An Adaptive DCT Based Intrusion Detection System

Morteza Zi Hayat  
School of Electrical and Computer Engineering  
University of Tehran  
Tehran, Iran  
zihayatm@ut.ac.ir

Mahmoud Reza Hashemi  
School of Electrical and Computer Engineering  
University of Tehran  
Tehran, Iran  
rhashemi@ut.ac.ir

Abstract— Anomaly based intrusion detection is a critical research area, since it does not require any prior knowledge of attack signatures in order to detect them. Typically, intrusion detection systems (IDSs) consider a fix learning model, and generally do not contemplate any memory constraint. These assumptions may not be valid in real world scenarios, where the learning model may evolve and the available memory is restricted. In this paper a cluster based method has been proposed which uses Discrete Cosine Transform (DCT) to build an effective and compact model of normal data. This model is then used for intrusion detection in environments where there is a concept drift. The proposed method has been evaluated with the KDD99 intrusion detection dataset. Simulation results indicate the superiority of the proposed method when compared to a state-of-the-art IDS.

Keywords-component: IDS, Novelty detection, DCT.

I. INTRODUCTION

The design and development of an effective technique which can detect intruder packets is a significant challenge. Intrusion Detection Systems (IDSs) have become the main mechanism to provide a secure network for organizations. IDSs try to identify unauthorized abnormal activities in a particular system. IDSs can be categorized into two main categories: misuse based IDS and anomaly based IDS [1]. A misuse based IDS uses labeled data to build the normal model. As a result it can only detect known attacks. On the other hand, an anomaly based IDS builds its model based on the normal packets and try to discriminate the attacks from normal packets without considering any predefined pattern. Hence, it is able to detect new attacks as well.

An attack in the network can be considered as a novel concept. Moreover, some novelty detection methods work based on one-class classification [2]. These methods build their initial model using a specific or normal data. Hence, a novelty detection method can be applied to intrusion detection as an anomaly based method. But applying a novelty detection technique in the network traffic environment should consider some new challenges. First, network packets arrive at a high speed and high volume, making it impractical to store all the data in memory. Second, the normal patterns change with time. This problem has been addressed by concept drift detection in the literature [3]. Another common challenge is the lack of labeled data in real world.

In this paper, a new IDS scheme has been proposed that uses an effective and compact knowledge about the normal data using Discrete Cosine Transform (DCT). The proposed method is an incremental approach which can work under mentioned challenges to detect and handle concept drift.

The rest of the paper is organized as follows. Section II introduces related works that have addressed the mentioned challenges in IDS. The proposed scheme is introduced in section III. Section IV presents the experimental results. Finally the paper concludes in section V.

II. RELATED WORKS

This section reviews some of the recent works on intrusion detection. We first review works that use a domain transform and then recent researches that have addressed some or all of the mentioned challenges.

There are many IDS methods that use a domain transform due to its potential to find novel and unknown intrusions [4]. The most popular transform is Discrete Wavelet Transform (DWT) [4,5,6,7]. Some other papers have also applied DCT [8].

In [4] a complete network anomaly detection approach based on DWT and the system identification theory (ARX) has been proposed. This paper introduces a traffic prediction model which measures the difference between normal and anomalous traffic. After determining anomalous traffics, it uses an outlier detection method to identify the location of each attack in the anomalous traffic individually. However, this method cannot handle concept drift.

In [8] an image based IDS has been proposed. In this study, they consider the network traffic as a sequence of images. Network attacks are detected using image and video processing techniques. It decreases the amount of stored data by detecting attacks in the DCT domain. Although it can detect the attacks effectively, but it has not considered the concept drift problem.

OLINDDA, a cluster based novelty detection method which can detect drifting concepts and novelty classes [9] has been applied to detect network attacks in [10]. OLINDDA uses distance based measures such as cluster radius to detect the normal packets and discriminate concept drift from the attacks. Same as its initial version [9], it uses distance based methods to detect attacks. The main problem
of a distance based method is that its decision is made based on whole cluster. Moreover, it assumes that the clusters have a spherical shape.

In [11] a new method to classify the network attacks has been proposed. It assumes that the underlying data distribution changes over time. It computes cohesion among unlabeled testing instances and their separation from training instances to detect novel classes. However, this work is a supervised method and requires labeled data.

In [12] an ensemble based IDS has been proposed which can detect the attacks in a concept drifting environment. It uses a self similarity measure to find changes in classifier distribution. If a new classifier differs from previous ones, this means that a concept drift has occurred. This method can handle concept drift effectively but it uses labeled datasets.

To summarize, most existing methods have not considered all the challenges that exist in an IDS environment. To the best of our knowledge, [10] is the only work that has addressed all the challenges. As it will be explained in the coming sections, the proposed method uses deviation instead of distance to detect the attacks and the cluster energy instead of just its center.

III. PROPOSED APPROACH

As mentioned above, this paper applies the previously introduced DETECTNOD (DiscrETE Cosine Transform based NOvelty and Drift detection) method [13] for intrusion detection. DETECTNOD is a cluster based method which can detect novel classes as well as concept drift. It consists of two main phases, in the first phase it tries to build an effective and efficient generative model from the normal data, and then in the second phase, it not only discriminates normal data from attacks, but is also able to detect and deal with concept drift. The proposed method uses deviation instead of distance to detect the attacks and the cluster energy instead of just its center.

A. Phase 1: Constructing the generative model

In the first phase, using the K-means algorithm, normal clusters are built. As mentioned before, it is not practical to maintain all the clustered data in the memory. One solution is to extract necessary information from the normal clusters. Moreover, it is better to ascertain the label of an instance based on its deviation on the neighborhoods of the closest cluster instead of whole cluster.

Therefore, DETECTNOD works as follows:

First, each cluster is divided into sub-clusters such that the closest sub-cluster to an instance is an approximation of the nearest K-neighbor set. The number of sub-clusters is computed as follows:

$$M_i = \frac{N_{C_i}}{K-\text{Neigh}} \quad i: 1, \ldots, \text{Number of clusters} \quad (1)$$

where K-neigh is the number of neighbor sets. Its value is determined by the user. $N_{C_i}$ is the number of examples in cluster $C_i$.

B. Phase 2: Online and unsupervised intrusion detection

1) Detecting the normal packets

In this phase, the network packets are divided into equal sized blocks. When a new block arrives, each packet is analyzed individually. DETECTNOD uses the obtained model in the previous phase to detect the normal packets. It holds packets that are not determined to be normal in a memory which is called “unknown memory”.

To discriminate the normal packets from the attacks, first, the closest sub-cluster to the instance is found. This represents an approximation of the instance’s nearest neighborhood set.
DETECTNOD measures the deviation of the instance from it. If this deviation exceeds from a predetermined threshold, the instance is considered as an unknown packet, otherwise it is a normal packet. The following expression illustrates it:

\[ Dev_P = \left( \sum_{i=1}^{NumDC} (OldKNeighbors Model_i - NewKNeighbors Model_i)^2 \right) \]

where \( Dev_P \) shows the amount of deviation, \( OldKNeighbors Model \) is the closest sub-cluster and \( NewKNeighbors Model \) is the closest sub-cluster after joining the instance.

The threshold can be defined as the maximum deviation in a normal cluster that can still be considered as normal. If the distribution of sub-clusters is Gaussian, we use \( Dev_{P_{\mu+3\sigma}} \) (where \( \mu \) is the mean of the closest sub-cluster and \( \sigma \) is its standard deviation) as the threshold; otherwise the threshold is defined as the value of \( Dev_P \), where \( P \) is the farthest point in the sub-cluster.

2) Discriminating the drifting concepts from attacks

The unknown memory consists of two possible packets: Concept drift packets that are new trends in the normal model, and Attacks that demonstrate an abnormal behavior.

At the end of each block, the unknown packets are clustered using K-means. The obtained clusters are interpolated and then DCT transformed to build their unknown generative models.

In order to discriminate concept drift from the attacks, DETECTNOD computes the distance between the unknown clusters and the normal sub-clusters. It is done by Euclidean distance which is an effective distance measure. The minimum distance belongs to the closest sub-cluster to each unknown cluster. If this distance is more than a threshold, it is considered as an attack behavior; otherwise it is a concept drift. We define the threshold as follows: The maximum distance between the sub-clusters in normal clusters represents the maximum acceptable dissimilarity. Hence, this distance can be considered as the threshold. Fig.3 illustrates this threshold visually (see “Dist” in fig.3). In this figure \( S_1 \) is considered as a concept drift while \( S_2 \) is an attack concept.

3) Concept drift management

The detected new normal trends should be learned for detecting the attacks in the future. On the other hand, the memory constraints do not allow extending the normal model. So using a time stamp strategy, DETECTNOD substitutes the detected new normal trends with the oldest sub-cluster. Each sub-cluster has a counter which determines its last access time. The oldest normal pattern is the least used sub-cluster in the normal detection phase.

IV. EXPERIMENTAL RESULTS

DETECTNOD has been evaluated with a common dataset in intrusion detection community. In the following we discuss the dataset; the obtained rates and the effectiveness of our concept drift management strategy.

A. Dataset

We have performed the experiments on the KDD99 network intrusion detection datasets [16]. It includes four main types of intrusions: DOS (391458 examples), Probe (4107), U2R (52) and R2L (1126). This dataset consists of 41 features, where only four features are nominal. Nominal attributes have been transformed into numerical attributes using binary encoding.

Same as [10], we set 97277 packets of normal packets as a training data. The testing data consists of the remaining 9727 normal packets plus the attack data.

B. Performance Measures

We use three performance measures for evaluation:

1) False-abnormal error rate (eFAbn)

This measure computes the effectiveness of the method for normal detection [10]:

\[ eFAbn = \frac{FAbn}{Nor} \]
where FAbn is the number of normal packets which were incorrectly determined to be an attack, and Nor is the number of normal packets in the testing set.

2) False-normal error rate (eFNor)

The second measure concentrates on the efficiency of the method for attack detection [10]:

\[
eFNor = \frac{FNor}{Abn}
\]

where FNor is the number of attacks which were incorrectly determined to be normal and Abn is the number of attacks in the testing set.

3) Total cost of errors

The two mentioned measures focus on the normal or the attack separately. The third measure is an asymmetric measure which computes the efficiency of the method in both attack and normal detection.

According to [17] “The cost of a false negative error is much higher than that of a false positive error because an organization may suffer from various security incidents compromising confidentiality, integrity, and availability when not detecting real attacks.”

Using this measure we can apply these asymmetric costs for evaluation. This measure is described as follows [17]:

\[
\text{Total Cost} = f(\text{false negative error}, \text{false positive error}) = \sum_{i=1}^{n} W_i \times C_i
\]

where \( W_i \) is the weight for each \( C_i \), and \( C_i \) is the costs for each error \( i \).

The weight for each \( C_i \) depends on the characteristics of an organization. Same as [17], in this experiment we assumed the cost of false abnormal error rate is at least five times greater than the false normal error rate. Therefore:

\[
\text{Total Cost} = (\text{False Abnormal Error Rate}) + 5 \times (\text{False Normal Error Rate})
\]

C. Comparison method and obtained rates

To investigate the efficiency of DETECTNOD, it has been compared with OLINDDA, as one of the most recent and most similar method to ours, which has been proposed in [9] and later has been used for intrusion detection in [10].

As mentioned before, DETECTNOD has several thresholds that should be predetermined experimentally. For our simulations they have been determined as follows: i) Block Size = 5k, ii) K-neigh = 500, iii) NumDCT = 15.

We compute the performance measures for DETECTNOD in two ways: DETECTNOD without update, in this way DETECTNOD can just detect the attacks and cannot handle concept drift. DETECTNOD with update, it can handle concept drift as well as attack detection. Table I shows the obtained results at the end of experiments.

As table I presents, DETECTNOD works more effectively in comparison with OLINDDA.

<table>
<thead>
<tr>
<th>Method</th>
<th>Performance Measures</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>eFAbn</td>
</tr>
<tr>
<td>OLINDDA (without update)</td>
<td>0.18</td>
</tr>
<tr>
<td>DETECTNOD (without update)</td>
<td>0.16</td>
</tr>
<tr>
<td>DETECTNOD (with update)</td>
<td></td>
</tr>
</tbody>
</table>

The “Total Cost” column of table I declares that DETECTNOD has a significant difference with OLINDDA in terms of cost savings.

Table II shows the distribution of the real classes at the end of simulations. In this table “NorClass” indicates the rate of packets that have been assigned to the normal class using the model. In the normal class a high “NorClass” value indicates high normal detection rate but in other classes which are not normal, a high value means that the method has assigned the attacks as normal incorrectly. “ExtNovClass” is the rate of packets which have been considered as attacks and “Unknown” represents the rate of packets which have stayed in the unknown memory at the end of the simulations.

<table>
<thead>
<tr>
<th>No. Examples</th>
<th>Class</th>
<th>Method</th>
<th>NorClass</th>
<th>ExtNovClass</th>
<th>Unknown</th>
</tr>
</thead>
<tbody>
<tr>
<td>9727</td>
<td>Normal</td>
<td>DETECTNOD</td>
<td>0.83</td>
<td>0.16</td>
<td>0.00</td>
</tr>
<tr>
<td>391458</td>
<td>DOS</td>
<td>DETECTNOD</td>
<td>0.007</td>
<td>0.993</td>
<td>0.00</td>
</tr>
<tr>
<td>4107</td>
<td>Probe</td>
<td>DETECTNOD</td>
<td>0.24</td>
<td>0.76</td>
<td>0.00</td>
</tr>
<tr>
<td>1126</td>
<td>R2L</td>
<td>DETECTNOD</td>
<td>0.12</td>
<td>0.88</td>
<td>0.00</td>
</tr>
<tr>
<td>52</td>
<td>U2R</td>
<td>DETECTNOD</td>
<td>0.13</td>
<td>0.87</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table II indicates that DETECTNOD outperforms OLINDDA significantly in detecting the several types of attacks. In this table, we marked the technique that has the best obtained rates for each class type by a shaded area.

Table III compares the “NorClass” value for the Normal class in table II with accuracy in the best result of KDD99 and “ExtNovClass” of the attack classes in this table with the obtained detection rates in KDD99 competition.

Although it is not fair to compare DETECNOD as a one-class classification with these results but Table III asserts the superiority of DETECTNOD in three types of attacks. In the remaining two states (Normal class and Probe class) although DETECNOD does not work better but the results are comparable.
TABLE III COMPARISON WITH THE BEST RESULTS IN KDD99

<table>
<thead>
<tr>
<th>Method</th>
<th>Normal</th>
<th>Dos</th>
<th>Probe</th>
<th>R2I</th>
<th>U2R</th>
</tr>
</thead>
<tbody>
<tr>
<td>DETECTNOD</td>
<td>83</td>
<td>99.7</td>
<td>76</td>
<td>88</td>
<td>87</td>
</tr>
<tr>
<td>KDD99</td>
<td>99.5</td>
<td>97.1</td>
<td>83.3</td>
<td>3.4</td>
<td>13.2</td>
</tr>
</tbody>
</table>

D. Efficiency of the concept drift management strategy

In this section we show the efficiency of DETECTNOD in dealing with concept drift during the testing phase.

DETECTNOD is an unsupervised method, so it does not have any information about the real labels of the packets, consequently it should learn based on the labels which are predicted during the testing phase. Fig. 4 and 5 show the effect of concept drift management on eFAbn and eFNor.

![Figure 4. Effect of concept drift management on eFAbn](image)

![Figure 5. Effect of concept drift management on eFNor](image)

Generally, DETECTNOD with update works more efficiently, but sometimes it demonstrates some inefficiency. (See the value of eFAbn in fig.4 and eFNor in fig.5 between 22\textsuperscript{th} to 30\textsuperscript{th} block). In this state, concept drift has been detected incorrectly, so applying detected trends in the model is misleading. But using our method after awhile, DETECTNOD replaces them with the new detected trends correctly.

V. CONCLUSION

In this paper, an unsupervised novelty detection method has been applied for detecting network attacks. The proposed method is capable of dealing with concept drift. We applied DCT to build a condensed model which can detect the attacks effectively. Moreover, using DCT coefficients the proposed method could handle concept drift during the simulation.

Experimental results indicate that the method is able to outperform existing methods in terms of both error rate and attack costs.

REFERENCES


