Journal of Convergence Information Technology
Volume 5, Number 1, February 2010

Anomaly Detection Analysis of Intrusion Data using Supervised & Unsupervised Approach

Prasanta Gogoi, B Borah and D K Bhattacharyya
Department of Computer Science & Engineering,
Tezpur University, Napaam, Tezpur, India
E-mail: {prasant, bgb, dkb} @tezu.ernet.in
doi: 10.4156/jcit.vol5.issue1.11

Abstract

Anomaly based network intrusion detection (ANID) is an important problem that has been researched within diverse research areas and various application domains. Several anomaly based network intrusion detection systems (ANIDS) can be found in the literature. Most ANIDSs employ supervised algorithms, whose performances highly depend on attack-free training data. However, this kind of training data is difficult to obtain in real world network environment. Moreover, with changing network environment or services, patterns of normal traffic will be changed. This leads to high false positive rate of supervised ANIDSs. Using unsupervised anomaly detection techniques, however, the system can be trained with unlabeled data and is capable of detecting previously unseen attacks. We have categorized the existing ANIDSs based on its type, class, nature of detection/ processing, level of security, etc. We also enlist some proximity measures for intrusion data analysis and detection. We also report some experimental results for detection of attacks over the KDD'99 dataset.

Keyword

Anomaly detection, NIDS, HIDS, ANIDS

1. Introduction

The intrusion detection is one of the important components of infrastructure protection mechanism. It is a type of security management system for computers and networks. Intrusion can be defined as a set of actions aimed to compromise the computer security goals like confidentiality, integrity and availability [1] of resources. Intrusion can occur from outside or inside of the system to be protected. The external intruders are unauthorized users of the machines they attacked, and whereas internal intruders have permission to access the system, but do not have privileges for the root or super user mode. Based on the type of analysis carried out, intrusion detection systems (IDS) can be either *signature-based* or *anomaly-based*. Signature-based schemes (also denoted as *misuse-based*) seek defined patterns, or signatures, within the analyzed data. For this purpose, a signature database corresponding to known attacks is specified a priori. On the other hand, anomaly-based detectors attempt to estimate the 'normal' behavior of the system to be protected, and generate an anomaly alarm whenever the deviation between a given observation at an instant and the normal behavior exceeds a predefined threshold. Depending on the information source considered, an intrusion detection system (IDS) may be either host-based intrusion detection system (HIDS) or network-based intrusion detection system (NIDS). A HIDS analyzes events such as process identifiers and system calls, mainly related to Operating System (OS) information. On the other hand, an NIDS analyzes network related events like- traffic volume, IP addresses, service ports, protocol usage, etc. To test and compare different IDSs, KDDcup 1999 intrusion dataset is a publicly available benchmark data.

The paper is structured as follows. In section 2, we discuss the related work. Section 3 describes various aspects of anomaly based NIDS. In section 4 we explain some popular and relevant ANIDSs. Section 5 describes the details of our experimental results using supervised and unsupervised approach. In section 6, we suggest some possible future directions of the investigation.

2. Related Work

Anomaly based intrusion detection approaches can be basically categorized into two: *supervised* and *unsupervised*. However, a combination of both also can be found to exist, which can be referred as *semi-supervised* approach. Next, we report a selective survey on the existing supervised and unsupervised approaches. Techniques trained in supervised mode assume the availability of a training data set which has labeled instances for normal as well as anomaly class. Typical approach in such cases is to build a predictive model for normal vs. anomaly classes. Any unseen

data instance is compared against the model to determine which class it belongs to. There are two major issues that arise in supervised anomaly detection. First, the anomalous instances are far fewer compared to the normal instances in the training data. Issues that arise due to imbalanced class distributions have been addressed in the data mining and machine learning literature [2]. Second, obtaining accurate and representative labels, especially for the anomaly class is usually challenging. A number of techniques have been proposed that inject artificial anomalies in a normal data set to obtain a labeled training data set [3]. Other than these two issues, the supervised anomaly detection problem is similar to building predictive models. In most circumstances, labeled data or purely normal data is not readily available since it is time consuming and expensive to manually classify it. Purely normal data is also very hard to obtain in practice, since it is very hard to guarantee that there are no intrusions when collecting the network traffic. To address these problems, unsupervised anomaly detection approach is adopted. The unsupervised approach [4] of anomaly detection does not require training data, and thus are most widely applicable. The approach takes as input a set of unlabelled data and attempts to find intrusions buried within the data. After these intrusions are detected, it can be used to train a misuse detection method or a traditional anomaly detection method using the data. Many semi-supervised [5] techniques, assume that the training data has labeled instances for only the normal class, can be adapted to operate in an unsupervised mode by using a sample of the unlabeled data set as training data. Such adaptation assumes that the test data contains very few anomalies and the model learnt during training is robust to these few anomalies.

2.1 Various Anomaly Detection Approaches

A variety of approaches were discussed in *Table-1* which can be applicable in different anomaly based intrusion detection techniques to build classifiers. Based on our survey it has been observed that

(i) If the training or labeled instances for normal as well as anomaly classes are available, supervised approach can be found to be effective in the detection of known attack.

(ii) In case of non-availability of labeled or purely normal data, unsupervised approach of anomaly detection can be found to be effective in the detection of known as well as unknown attack. However, the rate of false positive is more in case of this approach.

(iii) In case of unsupervised anomaly detection using clustering approach, proximity measures play an important role. An appropriate use of proximity measure increases the effectiveness of a clustering based ANIDS.

3. Aspects of Anomaly Based NIDS

In this section, we report some of the important aspects of anomaly detection. Generally, an anomaly based intrusion detection problem is formulated based on the following aspects: (i) types of anomalies, (ii) nature and type of the input data, (iii) appropriateness of similarity/dissimilarity measures (iv) importance of data labeling and (v) reporting of anomalies etc. Next, we discuss each of these aspects, in brief.

3.1 Types of Anomalies

Based on the nature, context, behavior or cardinality, anomalies are generally classified into following three categories [5]:

(i) *Point Anomalies:* This simplest type of anomaly can be defined as an instance of an individual data which has been found to be anomalous with respect to the rest of data. In the majority of applications, this type of anomaly can be found to occur and a good amount of research can be found to address this issue.

(ii) *Contextual Anomalies*: This type of anomaly (also known as conditional anomaly [19]) is defined for a data instance in a specific context. Generally, here the notion of a context is induced by the structure in the data set. Basically, two sets of attributes can be found to be responsible for defining each data instance belonging to this type of anomalies: Contextual attributes and Behavioral attributes. Contextual attributes determine the context (or neighborhood) for that instance. For example, in stock exchange or gene expression time series dataset, time is a contextual attribute that helps to specify the position of the instance on the entire sequence. However, the behavioral attributes are responsible for the non-contextual characteristics of an instance. For example, in a spatial data set describing the average number of people infected by a specific disease in a country,

Table 1: List of Intrusion Detection Approaches

Approach Name	Approach Description	Detection Mode	Application Area	Example IDSs
Statistical based	Attempts to fit a statistical model (usually for normal behaviour) to the given data and then apply an inference test to check whether an unseen instance occuring or not.	Unsupervised	Network	HIDE [6]
Bayesian network based	Estimates the posterior probability of observing a class label (from a set of normal class labels and the anomaly class label), in a given test data instance. The class label with largest posterior is chosen as the predicted class for the given test instance. The zero probabilities, especially for the anomaly class, are smoothed using Laplace Smoothing.	Supervised	Network	ADAM [7]
Knowledge-based	Events are checked against predefined rules or patterns of attack. The goal is to employ representations of known attacks that are general enough to handle actual occurrences of the attacks.	Unsupervised	Host/Network	NSOM [8]
Fuzzy set theoretic	Fuzzy rules are considered for high level description of patterns of behavior found in the data. Once the appropriate rules are identified, they must be combined using fuzzy reasoning to determine an overall solution.	Unsupervised	Network	FIRE [9]
GA based	Genetic algorithms are used to evolve simple rules for network traffic. These rules are used to differentiate normal network connections from anomalous connections. These anomalous connections refer to events with probability of intrusions. The rules stored in the rule base.	Unsupervised	Network	NEDAA [10]
ANN based	The manner in which the neurons of a neural network are structured is intimately linked to the learning algorithm used to train the artificial neural network (ANN). The learning system detects or categorizes persistent features without any feedback from the environment to aid in data clustering, feature extraction, and similarity detection.	Unsupervised	Network	NNID [11]
HMM based	HMM (Hidden Markov Models) correlates observations with hidden states that factor in the system design where observation points are optimized using an acceptable set of system-wide intrusion checkpoints while hidden states are created using explicit knowledge of probabilistic relationships with these observations. These relationships, which are also called profiles, are hardened and evolved with the constant usage of the multiple and independent systems. If observation points can be standardized, then the problem of intrusion predictability can be reduced to profiling the existing and new hidden states to standard observations.	Unsupervised	Network	POMDP[12], NHMMIDS[13]
SVM based	SVM conceptually implement the idea of input data being two sets of vectors being mapped to a high dimensional feature space through some non-linear mapping. A separating hyperplane is constructed in the feature space which maximizes the margin between the two data sets.	Supervised	Network	[14]
Association mining based	Association rule mining has been used for one-class anomaly detection by generating rules from the data in an unsupervised fashion. Association rules are generated from a categorical data set. To ensure that the rules correspond to strong patterns, a support threshold is used to prune out rules with low support.	Unsupervised	Network	ADAM [7]
Clustering based	Clustering is often referred to as unsupervised classification. It does not require an initial dataset and the strength of the system lies within the algorithm itself. Several algorithms have been proposed[15]. Single-Link Clustering algorithm, k-means (Squared error Clustering), hierarchical clustering algorithm to mention a few [7] [15]- [16]	Unsupervised	Network	ADMIT [17]
KNN based	KNN (K Nearest neighbor) based anomaly detection techniques require a distance or similarity measure defined between two data instances. Distance (or similarity) between two data instances can be computed in different ways. The measures are typically required to be positive-definite and symmetric.	Unsupervised	Network	DNIDS[18]

the amount of infection at a specific location can be defined as a behavioral attribute.

(iii) *Collective Anomalies*: It can be defined as a collection of related data instances found to be anomalous with respect to the entire data set. The individual data instances in a collective anomaly may not be anomalies by themselves, but their occurrence together as a collection is anomalous. For example, consider a sequence of actions occurring in a computer as shown below:

. . . http-web, buffer-overflow, http-web, http-web, smtp-mail, ftp, http-web, ssh,
. . . smtp-mail, http-web, ssh, buffer-overflow, ftp, http-web, ftp, smtp-mail, http-web

The highlighted sequence of events (buffer-overflow, ssh, ftp) correspond to a typical web based attack by a remote machine followed by copying of data from the host computer to remote destination via ftp. It should be noted that this collection of events is an anomaly but the individual events are not anomalies when they occur in other locations in the sequence. It will be worth mentioning that unlike point anomalies, collective anomalies occur only in a relevant dataset (i.e. the dataset with which the instances are found to be related). However, in case of occurrence of contextual anomalies, rather it depends on the availability of the context attributes in the data.

3.2 Types of Data

A key aspect of any anomaly detection technique is the nature of the input data. Input is generally a collection of data instances (also referred as object, record, point, vector, pattern, event, case, sample, observation, entity) [20]. Each data instance can be described using a set of attributes (also referred to as variable, characteristic, feature, field or dimension). The attributes can be of different types such as binary, categorical or continuous. Each data instance might consist of only one attribute (univariate) or multiple attributes (multivariate). In the case of multivariate data instances, all attributes might be of same type or might be a mixture of different data types. The nature of attributes determines the applicability of anomaly detection techniques. For example, for statistical techniques different statistical models have to be used for continuous and categorical data. Similarly, for nearest neighbor based techniques, the nature of attributes would determine the distance measure to be used. Often, instead of the actual data, the pair wise distance between instances might be provided in the form of a distance (or similarity) matrix. In such cases, techniques that require original data instances are not applicable e.g. many statistical and classification based techniques.

3.3 Proximity measures

Distance or similarity measures are essential to solve many pattern recognition problems such as classification, clustering, and retrieval problems. From the scientific and mathematical point of view, distance is defined as a quantitative degree of how far apart two objects are. Synonyms for distance include dissimilarity. Those distance measures satisfying the metric properties are simply called metric while other non-metric distance measures are occasionally called divergence. Synonyms for similarity include proximity and similarity measures are often called similarity coefficients. The choice of distance/similarity measures depends on the measurement type or representation of objects. Following subsections explain proximity measure for numeric, categorical and mixed type data.

3.3.1 Proximity measures for Numeric Data

The probability density function represents a probability distribution [21]. In *Table-2* various distance/similarity measures are enumerated according to their syntactic similarities. Here, D and S represent distances and similarities respectively. The distance /similarity measures are based on pdf X and Y.

3.3.2 Proximity measures for Categorical Data

Computing similarity between categorical data instances is not straightforward owing to the fact that there is no explicit notion of ordering between categorical values. The simplest way to find similarity between two categorical attributes is to assign a similarity of 1 if the values are identical and a similarity of 0 if the values are not identical. Several data-driven similarity measures have been proposed [22] for categorical data. The behavior of such measures directly depends on the data. *Table 3* gives the mathematical formulae for the measures. The various measures described in *Table 3* compute the per-attribute similarity $S_k(X_k; Y_k)$ as shown in column 2 and compute the attribute weight w_k as shown in column 3. In *Table 3*, for the sake of notation, consider a categorical data set D containing N objects, defined over a set of d categorical attributes where A_k denotes the k-th attribute. $S_k(X_k; Y_k)$ is the per-attribute similarity between two values for the categorical attribute A_k.

Table 2: Proximity Measures for Numerical data

Group	Measures	Measures
G1*	$Euclidean, L_2\ D_{Euc} = \sqrt{\sum_{i=1}^{d} \lvert X_i - Y_i \rvert^2}$ (1)	$Minkowski, L_p\ D_{Mk} = \sqrt[p]{\sum_{i=1}^{d} \lvert X_i - Y_i \rvert^p}$ (2)
G2*	$S\phi rensen,\ D_{sor} = \frac{\sum_{i=1}^{d} \lvert X_i - Y_i \rvert}{\sum_{i=1}^{d} (X_i + Y_i)}$ (3)	$Lorentzian,\ D_{Lor} = \sum_{i=1}^{d} \ln(1 + \lvert X_i - Y_i \rvert)$ (4)
G3*	$Intersection,\ S_{IS} = \sum_{i=1}^{d} min(X_i, Y_i)$ (5)	$D_{non-IS} = 1 - S_{IS} = \frac{1}{2} \sum_{i=1}^{d} \lvert X_i - Y_i \rvert$ (6)
	$Czekanowski,\ S_{Cze} = \frac{2 \sum_{i=1}^{d} min(X_i, Y_i)}{\sum_{i=1}^{d} (X_i + Y_i)}$ (7)	$D_{Cze} = 1 - S_{Cze} = \frac{\sum_{i=1}^{d} \lvert X_i + Y_i \rvert}{\sum_{i=1}^{d} (X_i + Y_i)}$ (8)
G4*	$Jaccard,\ S_{Jac} = \frac{\sum_{i=1}^{d} X_i Y_i}{\sum_{i=1}^{d} X_i^2 + \sum_{i=1}^{d} Y_i^2 - \sum_{i=1}^{d} X_i Y_i}$ (9)	$D_{Jac} = 1 - S_{Jac} = \frac{\sum_{i=1}^{d} (X_i - Y_i)^2}{\sum_{i=1}^{d} X_i^2 + \sum_{i=1}^{d} Y_i^2 - \sum_{i=1}^{d} X_i Y_i}$ (10)
G5*	$Fidelity,\ S_{Fid} = \sum_{i=1}^{d} \sqrt{(X_i Y_i)}$ (11)	$Bhattacharyya,\ D_B = -\ln \sum_{i=1}^{d} \sqrt{(X_i Y_i)}$ (12)
G6*	$Squared\ Euclidean,\ D_{sqe} = \sum_{i=1}^{d} (X_i - Y_i)^2$ (13)	$Pearson\ \chi^2,\ D_p(X,Y) = \sum_{i=1}^{d} \frac{(X_i - Y_i)^2}{Y_i}$ (14)
G7*	$Kullback\ Leibler\ D_{KL} = \sum_{i=1}^{d} X_i \ln \frac{X_i}{Y_i}$ (15)	$Jeffreys\ D_J = \sum_{i=1}^{d} (X_i - Y_i) \ln \frac{X_i}{Y_i}$ (16)
G8*	$Taneja\ D_{TJ} = \sum_{i=1}^{d} \left(\frac{X_i + Y_i}{2} \right) \ln \left(\frac{X_i + Y_i}{2\sqrt{X_i Y_i}} \right)$ (17)	$Kumar\ Johnson\ D_{KJ} = \sum_{i=1}^{d} \frac{(X_i^2 - Y_i^2)^2}{2(X_i Y_i)^{\frac{3}{2}}}$ (18)

*G1-L_p $Minkowski$,*G2-L_1,*G3-$Intersection$,*G4-$Inner Product$,*G5-$Fidelity$,
*G6-$Squared\ L_2$ or χ^2,*G7-$Shannon's\ entropy$, *G8-$Combinations$

Table 3: Similarity Measures for Categorical data [22]

Measures	$S_k(X_k, Y_k)$	w_k, k=1...d	Comments
Overlap	$\begin{cases} 1 & \text{if } X_k = Y_k \\ 0 & \text{otherwise} \end{cases}$	$\frac{1}{2}$	It simply counts the number of attributes that match in the two data instances.
Eskin	$\begin{cases} 1 & \text{if } X_k = Y_k \\ \frac{n_k^2}{n_k^2+2} & \text{otherwise} \end{cases}$	$\frac{1}{d}$	It gives more weightage to mismatches that occur on attributes that take many values.
IOF	$\begin{cases} 1 & \text{if } X_k = Y_k \\ \frac{1}{1+\log f_k(X_k) x \log f_k(Y_k)} & \text{otherwise} \end{cases}$	$\frac{1}{d}$	It assigns lower similarity to mismatches on more frequent values.
OF	$\begin{cases} 1 & \text{if } X_k = Y_k \\ \frac{1}{1+\log \frac{N}{f_k(X_k)} x \log \frac{N}{f_k(Y_k)}} & \text{otherwise} \end{cases}$	$\frac{1}{d}$	It gives the opposite weighting of the IOF measure for mismatches, i.e., mismatches on less frequent values are assigned lower similarity and mismatches on more frequent values are assigned higher similarity.
Lin	$\begin{cases} 2\log \hat{p}_k(X_k) & \text{if } X_k = Y_k \\ 2\log(\hat{p}_k(X_k) + \hat{p}_k(Y_k)) & \text{otherwise} \end{cases}$	$\frac{1}{\sum_{i=1}^{d} \log \hat{p}_i(X_k) + \log \hat{p}_i(Y_i)}$	It gives higher weightage to matches on frequent values, and lower weightage to mismatches on infrequent values.
Goodall1	$\begin{cases} 1 - \sum_{q \in Q} p_k^2(q) & \text{if } X_k = Y_k \\ 0 & \text{otherwise} \end{cases}$	$\frac{1}{d}$	This measure attempts to normalize the similarity between two objects by the probability that the similarity value could be observed in a random sample of two points. This measure assigns higher similarity to a match if the value is infrequent than if the value is frequent.

3.3.3 Proximity measures for Mixed type Data

Mixed datasets include categorical and numeric values. A common practice to cluster mixed dataset is to transform categorical values into numeric values and then proceed to use a numeric clustering algorithm. Another approach is to compare the categorical values directly, in which two distinct values result in distance 1 while identical values result in distance 0. Nevertheless, these two methods do not take into account the similarity information embedded between categorical values. Consequently, the clustering results do not faithfully reveal the similarity structure of the dataset [23]. Huang [24] proposed a new incremental clustering algorithm for mixed datasets, in which the similarity information embedded between categorical attribute is considered during clustering. Each attribute of the data is associated with a distance hierarchy, which is an extension of the concept hierarchy [25] with link weights representing the distance between concepts. The distance between two mixed data patterns is then calculated according to distance hierarchies. It is worth mentioning that the representation scheme of distance hierarchy can generalize some conventional distance computation schemes including the simple matching and the binary encoding for categorical values, and the subtraction method for continuous values and ordinal values [24].

3.4 Data Labels

The labels associated with a data instance denote if that instance is normal or anomalous. It should be noted that obtaining labeled data which is accurate as well as representative of all types of behaviors, is often prohibitively expensive. Labeling is often done manually by a human expert and hence requires substantial effort to obtain the labeled training data set. Typically, getting a labeled set of anomalous data instances which cover all possible type of anomalous behavior is more difficult than getting labels for normal behavior. Moreover, the anomalous behavior is often dynamic in nature e.g. new types of anomalies might arise, for which there is no labeled training data. Based on the extent to which the labels are available, anomaly detection techniques can operate either in supervised anomaly detection approach or unsupervised anomaly detection approach.

3.5 Reporting of anomalies

An important aspect for any anomaly detection technique is the manner in which the anomalies are reported. Typically, the outputs produced by anomaly detection techniques are one of the following two types:

(i) *Scores:* Scoring techniques assign an anomaly score to each instance in the test data depending on the degree to which that instance is considered as an anomaly. Thus the output of such techniques is a ranked list of anomalies. An analyst may choose to either analyze top few anomalies or use a cut-off threshold to select the anomalies.

(ii) *Labels*: Techniques in this category assign a label (normal or anomalous) to each test instance. Scoring based anomaly detection techniques allow the analyst to use a domain specific threshold to select the most relevant anomalies. Techniques that provide binary labels to the test instances do not directly allow the analysts to make such a choice, though this can be controlled indirectly through parameter choices within each technique. A possible labeling strategy [26], measure the physical properties of the clusters to interpret their nature. Cluster quality indices are also used to identify the presence of a massive attack. In this strategy, Dunn's index [27] and C-index [28] are found suitable.

4. Existing ANIDS: A Selected Review

This section reports some of the popular and relevant ANIDSs which attempt to detect the non-conforming patterns over network data using statistical, data mining or knowledge based approaches.

(i) **ADAM [7]**: Automated Data Analysis and Mining basically provides a test bed for using data mining techniques to detect intrusions. ADAM exploits a combination of association rules mining and classification techniques to discover 6 attacks in a TCPdump audit trail. First, ADAM builds a repository of "normal" frequent itemsets that holds during attackfree periods. It does so by mining data that is known to be free of attacks. Secondly, ADAM runs a sliding-window based on-line algorithm that finds frequent itemsets in the connections and compares them with those stored in the normal itemset repository, discarding those that are deemed normal. With the rest, ADAM uses a classifier which has been trained to classify the suspicious connections as either known type of attack or an unknown type or a false alarm.

(ii) **MINDS [29]**: The Minnesota Intrusion Detection System is a data mining based system for detecting network intrusions. Input to MINDS is Netflow version 5 data collected using flow-tools. Flow-tools only capture packet header information (i.e., it does not capture message content), and builds one way sessions (flows). The analyst uses MINDS to analyze these data files in a batch mode. The reason for running the system in a batch mode is not due to the time it takes to analyze these files, but it is convenient for the analyst to do so. Before data is fed into the anomaly detection module, a data filtering step is executed to remove network traffic that the analyst is not interested in analyzing. The first step of MINDS is to extract the important features that are used in the data mining analysis. Then, it summarizes the time-window based features. After the feature construction step, the known attack detection module is used to detect network connections that corresponds to attacks for which signatures are available, and then to remove them from further analysis. Next, the data is fed into the MINDS 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. MINDS 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.

(iii) **JAM [30]**: The main idea in JAM (Java Agents for Metalearning), is to generate classifiers using a rule learning program on training data sets of system usage. The output from the classifier, i.e. a set of classification rules, is used to recognize anomalies and detect known intrusions. Specifically, the approach of using classifiers is tested on two sets of data: one from attacks that uses sendmail, the other from network attacks, using TCPdump. Sendmail data consists of two sets of traces, one with normal and one with abnormal data. The training data is fed to RIPPER (Repeated Incremental Pruning to Produce Error Reduction) [30], a rule-learning program. RIPPER rules classify the training data into the two classes "normal" and "abnormal". Each trace is then post-processed by comparing it with the RIPPER predictions, in order to filter out spurious prediction errors. The rationale for the post-processing scheme is that an actual intrusion is characterized by a majority of abnormal adjacent call sequences. A second experiment consists of computing classification rules using only the normal traces. In this case, in order to detect intrusions, the confidence information associated with the generated rules is used. Each trace is given a score according to whether a trace submitted to the classifier at run time violates one of the generated rules. In this case, the trace score is incremented in proportion to the confidence of the violated rule. Given a long trace with many sequences,

scores are assigned to each sequence, and the average score is used to decide whether the sequence represents an intrusion. Pre-processing is applied to the TCPdump raw data and then RIPPER is applied to the data.

(iv) **DNIDS [18]:** Dependable Network Intrusion Detection System is based on the Combined Strangeness and Isolation measure K-Nearest Neighbor (CSI-KNN) algorithm. The DNIDS can effectively detect network intrusions while providing continued service even under attacks. The intrusion detection algorithm analyzes different characteristics of network data by employing two measures: strangeness and isolation. Based on these measures, a correlation unit raises intrusion alerts with associated confidence estimates. In the DNIDS, multiple CSI-KNN classifiers work in parallel to deal with different types of network traffic. An intrusion tolerant mechanism monitors the classifiers and the hosts on which the classifiers reside and enables the IDS to survive component failure due to intrusions. As soon as a failed IDS component is discovered, a copy of the component is installed to replace it and the detection service continues. DNIDS consists of the following components: Sensors, Detectors, Alert Agents (AA), Maintenance Agents (MA), the Manager and the Console. Sensors capture the network packets from a network segment and transform them into connection-based vectors. The Detector is a collection of CSI-KNN classifiers that analyze the vectors supplied by the sensors. Sensors and Detectors are intrusion detection components. The Manager, Alert Agents, and Maintenance Agents are designed for intrusion tolerance and installed on a secure administrative server called Station. The Manager has knowledge of the resources of the NIDS, generates mobile agents (AA and MA) and dispatches them to execute tasks.

(v) **HIDE [6]:** Hierarchical Intrusion Detection is an anomaly based network intrusion detection system, with hierarchical architecture, that uses statistical models and neural network classifiers to detect attacks. HIDE is a distributed hierarchical application, which consists of several tiers with each tier containing several Intrusion Detection Agents (IDAs). IDAs are IDS components that monitor the activities of a host or a network. Different tiers correspond to different network scopes that are protected by agents affiliated to them. The probe layer collects the network traffic of a host or a network, abstracts the traffic into a set of statistical variables to reflect the network status, and periodically generates reports to the event preprocessor. The event preprocessor layer receives reports from both the probe and IDAs of lower tiers, and converts the information into the format required by the statistical model. The statistical processor maintains a reference model of the typical network activities, compares the reports from the event preprocessor to the reference models, and forms a stimulus vector to feed into the neural network classifiers. The neural network classifier analyzes the stimulus vector from the statistical model to decide whether the network traffic is normal or not. The post processor generates reports for the agents at higher tiers. At the same time, it may display the results through a user interface. HIDE can reliably detect UDP flooding attacks with attack intensity as low as five to ten percent of background traffic.

(vi**) NSOM [8]:** Network Self-Organizing Maps is a network based Intrusion Detection System (IDS) using Self-Organizing Maps (SOM). NSOM could be classified as an anomaly detection system. Here the attempt is made to detect anomalies by quantifying the usual or acceptable behavior and flags accordingly the other irregular behavior as potentially intrusive. The system uses a structured SOM to classify real-time Ethernet network data. NSOM classifies network traffic in real-time. It continually collect network data from a network port, preprocesses that data and select the features suitable for classification. Then it starts the classification process - a chunk of packets at a time - and then sends the resulting classification to a graphical tool that portrays the activities that are taking place on the network port dynamically as receive more packets. The hypothesis is that routine traffic that represents normal behavior would be clustered around one or more cluster centers and any irregular traffic representing abnormal and possibly suspicious behavior would be clustered outside of the normal clustering. The system can be found capable to classify regular vs. irregular and possibly intrusive network traffic for a given host.

(vii) **FSAS [31]:** Flow-based Statistical Aggregation Schemes for Network Anomaly Detection (FSAS) is a layered architecture, which includes two main parts: feature generator and flow-based detector. In the feature generator, the event preprocessor module collects the network traffic of a host or a network. The event handlers generate reports to flow management module. The flow management module efficiently determines if a packet is part of an existing flow or should generate a new flow key. According to different flow key, this module aggregates flows together based on their flow keys, and dynamically updates per-flow accounting measurements. The event time module periodically calls feature extraction module to converts the statistic information of flows into the format required by the statistical model. The probe module

collects the network traffic, abstracts the traffic into a set of statistical parameters to reflect the network status. The feature scoring metric calculates the probability scores of these features by comparing the features with the reference model generated by past normal and attack users. The probability scores are measurements indicating how likely for a feature to take the observed value. The neural network classifier classifies the score vectors to prioritize flows with maliciousness. The higher maliciousness of a flow means the flow has the higher possibility of attacker. The feature analyzer module identifies the attack by detecting abrupt network behavior changes. The detector part includes a neural network detector (NND) which scores each flow according to reference model. The extracted flow features are then scored based on the information of the reference models. Since the actual legitimate flow distribution during attack is unknown, the approach needs to establish a reference profile of normal users and the reference model of attackers from the past traffic. While attack packets arrive continuously and the true flow profile changes as new packets arrive, the reference profile is updated and flow is scored at the same time. Detecting intrusions involves the multi-variants classifications based on the monitored features. Generally, intrusion detection systems are designed to use as many features as the designers could think they might be useful. In FSAS provided 22 features, which are relevant in DoS attack detection.

(viii) **N@G [32]:** N@G(Network at Guard) is a Hybrid Intrusion Detection System(IDS) having capabilities of both misuse and anomaly detection. N@G has both network and host sensors. The Statistical Anomaly based Intrusion Detection is pursued using the Chi-Square technique where various network protocol parameters is profiled to detect intrusions. It has four detection methodologies viz., Signature based detection, Network Access Policy Violation and Protocol Anomaly Detection as a part of its network sensor and it has Audit trail analysis, Log analysis, statistical analysis and host access policy violation components as the part of host sensor. The system has a separate IDS Server, i.e. basically a management console to aggregate alerts from various sensors with user interface, middle-tier and data management component. It provides real-time protection against malicious changes to network settings on client computers, which can include unsolicited changes to the Windows Hosts file, Windows Messenger service, as well as providing Layered Service Provider (LSP) and Domain Name Server (DNS) protection. N@G can dynamically apply access controls to routers (Cisco) to actively block the network attacks and can forge a TCP reset packet to both ends of the connection to stop detected intrusion attempts. N@G can send SNMP traps for the detected intrusions to any Network Management System. N@G has the capability to send intrusion alert information through email, send alarms to the N@G-MS and allows user to execute custom scripts. N@G incorporates IDMEF (IETF's Intrusion Detection Message Exchange Format), which facilitates third party inter-operability for communicating information about intrusions in a standard format.

(ix**) FIRE [9]:** The Fuzzy Intrusion Recognition Engine (FIRE) is an anomaly-based intrusion detection system that uses fuzzy logic to assess whether malicious activity is taking place on a network. The system combines simple network traffic metrics with fuzzy rules to determine the likelihood of specific or general network attacks. These metrics are then evaluated as fuzzy sets. Fuzzy rules are developed for some common intrusion detection scenarios. It relies on fuzzy network traffic profiles as inputs to its rule sets. It uses simple data mining techniques to process the network input data and help expose metrics that are particularly significant to anomaly detection. FIRE consists of the three types of components. The network data collector (NDC) is a promiscuous network data sniffer and recorder. It reads raw network packets off the wire and stores them on disk. The next component, the network data processor (NDP), summarizes and tabulates the raw packet data in carefully selected categories. In a sense, an NDP performs a kind of data mining on the collected packets. The NDP merges these summaries and tables with past data and stores them on disk. Next, the NDP compares the current data with the historical mined data to create values that reflect how the new data differs from what was observed in the past. These values are "fuzzified" to produce the fuzzy inputs needed by the Fuzzy Threat Analyzer (FTA). The resulting fuzzy inputs from the NDPs are called "fuzzy alerts" because they represent an alert condition to a degree. The Fuzzy Threat Analyzer combines the inputs from one or more FDPs to create composite inputs. Individual NDPs can be given greater or less influence on the results by assigning them different weights. Finally, the output from the fuzzy system executed by the FTA leads to fuzzy alerts that are sent to the security administrator for response. FIRE uses a fuzzy analysis engine to evaluate the fuzzy inputs and trigger alert levels for the security administrator. The FIRE IDS can detect a wide range of common attack types.

(x**) NFIDS [33]:** NFIDS is an anomaly based network intrusion detection system based on Neuro-Fuzzy

approach with a hierarchical architecture that is based on the autonomous agent's architecture. The system consists of three tiers. Different tiers correspond to different network scopes that are protected by agents affiliated to them. Tier-I contains several Intrusion Detection Agents (IDAs). IDAs are IDS components that monitor the activities of a host or a network and report the abnormal behavior to tier-II. Tier-II agents detect the network status of a LAN based on the network traffic that they observe as well as the reports from the tier-I agents within the LAN. Tier-III combines higher-level reports, correlate data, and send alarms to the User Interface. There were four main types of agents developed for this system: TCPAgent, which monitors TCP connections between hosts and on the network, UDPAgent, which looks for unusual traffic involving UDP data, ICMPAgent, which monitors ICMF traffic and PortAgent, which looks for unusual services in the network.

4.1 Discussion

A comparative study on those ANIDSs reported in the previous subsection is made based on parameters like detection type (host based, network based or both), class of detection approach (misuse, anomaly or both), nature of detection (online or offline), nature of processing (centralized or distributed), data gathering mechanism (centralized or distributed) level of security (low, high or medium) and approach of analysis. Comparisons are illustrated in *Table-4*. It can be observed that

(i) An ANIDS with high accuracy (least false alarm) often can be found failed to operate in real time. However, with a compromised false alarm rate, several effective ANIDSs can be found in the literature.

(ii) In case of unsupervised ANIDS, faster incremental clustering techniques can be found suitable. Use of appropriate proximity measure to handle higher dimensionality as well as mixed type data can help to improve the effectiveness of such techniques to a great extent.

(iii) An ANIDS should be capable of not only to detect unknown attack but also should be capable of modifying the normal as well as on attack profile.

Table 4: List of Anomaly based NIDS

Name of IDS	Year of Publish	Types of Detection (H/N/Hy*)	Class (M/A/B*)	Nature of Detection (R/N*)	Nature of Processing (C/D*)	Data Gathering Mechanism (C/D*)	Level of Security (L/H/Md*)	Approach
JAM [30]	1995	N	A	N	C	C	L	Classification
FIRE [9]	2000	N	A	N	C	C	H	Fuzzy Logic
ADAM [7]	2001	N	A	R	C	C	L	Association Rule
HIDE [6]	2001	N	A	R	C	D	L	Statistical &Neuralnet
NSOM [8]	2002	N	A	R	C	C	L	Neuralnet
MINDS [29]	2003	N	A	R	C	C	H	Outlier
NFIDS [33]	2003	N	A	N	C	C	L	Neuro Fuzzy Logic
N@G [32]	2003	Hy	B	R	C	C	H	Statistical
FSAS [31]	2006	N	A	R	C	C	L	Statistical
DNIDS [18]	2007	N	A	R	C	C	L	K-NN

*H-Host/N-Network/Hy-Hybrid, M-Misuse/A-Anomaly/B-Both, R-Realtime/N-Non realtime, C-Centralised/D-Distributed, L-Low/H-High/M-Moderate)

Table 5: KDD'99 Intrusion Dataset

Records	Total Features	Discrete or continuous Features	Categorical Features	Categories of Attack	Types of Attack
311029	41	38	3	4	37

5. Attack Detection over KDD'99 Dataset: An Experimental Analysis

In 1999, recorded network traffic from the DARPA 98 Lincoln Lab dataset [34] was summarized into network connections with 41-features per connection as shown in *Table-5*. This formed the KDD'99 intrusion detection benchmark in the International Knowledge Discovery and Data Mining Tools Competition [35].

5.1 Attack Detection using Unsupervised Approach

Using unsupervised anomaly detection approach, the system can be trained with unlabeled data and can be made capable of detecting previously unseen attacks. The approaches in this category make the implicit assumptions that the majority of the network connections are normal traffic, only *x*% of traffic are malicious [36] and attack traffic is statistically different from normal traffic [37]. If any of the assumptions fails, the performance of the approaches may deteriorate, for example, such techniques may suffer from high false alarm rate. There exists a large number of clustering algorithms in the literature [5], which can be considered for unsupervised approach of anomaly detection of intrusions. In the subsequent subsections, we report an experimental analysis of three existing clustering based intrusion detection techniques i.e. ADWICE [38], CatSub [39] and Fuzzy c-means [40].

5.1.1 ADWICE [38]

The ADWICE (Anomaly Detection With fast Incremental Clustering) model consists of a number of clusters, a number of parameters, and a tree index in which the leaves contain the clusters. The condensed information of data points are denoted as cluster features *(CF)*. A cluster feature is a triple $CF = (n; S; SS)$ where n is the number of data points in the cluster, S is the linear sum of the n data points and SS is the square sum of all data points. Given n d-dimensional data vectors v_i and a cluster CF representing $\{v_i | i = 1,.., n\}$, the centroid v_0 and radius $R(CF)$ are defined as:

$$v_0 = \frac{\sum_{i=1}^{n} v_i}{n}, \; R(CF_i) = \sqrt{\frac{\sum_{i=1}^{n} (v_i - v_0)^2}{n}} \quad (19)$$

where R is the average distance from member points in the cluster to the centroid and is a measure of the tightness of the cluster around the centroid. The distance between a data point v_i and a cluster CF is the Euclidian distance between v_i and the centroid, denoted by $D(v_i, CF)$ while the distance between two clusters CF_i and CF_j is the Euclidian distance between their centroids, denoted as $D(CF_i, CF_j)$. If two clusters $CF_i = (n_i, S_i, SS_i)$ and $CF_j = (n_j, S_j, SS_j)$ are merged, the CF of the resulting cluster may be computed as $(n_i + n_j, S_i + S_j, SS_i + SS_j)$. A leaf node contains at most LS (leaf space) entries, each of the form (CF_i) where $i = \{1,.,LS\}$. Each CF_i of the leaf node must satisfy a threshold requirement (TR) with respect to the threshold value T which is evaluated to see if a cluster may absorb more data points.

During the detection, the model is searched for the closest cluster CF_i. Then the distance $D(v,, CF_i)$ from the centroid of the cluster to the new data point v is computed. Informally, if D is small, i.e. lower than a limit, v is similar to data included in the normality model and v should therefore be considered normal. If D is large, v is an anomaly.

5.1.2 CatSub [39]

It assumed that any activity deviate from the known normal activity is malicious. The whole dataset is clusterised using the subspace-based incremental clustering method (CatSub) along with a set of cluster seeds for detecting normal records. Non-seeded clusters will contain attacks. The clusters are then labeled as either intrusive or normal traffic.

The given dataset $X = \{X_1, X_2, .., X_n\}$ contains n objects, each described by d categorical attributes $A_1, A_2, ..., A_d$ having finite and discrete valued domains $D_1, D_2, .., D_d$. Let, the i-th and k-th objects be such that $\forall j \in 1,2,..d : x_{ij} = x_{kj}$ i.e. the two objects have a common value in each of the attributes. An attribute having a common value over a set of objects is called a matching attribute. A specified minimum number (*MinAtt*) of such matching attributes are needed to form a cluster. A cluster should also contain at least *MinObj* number of objects. Let, a set of matching

attribute and value pairs be represented by the set : $MAV = \{(j, v) \mid j \in \{1,2..d\}, v \in D_j\}$ The set *MAV* together with the set of objects $T \in \{1,2...n\}$ represent a cluster *C*, which is the set : $C = \{T, MAV\}$. Subspace based similarity function between two clusters *C* and *C′* , *Sim*(C, *C′*) is defined as:

$$\left.\begin{array}{l} 0 \text{ , if } |C.MAV| - m \geq \delta \\ m - MinAtt + \frac{1.0}{(2.0+|C.MAV|-m)}, \text{ otherwise} \end{array}\right\} \quad (20)$$

where, m = |C.MAV ∩ *C′* .MAV| is the cardinality of the set of matching attributes that remains if *C′* is merged with *C*. The expression (|C.MAV| - m) computes the reduction in number of matching attributes after the merger. This reduction should be less than a specified threshold, *d*, otherwise the similarity value returned should be set to zero.

The detection is done in two phases. In phase *1* , the entire dataset is clustered with higher similarity threshold to find clusters with large subspaces. Some cluster seeds are also used for detecting normal records. All the non-seeded clusters will contain attacks (mostly DoS) and seeded clusters will contain the normal instances. The remaining outliers that contain both normal and attack records, are passed to the second phase for labeling. In phase *2*, the outliers are clustered again with a lower similarity threshold value. Cluster seeds are used for detecting normal records as before. The clusters formed (non-seeded) are treated as attacks. Remaining outliers are also treated as attacks.

5.1.3 Fuzzy c-means [40]

Attack detection is performed using Fuzzy *c*-means algorithm. Fuzzy *c*-means is a method of clustering which allows one piece of data belongs to two or more clusters. The input to the algorithm is *N*, the data records and *m*, the fuzziness value. Its steps are:

Step 1. Initialize μ, the utility matrix with random values between zero and one ; but with the sum of all fuzzy membership table elements for words, the sum of the memberships of a record for all clusters must be one.

Step 2. Calculate an initial value for J using

$$J = \sum_{i=1}^{N}\sum_{j=1}^{C} \mu_{ij}^{m} |x_i - c_j|^2$$

Step 3. Calculate the centroids of the clusters C_j

$$C_j = \frac{\sum_{i=1}^{N} \mu_{ij}^{m} x_i}{\sum_{i=1}^{N} \mu_{ij}^{m}}$$

Step 4. Calculate the fuzzy membership table using

$$\mu_{ik} = \frac{1}{\sum_{k=1}^{C} \left(\frac{|x_i - c_j|}{|x_i - c_k|}\right)^{\frac{2}{m-1}}}$$

Step 5. Recalculate J.

Step 6. Goto step 3 until a stopping condition was reached.

5.1.4 Experimental Evaluation based on KDDcup99 Dataset

The ADWICE requires data to be numeric. Non-numeric data are therefore assumed to be transformed into numeric format by pre-processing. Having a large number of attack types and a large number of features to consider, KDD'99 data set can thus work as a proof of concept for the distinguishing attributes of the algorithm to test the real-time properties. The anomaly detection is 95% with false positive (FP) rate 2.8%. The result for ADWICE over KDD'99 dataset is given in *Table-6*.

Table 6: Attacks Detection using Unsupervised Method over KDDCUP99 dataset

Attack Category	ADWICE			Catsub			FCM		
	Records	Detection	Accuracy (%)	Records	Detection	Accuracy (%)	Records	Detection	Accuracy (%)
Normal	60593	57563	95	60593	49813	82.21	60593	58169	96
DOS	229853	227554	99	229853	229806	99.97	227554	229806	99
U2R	228	200	88	70	0	0	70	0	0
R2L	16189	5018	31	16347	13458	82.33	16347	13568	83
Probe	4166	4124	99	4166	1523	36.56	4166	3874	93
FP		1696	2.8(%)		8185	13.51(%)		3313	5.47(%)

In Catsub, KDDcup'99 dataset was considered for testing. It shows an average detection rate of 82% with FP rate 13%. The result is in *Table-6*.

Similarly, experiments were carried out for Fuzzy c-means over the KDD'99 dataset. The categorical attributes of the data were converted to numerical data and were clusterised into attack and normal categories. The detection rate for normal is 96% with FP rate 5.47%. The results are shown in *Table-6*.

5.2 Attack Detection using Supervised Approach

Supervised anomaly detection in network intrusion detection, basically uses purely normal instances as training data to build a predictive model of normal class as well as anomaly class. Any unknown data instance is compared with this model to determine its class. A large number of supervised anomaly detection approaches exist in literature [41]. A machine learning based algorithm DGSOT [14] and a clustering based algorithm K-Means+ID3 [42] are discussed next.

Support vector machines (SVM)[43] have been used as anomaly intrusion detectors. The training time of SVM, prohibits its real-time usage in intrusion detection. With regard to the training time of SVM, random sampling has been used to enhance the training of SVM [44]. The DGSOT (dynamic growing self-organizing tree) algorithm, top-down clustering, builds the hierarchical tree iteratively in several epochs/iterations. After each epoch, new nodes are added to the tree based on a learning process. To avoid the computation overhead of building the tree, after each epoch SVM is trained on the nodes of both trees. The support vectors of the classifier are used as prior knowledge for the succeeding epoch in order to control the tree growth. Specifically, support vectors are allowed to grow, while non-support vectors are stopped.

The results of DGSOT algorithm and SVM based on randomly selected 14% from the overall training data set of 1998 DARPA data are given in *Table-7*. In this table, diagonal values represent the number that is correctly classified. The last three columns represent detection accuracy, false positive (FP) rate and false negative (FN) rate. K-Means+ID3 is a supervised anomaly detection method by cascading k-Means clustering and the ID3 decision tree learning methods for classifying anomalous and normal activities in a computer network system. The *k*-Means clustering method first partitions the training instances into *k* clusters using Euclidean distance. On each cluster, representing a density region of normal or anomalous instances, build an ID3 decision tree. The decision tree on each cluster refines the decision boundaries by learning the subgroups within the cluster. To obtain a final decision on classification, the decisions of the *k*-Means and ID3 methods are combined using two rules: the Nearest-neighbor rule and the Nearest-consensus rule. The results show that the detection accuracy of the *k*-Means+ID3 method is as high as 96.24 % at FP rate 0.03 % on the Network Anomaly Data (NAD), which is feature extracted from the 1998, 1999, and 2000 MIT-DARPA network traffic [45, 46, 47].

Experimentation was carried out for attack classification over 10% corrected KDD'99 intrusion data using LIBSVM [48] (one-class SVM). The result is shown in *Table-8* with detection accuracy rate of 92.25% and FP rate 7.7%.

Table 7: Attack classification using SVM + DGSOT

	Normal	DOS	U2R	R2L	Probe	Accuracy (%)	FP (%)	FN (%)
Normal	45616	541	92	267	1397	95	3	5
DOS	316	21145	0	48	211	97	2	3
U2R	4	0	4	9	0	23	100	76
R2L	793	0	408	1024	103	43	24	56
Probe	112	1	0	0	1156	91	60	9

Table 8: Attack classification using LIBSVM

Attacks	Records	Detection	Accuracy (%)
Normal	60592	55897	92.25
DOS	229853	222381	96.74
U2R	4166	4155	99.23
R2L	16189	15343	94.78
Probe	228	226	99.42
FP	4695	7.7(%)	

Table 9: FP rate in Supervised vs. Unsupervised Approach

Unsupervised Approach	FP (%)	Supervised Approach	FP (%)
ADWICE	2.8	K-Means+ID3	0.3
CatSub	13.51	DGSOT+SVM	3.0
Fuzzy *c*-means	5.47	LIBSVM	7.7

5.3 Discussion

Both the supervised and unsupervised approaches for anomaly detection can be found to be effective in detecting known attacks. The attack detection rate of unsupervised approach has been found to be higher than the supervised one. However, rate of false positive (FP) in unsupervised approach is more compared to the supervised approach. Our experimental study over KDD'99 dataset also indicates that attack detection by supervised approaches have low false positive (FP) rate over unsupervised approach. Though, the supervised approach can provide a better result, in most circumstances, a labeled dataset or purely normal data is not available. So, by including a post-processing step for alarm diagnosis (in a feedback mode) with an appropriate unsupervised approach, an effective and adaptive solution can be developed for unknown attack detection.

6. Conclusion

Anomaly detection can detect novel attacks to increase the detection rate. Compared to supervised approaches, unsupervised approach breaks the dependency on attack-free training datasets. The performance of unsupervised anomaly detection approaches achieve higher detection rate over supervised approach. Also, unsupervised approach have high false positive rate over supervised approach. Using unsupervised anomaly detection techniques, however, the system can be trained with unlabeled data and is capable of detecting previously unseen attacks. Using the results, we provided some insights to the likely requirements of a good unsupervised anomaly detection scheme in the application of network intrusion detection systems.

7. References

[1] R. Heady, G. Luger, A. Maccabe, and M. Servilla, "The architecture of a network level intrusion detection system," Computer Science Department, University of New Mexico, Tech. Rep., 1990.

[2] M. V. Joshi, R. C. Agarwal, and V. Kumar, "Mining needle in a haystack: classifying rare classes via two-phase rule induction," in Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM Press, 2001, pp. 293–298.

[3] J. Theiler and D. M. Cai, "Resampling approach for anomaly detection in multispectral images," in Proceedings of SPIE, vol. 5093. SPIE, 2003, pp. 230–240.

[4] S. Zanero and S. M. Savaresi, "Unsupervised learning techniques for an intrusion detection system," in Proceedings of the 2004 ACM symposium on Applied computing, 2004, pp. 412 – 419.

[5] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," ACM Computing Surveys, Tech. Rep., to appear Sept. 2009.

[6] Z. Zhang, J. Li, C.N.Manikopoulos, J. Jorgenson, and J. Ucles, "Hide: a hierarchical network intrusion detection system using statistical preprocessing and neural network classification," in Proceedings of 2001 IEEE Man Systems and Cybernetics Information Assurance Workshop, 2001.

[7] B. Daniel, C. Julia, J. Sushil, and W. Ningning, "Adam: a testbed for exploring the use of data mining in intrusion detection," SIGMOD Rec., vol. 30, no. 4, pp. 15–24, 2001.

[8] K. Labib and R. Vemuri, "Nsom: A tool to detect denial of service attacks using self-organizing maps," Department of Applied Science University of California, Davis, California, U.S.A., Tech. Rep., 2002.

[9] J.E.Dickerson, "Fuzzy network profiling for intrusion detection," in Proceedings of the 19th International Conference of the North American Fuzzy Information Processing Society (NAFIPS), 2000, pp. 301– 306.

[10] S. Chris, L. Pierce, and S. Matzner, "An application of machine learning to network intrusion detection," in Proceedings of the 1999 Annual Computer Security Applications Conf. (ACSAC). Phoenix,Arizona: IEEE, Computer Society Press, Los Alamitos, California, may 1999, pp. 371–377. [Online]. Available: http://www.acsac.org/1999/papers/fri-b-1030-sinclair.pdf (30 Oct. 2003)

[11] J. Ryan, M. Lin, and R. Miikkulainen, "Intrusion detection with neural networks," Advances in Neural Information Processing Systems, vol. 10, 1998.

[12] T.Lane, "A decision-theoretic, semi-supervised model for intrusion detection," University of New Mexico, Tech. Rep., 2004.

[13] M. Hashem, "Network based hidden markov models intrusion detection systems," IJICIS, vol. 6, no. 1, JANUARY 2006.

[14] L. Khan, M. Awad, and B. Thuraisingham, "A new intrusion detection system using support vector machines and hierarchical clustering," VLDB, vol. 16, pp. 507–521, October 2007.

[15] J. Anderson, "Computer security threat monitoring and surveillance," James P Anderson Co., Fort Washington, Pennsylvania, Tech. Rep., Apr 1980.
[16] B. Neumann, "Knowledge management and assistance systems," 2007.
[17] K.Sequeira and M.Zaki, "Admit: anomaly-based data mining for intrusions," in Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002, pp. 386– 395.
[18] L. V. Kuang, "Dnids: A dependable network intrusion detection system using the csi-knn algorithm," Master's thesis, Queen's University Kingston, Ontario, Canada, Sep 2007.
[19] X. Song, M. Wu, C. Jermaine, and S. Ranka, "Conditional anomaly detection," Transactions on Knowledge and Data Engineering, IEEE, vol. 19, no. 5, pp. 631–645, 2007.
[20] P. N. Tan, M. Steinbach, and V. Kumar, Introduction to Data Mining. Addison-Wesley, 2005.
[21] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," International Journal of Mathematical Models and Methods in Applied Science, vol. 1, no. 4, November 2007.
[22] S. Boriah, V. Chandola, and V. Kumar, "Similarity measures for categorical data: A comparative evaluation," in Proceedings of the 8th SIAM International Conference on Data Mining, 2008, pp. 243–254.
[23] C.C.Hsu and S.H.Wang, "An integrated framework for visualized and exploratory pattern discovery in mixed data," IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 2, pp. 161 – 173, 2005.
[24] C. C. Hsu and Y. P. Huang, "Incremental clustering of mixed data based on distance hierarchy," in Expert Systems with Applications, vol. 35. http://www.sciencedirect.com: Elsevier Science Ltd., 2008, pp. 1177 – 1185.
[25] G.L.Somlo and E.H.Adele, "Incremental clustering for profile maintenance in information gathering web agents," in Proceedings of the 5th International Conference on Autonomous Agents, 2001, pp. 262 – 269.
[26] R. Storløkken, "Labelling clusters in an anomaly based ids by means of clustering quality indexes," Master's thesis, Faculty of Computer Science and Media Technology Gjøvik University College, 2007.
[27] J. Dunn, "Well separated clusters and optimal fuzzy partitions," Journal of Cybernetics, vol. 4, pp. 95–104, 1974.
[28] L. Hubert and J. Schultz, "Quadratic assignment as a general data-analysis strategy," British Journal of Mathematical and Statistical Psychology, vol. 29, pp. 190–241, 1976.
[29] L. Ertöz, E. Eilertson, A. Lazarevic, P. ning Tan, V. Kumar, and J. Srivastava, "Chapter 3 minds- minnesota intrusion detection system."
[30] W. W. Cohen, "Fast effective rule induction," in Proceedings of the 12th International Conference on Machine Learning, Lake Taho, CA, 1995.
[31] S. Song, L. Ling, and C. N. Manikopoulo, "Flow-based statistical aggregation schemes for network anomaly detection," in Proceedings of the IEEE International Conference On Networking, Sensing., 2006.
[32] N. Subramoniam, P. S. Pawar, M. Bhatnagar, N. Khedekar, S. Guntupalli, N. Satyanarayana, V. A. V. P. K. Ampatt, R. Ranjan, and P. S. Pandit, "Development of a comprehensive intrusion detection system- challenges and approaches," in Proceedings of the ICISS 2005, Kolkata,India, 2005, pp. 332–335.
[33] M. Mohajerani, A. Moeini, and M. Kianie, "Nfids: A neuro-fuzzy intrusion detection system," in Proceedings of the 10th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Dec 2003, pp. 348–351.
[34] I. S. T. G. MIT Lincoln Lab., "The 1998 intrusion detection off-line evaluation plan," March 1998. [Online]. Available: http://www.11.mit.edu/IST/ideval/docs/1998/id98-eval-11.txt
[35] U. of California, "Kdd cup 1999 data," 1999. [Online]. Available: http://www.kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[36] L.Portnoy, E.Eskin, and S.J.Stolfo, "Intrusion detection with unlabeled data using clustering," in Proceedings of The ACM Workshop on Data Mining Applied to Security, 2001.
[37] H.S.Javitz and A.Vadles, "The nides statistical component: Description and justification," SRI International, Tech. Rep., 1993.
[38] K. Burbeck and S. Nadjm-Tehrani, "Adwice - anomaly detection with real-time incremental clustering," in Proceedings of Information Security and Cryptology - ICISC 2004, vol. 3506/2005. Berlin, Germany: Springer Berlin / Heidelberg, May 24 2005, pp. 407–424.
[39] B. Borah and D. K. Bhattacharyya, "Catsub: A technique for clustering categorical data based on subspace," The Icfai University Journal of Computer Sciences, April 2008.
[40] J. C. Dunn, "A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters," Journal of Cybernetics, vol. 3, pp. 32 – 57, 1973.
[41] P. Laskov, P. Dussel, C. Schafer, and K. Rieck, "Learning intrusion detection: Supervised or unsupervised ?" in Proceedings of Image Analysis and Processing -ICIAP 2005, vol. 3617 / 2005. Berlin , Germany: Springer Berlin / Heidelberg, November 18 2005, pp. 50–57.
[42] S. R. Gaddam, V. V. Phoha, and K. S. Balagani, "K-means+id3: A novel method for supervised anomaly detection by cascading k-means clustering and id3 decision tree learning methods," IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 3, March 2007.
[43] S.Mukkamala, G.Janoski, and A.Sung, "Intrusion detection: support vector machines and neural networks," in Proceedings of the IEEE International Joint Conference on Neural Networks (ANNIE), 2002, pp. 1702–1707.
[44] D.Tufis, C. Popescu, and R. Rosu, "Automatic classification of documents by random sampling," in Proceedings of the Romanian Acad. Ser., 2000, pp. 117 – 127.

[45] R. P. Lippman, D. J. Fried, I. Graf, J. Haines, K. Kendall, D. McClung, D. Weber, S. W. D. Wyschogrod, R. K. Cunningham, and M. A. Zissman, "Evaluating intrusion detection systems: The 1998 darpa off-line intrusion detection evaluation," in Proceedings of DARPA Information Survivability Conference and Exposition (DISCEX '00), January 2000, pp. 12–26.

[46] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, , and K. Das, "The 1999 darpa off-line intrusion detection evaluation," in Proceedings of Third International Workshop Recent Advances in Intrusion Detection (RAID '00), Toulouse, France, October 2000, pp. 162–182.

[47] J. Haines, L. Rossey, R. P. Lippman, and R. K. Cunningham, "Extending the darpa offline intrusion detection evaluation," in Proceedings of DARPA Information Survivability Conference and Exposition (DISCEX '01), June 2001.

[48] "Libsvm," http://www.csie.ntu.edu.tw/cjlin/libsvm/.