LSC-CHAM: A New Algorithm for Intrusion Detection Systems

Fadi Tashtoush                Rehab Duwairi
fadimast@just.edu.jo          rehab@just.edu.jo

Department of Computer Information Systems
Jordan University of Science and Technology
Irbid, Jordan,
Fax: +962-2-7095046

Abstract

This paper introduces a new classifier, called LSC-CHAM, for intrusion detection systems. This classifier is based on the Local Sparsity Coefficient-mine algorithm (referred to as LSC-mine) [AE2004] and the CHAMELEON algorithm [KHK98]. The LSC-mine and the CHAMELEON were integrated to produce a new classifier that outperforms either of them when taken individually. Experiments were performed on a sample obtained from DARPA 98 dataset; this sample consists of 50,000 connections; about 10% of these connections are infected with intrusions. Conducted experiments show that the detection rate of LSC-CHAM is 75.12% and the false alarm rate is equal to 2.468%. The LCS-CHAM classifier performed well when compared with other known classifiers such as the SVM classifier.

Keywords: Intrusion Detection, Data Mining, Outlier Detection, Network Security.

1- Introduction

The intrusion detection problem may be considered as a problem of detecting abnormal elements inside packet traces, that is recognizing abnormal patterns inside the network flow. According to this premise, developers of intrusion detection (ID) systems categorize these systems into either misuse intrusion detection systems [Ilgu92, KS94, CG2003] or anomaly detection systems [LS98, LS2000].

Misuse intrusion detection systems present an accurate performance in detecting pre-known abnormal activities; data are being labeled into either normal or intrusion; a learning algorithm is used to predict intrusions. These systems require updating attacks’ signatures periodically to maximize the probability of detecting novel attacks. However; it does not have the capability of deducing novel attacks, hence; anomaly ID systems were developed to overcome the shortages of misuse ID systems, anomaly ID systems use data mining techniques to detect pre-defined attacks as well as discovering novel attacks.

A main drawback of anomaly-based systems is the high number of generated false alarms due to detecting novel normal behavior. Anomaly detection systems are divided into either supervised approaches where a prior knowledge is necessary to accomplish correct classification [LS2000] or unsupervised approaches where no prior knowledge is used in detecting the intrusions [Juli2003].

This paper contributes a new anomaly-based and host-based detection technique that is called LSC-CHAM classifier; this technique merges the work and modifies some parts of the

1 Please send all correspondences to second author
local sparsity coefficient-mine (abbreviated as LSC-mine) and the CHAMELEON classifiers [AE2004, KHK98]. The first classifier is aimed at detecting anomalous behavior of the network by assigning a coefficient ratio for every connection; this ratio expresses the degree of outlier-ness of the connection. The second classifier is induced from the CHAMELEON algorithm [KHK98]; it produces a feedback generation stage to reduce as much as possible the false alarm rate incurred during the first stage.

This paper is organized as follows: section 2 highlights previous research in the intrusion detection field. Section 3 introduces the idea beneath the LSC-CHAM classifier in addition to the major phases of this classifier. Section 4 describes the conducted experiments and states a comparison between the introduced classifier and other recent algorithms used in ID systems. Section 5 includes conclusions and final remarks of this paper.

2-Literature review

2.1 Attack Taxonomies

Building eligible and reliable intrusion detection systems requires the full understanding to the nature, work and effects of the initiated attacks, therefore; numerous taxonomies have been introduced mainly to facilitate the building process. Purdue University students developed several taxonomies [KS94, Asla95, Krsu98] to classify attacks' signatures and computer vulnerabilities.

Lincoln taxonomy [Kend99, L+2000] has been considered as a turn point in this field, the relation between this taxonomy and the introduction of DARPA98 dataset makes this taxonomy as one of the most important taxonomies in the literature. It depends on the attack effect when exploited in the computer or network systems; Lincoln taxonomy comprises of four main categories; namely: Denial of Service attacks (DoS), Probing attacks, User to Root attacks (U2R), and Remote to local attacks (R2L).

A new methodology that was based on dimensions rather than singular predefined labeled classes was introduced in [HH2005]. This methodology overcomes the generality of previously addressed taxonomies. It consists of four main dimensions which are: base, target, type of vulnerabilities or exploits, and existence of payload within a defined attack dimensions.

A completely different taxonomy emphasizes that attacks might be classified depending on the defense mechanism that was carried out to discover them. A defense-centric taxonomy [KMT2004] analyzes the system calls manifestation and categorizes attacks into: foreign symbols, minimal foreign sequence, dormant sequence and non-anomalous sequences.

2.2 Intrusion Detection Systems and Techniques

ID systems are mainly classified into misuse-based or anomaly-based detection systems. STAT misuse-based systems series were invented by the University of California researchers; these systems depend on the basis of representing system situations as a set of transitions and the change from a certain situation to another is represented as a change from one transition to another [Ilgu92]. USTAT, NSTAT, NETSTAT as well other systems and languages were developed under the same concept [Ilgu93, EVK2002, VK98].

SRI laboratories established three anomaly-based ID systems; IDES [LG+92], NIDES [JV94, AF+95] and EMERALD [PN97]; these systems depend mainly on the statistical tools' components to derive the deviation between the already trained data and currently received data. IDES and NIDES are host–based ID systems, but EMERELD is a network-based ID system.

Recent ID systems have been based on data mining techniques to detect intrusions by training records to recognize normal behavior, during testing phase; novel intrusions are
learned and signatures are extracted using different methodologies. ADAM [BC+2001], ADMIT [SZ2002] and CIDS [DG2005] are some of many anomaly-based ID systems that deploy data mining classifiers into their work.

2.3 Data Mining Techniques for Intrusion Detection

Numerous data mining techniques have been applied to enrich the ID system paradigm. K-nearest neighbor conceptuality and its variant algorithms have emerged as a powerful classification model in the DM field. A k-nearest neighbor algorithm is addressed in the intrusion detection paradigm through a model [RRS2000] that computes the distance between incoming data streams and other existing points, according to the calculated distance. The k-nearest neighbor algorithm ranks the streams and determines the position of the new stream according to these ranks. The aim from this work is to discover outliers, which are considered as intrusions.

Local outlier factor algorithm [B+2000] and unsupervised support vector machine abbreviated as SVM [SP+2001] algorithm are implemented to discover intrusions [LO+2003], this research compares these algorithms with k-nearest neighbor [RRS+2000], conducted results observes that both local outlier factor and SVM outperforms other algorithms, SVM has a better performance than local outlier factor algorithm despite its difficult implementation.

3. LSC-CHAM Classifier: a New Intrusion Detection System

This work contributes a new classifier that forms an anomaly-based and host-based intrusion detection system. This classifier consists of two main stages; the first stage applies the local sparsity ratio-mine algorithm (LSC-mine) [AE2004] to discover outlier connections. These connections are then passed to a second phase, which aims at defining attacks' types and extracting attack signatures by performing the CHAMELEON hierachal clustering algorithm. The following subsections demonstrate the stages of LSC-CHAM classifier.

3.1 The Preprocessing Stage

The preprocessing stage starts when an input file (its extension is .tcpdump) is sniffed by the Ethereal software 0.10.11; this software is released under the GNU general public license. This stage induces all necessary information required in the feature construction process; the information of each packet is called intrinsic features, which are subsequently utilized to derive the DARPA 98 dataset features. These intrinsic features include source IP address, destination IP address, service, arrival time, duration, packet length, capture length, header length, total length, reserved bit, do not fragment bit, protocol type, header checksum, retransmission flag, sequence number, acknowledgment number, source port, destination port, CWR bit, urgent bit, acknowledgment bit, push bit, reset bit, SYN error bit, and finish bit.

3.2 Feature Extraction Process

The feature extraction stage extracts all relevant information from the intrinsic features; there are three main types of features; content-based, time-based and connection-based features [LS98, L+2000]. Content-based features correspond to the information extracted from every individual connection; this information helps in determining U2R and R2L attacks [LO+2003, LS98]. On the other hand, time-based features provide suitable information about the incoming connections within the last $\varepsilon$ seconds. This category of features improves the detection of many DoS and probing attacks [LO+2003, LS98]. Finally, connection-based
features differ from time-based features only in the extraction criteria - instead of extracting features for incoming connections within the last $\gamma$ seconds, the system extracts features for the last $\gamma$ connections. Usually $\gamma$ is equal to 100 connections.

The extraction process is carried out through the following steps:

1. Convert the set of incoming packets into connections.
2. Extract features and store them on external files.
3. Normalize the features using the following formula:

$$v' = \frac{v - \min_A}{\max_A - \min_A} \cdot (newMax_A - newMin_A) + newMin_A$$

Where:

- $v'$ is the normalized value of $v$.
- $\min_A$ and $\max_A$ are the lowest and the highest values of attribute A in the original domain.
- $newMin_A$ and $newMax_A$ are the new lowest and the highest values of attribute A in the new (mapped to) domain respectively.

4. Reduce the dimension of extracted features. Dimension reduction is accomplished in this work by following performance ranking technique; this technique concludes that the number of the most important features is equal to 17 features out of 41 features extracted from the DARPA 98 dataset.

### 3.3 LSC-mine stage steps

LSC-mine algorithm detects outliers through assigning a value called Local Sparsity Coefficient to every connection in the space; this value expresses the density of the connection's area, the following steps denote the major steps of this algorithm:

1. Compute the actual distances between every pair of connections using the Euclidean distance (formula 3.2)

$$\text{Dist} \ (p, q) = \sqrt{\sum_{i=1}^{N} (p_i - q_i)^2} \ ----------------------------- (3.2)$$

Where N is the number of features.

2. Compute the $k$-distance value for every connection; this value is equal to the maximum distance between the most minimum $k$ connections distances with the current connection. All connections' labels are stored in a set named as $k$-nearest neighbor and the cardinality of this set is computed.

3. Compute the Local sparsity ratio (LSR) for every connection using formula 3.3. This formula denotes the degree of density around the connection.

$$\text{LSR} = \frac{|N_k(p)|}{\sum_{o \in N_k(p)} \text{distof} N_k(p)} \ ----------------------------- (3.3)$$
4. Compute the pruning factor (PF) using the formula 3.4 for every connection; this value helps reduce the number of examined connections, which have a potential evidence of not being an outlier connections.

\[ LSR = \frac{\sum |N_K(p)|}{\sum \text{distof} N_K(p)} \]  

...(3.4)

When \( LSR(p) < PF(p) \) then the connection \( p \) has a strong evidence of not being an outlier, therefore, remove \( p \) from all k-nearest neighbor candidate set.

5. Compute the Local Sparsity Coefficient (LSC) to decide the degree of outlier-ness of the current connection, formula 3.5 shows the rule of LSC ratio.

\[ LSC_K(p) = \sum_{o \in N_K(p)} \frac{LSR_K(o)}{N_K(p)} \]  

...(3.5)

6. Compute the delimiter ratio (R) that measures the deviation of current LSC ratio from previous values of LSC ratios.

\[ R = \frac{\sigma(LSC_{values}(C_{1 \rightarrow p}))}{\sigma(LSC_{values}(C_{1 \rightarrow p-1}))} \]  

...(3.6)

The connections with the highest delimiter ratio correspond to the most possible outliers.

3.4 The CHAMELEON and Feedback Generation Stage

The CHAMELEON and feedback generation stage executes the following phases:

1. Drawing the k-nearest neighbor graph phase:
   a. Decide the value of \( k \); this number carries out the cluster’s density degree.
   b. An edge is drawn between two points \( p \) and \( q \) that corresponds to the distance between them; the purpose of this edge is to indicate that \( q \) is one of the nearest neighbors of \( p \).
   c. An adjacency matrix is created for further computations.

2. Graph partitioning phase.
   The graph partitioning phase accepts the k-nearest neighbor graph and produces many sub-clusters by splitting the graph, the split method proposed in this paper is considerably changed from the original algorithm and a new graph partitioning technique is introduced; it consists of the following steps:
I. Create a maximal independent set \( N_i \); this set contains points with disconnected edges. Any point that is not included in \( N_i \) will be inserted to the complement set \( N_c \). The set \( E_c \) contains all edges that connect nodes in different domains.

II. \textbf{REPEAT}

1. If (node \( i \) is not connected to all other nodes in the sub-cluster) AND (node \( i \) has the highest degree (that have the maximum number of edges)) then
   a. Select node \( i \) from \( N_i \) and consider it as the partition center.
   b. Every node in \( N_i \) will be included in the domain of node \( i \).
   c. Remove edges that belong to \( E_c \) from the graph.
   d. Label distinct clusters.
2. Else If (node \( i \) is connected to all other nodes in the sub-cluster) then
   a. Select the node with the second highest degree.
   b. Perform steps from 1-a to 1-d.
3. Any node remains outside \( N_i \) and \( N_c \) is stored in \( N_i \)

UNTIL \( N_i \) cannot contain more than one element AND the number of nodes in each domain is less than or equal to the value of the \( k \) AND each split graph maintains the \( k \)-nearest neighbor graph.

3. \textbf{Merge sub-clusters:}

Let the number of edges linked between two clusters before starting the partitioning phase is denoted by \( EC(C_1, C_2) \); the number of cutting edges is equal to the number of edges existed in the original \( k \)-nearest neighbor graph but have been eliminated during the partitioning phase (this value denoted by \( ES(C_1, C_2) \)); total weight denoted as weight \( (E_c) \) of any set of edges is equal to the summation of individual weights; the average weight \( \text{avg}(E_c) \) is equal to the summation of individual weights divided by the number of edges in the set. The merge phase consists of the following steps:

a. Compute relative inter-connectivity (IR) between any two sub-clusters.

\[
IR(C_1, C_2) = \frac{\text{weight}(EC(C_1, C_2))}{\text{weight}(EC(C_1)) + \text{weight}(EC(C_2))} \quad \text{……..(3.7)}
\]

b. Compute the inter-closeness (RC) between any two sub-clusters:

\[
RC(C_1, C_2) = \frac{\text{avg}(EC(C_1, C_2))}{\#V(C_1) \cdot \frac{\text{avg}(ES(C_1))}{\#V(C_1 \cup C_2)} + \#V(C_2) \cdot \frac{\text{avg}(ES(C_2))}{\#V(C_1 \cup C_2)}} \quad \text{……(3.8)}
\]

c. For all pairs, the merging procedure computes the following function:

\[
f(C_1, C_2) = IR(C_1, C_2)^\alpha \cdot RC(C_1, C_2) \quad \text{………..(3.9)}
\]

\( \alpha \) is a user defined parameter, when \( \alpha > 1 \) then the user is interested in merging clusters with high connectivity despite the closeness of the two clusters. In
contrast, if $\alpha < 1$ then the user tends to merge clusters that are close to each other.

4. Feedback generation process phase:

After the merge phase is completed, any unmerged connections (including the ones that form small sub-clusters with very low growth factor) are considered as outliers and consequently treated either as a false alarm connection or a connection corresponding to a completely new attack type. These connections constitute the input of the feedback generation process phase; connections are re-weighted by reducing the features’ weights by a constant ratio. Depending on the level of every feature, distances are recalculated and LSC-mine as well as CHAMELEON stags are repeated until the reduction of weights reaches a specified threshold.

4. Experiments and Result Analysis

To prove the eligibility and competence of the proposed work, several experiments have been conducted on DARPA 98 dataset for ID systems prepared by MIT institution. The two main investigated parameters are the detection rate and false alarm rate.

4.1 DARPA 98 Dataset Description

IST group of Lincoln laboratories in MIT introduces the first comprehensive dataset for ID systems named as DARPA 98 dataset. This dataset was collected from a simulated network for US air force (LAN), the dataset is comprised of 9 weeks sniffed tcpdump data.

The dataset is infected by about 300 attacks; the number of attack types is 32 incurred in 7 different scenarios. The creators of DARPA 98 decided to consider the data of the first seven weeks as the training data and the data of the final two weeks are the testing data.

4.2 Experiments Description

Experiments were conducted on AOpen machines with 3200 MHz speed, 1 GB RAM, 1 MB cash memory and WINDOWS XP SP2. A sample of 50,000 connections is selected from the first five weeks data and is used in all experiments; about 10% of these connections are infected with 20 different intrusions from all types, the number of infected connections is equal to 1533.

4.3 LSC-mine Stage Experiments

4.3.1 Time Interval Specification Experiments

Detection and false alarm rates were measured for connections processed at four different time intervals, the time interval that will yield the highest detection rate and considerably low false alarm rate (less than 4%) is going to be adopted in the rest of the detection process. Experiments show that the detection rate reaches its highest value (76.29%) when the time interval is 4 seconds. For the same time interval (4 seconds) the false alarm rate reading its lowest value (3.702%), therefore, the 4-seconds interval will be adopted in this paper. Figure 1 depicts the result of these experiments. The value 7.5% was selected for the minpts because it satisfies a desirable tradeoff between high detection rates and low false alarm rates.
4.3.2 Minpts Specification Experiments

Four experiments were conducted to determine the percentage of minpts within 4 seconds time interval; minpts percentages range from 2.5% to 10%. Figure 2 shows how detection and false alarm rates were varied as the percentage of minpts is raised.

4.4 The CHAMELEON and Feedback Generation Results

4.4.1 K-value Specification Experiments

In order to determine the k-value precisely, a training phase is held, that is; a sample of 500 connections is chosen, 25% of these connections are normal connections and the rest of these connections are infected with intrusions. Normal connections are considered as falsely detected intrusions, the aim is to find the k-value that achieves the highest detection of these false alarms.

As in minpts percentage experiments; k-value is chosen to be 2.5%, 5%, 7.5% or 10% from the total number of incoming outliers from the LSC-mine stage. Figure 3 demonstrates the changes of detection rate of false alarm as the k-value increases; it shows that the highest detection rate of false alarms is achieved when k-value is equal to 10% from the total connections.
Variation of false alarm rates against variation of K-value

As a result of introducing the CHAMELEON and feedback generation stage; some connections that were correctly detected in the first stage were mis-clustered. Figure 1 also shows the decrease in the detection rate for different k-values. In this figure, the highest decrease of the detection rate (no more than 1%) is incurred when K-value is equal to 2.5%, this decrease rate is tolerable in terms of overall detection rate performance.

4.4.2 Feedback Generation Process Experiments

At 4 seconds time interval and 7.5% minpts, the number of detected connections is 3916 connections and the number of false alarms equals 1851 connections, the overall entering connections to the CHAMELEON and feedback generation stage equals 5767 connections. Table 3 highlights the false alarm rates before and after performing the CHAMELEON and feedback generation stage. The false alarm is reduced from 3.702% to 2.468%, which is a very significant improvement.

Table 3: False alarm connections and rates

<table>
<thead>
<tr>
<th></th>
<th>Number of false alarm connections before CHAMELEON stage</th>
<th>Number of false alarm connections after CHAMELEON stage</th>
<th>False alarm rate before CAMELEON stage</th>
<th>False alarm rate after CAMELEON Stage</th>
</tr>
</thead>
<tbody>
<tr>
<td>First week</td>
<td>184</td>
<td>110</td>
<td>0.061333333</td>
<td>0.036666667</td>
</tr>
<tr>
<td>Second week</td>
<td>169</td>
<td>95</td>
<td>0.056333333</td>
<td>0.031666667</td>
</tr>
<tr>
<td>third week</td>
<td>518</td>
<td>354</td>
<td>0.043166667</td>
<td>0.0295</td>
</tr>
<tr>
<td>Forth week</td>
<td>400</td>
<td>295</td>
<td>0.033333333</td>
<td>0.024583333</td>
</tr>
<tr>
<td>Fifth week</td>
<td>580</td>
<td>380</td>
<td>0.029</td>
<td>0.019</td>
</tr>
<tr>
<td>Average false alarm rate</td>
<td></td>
<td></td>
<td>0.03702</td>
<td>0.02468</td>
</tr>
</tbody>
</table>

The gain of reducing the false alarm rate came at the cost of decreasing the detection rate by no more than 1%. To sum up, the developed intrusion detection technique achieves 75.12% detection rate with 2.468% false alarm rate, which has not been achieved by earlier work in ID systems.

4.5 A Comparison between the LSC-CHAM and Recent ID Techniques.

This section introduces a comparison between the results of four previously proposed techniques and LCS-CHAM classifier; these techniques are local outlier factor technique (abbreviated as LOF) [B+2000], Support vector machine technique (abbreviated as SVM) [SP+2001], K-nearest neighborhood technique (abbreviated as K-NN algorithm) [RRS2000]...
and Mahalanobis distance-based technique. A comparison between the above four techniques using a sample of DARPA 98 dataset was carried out in [LO+2003]. The comparison reported in this work concentrates on assessing detection rate only because false alarm rate was fixed to 2% in previous researches [LO+2003].

Figure 4 shows that SVM has the highest detection rate although it has the largest false alarm rate (larger than 4%), which is completely unacceptable. Excluding SVM technique, because it has a high false alarm rate, the work reported in this paper contributes the highest detection rate (75.12%) when compared to LOF (68%), K-NN (73.7%) and Mahalanobis (57.9%). The main drawback of the LSC-CHAM technique is its high time complexity because of computing the distances between every two connections in the LSC-mine stage; however, computing distances is inherited from the LSC-mine algorithm [AE+2004].

Figure 4: A comparison of overall detection rates

5. Conclusions and Final Remarks

The work reported in this paper contributes a new intrusion detection classifier named LSC-CHAM; this classifier consists of two main stages: the LSC-mine stage and the CHAMELEON and feedback generation process stage. LSC-mine classifier computes the LSC value to find outliers, which comprise infected connections.

The role of the CHAMELEON and feedback process becomes important when the false alarm rate has to be reduced; the CHAMELEON algorithm reforms incoming outliers from the previous stage to extract outliers of outliers. Feedback generation process starts in re-weighting connection's features several times to adjust the position of the connection. In the CHAMELEON stage, a new graph partitioning algorithm was proposed to facilitate the splitting phase of the algorithm.

A sample of 50,000 connections are selected from DARPA 98 dataset, about 10% of these connections are infected with 20 different intrusions. Two main parameters are investigated, the detection rate and false alarm rate. At the LSC-mine stage, the size of the time interval is set to be 4 seconds and the minpts percentage is set to 7.5% of the number of all connections; these numbers were justified experimentally. The measured detection rate at the end of this stage is equal to 76.29% and the false alarm rate is equal to 3.702%.

The k-value of the CHAMELEON and feedback generation process must be experimentally determined; and it was set to 2.5% of the number of received connections from the LSC-mine stage. The false alarm rate is reduced to 2.468%, which is very good. However, the detection rate is decreased to 75.12% from 76.29%, which is tolerable when the false alarm rate has to be reduced.

The results of this technique are compared to four other outlier detection techniques – LOF, K-NN, Mahalanobis and SVM techniques - the comparison shows that LSC-CHAM contributes the highest detection rate except when compared to SVM technique. However, SVM technique produces unacceptable levels of false alarm rates (larger than 4%). The LSC-CHAM produces relatively the same rate of false alarms produced by LOF and K-NN techniques.
References


