect known attacks easily. It logs the intrusion packets in an alert database, so that other NIDS can communicate the message with each other. Non-intrusion packets are processed for...
anomaly detection and it is used by Bayesian Classifier to predict the class label (normal or intrusion) by comparing it with behavior base.

C. Working Principle of the proposed NIDS

In the first phase (Fig. 3), the NIDS read the NSL-KDD training data set. The records are classified into normal or intrusion classes by applying Naïve Bayes Eq.(3). The training data set are reduced by removing the records that generates false alarm and the NIDS is retrained on the reduced data set and the performance is again measured. These reduction processes are repeated until the performance reaches 100%. The aim of the training phase is to identify the records from the training data set that provides 100% accuracy.

The main aim of the proposed NIDS is to identify HTTP data packets only. So in the second phase, HTTP records are selected from the training data set by removing features 2, 3 and 4 from the 41 features of the NSL-KDD data set. These features include protocol, service and flag. These features are removed because all the HTTP records have identical protocol (TCP), service (HTTP) and flag (SF). In the proposed algorithm, the features are selected based on Information Gain. In this process features are removed from both training and testing data set and then the performance of the classifier is measured using Naïve Bayes Classifier.

D. Proposed Algorithm

Input:
F = All the 41 features of NSL-KDD data set
N = Total number of records in the data set.
CA = classifier accuracy on entire data set.
Err = RMSE on entire data set.
Avg_TPR = average TPR on entire data set.
TH=Threshold value

First Phase
1) For each record Ri
   Do
   If Ri generates false alarm, then
   N = N – {Ri} // N contains the resultant training data set
   End

Second Phase
2) S={F}- {2,3,4} // Remove feature 2, 3 and 4 to identify only HTTP records
   For each feature Fi,
   i.  INF_i = compute information gain
   ii. If INF_i < TH
   iii. S=S - {Fi}
3) Call Naïve Bayes Classifier based on the selected features

IV. IMPLEMENTATION AND RESULT
The main aim of the proposed NIDS is to achieve 100% detection rate during the training phase. It means the training data must detect all the intrusion before implementing the testing phase. In the first phase of the NIDS, execute the algorithm on the training data as training as well as testing purpose. These steps are repeated until the accuracy reaches 100% i.e. no false alarm is generated.

In the first phase, KDD-Train 20% (25,192 records) has been taken as training well as testing data set. To implement Naïve Bayes Classifier all the 41 features must have numeric value. Table-1 shows the results of each step during execution of the first phase of the proposed algorithm. In step-0 the false-negative alarm is 0.52% and the false-positive alarm is 2.66%. The records causing these false alarms are removed from the training data set. After step-0, the data set reduced to 24,810 records and the test is repeated again. This process is repeated and in step-5 the TP rate and TN rate reaches 100%. After step-5, the data set reduced to 24,262 records and the resultant data set is treated as training data set.

Table-1. The experimental results of the first phase of the algorithm

<table>
<thead>
<tr>
<th>Step</th>
<th># of Attack Records</th>
<th># of Normal Records</th>
<th>Total</th>
<th>True positive (%)</th>
<th>False negative(%)</th>
<th>True negative (%)</th>
<th>False positive (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11431</td>
<td>13379</td>
<td>24810</td>
<td>97.34</td>
<td>2.66</td>
<td>99.48</td>
<td>0.52</td>
</tr>
<tr>
<td>1</td>
<td>11199</td>
<td>13379</td>
<td>24578</td>
<td>97.97</td>
<td>2.03</td>
<td>100</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td>10,934</td>
<td>13379</td>
<td>24313</td>
<td>97.63</td>
<td>2.37</td>
<td>100</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>10889</td>
<td>13379</td>
<td>24268</td>
<td>99.58</td>
<td>0.42</td>
<td>100</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>10883</td>
<td>13379</td>
<td>24262</td>
<td>99.94</td>
<td>0.06</td>
<td>100</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>10883</td>
<td>13379</td>
<td>24262</td>
<td>100.00</td>
<td>0.00</td>
<td>100</td>
<td>0.00</td>
</tr>
</tbody>
</table>

The second phase of the NIDS is to identify the features that can be used to detect the HTTP attacks. In this phase, features 2, 3 and 4 are removed from the training data set (phase-1 records). Features are selected having Information gain more than the threshold value. Table -2 shows the selected features based on different threshold value. Table -3 shows the results of the second phase of the proposed algorithm. The performance
of the algorithm is tested using NSL-KDD Test 21% (11,850 records) data set.

Table 2. Selected Features with higher Information Gain

<table>
<thead>
<tr>
<th>Threshold Value</th>
<th># of features selected</th>
<th>Selected features</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>25</td>
<td>1,2,3,5,7,9,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,38</td>
</tr>
<tr>
<td>0.05</td>
<td>23</td>
<td>1,2,3,9,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38</td>
</tr>
<tr>
<td>0.1</td>
<td>19</td>
<td>2,3,9,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,38</td>
</tr>
</tbody>
</table>

Table-3 Experimental results on the selected Features

<table>
<thead>
<tr>
<th>Step</th>
<th>No. of Features</th>
<th>Threshold Value</th>
<th>True positive (%)</th>
<th>False negative (%)</th>
<th>True negative (%)</th>
<th>False positive (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>38</td>
<td>0.00</td>
<td>97.76</td>
<td>2.24</td>
<td>98.993</td>
<td>1.007</td>
</tr>
<tr>
<td>1</td>
<td>25</td>
<td>0.01</td>
<td>99.54</td>
<td>0.46</td>
<td>98.993</td>
<td>0.51</td>
</tr>
<tr>
<td>2</td>
<td>23</td>
<td>0.05</td>
<td>99.7</td>
<td>0.30</td>
<td>99.84</td>
<td>0.16</td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>0.1</td>
<td>99.75</td>
<td>0.25</td>
<td>99.46</td>
<td>0.54</td>
</tr>
</tbody>
</table>

The Proposed classified based NIDS achieved good detection rate as compared with similar IDS. Table -4 shows the detection rate of the proposed NIDs along with some similar IDS.

Table 4. Comparison of IDS detection rates

<table>
<thead>
<tr>
<th>IDS Index</th>
<th>Intrusion Detection system</th>
<th>Detection Rate %</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDS I</td>
<td>Anomaly based IDS [10]</td>
<td>68.7</td>
</tr>
<tr>
<td>IDS II</td>
<td>IDS based on integration of Snort and bayes theorem [6]</td>
<td>96</td>
</tr>
<tr>
<td>IDS III</td>
<td>Tree based IDS classification algorithm [5]</td>
<td>97.49</td>
</tr>
<tr>
<td>Proposed IDS</td>
<td>HTTP based NIDS using Naive Bayes Classifier</td>
<td>99.75</td>
</tr>
</tbody>
</table>
V. CONCLUSION AND FUTURE WORK

To detect and handle all types of malicious activities in Cloud network, firewall is not an efficient solution. In this paper, we design the framework of NIDS integrating Naïve Bayes Classifier and Snort to identify the intrusions in Cloud Computing. Additionally, an algorithm has been proposed to classify the data set based on HTTP services. The objective of the NIDS is to enhance the performance by selecting only the important features that identify each attack and normal connections of HTTP services. The proposed NIDS shows significant performance, the TP = 99.75%, TN=99.46%, FP=0.54% and FN=0.25%. The performance is better as compared to similar IDS.

As a future work, the proposed NIDS can be installed in Cloud to protect network against real time intrusions. Additionally, it can be applied to other network services like TELNET and FTP.

REFERENCES


