Network Intrusion Detection System (NIDS) Using Data Mining Techniques

Sudhir N. Dhage, *Member, IEEE*, and Raman R. Bane, *Member, IEEE.*

***Abstract*--This paper introduces the Network Intrusion Detection System (NIDS), which uses a suite of data mining techniques to automatically detect the attacks against computer networks and systems. While the long-term objective of NIDS is to address all aspects of intrusion detection, this paper focuses on two specific contributions: (i) an unsupervised anomaly detection technique that assigns a score to each network connection that reflects how anomalous the connection is, and (ii) an association pattern analysis based module that summarizes those network connections that are ranked highly anomalous by the anomaly detection module. Experimental results show that our anomaly detection techniques are very promising and are successful in automatically detecting several intrusions that could not be identified using popular signature-based tools. Furthermore, given the very high volume of connections observed per unit time, association pattern based summarization of novel attacks is quite useful in enabling a security analyst to understand and characterize emerging threats.**

Index Terms--**anomaly detection, association pattern analysis, local outlier factor, network intrusion detection, novel attacks, time-window based features, connection-window based features etc.**

I. Introduction

TRADITIONAL methods for intrusion detection are based on extensive knowledge of attack signatures that are provided by human experts. The signature database has to be manually revised for each new type of intrusion that is discovered. A significant limitation of signature-based methods is that they cannot detect novel attacks. In addition, once a new attack is discovered and its signature developed, often there is a substantial latency in its deployment. These limitations have led to an increasing interest in intrusion detection techniques based upon data mining [1, 2], which generally fall into one of two categories: misuse detection and anomaly detection.

In misuse detection, each instance in a data set is labeled as 'normal' or 'intrusive' and a learning algorithm is trained over the labeled data. Research in misuse detection has focused mainly on detecting network intrusions using various classification algorithms [1], association rules [1] and cost sensitive modeling. Unlike signature-based intrusion detection systems, models of misuse are created automatically, and can be more sophisticated and precise than manually created signatures. In spite of the fact that misuse detection models have high degree of accuracy in detecting known attacks and their variations, their obvious drawback is the inability to detect attacks whose instances have not yet been observed. In addition, labeling data instances as normal or intrusive may require enormous time for many human experts.

Anomaly detection algorithms build models of normal behavior and automatically detect any deviation from it. The major benefit of anomaly detection algorithms is their ability to potentially detect unforeseen attacks. In addition, they may be able to detect new or unusual, but non-intrusive, network behavior that is of interest to a network manager. A major limitation of anomaly detection systems is a possible high false alarm rate. There are two major categories of anomaly detection techniques, namely supervised and unsupervised. In supervised anomaly detection, given a set of normal data to train on, and given a new set of test data, the goal is to determine whether the test data is 'normal' or 'anomalous'. Recently, there have been several efforts in designing supervised network-based anomaly detection algorithms, and other techniques that use neural networks, theoretic measures, network activity models, etc. Unlike supervised anomaly detection where the models are built only according to the normal behavior on the network, unsupervised anomaly detection attempts to detect anomalous behavior without using any knowledge about the training data. Unsupervised anomaly detection approaches are based on statistical approaches, clustering, outlier detection schemes [3], state machines [8], etc.

This paper introduces the Network Intrusion Detection System (NIDS), which uses a suite of data mining techniques to automatically detect the attacks against computer networks and systems. While the long-term objective of NIDS is to address all aspects of intrusion detection, in this paper we present details of two specific contributions:

(i) an unsupervised anomaly detection technique that assigns a score to each network connection that reflects how anomalous the connection is, and (ii) an association pattern analysis based module that summarizes those network connections that are ranked highly anomalous by the anomaly detection module.

Sudhir N. Dhage is with the Department of Computer Engineering, Sardar Patel Institute of Technology, Andheri, Mumbai, Mumbai University, Mumbai, India. (e-mail: sudhirdhage@gmail..com).

Raman R. Bane is with the Department of Computer Engineering, Sindhudurg S.S.P.M's College of Engineering, Kankavali, Mumbai University, Mumbai, India.

II. NIDS SYSTEM

The Network Intrusion Detection System (NIDS) is a data mining based system for detecting network intrusions. Fig. 1 illustrates the process of analyzing real network traffic data using the system. The input to NIDS is data collected using data capturing unit. This tool only capture packet header information (i.e., it does not capture message content), and build one way sessions (flows). Captured data are stored in flat files. The analyst uses NIDS to analyze these in a batch mode. The reason the system is running in a batch mode is not due to the time it takes to analyze these files, but because it is convenient for the analyst to do so. Running the system on a 10-minute file takes less than 3 minutes on a typical desktop computer. Before data is fed into the anomaly detection module, a data filtering step is performed by the analyst to remove network traffic that the analyst is not interested in analyzing. For example, data filtered may include traffic from trusted sources or unusual/anomalous network behavior that is known to be intrusion free.

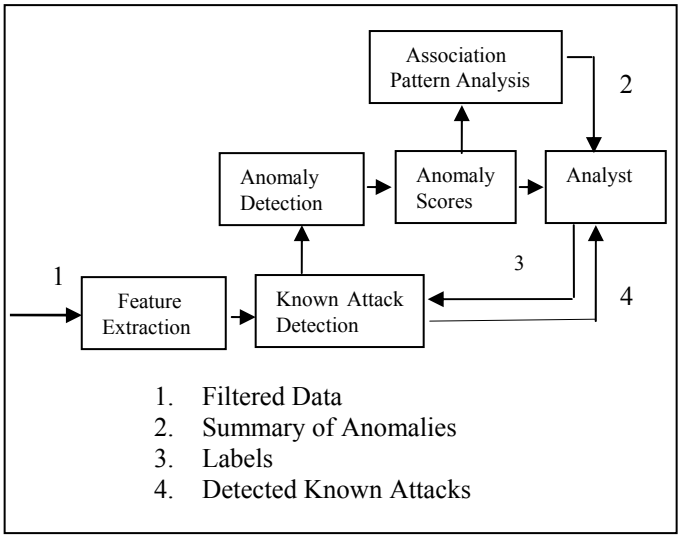

Fig 1 NIDS System

The first step in NIDS is extracting features that are used in the data mining analysis. Basic features include source and destination IP addresses, source and destination ports, protocol, flags, number of bytes and number of packets. Derived features include time-window and connection-windows based features. Time-window based features are constructed to capture connections with similar characteristics in the last seconds. These features are especially useful in separating sources of high volume connections per unit time from the rest of the traffic such as fast scanning activities. Table I summarizes the time-windows based features.

The connection-window based features are shown in Table II.

TABLE I
TIME WINDOW BASED FEATURES

Feature Name	Feature description
count-dest	Number of flows to unique destination IP addresses inside the network in the last T seconds from the same source
count-src	Number of flows from unique source IP addresses inside the network in the last T seconds to the same destination
count-serv-src	Number of flows from the source IP to the same destination port in the last T seconds
count-serv-dest	Number of flows to the destination IP address using same source port in the last T seconds

TABLE II
CONNECTION-WINDOW BASED FEATURES

Feature Name	Feature description
count-dest-conn	Number of flows to unique destination IP addresses inside the network in the last N flows from the same source
count-src-conn	Number of flows from unique source IP addresses inside the network in the last N flows to the same destination
count-serv-src-conn	Number of flows from the source IP to the same destination port in the last N flows
count-serv-dest-conn	Number of flows to the destination IP address using same source port in the last N flows

After the feature construction step, the known attack detection module is used to detect network connections that correspond to attacks for which signatures are available, and then to remove them from further analysis. For results reported in this paper, this step is not performed.

Next, the data is fed into the NIDS anomaly detection module that uses an outlier detection algorithm to assign an anomaly score to each network connection. A human analyst then has to look at only the most anomalous connections to determine if they are actual attacks or other interesting behavior.

NIDS association pattern analysis module summarizes network connections that are ranked highly anomalous by the anomaly detection module. The analyst provides a feedback after analyzing the summaries created and decides whether these summaries are helpful in creating new rules that may be used in the known attack detection module.

A. NIDS Anomaly Detection Module

NIDS anomaly detection module [9] assigns a degree of being an outlier to each data point, which is called the local outlier factor (LOF) [3]. The outlier factor of a data point is local in the sense that it measures the degree of being an outlier with respect to its neighborhood. For each data example, the density of the neighborhood is first computed. The LOF of a specific data example represents the average of the ratios of the density of the example and the density of its neighbors. To illustrate the advantages of the LOF approach, consider a simple two-dimensional data set given in Fig. 2. It is apparent that the density of cluster C_2 is significantly higher than the density of cluster C_1. Due to the low density of cluster C_1, for most examples q inside cluster C_1, the distance between the example q and its nearest neighbor is greater than the distance between the example P_2 and its nearest neighbor, which is from cluster C_2, and therefore example P_2 will not be considered as outlier.

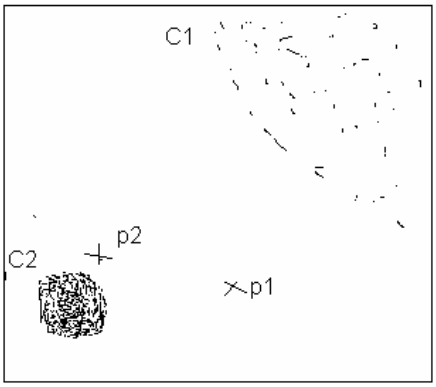

Fig. 2 2-D Outlier Example

Hence, the simple nearest neighbor approach based on computing the distances fail in these scenarios. However, the example p_1 may be detected as an outlier using the distances to the nearest neighbors. On the other hand, LOF is able to capture both outliers due to the fact that it considers the density around examples.

LOF requires the neighborhood around all data points to be constructed.

A.1) Density-Based Local Outlier Detection

For this type of outlier the density of the neighbors of a given instance plays a key role. Furthermore an instance is not explicitly classified as either outlier or non-outlier; instead for each instance a local outlier factor (LOF) is computed which will give an indication of how strongly an instance can be considered an outlier.

The following definitions are needed in order to formalize the algorithm to detect density-based local outliers:

(a) k-distance of an instance x. For any positive integer k, the *k-distance* of an instance x, denoted by *k-distance(x),* is defined as the distance *d(x, y)* between x and instance y ∈ *D* such that:

(i) for at least k instances y'∈ D – {x} it holds that
d (x, y') - d(x, y),
(ii) for at most k-1 instances y' ∈ D – {x} it holds that
d (x, y') < d(x, y).

(b) .k-distance neighborhood of an instance x. Given the k-distance of x, the k-distance neighborhood of x contains every instance whose distance from x is not greater than the k-distance; i.e. $N_{k-distance(x)}(x) = \{q \in D - \{x\} : d(x, q) \leq k - distance(x)\}$.

(c) reachability distance of an instance x w.r.t. object y.
Let k be a positive integer number. The reachability distance of an instance x with respect to the instance y is defied as-

$$reach - dist_k(x, y) = \max\{k - dist(y), d(x, y)\}.$$

(d) local reachability density of an instance x. It is given by –

$$lrd_{Minpts}(x) = \left[\frac{\sum_{o \in N_{Minpts}(x)} reach - dist_{Minpts}(x, o)}{\left| N_{Minpts}(x) \right|} \right]^{-1}$$

The *lrd* is the average *reachability distance* based on the *MinPts*-nearest neighbor of the instance x.

(e) local outlier factor of an instance x. The local outlier factor of x is defined as-

$$LOF_{Minpts}(x) = \frac{\sum_{o \in N_{Minpts}(x)} \frac{lrd_{Minpts}(o)}{lrd_{Minpts}(x)}}{\left| N_{Minpts}(x) \right|}$$

The density-based local algorithm to detect outliers requires only one parameter, *MinPts*, which is the number of nearest neighbors used in defining the local neighborhood of the instance. The LOF measures the degree to which an instance x can be considered as an outlier. Finally all the instances are ranked with respect to the maximum LOF value within the specified range. That is, the ranking of an instance x is based on:

$Max\{LOF_{MinPts}(x) / MinPtsLB \leq MinPts \leq MinPtsUB\}$

The Local Outlier Factor (LOF) algorithm [10] to detect density-based local outliers is as follows-

LOF Algorithms:-

Input:
klb and kub the lower and upper bounds of k-distance neighborhoods.
D a set of examples.
The number of top outliers.
Output:
lof a vector with local density factors
Let kdis-neighbors(D,k) return a matrix that containts the k-distances neighborhoods and their k-distances.
Let reachability(KDNeighbors) return the local reachability density of each p in D
Begin
1.lof ←NULL
2.for each k in {klb,..., kub} {
3. KDNeighbors ← kdis-neighbors(D,k)
4. lrddata reachability(KDNeighbors,k)
5. for each p in KDNeighbors
6. templof[i] ← sum(lrddata[o ∈ N(p)]) / lrddata[i])/ |N(p)|
7. lof max{lof , templof} }
8.return top(lof)
End
Output:lof

B. NIDS Module for Summarizing Anomalous Connections Using Association Rules

In the past decade, mining association rules has been the subject of extensive research in data mining. Techniques for mining association rules were originally developed to analyze sales transaction data, where analysts are interested to know what items are frequently bought together in the same transaction. In general, an association rule is an implication expression of the form X => Y, where X and Y are sets of binary features. An association rule can be used to predict the occurrence of certain features in a record given the presence of other features. For example, the rule {Bread, Butter} => {Milk} indicates that most of the transactions that contain bread and butter also involve the purchase of milk. The sets of items or binary features are known as item sets in association rule terminology

Given a set of records, the objective of mining association rules is to extract all rules of the form X =>Y that satisfy a user-specified minimum support and minimum confidence thresholds. Support measures the fraction of transactions that obey the rule while confidence is an estimate of the conditional probability P(Y|X). For example, suppose 10%of all transactions contain bread and butter, and 6 %of the transactions contain bread, butter, and milk. For this example, the support of the rule {Bread, Butter} => {Milk} is 6% and its confidence is 6%/10%= 60%. If the minimum support threshold is chosen to be 1% and the minimum confidence threshold is 50%, then this rule would be extracted by the association rule mining algorithm. In this example, the set {Bread, Butter, Milk} is also referred to as a frequent item set.

Association patterns, often expressed in the form of frequent item sets or association rules, have been found to be valuable for analyzing network traffic data [1]. These patterns can be used for the following purposes:

(i) To construct a summary of anomalous connections detected by the IDS. Often times, the number of anomalous connections flagged by IDS can be very large, thus requiring analysts to spend a large amount of time interpreting and analyzing each connection that has a high anomaly score. By applying association pattern discovery techniques, analysts can obtain a high-level summary of anomalous connections. For example, scanning activity for a particular service can be summarized by a frequent set:

$scrIP=X,\ dstPort=Y$

If most of the connections in the frequent set are ranked high by the anomaly detection algorithm, then the frequent set may be a candidate signature for addition to a signature-based system.

(ii) To construct a profile of the normal network traffic behavior in anomaly detection systems. As previously noted, an anomaly detection system requires some information about how the normal network traffic behaves in order to ascertain the anomalous connections. Association patterns can provide the necessary information by identifying sets of features that are commonly found in the normal network traffic data. For example, a Web browsing activity, (almost always on port 80 and uses the TCP protocol with a small number of packets) could generate the following frequent set:

$protocol=TCP,\ dstPort=80,\ NumPackets=3\ldots6$

In addition, association patterns generated at different time frames can be used to study the significant changes in the network traffic at various periods of time.

(iii) Recurrent patterns in normal or anomalous connections can serve as secondary features to be augmented to the original data in order to build better predictive models of the network traffic data.

C. Finding Discriminating Patterns

Eventually, the goal of mining association patterns is to discover patterns that occur regularly in the normal class or

anomaly class, but not both. To do this, we need a measure that can rank the patterns according to their discriminating power. NIDS allows the users to rank the discovered patterns according to various measures, as illustrated below.

Measures for ordering Patterns

Ratio= (c1/n1)/ (c2/n2)
Precision p = c1/ (c1+c2)
Recall r = c1/n1
F1 = (2*r*p)/(r+p)

Consider a set of features X that occurs c1 times in the anomalous class and c2 times in the normal class. Also, let n1 and n2 be the number of anomalous and normal connections in the data set. Assuming that we are only interested in finding profiles of the anomalous class, the ratio (c1/n1) to (c2/n2) would indicate how well the pattern could discern anomalous connections from normal connections. If the proportion of samples in each class is the same, i.e., n1 = n2, then the ratio measure is a monotone function of precision. Ratio or precision alone is insufficient because they often characterize only a small number of anomalous connections. In the extreme case, a rare pattern that is observed only once in the anomalous class and does not appear in the normal class will have a maximum value of ratio and precision, and yet, may not be significant. To account for the significance of a pattern, the recall measure can be used as an alternative. Unfortunately, a pattern that has high recall may not necessarily be discriminating. The F1-measure, which is the harmonic mean of precision and recall, provides a good trade-off between the two measures.

D. Grouping the Discovered Patterns

It is worth noting that some of the extracted patterns can describe a similar set of anomalous connections. For example, a probe or scan may give rise to multiple patterns that are very similar to each other (e.g., these patterns may involve the same source IP address and port number, but different destination IP addresses). Thus, it is useful to group together the related patterns before presenting them to the analysts. The overall architecture of our association analysis module is shown in Fig. 3.

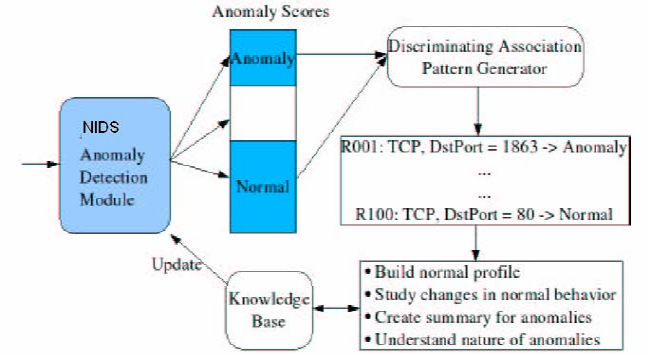

Fig. 3 NIDS Association Analysis Module

As previously noted, NIDS would use the anomaly scores of the connections to determine whether a connection belongs to the normal or attack class. In our experiments, we choose connections that have the top 10% anomaly score to be the anomaly class and the bottom 30% anomaly score to be the normal class. Connections with intermediate anomaly scores are ignored. Next, the association pattern generator is applied to each class and the patterns are ranked according to the various measures described above. The extracted patterns can be used to create summaries and profiles for normal and anomalous connections. Once the profile for the attack class is created, a follow-up analysis is often performed to study the nature of the anomalous connections. A typical follow-up analysis involves connecting via telnet to the suspected computer at the specific port and examining the returned information. Another possibility of analyzing the suspected computer is to start capturing packets on that machine at the particular port and to investigate the contents of the packets.

III. EVALUATION OF ATTACK SUMMARIES ON NETWORK DATA

In the following, we present several association patterns found by the NIDS summarization module

Example 1

srcIP=IP1, dstPort=80, Protocol=TCP, Flag=SYN, NumPackets=3, NumBytes=120. . . 180 (c1=256, c2=1)

srcIP=IP1, dstIP=IP2, dstPort=80, Protocol=TCP, Flag=SYN, NumPackets=3, NumBytes=120. . . 180 (c1=177, c2=0)

The first rule indicates that the source of the anomalous connections originates from IP1, the destination port is 80, the protocol used is TCP with tcpflags set to SYN, the number of packets is 3, and the total number of bytes is between 120 and 180. Furthermore, this pattern is observed. 256 times (c1 = 256) among the anomalous connections and only once (c2=1) in the normal connections. Therefore, it has a high ratio and precision, which is why it is ranked among the top few patterns found by the system.

At first glance, the first rule indicates a Web scan since it appears mostly in the anomaly class with a fixed source IP address but not with a fixed destination IP address. However, the second rule suggests that an attack was later launched to one of the specific machines since the pattern originates from the same source IP address but has a specific destination IP address and covers only anomalous connections. Further analysis confirms that a scan has been performed from the source IP address IP1, followed by an attack on a specific machine that was previously identified by the attacker to be vulnerable.

Example 2

dstIP= IP3, dstPort=8888, Protocol=TCP (c1=369, c2=0)

dstIP=IP3,dstPort=8888,Protocol=TCP,Flag=SYN(c1=291, c2=0)

This pattern indicates a high number of anomalous TCP connections on port 8888 to a specific machine.

Example 3

srcIP=IP4, dstPort=27374, Protocol=TCP, Flag=SYN, NumPackets=4, NumBytes=189200 (c1=582, c2=2)

srcIP=IP4, dstPort=12345, NumPackets=4, NumBytes= 189200 (c1=580, c2=3)

srcIP= IP5, dstPort=27374, Protocol=TCP, Flag=SYN, NumPackets=3, Num-Bytes=144 (c1=694, c2=3)

The patterns above indicate a number of scans on port 27374 (which is a signature for the Sub Seven worm) and on port 12345 (which is a signature for the NetBus worm).

IV. CONCLUSION AND FUTURE WORK

The overall objective of the NIDS is to develop on-line and scalable data mining algorithms and tools for detecting attacks and threats against computer systems and it giving satisfactory results compared to standard signature based tools. Improvement is necessary because of following challenges. First, data generated from network traffic monitoring tends to have very high volume, dimensionality and heterogeneity, and there is a need for high performance data mining algorithms that will scale to very large network traffic data sets. Second, network data is temporal (streaming) in nature, and development of algorithms for mining data streams is necessary for building real-time intrusion detection system. Third, cyber attacks may be launched from several different locations and targeted to many different destinations, thus creating a need to analyze network data from several network locations in order to detect these distributed attacks. Therefore, development of a cooperative and distributed intrusion detection system for correlating suspicious events among multiple participating network sites to detect coordinated attacks will be one of the key components.

V. REFERENCES

[1] Daniel Barbara, Ningning Wu and Sushil Jajodia Detecting novel network intrusion using bayes estimators. In Proceedings of First SIAM Conference on data mining Chicago, 2001.

[2] Eric Bloedorn, Alan D. Christiansen, William Hill, Clement Skorupka, Lisa M. Talbot, and Jonathan Tivel. Data mining for network intrusion detection: How to get started. Technical report, The MITRE Corporation, 2001.

[3] Markus Breunig, Hans-Peter Kriegel, Raymond T. Ng, and J¨org Sander. Lof: Identifying densitybased local outliers. In *Proceedings of the ACM SIGMOD Conference*, Dallas, TX, 2000.

[4] M. Joshi and V. Kumar. Credos: Classification using ripple down structure (a case for rare classes). In P*roceedings of 19th International Conference on Data Engineering*, Bangalore, India, 2003.

[5] Mahesh V. Joshi, Ramesh Agarwal, and Vipin Kumar. Mining needles in a haystack: Classifying rare classes via two-phase rule induction. In *Proceedings of ACM SIGMOD Conference on Management of Data*, Santa Barbara, CA, 2001.

[6] Richard P. Lippmann and Robert K. Cunningham. Improving intrusion detection performance using keyword selection and neural networks. *Computer Networks (Amsterdam, Netherlands: 1999)*, 34:597–603, 2000.

[7] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the ACM SIGMOD Conference*, pages 427–438, Dallas, TX, 2000.

[8] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang, and S. Zhou. Specification based anomaly detection: A new approach for detecting network intrusions. In *ACM Conference on Computer and Communications Security*, 2002.

[9] Aleksander Lazarevic, Levent Ert¨oz, Aysel Ozgur, Vipin Kumar, and Jaideep Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of Third SIAM International Conference on Data Mining*, San Francisco, 2003.

[10] Elio Lozano, Edgar Acu˜na. Parallel algorithms for distance-based and density-based outliers, *Data Mining, Fifth IEEE International Conference, Publication on 27-30 Nov. 2005.*

VI. BIOGRAPHIES

Sudhir N. Dhage was born in Maharastra, India, on May 25, 1971. He graduated from the Govt. College of Engineering, Amravati, at University of Amravati, India. He is working as an Assistant Professor in Computer Engineering Department., Sardar Patel Institute of Technology, Andheri, Mumbai, at University of Mumbai, India. His research interests include Networking, Network Security, Parallel Processing and Language Processing. He is a student counselor of Computer Society of India, Mumbai Chapter, Mumbai. He is life member of CSI. Contact him at sudhirdhage@gmail.com.

Raman R. Bane is working as an Assistant Professor in Computer Engineering Department, Sindhudurg S.S.P.M's College of Engineering, Kankavali, at University of Mumbai, India. His research interests include Networking, Network Security, System Programming.