Host Based Intrusion Detection Systems Using Pattern Recognition

Abhinav Moudgil*, Ishit Mehta†, Shivam Khandelwal‡
International Institute of Information Technology
Hyderabad
Email: {*abhinav.moudgil, †ishit.mehta, ‡shivam.khandelwal}@research.iiit.ac.in

***Abstract*—Intrusions pose a serious security risk in a network environment. Although systems can be hardened against many types of intrusions, often intrusions are successful making systems for detecting these intrusions critical to the security of these system. Rare intrusion types, of which detection systems are unaware, are the most dicult to detect. Current methods and learning algorithms which rely on labeled data to train, generally can not detect these new intrusions. In addition, labeled training data in order to train misuse and anomaly detection systems is typically very expensive. We present an analysis of various classifiers over the KDD CUP 1999 dataset.**

***Keywords*—IDS, KDD**

I. Introduction

Network security has become more important to personal computer users, organizations, and the military. With the advent of the internet, security became a major concern and the history of security allows a better understanding of the emergence of security technology. The internet structure itself allowed for many security threats to occur. The architecture of the internet, when modified can reduce the possible attacks that can be sent across the network. Knowing the attack methods, allows for the appropriate security to emerge.

This paper concerns with intrusion detection systems and the related issue of identifying the important features contributing to detecting an anomaly or attack on a system. We show how various classifiers vary in terms of their training time and classification precision. The data we are using is the KDD 99 dataset developed for the KDD competition organised by DARPA in 1999 which is considered to be the de-facto standard for testing an IDS.
We begin by defining terms like attack, anomaly and IDS in the next section and then move onto feature selection and classification.

II. Defining Terms

As mentioned in [2], the terms - attack, anomaly and IDS are defined as follows:

A. Attack

In computer and computer networks an attack is any attempt to destroy, expose, alter, disable, steal or gain unauthorized access to or make unauthorized use of an asset.

B. Anomaly

Anomalies are deviation from normal user behavior. While, attacks are easier to predict and classify as they protray standard user behavior, it is often that the anomalies that create trouble. Also, if the system is exposed to a new type of attack, it has no idea of how to detect it.

C. Intrusion Detection System (IDS)

An intrusion detection system (IDS) is a device or software application that monitors network or system activities for malicious activities or policy violations and produces reports to a management station. IDS come in a variety of flavors and approach the goal of detecting suspicious traffic in different ways. There are network based (NIDS) and host based (HIDS) intrusion detection systems. NIDS is a network security system focusing on the attacks that come from the inside of the network (authorized users). Some systems may attempt to stop an intrusion attempt but this is neither required nor expected of a monitoring system. There are also Intrusion detection and prevention systems (IDPS), which are primarily focused on identifying possible incidents, logging information about them, and reporting attempts.

III. Dataset

The KDD Cup 1999 dataset [1] used for benchmarking intrusion detection problems is used in our experiment. The dataset was a collection of simulated raw TCP dump data over a period of nine weeks on a local area Network. The training data was processed to about five million connections records from seven weeks of network traffic and two weeks of testing data yielded around two million connection records. The training data is made up of 22 different attacks out of the 39 present in the test data. The known attack types are those present in the training dataset while the novel attacks are the additional attacks in the test datasets not available in the training data sets. The attacks types are grouped into four categories:

1) DOS: Denial of service e.g. syn flooding.
2) Probing: Surveillance and other probing, e.g. port scanning.
3) U2R: unauthorized access to local super user (root) privileges, e.g. buffer overflow attacks.
4) R2L: unauthorized access from a remote machine, e.g. dictionary attacks.

The 10% of the training dataset consisted of 494,021 records among which 97,277 (19.69%) were normal, 391,458 (79.24%)

TABLE I. CLASSIFICATION REPORT OF SVM

-	Precsion	Recall	F1 Score	Samples
Normal	93.8%	97.46%	95.38%	9840
Attack	99.37%	98.31%	98.84%	40160
Avg	98.19%	98.14%	98.15%	50000

DOS, 4,107 (0.83%) Probe, 1,126 (0.23%) R2L and 52 (0.01%) U2R connections. In each connection are 41 attributes describing different features of the connection and a label assigned to each either as an attack type or as normal.

IV. CLASSIFICATION FOR HIDS

A. Support Vector Machine

The data is first shuffled and then split with a 50-50 ratio into training data and validation data. The seed with which the data is shuffled is kept constant throughout all the classifiers in the future.
Initially we are labelling all the different types of attack as one label and the rest as normal.

1) Training: Due to lack of enough computational resources, we sampled 20% of the data consisting 100,000 samples. 50,000 of these samples with 41 features were used for training the model. A linear kernel was used for training and the regularization parameter c was kept as 1.

2) Testing: Half of the sampled data, viz. 50,000 samples, was used to test the trained model. An average accuracy of 98.19% was achieved. The classification report is as shown in Table 1.

B. Multilayer Neural Networks

10% of the dataset was used for training and another 2,000 samples were used for testing. The configuration of the neural network used:

$$41 - 25 - 25 - 5$$

The classification report and the confusion matrix are shown in Table 1 and 2 respectively.

The important thing to notice here is that the there are literally 0 U2R classifications. For the rest of the classifications, a decent precision value is obtained. Also, the training time for both SVM and MNN were large. The next section deals with reducing the training time by feature reduction.

TABLE II. CLASSIFICATION REPORT OF MNN

-	Precsion	Recall	F1 Score	Samples
DOS	100%	98.77%	99.38%	487
Normal	92.61%	96.07%	94.31%	509
Probing	98%	97.8%	97.9%	500
R2L	90.73%	96.36%	92.16%	502
U2R	0	0	0	28
Avg	93.97%	95.21%	94.58%	2026

TABLE III. CONFUSION MATRIX FOR MNN

-	DOS	Normal	Probing	U2R	R2L
DOS	501	0	1	8	0
Normal	0	458	4	18	0
Probing	1	11	470	7	0
R2L	4	15	4	494	0
U2R	0	17	0	13	0

V. FEATURE SELECTION

A. Principle Components Analysis

PCA is mathematically defined as an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by some projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on.

Consider a data matrix, $\underline{X}$, with column-wise zero empirical mean (the sample mean of each column has been shifted to zero), where each of the n rows represents a different repetition of the experiment, and each of the p columns gives a particular kind of datum (say, the results from a particular sensor).

Mathematically, the transformation is defined by a set of p-dimensional vectors of weights or loadings $w_{(k)} = (w_1, \ldots, w_p)_{(k)}$ that map each row vector $x_{(i)}$ of X to a new vector of principal component scores $t_{(i)} = (t_1, \ldots, t_p)_{(i)}$, given by

$$t_{k(i)} = x_{(i)} \cdot w_{(k)}$$

in such a way that the individual variables of t considered over the data set successively inherit the maximum possible variance from x, with each loading vector w constrained to be a unit vector.
We used 50,000 samples and reduced the number of features from 41 to 21 and 11. Then support vector machine classifier was used to classify the samples with reduced feature vectors. The precision of classification reduced at the cost of reducing the training time to nearly half. The results are showing in Figure 1 and 2.

B. Fast Feature Reduction

PCA is clearly impractical if a large dataset is used. To build an intrusion detection system which trains it model frequently as it is exposed new and more number of attacks and anomalies, we require a method which can find the principle features in the dataset faster than PCA, without making the system less robust.
Fast feature reduction(FFR) is one such method. Mathematically it can be described as follows.
Let $X = \{x_i\}_{i=1}^{N}$ be our intrusion detection dataset, containing N network connection samples containing all attack and normal connections. Each x_i is a vector with D dimension..

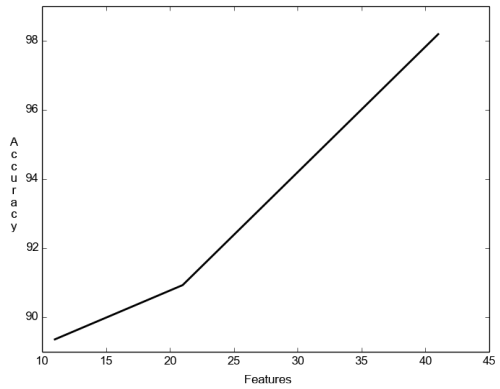

Fig. 1. PCA Accuracy

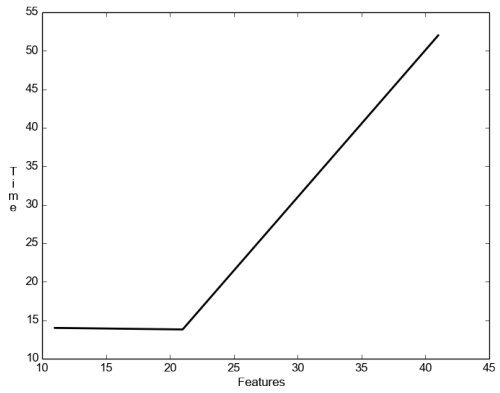

Fig. 2. PCA Time

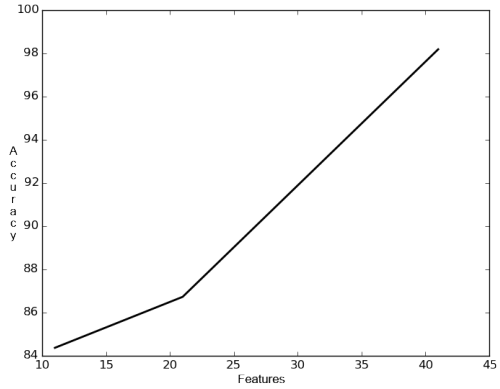

Fig. 3. FFR Accuracy

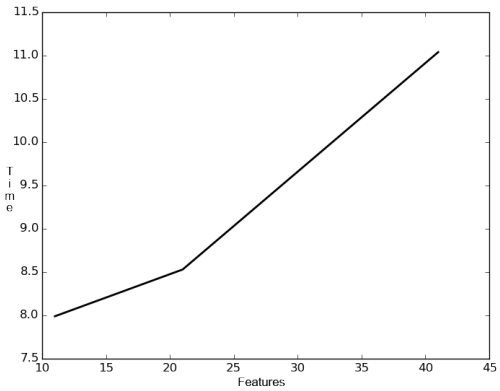

Fig. 4. FFR Time

Let $L = \{l_i\}_{i=1}^{N}$e our class labels for each sample. Also we have $l_i \in\ k\ where\ k = \{1, \ldots, K\}$ where we have K classes in our dataset with $K = 5$. x_{id}present the value of feature $d = \{1, \ldots, D\}$ in connection sample i. For each class first we add all data point x_i which are in same class, feature by feature:

$$S_{cd} = \{for\ all\ x_i\ where\ l_i = c \mid \frac{\sum x_{id}}{n_c}\}$$

Where n_c is the number of samples in class c and S_{cd} is mean of connection sample x_i along feature d whose class is c and $c \in k$. We put all S_{cd} in a vector and call it:

$$S_d = [S_1\ S_2\ \ldots S_{kd}]$$

After that we need to calculate how much S_d is scattered along each feature. In other we words we need to measure confusion of S_d for each feature. For this we need variance of S_d:

$$V_d = Var(S_d) = E[(S_d - \mu_d)^2]$$

Where μ_d is mean of all values in S_d and V_d is variance of S_d. Items in $V = \{V_d\}_{d=1}^{D}$ with high value represeting degree of turbulence in feature d among all classes. High turbulence represents power of corresponding feature in class separation. Now we can sort V with descending order and select t feature index for classification where t is numer of desired features that user has defined to be selected. The results are shown in Table IV and Fig. 3 4.

VI. Ensemble of Classifiers

The problem with classifers used in previous section is that there were a lot of false negatives for U2R classification. We tackled this problem by using an ensemble of 3 classifiers:

- Decision Tree
- Random Forests
- Gaussian Naive Bayes

TABLE IV. FFR Results

FFR	1.49s
SVM Training	2h 35m
Classification	35m
Accuracy	87.33%

TABLE V. CLASSIFICATION REPORT OF ENSEMBLE CLASSIFER

-	Precsion	Recall	F1 Score	Samples
Normal	100%	100%	100%	193340
DOS	100%	100%	100%	48091
Probing	99%	99%	99%	2011
R2L	99%	96%	98%	554
U2R	71%	68%	69%	25

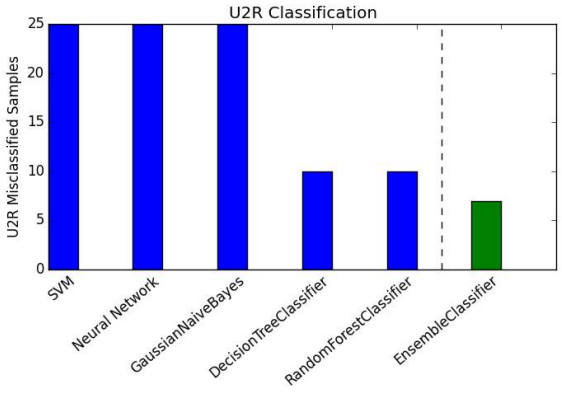

Fig. 5. U2R Misclassification

The dataset was split into test data and train data after shuffling using the same seed used in SVM and MNN. 2,50,000 samples were used for training and the rest were used for validation. A soft voting scheme was to make the final classification decision by considering the majority of the votes of the three classifiers. The results are as show in Table V.

VII. EXISTING CRITICISM OF THE KDD DATASET

Criticism of the KDD datasets has grown over the last ten years, and centres on the dated nature of the underlying experiment. These datasets represented the first real attempt to provide a benchmark for intrusion detection research and undeniably had a seminal impact in the field. At the time, there was no standardised way of evaluating IDS performance and DARPA attempted to remedy this through collaboration with various academic institutions.

The KDD experiments used a Solaris-based system to collect a wide range of data. Given the dynamism of computer technology, however, the relevance of the KDD data rapidly diminished in practical terms. The Solaris operating system was selected largely because of its ease of integration and monitoring; whilst a perfectly functional OS, Solaris did not and does not have a large market share of the computer industry. Logically, any results produced on this operating system hence have limited generalisability to the cyber community as a whole.

Furthermore, the technology of the late 1990s has very little resemblance to the technology of the current era. Operating systems have undergone many fundamental revisions in that time, and innovative development has drastically changed the basic nature of computer systems. Even something as simple as the conversion from 32 bit architecture to 64-bit systems has a profound effect on the exploitability of contemporary systems.

VIII. CLASSIFICATION FOR NIDS

A. HTTP Dataset

The HTTP dataset CSIC 2010 contains thousands of web requests automatically generated. It can be used for the testing of web attack protection systems. It was developed at the Information Security Institute of CSIC (Spanish Research National Council).

The HTTP dataset CSIC 2010 contains the generated traffic targeted to an e- Commerce web application developed at our department. In this web application, users can buy items using a shopping cart and register by providing some personal information. As it is a web application in Spanish, the data set contains some Latin characters.

The dataset is generated automatically and contains 36,000 normal requests and more than 25,000 anomalous requests. The HTTP requests are labeled as normal or anomalous and the dataset includes attacks such as SQL injection, buffer overflow, information gathering, files disclosure, CRLF injection, XSS, server side include, parameter tampering and so on. This dataset has been successfully used for web detection in previous works [4, 5, 6, 7, 8, 9].

The traffic is generated following the next steps:

First, real data are collected for all the parameters of the web application. All the data (names, surnames, addresses, etc.) are extracted from real databases. These values are stored in two databases: one for the normal values and other for the anomalous ones. Additionally, all the public available pages of the web application are listed.

Next, normal and anomalous requests are generated for every web page. In the case that normal requests have parameters, the parameter values are filled out with data taken from the normal database randomly. The process is analogous for anomalous requests, where the values of the parameters are taken from the anomalous database.

Three types of anomalous requests were considered:

1) Static attacks: try to request hidden (or non-existent) resources. These requests include obsolete files, session ID in URL rewrite, configuration files, default files, etc.
2) Dynamic attacks: modify valid request arguments: SQL injection, CRLF injection, cross-site scripting, buffer overflows, etc.
3) Unintentional illegal requests. These requests do not have malicious intention, however they do not follow the normal behavior of the web application and do not have the same structure as normal parameter values (for example, a telephone number composed of letters).

The dataset has 223585 instances with 18 attributes.

B. Decision Trees

In Decision trees, the nodes are split using a single feature, obtaining binary split at each level. The decision of choosing

TABLE VI. CLASSIFICATION REPORT OF DECISION TREES

-	Precsion	Recall	F1 Score	Samples
Anomaly	66%	91%	77%	59847
Normal	82%	46%	59%	51946
Avg	73%	70%	69%	111793

the right feature for splitting was based on the entropy gain respective to each feature. Let S be a data set and A a feature with values $v \in V$, and let E denote Shannons entropy function. Moreover, let S_v denote the subset of S for which the feature A has the value v. The gain of a split along the feature A, denoted $G(S, A)$ is:

$$G(S, A) = E(S) - \sum_{v \in V} \frac{|S_v|}{|S|} E(S_v)$$

The results obtained are show in Table VI.

IX. CONCLUSION

We started out with SVM and MNN and obtained decent results with the only problem of U2R being missclassified. Also, the time taken for model training and classification were extraordinaly huge. So, feature reduction using PCA made sense. But, as PCA works with high dimensional matrices, it takes time. So a less robust and accurate method called FFR was used. This still didn't solve the problem of U2R missclassification which was solved by using an enseble of classifiers. Further we built a Network based Intrusion Detection System using Decision Trees.

ACKNOWLEDGMENT

We would like to thank the TAs for their mentorship. We would also like to thank Prof. Avinash Sharma for teaching us Statistical Methods in Aritficial Intelligence.

REFERENCES

[1] Mukkamala, S.; Janoski, G.; Sung, A., "Intrusion detection using neural networks and support vector machines," in Neural Networks, 2002. IJCNN '02. Proceedings of the 2002 International Joint Conference on , vol.2, no., pp.1702-1707, 2002

[2] Parsazad, S.; Saboori, E.; Allahyar, A., "Fast Feature Reduction in intrusion detection datasets," in MIPRO, 2012 Proceedings of the 35th International Convention , vol., no., pp.1023-1029, 21-25 May 2012

[3] CUP, K., Intrusion Detection Dataset. 99.