A NOVEL TWO-STAGE CLASSIFIER WITH FEATURE SELECTION FOR INTRUSION DETECTION

A DISSERTATION

Submitted by

AJAY KAMATH

(PG/FT/132112207)

In partial fulfilment of requirements for the award of Degree of

MASTER OF TECHNOLOGY

In

INFORMATION SECURITY

Under Esteem Guidance Of

Dr. Praveen Kaushik
(Asst. Professor in Dept. of CSE & IT)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
MAULANA AZAD NATIONAL INSTITUTE OF TECHNOLOGY,
BHOPAL (M.P.) 462051
JUNE 2015

CANDIDATE'S DECLARATION

I hereby declare that the work, which is presented in this thesis, entitled "**A novel two-stage classifier with feature selection for Intrusion Detection"** in partial fulfilment of the requirements for the award of the degree of **Master of Technology** in Information Security, submitted in the **Department of Computer Science & Engineering,** Maulana Azad National Institute of Technology Bhopal is an authentic record of my own work carried out from Jan 2015 to June 2015 under the noble guidance of my thesis guide **Dr. Praveen Kaushik,** Department of CSE & IT, MANIT Bhopal.

The further declaration is that the matter embodied in this thesis has not been submitted by me for the Award of any other degree.

AJAY KAMATH

CERTIFICATE

This is to certify that the thesis entitled "**A novel two-stage classifier with feature selection for Intrusion Detection",** being submitted by **AJAY KAMATH, Sch. No: 132112207** to the Maulana Azad National Institute of Technology, Bhopal, for the award of the degree of **MASTER OF TECHNOLOGY,** is a bonafide record of work carried out by him under my supervision. The contents of this thesis have not been submitted and will not be submitted to any other Institute or University for the award of any degree or diploma.

Dr. Praveen Kaushik,

Assistant Professor,

Department of CSE & IT,

MANIT, Bhopal.

ACKNOWLEDGEMENTS

I would like to express my deep gratitude and indebtedness to my supervisor **Dr. Praveen Kaushik** for the encouragement and guidance. His invaluable meticulous and critical supervision in each and every phase of this work inspired me in innumerable ways.

I also express my gratitude to **Dr. Nilay Khare,** HOD, Department of Computer Science & Engineering and Information Technology, who encouraged me in many ways and provided all the help that he could as the head of the Department, for a smooth completion of my research work.

I express my special gratitude to **Dr. Appu Kuttan K.K,** Director, MANIT, for permitting us to utilize all the necessary facilities of the college.

Special acknowledgements are due to all the faculty members of the CSE & IT Departments and other supporting staff for helping during my M.Tech course.

Finally, I wish to express my sincere gratitude to my family members for their patience and sacrifice throughout the work. This thesis is dedicated to them as a token of my gratitude and love.

AJAY KAMATH

ABSTRACT

A Network Intrusion Detection System (NIDS) is a mechanism that detects illegal and malicious activity inside a network. Network anomaly detection is an important and dynamic topic of research. It involves comparing the suspicious malicious activity with the normal behaviour of the system. Anomaly based Intrusion detection systems use machine learning techniques to detect novel attacks. Classifiers are predictors that estimate the class label of an attack based on prior training and learning models. While it can be seen that DoS (Denial of Service) and probe attacks are filtered with reasonable accuracy, the detection rate fails miserably for R2L (Remote-to-Local) and U2R (User-to-Root) attacks. This dissertation aims to improve the accuracy of the above mentioned attacks by proposing a two-stage classifier with feature selection used in the second stage of classification. The first stage uses a simple NB classifier with all the trained features. Feature selection is the process of filtering out useful and relevant features that contribute to an attack class. The proposed feature selection technique is based on Genetic Algorithm with entropy based weights used for giving importance to each feature in the fitness function. Experiments were conducted on the NSL-KDD dataset using the WEKA machine learning tool. It can be seen that the detection rate of R2L and U2R improves significantly with 86.2% and 95% enhancement in the second stage of classification. This dissertation also compares the proposed feature selection technique with the existing filter methods and also inspects the accuracies of other classifiers.

Keywords

Intrusion Detection System (IDS), anomaly detection, Machine Learning, Naïve Bayes classifier, Feature selection, Genetic Algorithm

LIST OF FIGURES

LIST OF TABLES

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Intrusion Detection

With the advancement in network based services and information transfer, it is highly imperative to provide security. Generally, when an application is connected to a network, it becomes vulnerable to attacks by malicious intruders who try to steal information. An intruder masquerading as a legitimate user tries to steal information and compromise the security**.**

Organizations generally have certain amount of critical data and applications that need to be secured to protect their confidentiality, integrity and availability. Intruders can be illegitimate users trying to gain access to a network, running harmful scripts or destroying important files or a legitimate user trying to gain super-user privileges to perform malicious activities. Intrusions can also be an attacker scanning all open ports, IP addresses for any vulnerability as well as denying a legitimate user an access to a particular service. While organizations employ firewalls to detect Viruses, Trojans, Worms, harmful software and application data, detection of novel intrusions that do not have any known patterns or signatures remains a major issue. Firewalls detect only known virus patterns and help the network security administrator to monitor the network. The databases of viruses need to be updated regularly to prevent the system from lagging behind in the security aspects. Each organization develops a risk assessment model to have an in-depth idea of the level and the nature of modern network security threats, determining their vulnerabilities as well as the deployment of firewalls and other prevention systems.

According to [5], Intrusion Detection is the process of monitoring the events occurring in a computer system or network and analysing them for signs of intrusions, defined as attempts to compromise the integrity, availability and the confidentiality of information or to bypass the security mechanisms of a computer or network. With the ever increasing dependence on information systems and the use of intra-network connectivity within the organizational network, an Intrusion Detection System (IDS) protects the system from threats. It raises alarms when any suspicious activity is found which can then be supervised upon by the network security administrator or the Intrusion Prevention System (IPS). With bulky data commonly traversing across the network, the amount of alerts generated by these systems is very high. Moreover, these systems are not perfect in detecting attacks with increasing number of novel attacks found every day. They are also optimized with latest state of the art techniques to automate and mitigate the task of the security expert. Nowadays, organizations use Intrusion Detection Systems coupled with a firewall to secure the network.

1.2 Deployment of IDS

Depending upon the operating environment, the network infrastructure and the resources available within an organization, it is necessary to evaluate the deployment strategies for an IDS sensor software [6]. IDS require adequate and appropriate human interaction at all times. The alarms generated by an IDS must be monitored by a security administrator to take appropriate action. IDS works along with an IPS (Intrusion Prevention System) that judges the severity of an alarm and ideally responds to an intrusion. (Intrusion Response System). It takes into consideration the risk assessment model developed by the organization. The security policies, plans and procedures must be known to all personnel handling IDS output to take appropriate action [4]. Based on the deployment of sensors, IDS can be classified in Network based IDS and Host based IDS [1].

Deployment of Network Based IDS

IDS sensors monitor the network traffic to find malicious activities. They perform ingress filtering as well as intra-network monitoring. They can be deployed at various points along the network infrastructure, along the main switch, or before the router connecting two LANs as well as in the DMZ (Demilitarized Zone). The deployment of IDS sensors in the networked LAN can be shown in Figure 1.

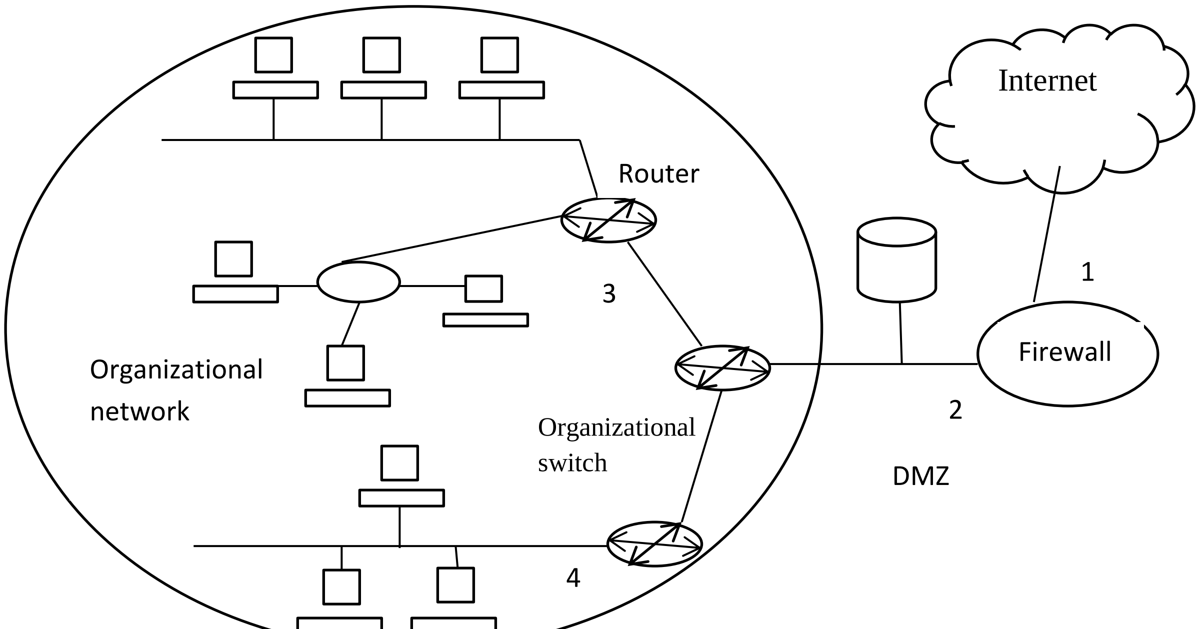


Fig. 1 Deployment strategies for NIDS

Location 1: Outside the DMZ, deploying the IDS here will help in identifying the attacks directly as it is directly connected to the outside internet. The ingress traffic will be given to the firewall to provide extra level of protection.

Location 2: In the DMZ (Behind external firewall), deploying the IDS here will help in identifying the flaws in the network firewall and even if ingress traffic is not recognized, it can help understand the outgoing traffic results from a compromised server.

Location 3: Between different LANs. Helps monitor the traffic between different intra-networked LANs having different policies. It can provide access control with different levels of security for each department.

Location 4: Within a LAN. This is perhaps the easiest form of deployment for an IDS. i.e. Behind the LAN router. It will filter the incoming and the outgoing traffic from the LAN, helping in securing the LAN from the external world.

Deployment of Host Based IDS

A Host Based IDS typically monitors the activities of a single host. It acts in conjunction with a firewall or anti-virus software to detect attacks. It can be used to capture the ingress traffic of a standalone application server as well as to examine the behaviour of a host. It can provide enhancement in the levels of security for our system. It will become extremely tedious, computationally expensive as well as costly for organizations to deploy an IDS sensor for each host. As a result, these are used only for select hosts and honeypot systems as a tool to inspect the traffic.

1.3 Types of Intrusion Detections

There are two main types of intrusion detection methodologies. Signature-based Detection (SD) and Anomaly-based Detection (AD).

A signature is a pattern or a string that corresponds to a known attack or threat [1]. *Signature Detection (SD)* or misuse detection compares the incoming packets with the existing database of attacks. It uses pattern matching algorithms and rule based approaches to find a match in the previously defined rules. This method has a very low false positive rate. False positives are the number of false alarms or the number of normal activities that are classified as attacks. This technique is the same as most of the antivirus software and firewalls that detect known patterns. The main disadvantage of this detection is that it cannot detect previously known attacks.

Signature Detection is most commonly deployed along with firewalls and other thwarting mechanisms to enhance their performance. According to [4], signature detectors were initially rule-based expert systems, however, with the gradually increase in the number of attacks, these detectors were not suitable to perform real-time detection. As the rule set becomes larger and larger, the comparison speed of these expert systems decreases. Fuzzy rule based systems and the use of other Artificial Intelligence techniques have made an improvement in the pattern matching techniques developing new rule sets based on existing rules to detect newer attacks. SNORT and P-BEST are most commonly used misuse detectors. Figure 2 shows the components of SNORT.

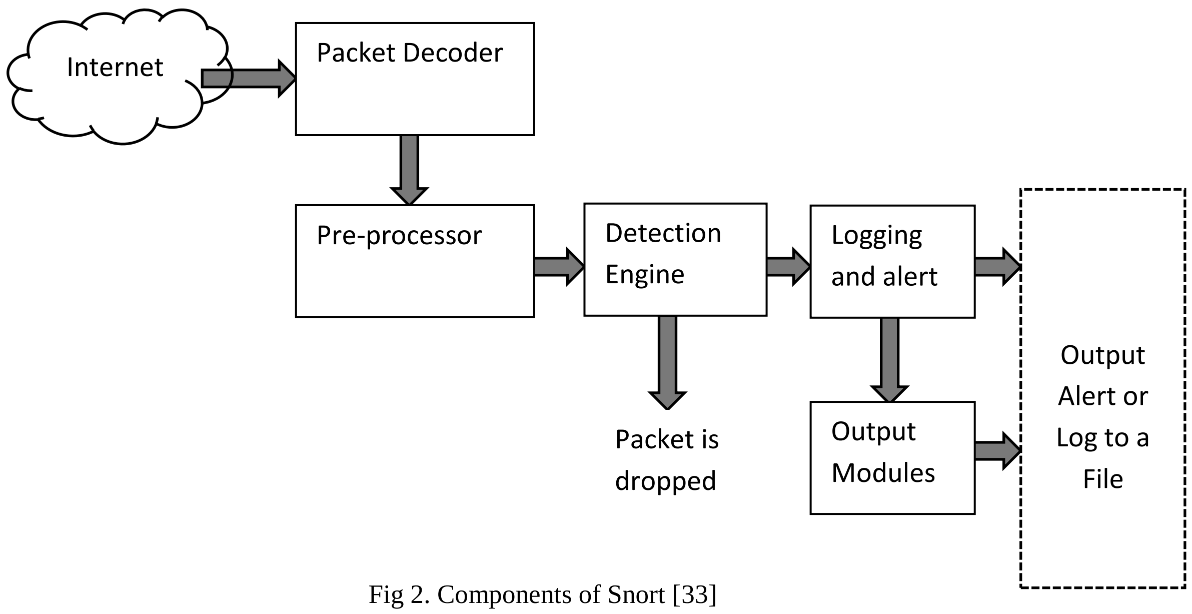


Fig 2. Components of Snort [33]

SNORT is a rule based Network Intrusion Detection System that uses fast pattern matching algorithms to detect known patterns in packets [33]. These rules are created with the help of human interaction and stored in databases for comparison by the Snort detection engine. The pre-processor component performs initial packet processing such as the analysis of the IP header, feature identification as well as separation of the packet payload. The detection engine compares the packets with the existing rule set and generates alert output into a log file or on the console. SNORT can also be run in inline mode where it can prevent attacks (IPS). The main disadvantage of using misuse detection is their constant need for updating the rule database.

In stark contrast to this approach is the *Anomaly Detection (AD)* method which assesses the incoming packets for any abnormality. An Anomaly is a deviation from the normal behaviour. It compares the packet with the normal behaviour that is predefined. For e.g. If the number of failed login attempts is more than the specified threshold, then it can be termed as an anomaly. The main advantage of using this detection is that it can detect previously unknown attacks.

The main problem in anomaly based detection is the definition of the normal behaviour of the system. For example, consider an employee of an organization who works for five days a week from 9:00 am to 5:00 pm. He logs in into his system at 9:00 am using a password which is only known to him. Here, an anomaly could be an 'unknown' user logging into his system at 8:00 pm late outside office hours, or a user logging in on a holiday or any failed number of login attempts based on organizational protocols. Sometimes, even that employee could have forgotten his correct password thereby having a failed login attempt anomaly. However, since he is a legitimate user, the system should grant him privileges to obtain his password instead of raising an alarm of a suspicious intrusion. Anomalies in behaviour can be identified by analysing audit data or by conducting a 'forensics' of the network and servers. Sometimes the data collected by Honeypot systems and historical data are used to create rules and protocols to establish normal behaviour. Some processes monitor the statistical measures like the amount of rejection rate (error rate), network bandwidth, CPU utilization or the number of requests to an application server. There measures are currently used in commercial systems because their identification of flaws is a straightforward process. These are threshold based anomaly detectors.

The actual use of anomaly detectors is in detecting unknown attacks. According to [34], unknown attacks can be variations in previously known attacks. Denial of Service (DoS) attacks thwarts a legitimate user from accessing a particular resource or service. For example, flooding the network with unwanted packets will clog the bandwidth, thereby hindering the smooth flow of traffic across the network. Similarly, SYN Flooding, ARP poisioning are variants of DoS attack. The ping of death (PoD) attack divides a packet whose length is greater than 65535 bytes into various fragments which will flood the destination with limited buffer length. All attacks are a result of flaws in protocol configuration. One solution to prevent the occurrence of these attacks is to make a perfect protocol that is robust in terms of security. However, when this happens performance of the network degrades significantly. Most of the attacks are due to improper exception handling by the protocols. Hence, scrutinizing exceptions in the network protocols helps in detecting unknown attacks.

Another important type of anomalies are the probe attacks. i.e Stealth scanning of the network for open IP addresses and ports. Usually, when an attacker wants to carry out an illegal activity or hack into a system, he has to carry out a reconnaissance of the network. He may find out all the open network address points through which he can attack, find out the OS running on the systems, run various tools to 'probe' the network. Nmap, TCP Stealth port scanning, FTP Bounce attack, SATAN tool find flaws in the protocol design to probe the network. While this attack type is not dangerous, its detection will help in building better network models, protocols and develop secure organizational rules.

Sometimes, anomalies can be detected by analysing the number of files created in the root directory, or the number of outbound commands in an FTP session. For example, if the FTP Server is not write protected and the root directory is owned by the FTP account, then the attacker can write malicious scripts from the remote machine to gain local access to the system [3]. To detect this attack, it is necessary to check for anonymous FTP sessions and whether any files are created in the root directory. Some FTP Servers allow a guest to login and download the files from the server. However, if the guest account is allowed to write to the FTP server, then the attacker can upload *"warez"* copies of the illegal software to the FTP server and gain local access. This compromises the server and the attacker gains local access. The legitimate clients can download the illegal *"warez"* software on their machines to perform the warezclient attack. The previous attack is called as warezmaster. Regulating whether a file is being uploaded or downloaded in an FTP session can help detect abnormalities in behaviour.

While Signature based detection involves comparison with known signatures, better accuracy in detection can be achieved here than anomaly detection. A large number of detectors are based on probabilistic models which provide no 100% surety that an activity is malicious or a normal one. Hence, anomaly detectors are not deployed in real environments due to their non-robust performance. As a result, efforts are made by the research community to continuously improve the efficiency of detection and reduce the false alarm rate as well as to automate the task of alert classification and intrusion prevention.

With regards to what was discussed earlier, a general architecture of an IDS system is shown in Figure 3. The components of an IDS are discussed below.

Data Acquisition module**:** It is responsible for acquiring data from the network, either from the upper layer protocols, or raw network frames or from audit data to analyse patterns in behaviour.

Packet Decoder: Performs the initial pre-processing of the packet to obtain initial features. It strips all the headers off the payload and acquires the features to perform detection. These features can be related to basic packet information, the source bytes, the destination address to which the packet is to be transported etc. or temporal features such as the number of packets received from a particular source in one second, or even payload information such as the number of TCP SYN packets successfully received etc. as well as the content.

Detection Engine: Compares the features identified with existing rule set or with the normal behaviour of the system to identify anomalies. When an event is found to be malicious, then an alert will be generated and logged into the output module as well as given to the *response management block*.

***Intrusion Response System** (IRS)*: It is the response management block which obtains the alert, calculates the severity of the alert and takes adequate response.

Knowledge Base: This is the expert system that takes in the reference data from the detection engine and the expert knowledge on attacks to create new set of rules. It examines behavioural patterns in data to establish new models to identify attacks.

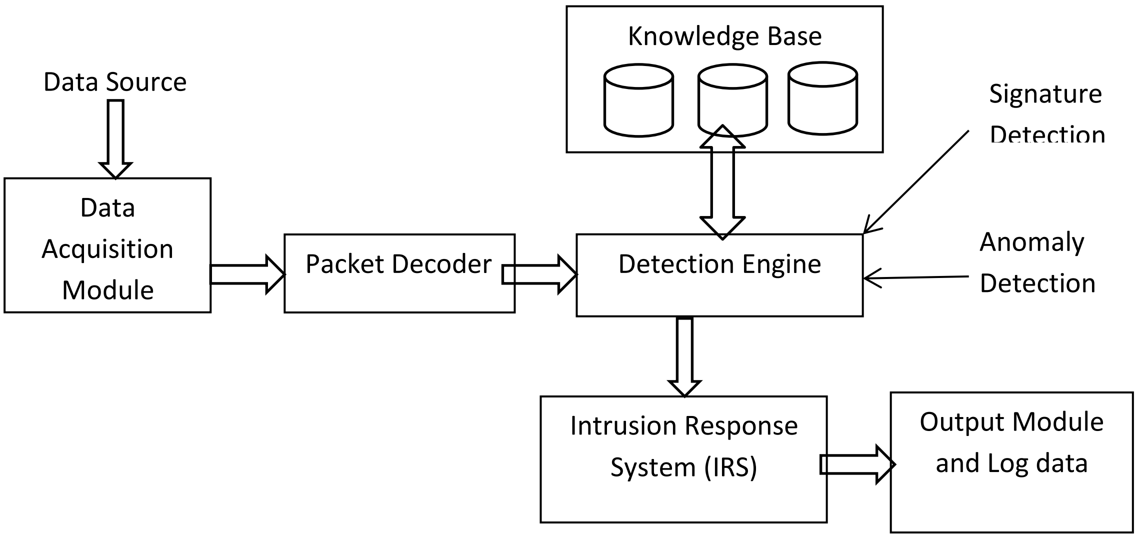


Fig. 3 Architecture of an Intrusion Detection System

1.4 Machine Learning in Anomaly Detection

Machine Learning is a vast field of Artificial Intelligence that builds models based on existing input data to make predictions or decisions instead of following a standard set of instructions to achieve a certain output. While expert systems are used in misuse detectors, anomaly based detection employs machine learners as well as other artificial intelligence techniques to estimate the class of an activity. i.e. whether an activity is an anomaly or not. The main idea behind using machine learning in anomaly detection is to develop predictive models based on the training dataset to perform the detection. The improvement should be so significant that the model estimates most of the novel malicious activities with a reasonable error rate (low false alarm rate). The learner should be well trained to include all different types of known attacks with large number of examples. Generally, the dataset is built using audit data from the routers, server logs or other maintenance records. One such dataset is the

DARPA dataset developed by the MIT Lincoln Laboratories in 1998 based on the monitoring of five weeks of raw TCP dump data [20]. They had used different Windows NT servers, BSD Solaris and Unix Servers and then simulated various attacks to obtain raw TCP dump data. The first three weeks were training data and two weeks of test data to test the accuracy of the learners. The DARPA dataset for Intrusion detection has become a benchmark dataset used by researchers to train their learner. Similarly datasets used are the KDD Cup 99' dataset, NSL-KDD dataset (2010) and the latest ISCX (2012) dataset. Many raw tcpdump data obtained by various organizations are also freely available online for research purposes.

Anomaly detectors use both *supervised learning* where the learners are trained with examples having class labels of attacks, as well as *unsupervised learning* where the learners themselves find its own structure of classifications. Each learning algorithm has its own set of advantages and disadvantages and the detection rate of attacks vary according to the type of classifier used. Researchers have built an ensemble of classifiers, used hybrid multi-stage classifiers as well as used optimization techniques to develop accurate algorithms. While researchers face tough competition between themselves in this field, the deployment of anomaly based detectors remains an open issue because of their stochastic nature of output obtained. Figure 4 gives a clear taxonomy of machine learning algorithms used in anomaly detection.

Based on whether the classes of attacks are provided or not, learners are classified as supervised learners or unsupervised learners. Most commonly used supervised learning algorithms are the Classification and Regression algorithms. Simple Naïve Bayes classifier, Bayesian networks, Hidden Naïve Bayes, Logistic Regression, Decision Trees such as ID3, C4.5, Random Tree classifiers, or CART (Classification and Regression Tree) etc. are used. Similarly Backpropagation based Neural Networks (ANN) and Support Vector Machines (SVM) classify an activity as an anomaly or a normal one. Unsupervised learners are the Clustering algorithms such as Simple K-Means, K-Nearest Neighbour (K-NN) etc. as well as Self-Organising Maps (SOMs) based Neural Networks finding patterns in training data to cluster in into useful classes. Stochastic processes such as the Markov process (MM), Hidden Markov Model (HMM) are used for detection of attacks. Another vast field of artificial intelligence is the field of Evolutionary computing. Genetic Algorithms (GA), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Simulated Annealing (SA) and other optimization techniques are used in hybrid classifier ensemble to either optimize the parameters of the main classifier, or to optimize the threshold for detection. Efforts are made since the 1990s to develop efficient models using hybrid, parallel or hierarchical structure of machine learners to achieve better results. One sun example of a statistical anomaly detector is HIDE presented in [7]. Hierarchical Intrusion DEtection (HIDE) system, detects network based attacks as anomalies using statistical pre-processing and neural network classification. The system was tested using Perceptron, Backpropagation (BP), Perceptron-backpropagation hybrid (PBH), Fuzzy ARTMAP, and Radial-based Function to find that BP and PBH provide efficient classification for data than the alternatives. Another example of a good non-commercial detector is PAYL (Payload-based anomaly detector) that can successfully detect outliers of network servers [8]. It applied a noise-reduced fuzzy support vector machine (fSVM) to improve the detection rate at lower false positive rate.

Depending upon the similarity in functioning, learning algorithms can be classified into the following categories:

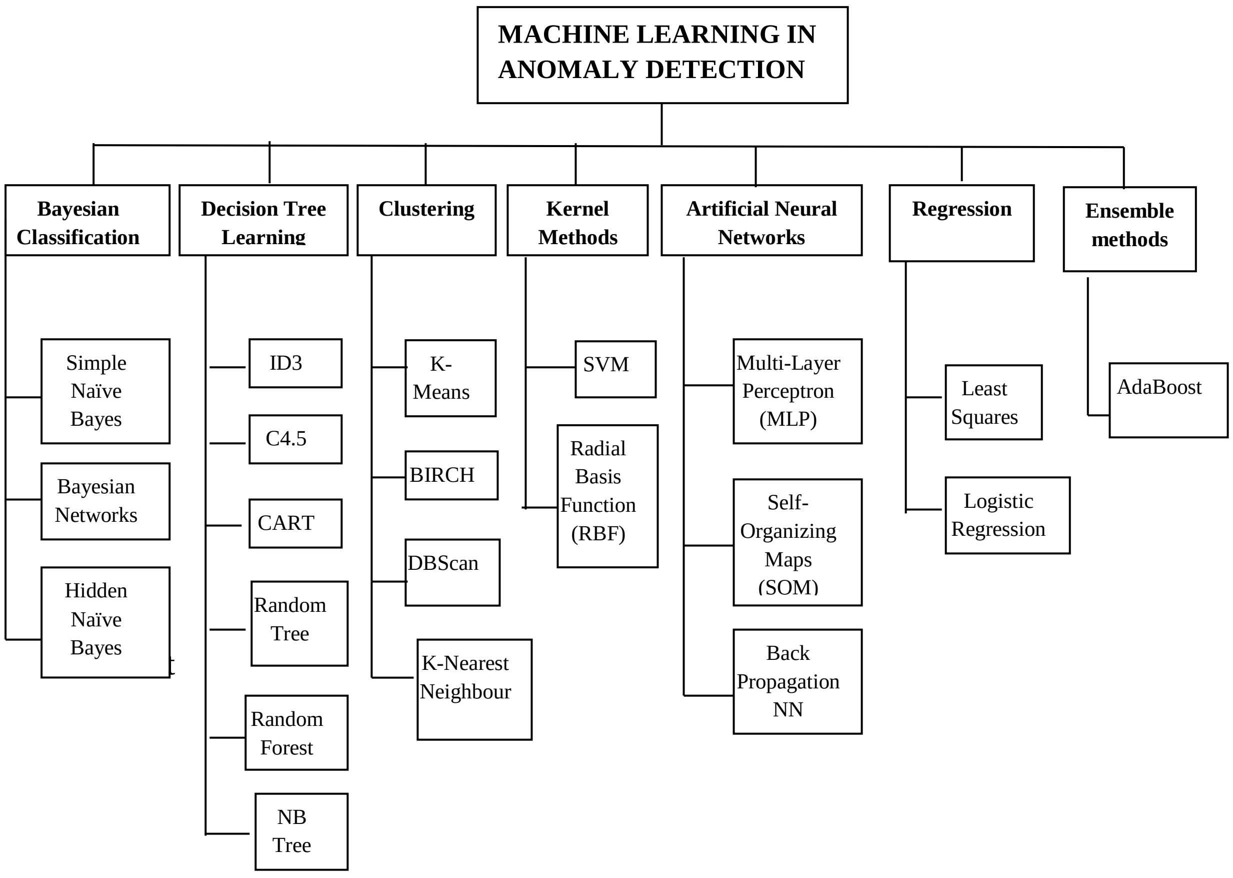


Fig. 4 Taxonomy of various machine learning algorithms used in anomaly detection

Bayesian Classification: They are stochastic techniques that explicitly employ the Bayes' Theorem of conditional probability to perform classification. Eg. Naïve Bayes, Bayesian Belief Network, Hidden Naïve Bayes (HNB). Bayesian networks are reliable classifiers in a multi-class setting. Simple NB algorithm assumes mutual independence amongst the attributes. Several techniques which calculate the conditional probability between the attributes to various complex Bayesian Networks have been proposed.

***Clustering*:** Use Centroid based or Hierarchical methods to organize data into groups of maximum commonality. Eg. K-Means, BIRCH Hierarchical clustering, DBScan algorithm. Clustering usually give less efficient results than classification due to the absence of labelled training examples. Clustering based learning is used to detect previously unknown attacks, although classification methods are very popular. The main task of trained clustering is to partition the training data into *k clusters* using the most commonly used Euclidean distance similarity. Several hybrid techniques inspired by evolutionary computing, TANN (Triangle Based Nearest Neighbour) based K-Means (K-Nearest Neighbour), or even decision trees are used in conjunction with clustering.

Kernel based methods: In such classifiers, it is assumed that the training data consists of only a single class. The main learning goal during training is to determine a kernel function which classifies the data points along the boundary into the single class and separates the outliers. This belongs to semi-supervised learning. SVM uses a maximum margin classifier separating the two classes (binary classification). Outlier detection is commonly employed using SVM. It takes a long time when the training dataset is very large. However, provides efficient results when combined with other classifiers. Optimization techniques like simulated Annealing, GA are used to adjust the parameters of SVM. Researchers have used decision trees, clustering and SVM in a Dynamically Growing Self-Organizing Tree (DGSOT) [35]. Clustering is done in parallel with the training of SVM to improve the training rates of SVM.

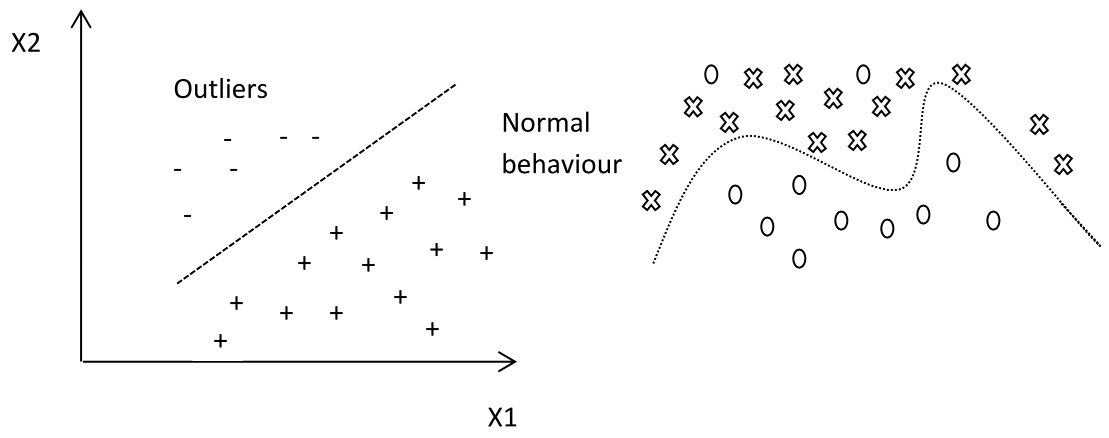


Fig 5. Linear Classification (Used by SVM) vs Non-linear classification

Outliers refer to the data points in the training set that are most unlikely to occur given the model of data.

Decision Tree learning: Most commonly used decision trees are the ID3 (Iterative Di-chotomizer 3), C4.5 (J48) tree, random tree, random forest, NB Tree, CART that can be "learned" by splitting the source feature set into subsets based on attribute value test. Each interior node represents one of the input variables, the edges to children for each of the possible values of that input variable. Classification trees are most commonly used in anomaly based IDS. The KDD Cup 99' contest winner used the C4.5 decision tree learner to achieve an overall average cost of 0.2331 per test sample with the detection rate of Normal, probe, DoS, U2R and R2L records being 99.5%, 83.3%, 97.1%, 13.2% and 8.4% respectively [29].

A simple decision tree for identifying a harmful connection on source_port 5603 with the ip_address attribute in the root node (high gain attribute) is given in the Figure 5.

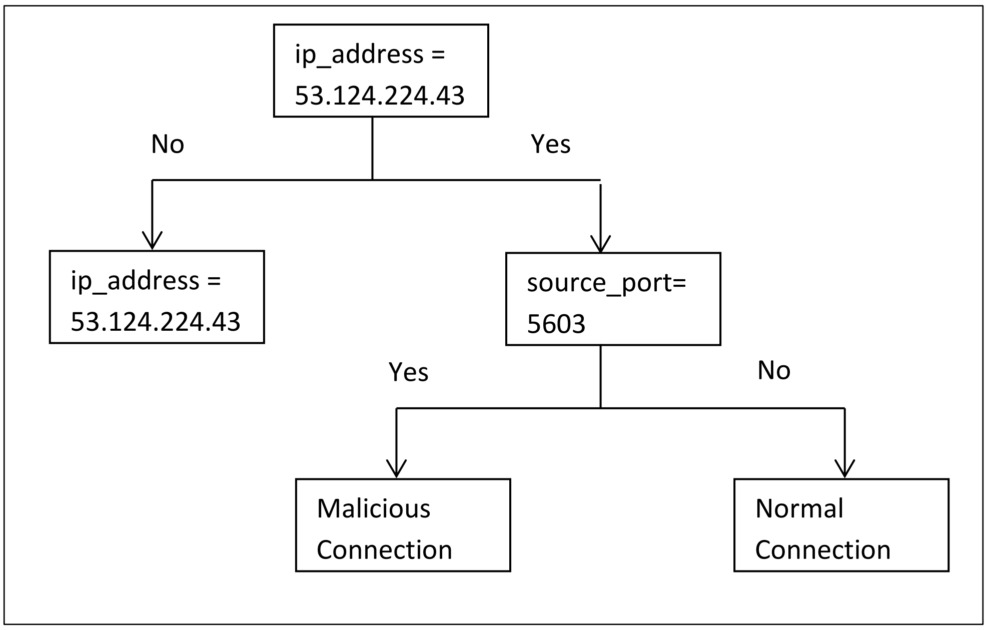


Fig 6. A simple Decision Tree example to find a malicious connection

Decision Tree are used with fuzzy logic to create fuzzy rules such as

if source_ip = 192.168.43.23 || dst_bytes ≥ 4000 || source_port ≥ 3000 || num_hot_ind ≥ 1 then connection= malicious

They can be used to write IDS and firewall rules along with identifying outliers.

Artificial Neural Networks: The brain constitutes interconnections of neurons that perform operations such as perception, pattern recognition, motor control etc. several times faster than a computer. A Neural network gains knowledge of the environment through learning which changes the weights or the interconnection strengths of the network to perform the desired classification task. Perceptron, Backpropagation NN and Self-Organizing Maps (SOM) are examples of Neural Networks. Self-Organizing Maps (SOM) are unsupervised learning algorithms. A detailed explanation of SOMs is given in [21].

Soft-Computing Approaches: Fuzzy Logic is used to analyse the likelihood of the occurrence of certain attacks. It is used to set the threshold for rule for specific attacks. Fuzzy association rules are used to describe normal and anomalous classes. With fuzzy logic, the intrusive activities can be clearly defined and false alarm rate can be reduced because intrusion activities are a form of uncertainty. Researchers have devised rules and those are converted to mathematical equivalents by the fuzzy expert system. Fuzzy rules are used in Markov Models (MM). Evolutionary computation techniques such as Genetic Algorithms (GA), Ant colony Optimization (ACO), Swarm Intelligence as well as Artificial Immune Systems (AIS) have been used by researchers all along.

1.5 Thesis Overview

It can be seen that anomaly based intrusion detection system is an exciting area of research because of its lack of efficient results with real-traffic. Artificial intelligence used to detect anomalies is a vast area with researchers developing innovative and contemporary techniques to expose novel attacks. As a result, it was necessary to narrow down this broad area by conducting a literature review of the existing techniques. This is discussed in the next Chapter 2. A theoretical foundation giving details about Naïve Bayes classifier, feature selection and genetic algorithm is also discussed in Section 2.2.

After a thorough investigation of the prevalent procedures in the field of IDS, several research gaps were identified and the problem statement for the dissertation was formulated in Chapter 3. The main objectives are stated in Section 3.3 with the intended outcomes in Section 3.4. The proposed algorithm to improve the security is discussed in Chapter 4 with the experimental setup and coding in Chapter 5. The results are obtained in Chapter 6 and compared with previously existing techniques. The end inferences from this undertaking are stated in Chapter 7.

CHAPTER 2

LITERATURE REVIEW

2.1 Related Work

There are many related works related to the design and functioning of IDS. Since 2000, a variety of machine learners, classifier ensembles and search and optimization techniques are used by researchers to classify activities as anomalies. A competition was conducted by Knowledge Discovery and Data Mining (KDD) in 1999. The main task of this competition was to build a network intrusion detector, a predictive model that was able to distinguish "bad" connections from "normal" connections [36]. Researchers use the dataset provide by DARPA, KDD99, ISCX12 etc. to construct their predictive classifiers. It was necessary to make an in-depth review of different techniques, understanding their advantages and limitations. This was done by studying review papers [1-2, 4].

Hesham Altwaijry and Saeed Algarny in [9] developed a Bayesian filter to improve detection rates of attacks in the KDD99 dataset. They calculated a score of each feature based on the frequency of occurrence for "normal" records and for "attack" records. They selected features with a higher score to test their filter. They also tested their filter using all features selected. It was discovered that the results obtained were worse when all features are selected. Features 23, 24 and 31 were found to be most relevant with the detection rate of R2L improved to 85.35% and a threshold value of 0.6. In [10], they proposed a multi-stage Naïve Bayes filter with each stage trained to detect a different type of attack. The multi-stage filter is shown in Figure 7. They discovered that the threshold for each stage can be changed depending upon the level of security required and also that the training model for each stage can be exercised with feature selection. While their work was related to reducing the false positives, the misclassified instances (where the actual class != predicted class, for supervised learning) is not sent to the next stage. Instead the normal records are sent to the next filtering stage. Their main aim was to identify anomalies and not classification of attacks according to type. This proposed method improved the overall detection rate to 96.85%.

In [13], Kok-Chin Kook Et.al used multiple Bayesian based classifiers, namely the Naïve Bayes classifier, expert-elicited Bayesian (EEBN) network and the Bayesian Network (BN) classifier using optimum feature set constructed using Correlation based feature selection (CFS) and the consistency subset evaluator. With respect to a single classifier, NB classifier provided the best results compared to the other two with an overall detection rate of 92.3%.

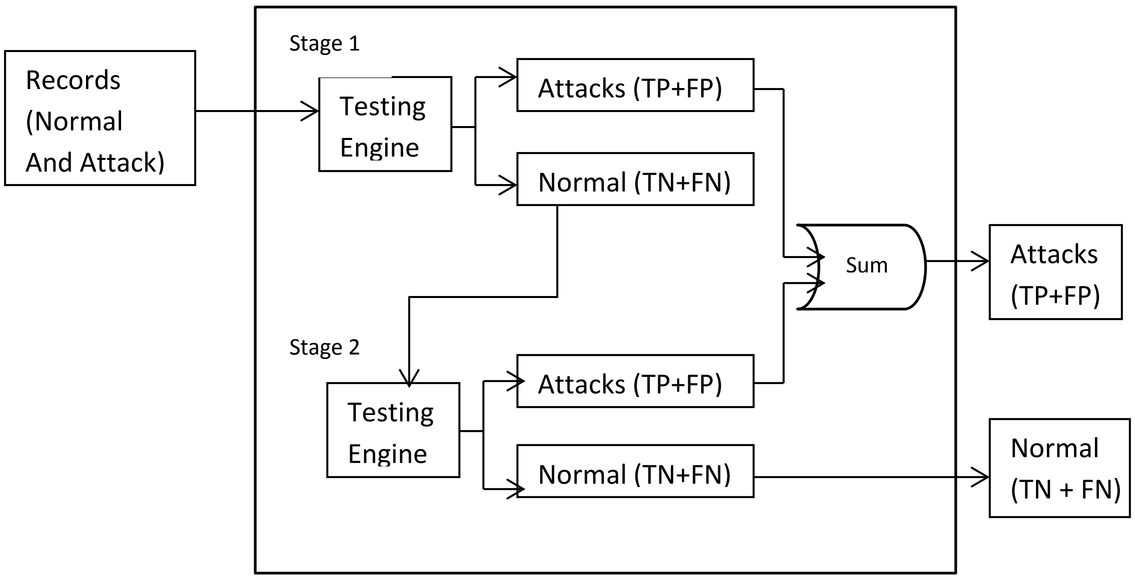


Fig 7. Improved multi-stage Bayesian Filter in [10].

On the other hand, for BN and EEBN, the detection rate was 90.3% and 91.8% respectively. The main problem in their work was that the detection rate of R2L and U2R was found to be very low, with NB classifier providing the best results. A dismal 12% for R2L and 19.7% for U2R. EEBN performed well in U2R category with 22.4% detection rate.

Researchers prefer to work with Bayesian classifiers because of its simplicity in construction and reasonable accuracy in detecting all attack types. In [12], Hui Lu and Jinhua Xu compared an unsupervised clustering method, Bayesian clustering, with the performance of Naïve Bayes and decision trees. They developed a three stage hybrid detection model to segregate attacks shown in Figure 8. In the first stage, the dataset was separated into three parts (DoS, probe and others) using a C4.5 decision tree algorithm. This main rationale behind this was to separate the R2L and U2R and normal records from the rest. In the second stage filtering, Simple NB classifier was used to separate U2R records from the R2L and normal records. It can be seen that with this approach, the accuracy of R2L and U2R was low. As a result, the third stage using Bayesian clustering approach was used to improve detection rate of R2L attack. It was seen that with this three level hybrid classifier, the detection rate of R2L and U2R reaches to 48.41% and 97.14% respectively on the KDD 99' dataset. The training of U2R and R2L records is a problem because of its low number of records in the dataset. It was seen that using multiple classifiers or hybrid classifiers in series or in parallel improved the detection rate significantly [11].

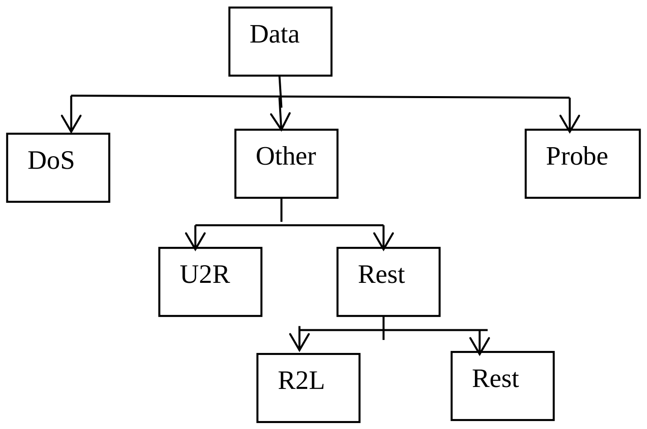


Fig. 8 A three stage hybrid detection model proposed in [12]

Guo Et.Al in [14], investigated the application of Feature Selection in IDS. They employed rough set theory in finding feature subset, and then used an optimization Genetic Algorithm with population clustering approach to find out the optimal subset from the remaining features. SVM was used as a classifier on the KDD99 dataset. Population clustering was used to establish a self-adaptive crossover and mutation rate in the Genetic Algorithm. K-Means algorithm with Euclidean distance was used to achieve this purpose. The feature selection algorithm selects 11 features; feature no. 1,2,4,5,6,11,22,23,31,33 and 35 for all attack types. The results were compared with other algorithm and it was found that the proposed method was more efficient as the detection rate has increased from 95.27% to 98.21% and the false positive rate reduced from 7.63% to 0.91%. They had used an evolutionary technique in computing the optimal feature subset with rigorous mathematical formulas to perform population clustering.

Zhou Et. Al in [15] (2007) used GA to calculate best feature subset and SVM to test the accuracy of the classifier. Some of the features are redundant and their removal in the data-preprocessing step greatly enhances the result. They used weight based parametric fitness function with the weights calculated from the frequency of occurrence in the dataset. Their optimal fitness value varied according to a parameter β. For $\beta = 0.9$, 5 features were selected. Them being 23,27,31,33 and 38. With $\beta = 0.9$, the overall detection rate reached 97.14%. Based on trials, the optimal value of β was found to be 0.5 where 26 features were selected and the detection rate reached 98.46%. It can be seen that though SVM is a good classifier, its detection rate is significantly low for the NSL-KDD dataset (69%) [18]. When the training set is large, it takes a long time to train the features.

The C4.5 decision tree algorithm can detect U2R and R2L attacks better than neural networks [22]. With respect to time, backpropagation based NN gave better results since it undergoes training, validation and testing phases. Researchers have used Linear Discriminate Analysis (LDA) [30] as features selection models, information gain or the Gain ratio to select features. Fuzzy techniques are used along with Hidden Markov Models (HMM) to efficiently classify abnormalities. In [22], GA was used to select features 2,3,4,5,6,27,33,40 and 41. The modified J48 (C4.5 algorithm) uses an intelligent agent to calculate the threshold for constructing the nodes of the tree. It was seen that this modified J48 classifier with optimal GA based feature selection model was better in detecting DoS and probe attacks with shorter training and testing times as well.

Dheeraj Pal and Amrita Parashar [24], used soft computing based approach to create rules to detect attacks. Information gain was used to select 15 most important features from the KDD99 Cup dataset. After normalization and fuzzification of the selected features using the triangular function, initial rules were created and they were stored as a decision tree.
For example,
feature 36 > 0.005 | feature 23 > 0.01 : ipsweep
{normal=0,buffer_overflow=0,loadmodule=0,smurf=0,ipsweep=72,multihop=0,ftp_write=0, nmap=0,Neptune=0,warezclient=0,spy=0}

Within GA, a support-confidence based fitness function was introduced and a rule was selected if its value is greater than the threshold (0.5). Their research was restricted to creating rules to detect anomalies and not attack classification. It was seen that the detection rate of attacks was 91.88% with a high false positive rate of 18.06%. They also compared the trends in feature selection, comparing the number of attributes selected to the accuracy of detection. It could be seen that the accuracy is maximum when the number of features selected is around 20-25. This trend in feature selection can be a topic for further research purposes. They also compared their GA based rule creation technique with ANN approaches. They concluded that GA is a form of unsupervised learning as compared to ANN. With neural networks we need to assign the weights to nodes and provide some learning, checking the output regularly via the backpropagation algorithm. However, with GA, we need not worry about this as it will provide the globally optimal solution. The accuracy of GA and thus the rule creation depends upon the definition of the fitness function. In [31], rules were created for detecting warezclient and the warezmaster attack. Salah Benaicha Et.Al [23] used GA based technique to create rules for attacks.

Ibrahim Et. Al [21], used Self-Organising Maps (SOM) on the KDD99, NSL-KDD(2010) dataset to provide an in-depth comparison of ANN based approaches. With the KDD99 dataset, the proposed SOM based algorithm had an accuracy of 92.37% whereas for the NSL-KDD dataset, it was 75.49%. Panda (2010) had used Multinomial Naïve Bayes + N2B to obtain a success rate of 38.89%. Because of its high speed and faster conversion rate, SOM can be used in multi-stage detection. SOMs are also powerful on static networks than dynamic ones because dynamic networks have memory. They can be trained to learn sequential or time varying patterns.

Deshmukh Et. Al [17] used preprocessing methods such as feature selection and discretization and compared the efficiencies of the NB, HNB and the NB Tree classifiers. A Fast Correlation based Feature selection technique (FCFS) was implemented and equal-width discretization converted the continuous values of a feature to nominal values. Discretization improves the accuracy of NB classifiers. Using CFS, 12 features were selected. The detection rate of NB tree was found to be maximum with 94.6% as compared to 88.2% for NB and 93.4% for HNB; however it takes longer time to construct the model.

Bhupendra Ingre and Anamika Yadav in [27] used ANN to evaluate the NSL-KDD dataset. Their ANN based classifier is shown in Figure 9.

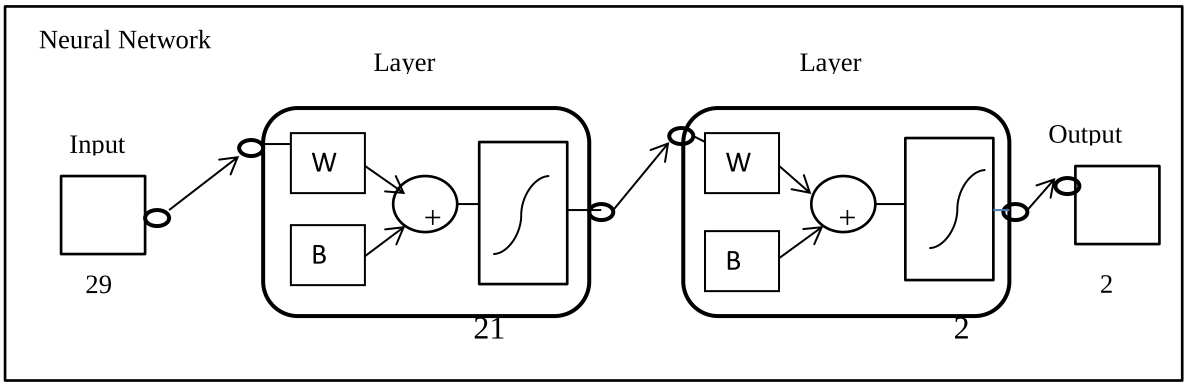


Fig. 9 Architecture of the Neural Network as proposed in [27]

Proposed method uses Levenberg-Marquardt (LM) and BFGS quasi-Newton backpropagation for updating the weight and bias. There are initially 29 neurons corresponding to 29 features in the set with the hidden layer using 21 neurons. For binary classes and 5 classes, output layer's neuron is changed with 2 and 5 respectively. Since there are very low U2R and R2L records in the dataset, it was necessary to maintain equality. For speeding up the training times and to maintain equality, only a set of normal and other class records were selected. With the proposed backpropagation approach, the overall accuracy achieved was 79.2%. The detection rates of DoS, probe, R2L and U2R were obtained to be 77.77%, 76.6%, 34.6% and 10.5% respectively. While DoS and probe attacks are classified reasonably well for NSL-KDD99 dataset, it was the R2L and U2R attack classification which is the problem. For binary classification (normal and attack class), the accuracy was 81.2% of detection with a false positive rate of 3.23%.

It can be seen that based on the vast review of existing techniques, the false positive rate can be reduced and the detection rate can be improved even more using multiple or hybrid classifiers and by using feature selection technique.

Feature Selection is an important domain area that could be used in the field of IDS. However, it was necessary to conduct a preliminary research of different feature selection techniques to understand their performance so as to determine its necessity of usage in my research. Megha Agarwal [16] compared the performance of different selection methods for the four attack classes. Removal of redundant and irrelevant features for a particular class is called feature selection or dimensionality reduction. It is different from feature extraction where new features are uncovered from existing features. Feature selection methods can be classified mainly into three types. Filter methods, Wrapper methods and embedded methods.

Removal of these redundant features will shorten the training times; improve the prediction accuracy rate of the classifier as well as simplify the classifier model construction [14-16]. Each feature selection technique employs a search algorithm for finding the subset and an evaluation function to give a score to that subset. Filter methods find an approximate subset whereas wrapper methods use a classifier to evaluate the subset. Wrapper methods give an optimal solution but take longer time to produce an output than the filter methods.

Correlation based Feature Selection (CFS) was first discussed by Mark Hall in 1999 in his PhD thesis [37]. It is based on the notion that all features that have a high correlation to the class attribute must be in the same subset whereas those features in the same subset must have low correlation amongst themselves. It gives a score to each subset called as the merit of the subset.

Greedy search technique such as the hill climbing, forward selection, backward selection, greedy stepwise search, best first search etc. are used to find feature subsets and CFS is used to find the optimal subset amongst them. Other important selection practices are ChiSquare subset evaluator, Filtered Wrapper technique, Information Gain (IG), Gain Ratio, LDA and Principle Component Analysis (PCA). In [16], the detection rate for GainRatioAttributeEval + Ranker algorithm is the maximum with 99.61% for Naïve Bayes classifier whereas the time for model construction was the least for InfoGainAttributeEval + Ranker algorithm.

Based on the study of existing techniques used by researchers, several shortcomings were identified in the field of IDS discussed in the next chapter.

2.2 Theoretical Foundation

2.2.1 Naïve Bayes Classifier

The Naïve Bayes classifier is a probabilistic estimator based on the Bayes theorem of conditional probability. Based on its usage, it can be Simple Naïve Bayes classifier, Hidden Naïve Bayes, Naïve Bayes updatable or a multi-nominal Naïve Bayes classifier. Henceforth Naïve Bayes classifier will also be abbreviated as NB classifier in future references in this thesis report. A simple NB classifier assumes mutual independence amongst the features. Some features are correlated to each other, after finding the correlation, each of these features has a hidden parent in HNB. Simple NB classifier is one of the simplest, intuitive, easy to construct and gives reasonably good accuracy. Hence it is one of the most used machine learners for IDS.

On the basis of conditional probability, Bayes theorem was formulated. The Bayes theorem states that given the prior probability of the occurrence of any event A when B has occurred, and the likelihood of the occurrence of event B, then it is possible to calculate the probability of the occurrence of event B when A has occurred. This can be given in formula (1).

$$P(B|A) = \frac{P(A|B) \times P(B)}{P(A)} \qquad \dots\dots(1)$$

$$posterior = \frac{prior \times likelihood}{P(A)}$$

For example, if a doctor knows the likelihood of occurrence of cancer P(cancer) based on previous results as well as he knows the symptoms that can cause cancer P(symptoms| cancer) , then he can guess the probability that a disease is cancer when he sees those symptoms i.e P(cancer | symptoms).

$$P(cancer|symptoms) = \frac{P(symptoms|cancer) \times P(cancer)}{P(symptoms)}$$

Consider a set of attributes X1,X2,X3….Xn with each contributing to the attack class C. Considering mutual independence amongst the attributes, calculating the prior probabilities and the likelihood of the occurrence of an attack in the attack class, we can estimate the probability of the occurrence of an anomaly (or attack) when a particular feature is significant. This is shown in formula (2). The denominator can be removed since it is a constant. Hence the problem would be to find the maximum posterior probability to estimate the class of an attack.

$$argmax(P(C|Xi)) = \sum_{i=1}^{N} P(Xi|C) \times P(C) \qquad \ldots.(2)$$

Bayesian based classifier is a supervised learning algorithm and is based on the Bayes' theorem of conditional probability. It calculates the probabilities for attack classes and in normal TCP traffic for different features. The classifier is trained by giving it classified traffic. It then corrects the probabilities for each feature. During the test phase, the classifier estimates the probabilities for each TCP connection and predicts the class of the each instance**.** For a discrete attribute, it calculates the prior probability based on the frequency of occurrence in the training set. For a continuous attribute, it calculates the mean and standard deviation and then uses a uniform Gaussian distribution to find the prior probability. WEKA provides a kernel estimator function, but it was not used in my thesis. Uniform distribution was used to find probabilities. Numeric estimation precision values are calculated using the analysis of training data. Hence, the classifier is not a UpdatableClassifier.

2.2.2 Feature Selection

Removing unwanted features (or attributes) from the feature set is called as feature selection or dimensionality reduction. Based on literature reviewed, it can be seen that certain features contribute to an attack more than the others. For example, inspecting the number of outbound commands in an FTP session can help detect warezclient and warezmaster attacks, the number of packets reaching a destination port from the same source can help in the detection of flooding attacks or even number of files created in the root directory can help in detecting U2R attacks. This does not mean that the others do not contribute to an attack. It just that their contribution is less, compared to the others. On the other hand, including some features in the model construction will fluctuate the bias of the estimator towards normal class and hence these attacks will not be detected at all (or they will get detected but not in the proper class). Feature selection is used by many researchers as discussed Section 2.1.

Removal of these redundant features will shorten the training times; improve the prediction accuracy rate of the classifier as well as simplify the classifier model construction. Each feature selection technique employs a search algorithm for finding the subset and an evaluation function to give a score to that subset. Filter methods find an approximate subset whereas wrapper methods use a classifier to evaluate the subset. Wrapper methods give an optimal solution but take longer time to produce an output than the filter methods. Embedded techniques used hybrid filter and wrapper methods to find the optimal subset. Most commonly feature selection technique is the Correlation based Feature Selection (CFS), Information Gain, Gain Ratio, Consistency Subset evaluator and Chi-squared attribute evaluator.

Genetic Algorithm

Genetic Algorithm is a model of machine learning which derives its behaviour from the process of evolution in nature. A population of candidate solutions to an optimization problem is evolved toward better solutions. Initially a population of randomly generated individuals is selected and they are operated upon by an iterative process to create a new population. Each newly generated population is called as a generation. The populations of individuals are represented by Chromosomes. Each chromosome is made up of genes. In essence, a genetic algorithm characteristically results in an offspring that are genetically identical to a parent. From the population of individuals, fitter parents are selected for crossover to produce fitter individuals. The fitness of an individual is determined by a known fitness function. It retains the fitter solutions in the population whereas removes the less fit solutions from the population.

With successive generations, we can see that the average fitness of the population increases and the algorithm can be terminated when the solution is found. Mutation is another operation performed on individuals and it is used to maintain the genetic diversity from one generation to the other. It involves arbitrarily changing the genes of a chromosome. It is necessary to consider two things in general: *exploration* and *exploitation*. Exploration means searching for the solution throughout the search space and is provided by the crossover operation while the latter involves producing better solutions within the local region and is provided by mutation.

While Genetic Algorithms are time consuming and tedious with regards to the efficient use of their parameters are concerned, this project uses GA in the training stage of classification. i.e. Only to find relevant features in the training phase.

The general functions used in Genetic Algorithm are:-

1. Set initial population
2. Encode population to chromosomes
3. Loop until number of generations is reached
   a. Calculate Fitness of chromosome
   b. Selection
   c. Crossover
   d. Mutation

4. Obtain optimal chromosome

Summary

Conducting preliminary research in this field led to the conclusion that while DoS and probe attacks were easy to detect, efforts were made to improve the detection accuracy rates of R2L and U2R attacks. It is seen that DoS and probe attacks involve many connections from a host or to the destination and hence easy to detect. However, the detection of the other attacks is possible through the analysis of the payload "content" of records. Nearly fifteen papers were discussed in this area to identify the current trends and advances in Intrusion Detection. Each technique is designed to solve a particular detection problem; some of them even combine hybrid approaches to find the solution. The potential research gaps were identified and are discussed in the next chapter.

After thoroughly studying the relevant papers related to this topic, it was found that a Naïve Bayes classifier produces reasonable accuracy and faster speed of execution makes it an ideal classifier to be used in intrusion detection. However, it still needs improvement which can be provided using feature selection. Naïve Bayes and feature selection were discussed in section 2.2. It gave a brief description of the classifier working as well as the theoretical foundation behind evolutionary computing (Genetic Algorithm).

CHAPTER 3

PROBLEM FORMULATION

3.1 Potential Research Gaps Identified

With regards to the techniques discussed in Chapter 2, several shortcomings were identified discussed below.

1. Most researchers use the outdated KDD99 dataset to test their detection algorithm. The KDD99 dataset is derived from DARPA dataset developed by MIT Lincoln Laboratories. DARPA lists a set of outdated attacks. New procedures, data availability mechanisms and applications are put to use with each passing day leading to newer vulnerabilities. These can be exploited by the attackers before the security analysts even realize the system shortcomings resulting in novel attacks on the systems. Even though KDD99 is a benchmark for testing anomaly based detectors, they do not provide a list of real novel attacks, as it happens in the real world.
2. In many datasets, the number of normal instances is large and that for attack instances is low. As a result, enough training is not provided for non-biased classifier results.
3. R2L and U2R have reasonable low detection rates and high false alarm rates because their detection mechanisms are embedded in the data portion of the packets. DoS and probe have reasonably good detection accuracies.
4. In anomaly detection, the activities are compared against the normal behaviour of the system; however, it is hard to determine the normal behaviour.
5. Determining a suitable and a fast feature selection method for each attack class is still an issue.
6. Improvement in the pre-processing methods, dimensionality reduction and extraction in the training set and their real-life deployment is still a major concern for researchers.
7. Reduction of false alarm rates and dynamic updation of user profiles is still a cause for concern for researchers.
8. An NIDS should inspect each packet at runtime without losing important data and its effectiveness depends upon the environmental setup and network topology.

3.2 Motivation behind the research

Providing data security is the prime concern of any organization. Be it service providers, banks, social networking sites or just a product developers, organizations need to protect customer data and applications. Consumers entrust organizations to behold their privacy while hackers or data attackers try to steal vital information such as bank account numbers, pin details or even render a service useless. Compromising such information could have disastrous consequences for the individual who will eventually file a lawsuit against the organization.

My job as a security expert is to prevent the occurrence of such data attacks. With due regards to the latest techniques developed to provide excellent service to the customer, it is said that hackers are a step ahead of the defence mechanisms. Hackers try to find loopholes in existing network protocols and mechanisms with the intention of spreading malice. Firewalls, antivirus software detect only known attacks. Improvements in IDS and other security mechanisms is an ongoing research field in computer science. Hackers trying to hack into a system (Remote-to-local, R2L) attacks and legitimate users trying to gain administrator privileges for malicious purposes (User-to-Root, U2R) are two types of attacks whose detection is difficult. The main motivation behind this research was to improve this detection rate of these so called difficult to detect novel attacks.

Improvement in the detection rate of these attacks will help an organization in finding and fixing existing loopholes as well as implementing preventive schemes to counter these attacks. Reduction of the false positive rate will help in better alert classification and generating better counter-responses for an Intrusion Response System (IRS).

3.3 Objectives of the project

1. To improve the detection accuracy rates of R2L and U2R attacks by using a two-stage Naïve Bayes Classifier with feature selection in the second stage.
2. To compare the detection rates for different machine learners in classifying R2L and U2R attacks.
3. To compare the detection rate of the proposed classification algorithm with the latest techniques.
4. To assess the competence and the proficiency of the proposed feature selection technique in calculating the optimal feature subset for R2L and U2R attacks.
5. Analysing the results given by CFS, Information Gain, Gain Ratio, Chi-Squared and Consistency Subset feature selection evaluators.

6. To examine the trends in the number of features selected to the performance of the detection algorithm.

3.4 Intended Outcomes

1. Naïve Bayes is the simplest and the quickest classifier out of the rest. Using it in a two-stage or even a multi-stage classification model should reduce the time complexity of detection.
2. With respect to the enhancement in the detection rate, a well-known proven feature selection model is used in the second stage of classification.
3. Using a soft-computing feature selection technique will not require any complicated hardware and reduce the complexity of model construction.
4. Entropy based weight calculation for each feature will give an estimate of the importance of each feature and the fitness function for Genetic Algorithm acts as the subset evaluator.
5. Exploration and exploitation of feature subset searching is done by crossover and mutation operators, thereby requiring no additional search technique.
6. Comparison of different dimensionality reduction mechanisms will give an approximate idea as to the best feature selection technique for R2L and U2R attacks.
7. Variation in the number of features selected to the performance of detection will help in selecting the optimal feature subset.
8. My dissertation will serve as a reference model for future multi-stage filters with the latter stages running in parallel.
9. Using the latest NSL-KDD dataset will help point out the shortcomings in algorithms which were tested on the outdated DARPA dataset or the KDD99 dataset.

CHAPTER 4

TWO-STAGE R2L AND U2R CLASSIFIER

4.1 Workflow

The proposed system to improve the detection rate of attacks is a two-stage classifier with feature selection used in the second stage. Simple NB classifier is used in both the stages of classification because of its speed of execution, ease of construction and reasonable accurate result rate. The second stage classifier is run in correspondence for each attack class to achieve parallelism along with the improvement in the true positive rate. Reduction of the false positive rate is an issue and to address this, the misclassified instances from the first stage are given as the input to the second stage. Figure 10 shows the workflow of the proposed two-stage classifier.

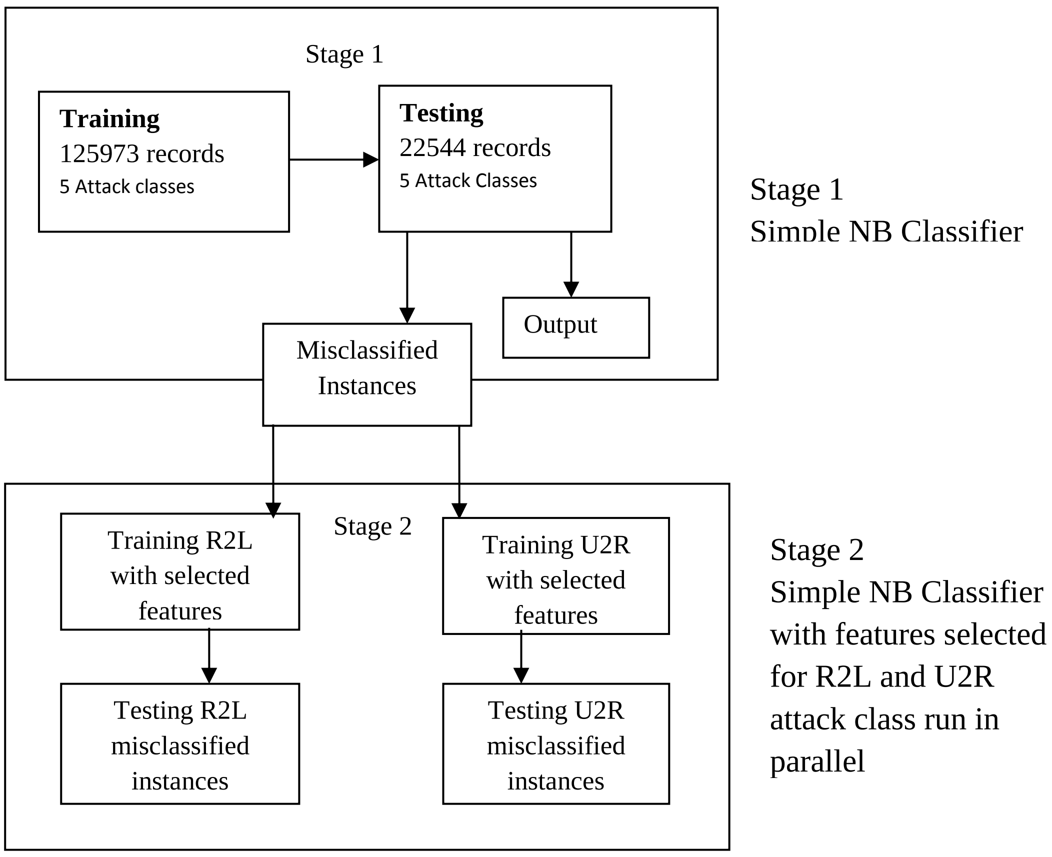

Stage 1
Training
125973 records
5 Attack classes
Testing
22544 records
5 Attack Classes
Stage 1
Simple NB Classifier
Output
Misclassified Instances
Stage 2
Training R2L with selected features
Training U2R with selected features
Testing R2L misclassified instances
Testing U2R misclassified instances
Stage 2
Simple NB Classifier with features selected for R2L and U2R attack class run in parallel

Fig. 10 Workflow of the proposed two-stage classifier

In the first stage, the classifier is trained with all features selected and then the trained filter is evaluated against the test set. The output obtained from the first stage consists of correctly identified instances and misclassified instances i.e. only those records whose predicted class value is not the same as the actual class value. The classifier accuracy is calculated. In the second stage, the training set is trained only for the R2L attacks and U2R attacks separately. The Naïve Bayes model filters the records into either two classes (R2L and others or U2R and others). In the second stage, the classifier is trained with the features selected by the Genetic Algorithm. The filter is evaluated against the misclassified records obtained from stage 1.

4.2 Analysis of the NSL-KDD dataset

The KDD99 cup dataset is a benchmark for testing many anomaly detection algorithms. It was prepared by the MIT Lincoln Labs that acquired nine weeks of raw TCP dump data from the environment they set up. They bombarded Windows NT, Solaris, Sun OS and Linux systems with multiple attacks over a nine week period. However, it is not a complete represented of real-life attacks since most of the background traffic is synthetically generated and not real burst traffic that is available nowadays.

Four Gigabytes of raw TCP data was processed into five million connection records. A connection represents a sequence of TCP packets starting and ending at well-defined times, during which data flows from the source to destination under some well-defined protocol [36]. Each connection is labelled as a 'normal' or an 'attack' record. Though it is mostly used by researchers, it suffers from several problems [19] given below:

1. The number of DoS and probe records are large and R2l and U2R attacks are low so that the classifiers are biased towards the more frequent records.
2. The existence of redundant and repeated records will tend to bias the result towards the more frequent records favouring the DoS and probe attacks.
3. It is outdated with newer and newer attacks discovered since 1999.

On account of these problems, KDD99 is not a suitable dataset for research purposes.
The NSL-KDD (2010) is a dataset which solved the shortcomings of its predecessor.

1. It does not include the redundant records so that the classifiers are not biased towards the more frequent records.
2. The detection of attacks in the test set is not biased towards certain types of records since duplicate records are removed from the test set.
3. The number of records of train and test set are reasonable and hence tests not need be conducted on small subset of the dataset.

The NSL-KDD is available online at [25]. My research work included attack classification and hence 5 attack classes are used.

The NSL-KDD dataset consists of 125973 records in the training set and 22544 records in the test set, with each record connection having 41 features (or attributes) divided into four categories: Basic features (Features 1-9), Content Based features (Features 10-22), Time-based Traffic features (Features 23-31) and Host-based Traffic features (Features 32-41). The detailed names of the features are given in Table 1.

ID	Name	ID	Name
1	Duration	22	is_guest_login
2	protocol_type	23	Count
3	Service	24	srv_count
4	Flag	25	serror_rate
5	src_bytes	26	srv_serror_rate
6	dst_bytes	27	rerror_rate
7	Dos	28	srv_rerror_rate
8	wrong_fragment	29	same_srv_rate
9	Urgent	30	diff_srv_rate
10	Hot	31	srv_diff_host_rate
11	num_failed_logins	32	dst_host_count
12	logged_in	33	dst_host_srv_count
13	num_compromised	34	dst_host_same_srv_rate
14	root_shell	35	dst_host_diff_srv_rate
15	su_attempted	36	dst_host_same_src_port_rate
16	num_root	37	dst_host_srv_diff_host_rate
17	num_file_creations	38	dst_host_serror_rate
18	num_shells	39	dst_host_srv_serror_rate
19	num_access_files	40	dst_host_rerror_rate
20	num_outbound_cmds	41	dst_host_srv_rerror_rate
21	is_host_login		

Table 1. List of features in NSL-KDD dataset

Each connection has a class label representing the attack type of the class and there are five attack types: Normal, DoS, U2R, R2L and Probe.

Denial of Service (DoS): In this type of attacks, legitimate users are denied access to a particular resource or service. For example flooding the bandwidth with unnecessary packets, flooding the server with spoofed SYN packets, Neptune, Ping of Death (PoD), Smurf attack, Arppoison etc. However, anomaly detectors can easily keep a track on these attacks since most of these attacks require analysing the temporal based features. Many packets will occur with a short period of time. It is the Distributed Denial of Service (DDoS) that is a cause for concern for security experts.

User-to-Root (U2R): A normal user trying to gain super-user privileges for malicious purposes. For example buffer overflow, loadmodule or by guessing the administrator password.

Remote-to-local (R2L): A remote attacker who does not have access to the local machine tries to gain local access. For example, anomalies can be in the form of incorrect password guessing for more than 5 times, gaining FTP root directory access through FTP commands and writing malicious scripts so that users download them etc. These attacks are the most difficult to detect since they require a thorough inspection of the payload.

Probe attacks: Stealth scanning and surveillance of the hosts on the network for open IP addresses and ports. They are not malicious in nature but network reconnaissance will help an attacker to gain insider information of the network.

In the NSL-KDD dataset there are 23 attack types belonging to these 4 classes with an additional 17 attack types in the test set. Table 2 all the different R2L and U2R attack types in the train and test set. Table 3 lists the number of records for each class.

Attack Type	**List of Attacks in Training Set**	**List of new Attacks in Test Set**
R2L	FTP Write, Guess Password, Imap, phf, spy, multihop, warezclient, warezmaster	Sendmail, named, snmpgetattack, snmpguess, xlock, xsnoop, worm
U2R	Buffer Overflow, Loadmodule, perl, Rootkit	Httptunnel, ps, xterm, sqlattack

Table 2. R2L and U2R attack types in the dataset

Class Type	**Number of records in Training Set**	**Number of records in Test Set**
Normal	67343	9711
DoS	45927	7458
U2R	52	200
R2L	995	2754
Probe	11656	2421
Total	125973	22544

Table 3. Number of records for each attack type

Analysis of R2L attacks

There are 8 R2L attacks in the training set given below:

Ftp-write: Remote FTP user creates .rhosts file in world writable anonymous FTP directory and obtains local login.

Guess password: An attacker tries to guess the password of a valid user using simple variants of account name over a telnet connection.
Imap: Remote buffer overflow using IMAP port leads to root shell.
Multihop: Multi-day scenario in which a user first breaks into one machine.
Phf: Exploitable CGI script which allows a client to execute arbitrary commands on a machine with a misconfigured web server.
Spy: Multi-day scenario in which a user breaks into a machine with the purpose of finding important information where the user tries to avoid detection. Uses several different exploit methods to gain access.
Warezclient: Users downloading illegal software which was previously posted via anonymous FTP by the warezmaster.
Warezmaster: Anonymoust FTP upload of Warez (usually illegal copies of copywrited software) onto FTP server.

There are three attacks based on misconfigured FTP servers. If the FTP Server is not write protected and the root directory is owned by the FTP account, then the attacker can write malicious scripts from the remote machine to gain local access to the system. To detect this attack, it is necessary to check for anonymous FTP sessions and whether any files are created in the root directory. Some FTP Servers allow a guest to login and download the files from the server. However, if the guest account is allowed to write to the FTP server, then the attacker can upload *"warez"* copies of the illegal software to the FTP server and gain local access. This compromises the server and the attacker gains local access. The legitimate clients can download the illegal *"warez"* software on their machines to perform the *warezclient* attack. The previous attack is called as *warezmaster*. Important features for detecting these attacks are hot, num_access_files, num_outbound_cmds, is_guest_login, same_srv_rate, dst_host_srv_count. In KDD feature complaint heuristic rules for R2L attack detection [31], Sabhani and Serpen have used the number of hot indicators along with the amount of data uploaded (dst_bytes) and downloaded (src_bytes) to create rules for warezmaster and warezclient attack.

The *guess password* attack, as the name suggests makes repeated guesses of usernames and passwords (num_failed_logins) making use of services like telnet, ftp, pop, rlogin and imap (service). The *multihop* attack is a multi-day attack in which the user tries to break into one machine using various techniques. Since there are just 7 instances of the multihop attack in the dataset, a statistical observation of the record connection implied that num of hot indicators, is_guest_login, count, srv_count, same_srv_rate, dst_host_count are important for detecting multihop attacks. The *spy* attack also tries to enter into a local machine (is_guest_login, logged_in) using several services (service). The *phf* attack exploits a badly written CGI script to execute commands with the privilege level of the http server. It can be identified by scrutinizing the http sessions.

The IMAP protocol allows legitimate users to access their mails from a mail server. If the IMAP server is allowed to run with root privileges, then it can perform some file manipulation on behalf of the user login. The attacker performs the buffer overflow attack to remotely execute malicious instructions on the IMAP server.

Attack Type	**Number of instances**
Training Set	
FTP Write	8
Guess Password	53
Imap	11
Multihop	7
phf	4
spy	2
warezclient	890
warezmaster	20

Table 4. Record distribution of R2L in train set

Attack Type	**Number of instances**
Testing Set	
FTP Write	3
Guess Password	1231
Imap	1
Multihop	18
Phf	2
Spy	0
Warezclinet	0
Warezmaster	944
Sendmail	14
Named	17
Snmpgetattack	178
Snmpguess	331
Xlock	9
Xsnoop	4
Worm	2

Table 5. Record distribution of R2L in test set

Analysis of U2R attacks

There are four U2R attacks in the training set:

Buffer Overflow: Vulnerability exploited is that of the boundary check of users' input data before copying it to a fixed length memory buffer. Attacker can execute data into the stream and remotely activate it to gain unauthorized access when buffer overflows.

Loadmodule: Loads 2 dynamically loadable kernel drivers into the currently running system to gain super-user access.

Perl: Exploits vulnerability in the perl script in which the interpreter does not relinquish super user privileges when changing user and group ID's.

Rootkit: Rootkit is a software that is installed in the system to gain administrator privileges once the machine has been compromised by some other method.

Attack Type	Number of instances
Training Set	
Buffer Overflow	30
Loadmodule	9
Perl	3
Rootkit	10

Table 6. Record distribution of U2R attacks in train set

Attack Type	No. of instances
Testing Set	
Buffer Overflow	19
Loadmodule	2
Perl	2
Rootkit	13
HttpTunnel	133
Ps	15
Sqlattack	2
Xterm	13

Table 7. Record distribution of U2R in test set

With the buffer overflow attack, the attacker can send insufficiently sized memory buffers and therefore possibly corrupt data and thus make a server crash. Analysing num_access_files, root shell or num_of_compromised conditions help in identifying buffer overflows [38]. Detection of loadmodules is possible through bottleneck verification with host based IDS or by using keyword spotting with IDS. Features such as number of hot indicators, is host login, number of root shells, number of access files, root shell compromised etc. play a vital role in identifying U2R attacks.

4.3 Feature Selection using Genetic Algorithm

The proposed feature selection technique is based on evolutionary computing. Genetic Algorithm combines the searching and evaluation of a feature subset thereby requiring no additional effort in finding the optimal subset for R2L and U2R attack class.

Genetic Algorithm requires a set of initial solutions through which it can work with. These solutions are arbitrary and provided through the training dataset. Since, feature must be selected for each class; the initial solutions for GA are the set of R2L and U2R attack instances. For calculation of feature subset for R2L attacks, the initial solutions are the 995 record connections whereas for U2R attacks, the initial solutions are the 52 records. Since feature number 2, 3 and 4 (protocol, service and flag respectively) are nominal values; they are removed from the algorithmic process and considered later on in the feature subset.

The first step in a Genetic Algorithm is the encoding. In the proposed approach, for each record connection, the value of a feature was converted to a bit binary gene value 1 or 0 indicating that a feature was selected or rejected respectively [15]. If the value of the feature is non-zero, then the gene value is 1 otherwise it is 0. For e.g. A record with values (0,tcp,ftp_data,SF,491,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,150,25,0.17,0.03,0.17 ,0,0,0,0.05,0) is encoded as 01000000000000000011000010011111000010. This encoding scheme is trivial, simple and one can easily identify as to the selected genes in a chromosomes. Hence the feature vector is a 38 bit vector with binary values.

According to [32], an ideal initial population size of about a 100 chromosomes gives a good solution to any optimization problem. Increasing the population size will only increase the computational overhead. However, since there are 8 attacks in R2L contributing to 995 record connections, it was necessary to consider each attack separately and find an approximate solution representing the aggregated feature subset for R2L attack class. Hence using each record connection as an initial solution is important.

Population size for R2L attack = 995

Population size for U2R attack = 52

4.3.1 Weight Calculation for fitness function

The next step is to calculate the fitness function to determine which chromosomes can be used for reproduction. The proposed feature selection technique based on Genetic Algorithm assigns weights to each feature to calculate the fitness of the chromosome. Weights are assigned using the concept of Information Theory.

Entropy

“Random sources contain more information”. An event which occurs frequently is easily predictable and hence will provide less information when it occurs. Conversely, an event which is random and less likely to occur will carry more information. This is the concept of information theory in a gist. The information that the source event provides is not the actual data contained in that event but is the frequency of occurrence when some dependent event occurs. The event is characterised by the probability distribution of the samples drawn from it. According to Shannon’s information theory, the probability of occurrence of an event along with its information content gives the average amount of information contained in that event.

Entropy is the average amount of information contained in each event. It gives the uncertainty of the event and is higher for more random sources [39]. The entropy for a random event X is calculated in Formula (3).

$$H(X) = -\sum_{i} P(x_{\mathrm{i}}) \log_2 P(x_{\mathrm{i}}) \qquad \ldots..(3)$$

Where, x_{i} is the probability that X takes a value i. P(X = i).

Entropy is commonly used in information gain to calculate the feature relevance with respect to the class attribute. Each feature carries a certain amount of information and provides a contribution to the class attribute. This information is not the actual value for that particular feature, but depends on the frequency with which a particular attribute is found to be contributing to a particular attack. The entropy of a feature has a minimum value of 0 and a maximum value of 1. When its value is 0, it means that the feature doesn’t occur in all record connection or that it occurs most frequently in all record connection. When the entropy has a value 1, it means the feature has a 0.5 probability of occurrence within all record connections.

The Entropy of a feature X can be derived from Formula (3) to form Formula (4)

$$H(X) = -1 \times \left[\left(\frac{n1}{n0+n1} \cdot \log_2 \frac{n1}{n0+n1}\right) + \left(\frac{n0}{n0+n1} \cdot \log_2 \frac{n0}{n0+n1}\right)\right] \qquad \ldots\ldots(4)$$

Where $n1$ represents the number of 1’s in a chromosome and $n0$ represents the number of 0’s in a chromosome. More the number of 1’s for a particular feature for all chromosomes, higher is the weight. It can be seen the curve for Entropy of a feature versus the probability distribution is strictly increasing till the probability reaches 0.5, after which it becomes a strictly decreasing function.

The curve for the entropy vs the probability distribution is shown in the Figure 11.

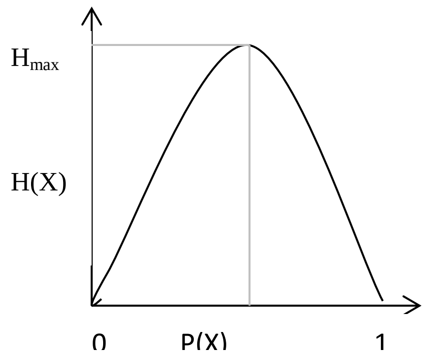


Fig. 11 Entropy of a feature vs the probability distribution

Calculating the entropies of all features for R2L and U2R attacks, the entropy vector is defined as:

For R2L attacks, entropy:

[0.9641 0.1951 0.9721 0 0 0.0209 0.9837 0.2959 0.4244 0.0377 0.0531 0.0114 0.0456 0.0743 0.0295 0.0743 0 0 0.8983 0 0 0.1128 0.1188 0.3001 0.3326 0 0.0604 0.1693 0 0 0.0531 0.9703 0.8256 0.9976 0.7473 0.6433 0.7473 0.3326]

For U2R attack, entropy:

[0.8667 0.7793 0.2351 0 0 0.1370 0.9957 0.1370 0.5159 0.9903 1 0 0.7062 0.9903 0.4566 0.1370 0 0 0 0 0 0.2351 0 0.2351 0.1370 0 0.5159 0 0 0 0.3912 0.8112 0.6646 0.7793 0 0.1370 0.5699 0.5699]

H_{max} represents the maximum entropy of a feature amongst others.

$H_{max}(R2L) = 0.9976$

$H_{max}(U2R) = 1$

The weights indicate the importance of a feature to an attack class. The feature selection algorithm will improve upon the weights to find the proper solution.

Formula (5) gives the weight of x^th feature.

$$w_x = \frac{H(X)}{2} \qquad if\ 0 < P(X=1) \leq 0.5$$

$$= \frac{2 \times H_{max} - H(X)}{2} \qquad if\ 0.5 < P(X=1) \leq 1$$

…… (5)

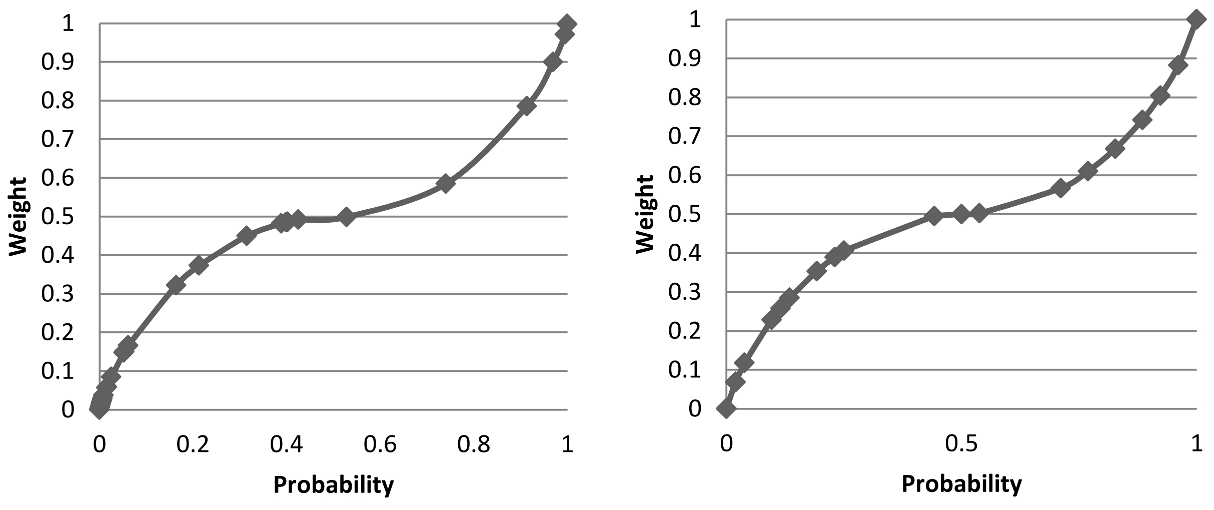
The entropy based weight vector for R2L and U2R attacks is given below:

For R2L attacks,

[0.4820 0.9000 0.4860 0 0 0.0104 0.4918 0.1479 0.7854 0.01889 0.02656 0.00572 0.02280 0.03719 0.0479 0.03719 0 0 0.44919 0.9976 0.9976 0.0564 0.0594 0.1500 0.1663 0.9976 0.0302 0.0846 0.9976 0.9976 0.9710 0.4851 0.5847 0.4988 0.3736 0.3216 0.3736 0.1663]

For U2R attacks,

[0.5666 0.6103 0.8824 0 0 0.0685 0.5021 0.0685 0.7420 0.4951 0.5 0 0.3531 0.4951 0.2283 0.0685 0 0 0 1 1 0.1175 0 0.1175 0.0685 1 0.2579 0 1 1 0.8043 0.4056 0.6676 0.3896 0 0.0685 0.2849 0.2849]



(a) For R2L attacks (b) For U2R attacks

Fig. 12 Weight distribution of features for (a) R2L attacks (b) U2R attacks

The curve for the weight of a feature versus the probability distribution for that feature P(X = 1) is given in Figure 12. This curve indicates a strictly increasing function with the weight of a feature assigned 0 when its probability of occurrence is 0 and weight increases as the probability increases. *For R2L attacks, the weight has a maximum value of 0.9976 whereas for U2R attacks, weight has a maximum value of 1.*

Based on weight, the fitness of a chromosome is calculated using Formula (6).

$$fitness = \frac{Lx}{N} + XW^{\mathrm{T}}$$

$$W = \frac{w_i}{\sum_{i=1}^{N} w_i} \qquad \ldots\ldots\ (6)$$

Lx is the number of 1's in the chromosome, X is the gene vector and W is the Weight vector. W_i is the weight of the i^{th} gene in the chromosome and N is the number of genes (features).

Theoretical Rationale behind the fitness function

Any chromosome is encoded in such a manner that when a feature is relevant to the attack, it is selected and has a value 1. When it is redundant and does not contribute any information to the attack class, it is removed and has a value 0. A statistical analysis of the dataset shows that most of the features that have a non-zero value contribute some information to the attack. No matter how trivial the information might be. However, since the dataset is in human readable format (For example. Feature no. 1, duration, units will be in seconds. 0-∞), when a feature has a value 0, then its contribution to an attack is redundant.

The proposed fitness function will work on sparse data i.e data which has many redundant values.

It is obvious that greater the number of 1s in a chromosome, higher is the fitness of a particular chromosome. The proposed feature selection algorithm is defined in such a way that after a certain number of generations, the fittest chromosome will act as a feature vector indicating which features are selected and which are not.

Lx counts the number of 1s in the chromosome.

In the second part of fitness function, the weights of each feature previously calculated are multiplied with individual genes in a chromosome.

The genes of the chromosome are multiplied with the *normalized weight vector* having values between 0 and 1 i.e individual weight values divided by the sum of all weights.

$Maximum\ fitness\ of\ a\ chromosome = 2$

$Minimum\ fitness\ of\ a\ chromosome = 0$

The subset converges to a solution when the average fitness of a generation improves. After certain number of generation, the fittest chromosome is rewarded and selected as the feature subset.

Selection, Crossover and Mutation

A fitter chromosome has a better chance of survival in the next generation. Chromosomes thrive to survive and reach the top. If better chromosomes are selected as parents, then their offsprings will also be fitter. This is the theory of biological evolution as proposed by Darwin. Several selection techniques such as tournament selection, fitness-proportionate selection and rank selection are employed.

In the proposed algorithm, *rank selection* is used. The chromosomes are sorted in descending order of fitness and ranks are given to each one of them. The top chromosome is given rank 1, the second best is given rank 2, and so on. Since, the lower ranked chromosomes do not contribute anything to the overall population; they can be replaced with the higher ranked chromosomes. This is called elitism strategy (Survival of the fittest). Top 10% of the chromosomes replace the lower ones. After this is done, parents are selected as adjacent chromosomes.

Crossover is the mating operation of chromosomes to create offsprings. *Single point crossover* is chosen with the crossover probability 1. The crossover probability simply indicates the ratio with which the parents are chosen for crossover. With a crossover probability of 1, all chromosomes are chosen as parents. Retaining some top chromosomes of the population is done by elitism (as explained in the selection process). Based on experimental evaluation, it was found that best results are obtained with the pivot at the 25th position. In essence, crossover operation creates new feature subsets to be evaluated by the fitness function. It is responsible for exploratory part of a search algorithm.

Mutation is responsible for exploitation i.e genetic diversity in the search space. Mutation probability is chosen to be 0.05 i.e Probability with which a particular gene is mutated is 0.05. For mutation, a random number generator is used for each bit. If the value of the random number generated is less than 0.05, then the gene is mutated; otherwise it is retained as it is. The parameters for GA are summarized in Table 8

.

Crossover Operation	**Mutation Operation**	Initial population (R2L)=995 Initial population (U2R)=52 No. of generations=300 Selection strategy used=rank selection Elitism and survival of fittest
Single Point crossover (Pivot= 25th position) Crossover probability = 1	Random number generator Mutation probability = 0.05	

Table 8. Description of parameters for GA

4.3.2 Pseudo Code for feature selection

The pseudo-code for feature selection is given below. encode[ ] stores the encoded initial solutions obtained from the dataset. chromosome[ ] is a vector which stores the 38 bit chromosomes used by the GA. children[ ] is a vector which stores the offspring resulting from the crossover operation.

Algorithm 1 Feature Selection

```
1: Separate R2L and U2R attack instances
2: Set population_size_r2l=995, population_size_u2r=52
3: pop[i][j] ← $i^{th}$ record and $j^{th}$ feature
4: Remove features 2, 3, 4 from Table 1
5: for i=0 to pop.length-1 do
6:      if value of $j^{th}$ feature > 0 then
7:          encode[i][j] ← 1
8:      else
9:          encode[i][j] ← 0
10:     end if
11: end for
12: for specific no. of generations do
13:     chromosome[i] ← encode[i][j]
14:     Calculate fitness[i] using Formula (6)
15:     Sort chromosome[i] based on fitness[i]
16:     Assign ranks to chromosome[i] in descending order
17:     Remove 10% of less fit chromosome[i]
18:     Copy 10% of best chromosome[i] to next generation
19:     for all chromosome[i] as parent do
20:         //Single Point Crossover on Parents
21          children[i] ← chromosome[i]
22:         children[i+1] ← chromosome[i+1]
23:         //Mutation
24:         for each bit do
25:             Generate random variable, $0 \le r \le 1$
26:             if $r \le 0.05$ then
27:                 flip bit
28:             else
29:                 retain bit
30:             end if
31:         end for
32:     end for
33: end for
```

The chromosome with highest fitness is the feature subset vector. In this vector, bit 1 indicates that a feature is selected, whereas bit 0 indicates that a feature is rejected. Include features 2,3,4 in the feature subset.

Working of the algorithm with example:

Consider four R2L records given as initial solutions to the genetic algorithm.

1. 0,tcp,ftp_data,SF,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,
   0,0,0,0,1,0,0,2,20,1,0,1,0.2,0,0,0,0,*r2l*
2. 60,tcp,telnet,S3,126,179,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,2,2,
   0.5,0.5,0.5,0.5,1,0,0,23,23,1,0,0.04,0,0.09,0.09,0.91,0.91,*r2l*
3. 15159,tcp,ftp,SF,350,1185,0,0,0,6,0,1,0,0,0,0,0,0,0,0,0,1,1,2,
   0,0,0,0,1,0,1,255,142,0.56,0.02,0,0,0,0,0,0,*r2l*
4. 1,tcp,ftp,SF,1238,2451,0,0,0,28,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0,
   0,0,0,1,0,0,255,123,0.48,0.02,0,0,0,0,0.01,0,*r2l*

---

1. *Remove features 2,3,4 (Protocol, service, flag)*

0,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,2,20,1,0,1,0.2,0,0,0,0,*r2l*
60,126,179,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,2,2,0.5,0.5,0.5,0.5,1,0,0,23,23,1,0,0.04,0,0.09,0.09,
0.91,0.91,*r2l*
15159,350,1185,0,0,0,6,0,1,0,0,0,0,0,0,0,0,0,1,1,2,0,0,0,0,1,0,1,255,142,0.56,0.02,0,0,0,0,0,0,
r2l
1,1238,2451,0,0,0,28,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0,0,0,0,1,0,0,255,123,0.48,0.02,0,0,0,0,0.01,0
,*r2l*

---

2. *Encoding (non-zero values as 1, zero values as 0 giving a 38 bit feature vector)*

01000000100000000000110000100111011 0000
11100011000000000000111111100111010 1111
11100010100000000000111000010111110 00000
11100010100000000000111000010011110 00010

---

3. Genetic Algorithm iterations start here…

Calculate fitness of each chromosome using Formula (6)

01000000100000000000110000100111011 0000	fitness= 0.6978
11100011000000000000111111100111010 1111	fitness=1.1784
11100010100000000000111000010111110 00000	fitness=0.9227
11100010100000000000111000010011110 00010	fitness=0.9988

Average fitness of the population= (0.6978+1.1784+0.9227+0.9988)/4 = 0.9494

Rank Selection (Sort the chromosomes according to descending order of fitness, give rank)

11100011000000000000111111100111010 1111	fitness=1.1784	Rank 1
11100010100000000000111000010011110 00010	fitness=0.9988	Rank 2
11100010100000000000111000010111110 00000	fitness=0.9227	Rank 3
01000000100000000000110000100111011 0000	fitness=0.6978	Rank 4

Remove chromosome with lowest fitness, retain top chromosome (Called elitism) Used as parents

11100011000000000001111111001110101111	Parent 1
11100010100000000011100001001111000010	Parent 2
11100010100000000011100001011111000000	Parent 3
11100011000000000001111111001110101111	Parent 4

Crossover operation (Parents mate with each other to create offspring) (single point crossover)
Pivot=25

```
111000110000000000011111 |11001110101111
111000101000000000111000 |01001111000010
111000101000000000111000 |01011111000000
111000110000000000011111 |11001110101111
```

Output:

```
111000110000000000011111 |11001110101111
111000101000000000111000 |01001111000010
111000101000000000111000 |01011111000000
111000110000000000011111 |11001110101111
```

Mutation operation (Randomly flip bits with certain probability)

```
11100011000000001001111111001010101111
11100010100100000011100001001101000010
11100010101000000011100001011111000000
11100011001001000001111111001010101111
```

This completes the first iteration of genetic algorithm. The output obtained will consist of mutated chromosomes in a new generation.

Calculate fitness of each chromosome and find the fittest chromosome.

11100011000000001001111111001010101111	fitness=1.1676
11100010100100000011100001001101000010	fitness=0.977
11100010101000000011100001011111000000	fitness=1.0831
11100011001001000001111111001010101111	fitness=1.2047

Average fitness of the population = (1.2047+1.0831+1.1676+0.977) =*1.1081*

Chromosome with highest fitness

11100011001001000001111111001010101111

Adding the three features 2,3,4

11111100011001001000001111111001010101111

This is the *optimal feature subset vector* as calculated by the proposed algorithm.
Features selected are
1,2,3,4,5,6,10,11,14,17,23,24,25,26,27,28,29,32,34,36,38,39,40,41
No. of features selected = 23

Summary

A two- stage classifier is constructed with feature selection used in the second stage of classification. The first stage uses a simple NB classifier trained with all the features given in Table 1. The second stage uses entropy based fitness function for giving weights (importance) in GA. The fitness function is calculated using Formula (6). The entire algorithm is explained with the help of an example taking 4 initial solutions into consideration. The misclassified instances are obtained by comparing the actual class (specified in the NSL-KDD dataset) and the predicted class (as estimated by the classifier) and then segregating only R2L and U2R misclassified records. The experimental setup and result analysis is done in the next section.

CHAPTER 5

EXPERIMENTAL SETUP AND IMPLEMENTATION

The proposed algorithm is built on two stages with the Naïve Bayes classifier:

1. Running all 41 features
2. Selected features

In the first stage, the filter is trained with all features present in the NSL-KDD dataset. The trained filter is tested against the test dataset consisting of 22544 records. The output of stage 1 classifier will be:

1. Records which are correctly classified. (actual class == predicted class).
2. Records which are misclassified. (actual class != predicted class)

In order to improve the detection rate of R2L and U2R attacks, the misclassified R2L and U2R records are applied to the next stage of filtering. It is expected that at least some incorrectly classified instances will be caught by the trained filter in the second stage.

The second stage filter is trained for only R2L and U2R attacks and is run in parallel. Training is provided by dividing the training dataset into 2 classes:

1. For R2L attacks, training set is divided into classes: R2L and others.
2. For U2R attacks, training set is divided into classes: U2R and others.

The filter is trained with only the selected features from the proposed feature selection algorithm.

It is tested against the misclassified instances obtained from stage 1.

The experimental setup is shown in Figure 13.

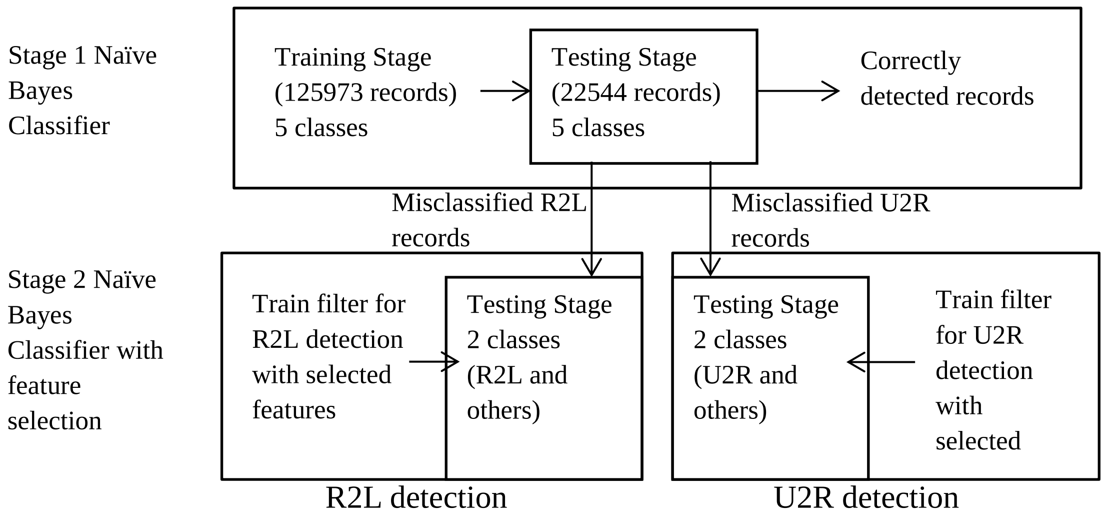

Stage 1 Naïve Bayes Classifier
Training Stage (125973 records) 5 classes
Testing Stage (22544 records) 5 classes
Correctly detected records
Misclassified R2L records
Misclassified U2R records
Stage 2 Naïve Bayes Classifier with feature selection
Train filter for R2L detection with selected features
Testing Stage 2 classes (R2L and others)
Testing Stage 2 classes (U2R and others)
Train filter for U2R detection with selected
R2L detection
U2R detection

Fig. 13 Experimental setup

The experiments were run on a Windows platform PC with Intel Core i5-3210 Processor running at 2.50 GHz and 4GB RAM. The classifications were performed in WEKA (Waikato Environment for Knowledge Analysis) version 3.7.12, a standard tool for running data mining tasks and machine learning algorithms [40].

5.1 Implementation of the feature selection algorithm

The feature selection algorithm was coded in java using Netbeans IDE version 8.0.2. WEKA classes were imported to fetch the dataset and segregate R2L and U2R records using tutorials [28].

Create new Project > include weka.jar in libraries > FeatureSelection.java

Import weka classes

```
import weka.core.Instances;
import java.io.*;
import weka.core.Instance;
import weka.core.Attribute;
import weka.filters.unsupervised.instance.RemoveWithValues;
import java.util.Random;
```

Algorithm has the following methods:

1. public void getDataSet(String path); //Retrieve the dataset
2. public void getParameters(Instances dataset, param_name); //Retreive parameters such as class index, no_of_instances, instance_no, instances_attributes etc.
3. public static String encode(Instances instance) //Used for encoding the dataset into chromosomes with the encoding scheme specified in Section 5.3.
4. public static double calculateWeight(String encode[ ]) //Calculates weights of individual features stored in weights_r2l[ ] and weights_u2r[ ].
5. public static double fitnessR2L(String encode[ ], weights_r2l[ ])
6. public static double fitnessU2R(String encode[ ], weights_u2r[ ]) //Calculates fitness of a chromosome using the fitness function defined in formula (4)
7. public static void selection(String encode[ ], int rank) //Use selection sort algorithm to sort the chromosomes and give rank
8. public static String crossover(String parents[ ], String children[ ]) //Crossover operation to form offsprings stored in children[i] and children[i+1] using parent[i] and parent[i+1].

9. public static String mutation(String children[ ], java.util.Random.rand) //Perform mutation with a random number generator.

After testing with different values of parameters given below, the best result was considered.

Mutation probability = 0.03, 0.05

Crossover probability = 1

Number of generations = 100,300,500

5.2 Implementation of two-stage classifier

```
//Importing WEKA classes in java code
import weka.core.Instance;
import weka.core.Instances;
import weka.classifiers.Evaluation;
import weka.classifiers.bayes.NaiveBayes;
import weka.classifiers.evaluation.output.prediction.PlainText;
import java.io.*;
```

Pseudo-code

```
//Training the stage 1 classifier
    1. Read training set
    2. Set class index ← attack class
    3. Create model ← (Classifier) new NaïveBayes( );      //Naive Bayes class in WEKA
    4. Build classifier on training set
    5. Save model in file as .model
    6. Evaluate model on training set
    7. Print summary

//Testing the stage 1 classifier
    1. Read testing set
    2. Set class index ← attack class
    3. Read model saved as .model
    4. Re-evaluate model on test set
    5. Print Confusion matrix, precision, recall of R2L and U2R attacks

//Obtain misclassified instances
    1. Evaluate model on test set
    2. Output predictions to .csv file
    3. Read .csv file, compare actual attack class with predicted class
    4. Output misclassified instances to r2l_misclassified.arff and u2r_misclassified.arff
```

//For stage 2 training and testing of the NB Classifier, same functions are called except that the training dataset is trained with the features selected by the proposed Genetic Algorithm.

Features are removed by using the filtering techniques provided in WEKA Explorer or by calling the deleteAttributeAt(int index) method provided in the weka.core.Instances class.

Sample code for obtaining misclassified instances

```
opstage1reader = new BufferedReader(new FileReader("D:\\Actual training and testing dataset\\Output models\\opstage1_predictions_preprocessed.arff"));
Instances opstage1=new Instances(opstage1reader);
double []actual=new double[opstage1.numInstances()];
double []predicted=new double[opstage1.numInstances()];
boolean select_instances[]=new boolean[opstage1.numInstances()];
 int count=0;
 for(int i=0;i<opstage1.numInstances();i++)
 {
          actual[i]=opstage1.instance(i).value(1);
          predicted[i]=opstage1.instance(i).value(2);
         // System.out.println(actual[i]+"\t"+predicted[i]);
       // System.out.println(opstage1.instance(i).value(0)+"\t"+actual[i]+"\t"+predicted[i]);
          if((actual[i]!=predicted[i])&& actual[i]==2)
          {
                count++;
                select_instances[i]=true;

          }
          System.out.println(i+"\t"+select_instances[i]);
 }
       /*
       The next stage classifier is also a Naive Bayes classifier with only certain features selected
        */
 BufferedReader test=new BufferedReader(new FileReader("D:\\Actual training and testing dataset\\Training and testing with normal and attack classes\\Testing_nosuccpred.arff"));
 Instances testing=new Instances(test);
 testing.setClassIndex(testing.numAttributes()-1);
 for(int i=testing.numInstances()-1;i>=0;i--)
 {
          if(!select_instances[i])
                       testing.remove(i);

 }
```

CHAPTER 6

RESULT ANALYSIS

WEKA tool produces an output whose analysis can lead to important conclusions. The following expressions are used to analyse classifier output.

True Positives (TP): Percentage of correctly detected R2L and U2R records. (Actual class = Predicted class)

False Positive (FP): Percentage of records belonging to some other attack class that are classified as either R2L or U2R records (False Alarm).

True Negative (TN): Percentage of records belonging to some other attack class that are correctly detected.

False Negative (FN): Percentage of records belonging to R2L and U2R attack class that are not detected.

A confusion matrix visualizes the performance of any classification algorithm displaying the number of true positives, false positives, true negatives and false negatives that a classifier achieves. This is shown in Figure 14.

	Predicted Class	
Actual Class	True Positive (TP)	False Negative (FN)
	False Positive (FP)	True Negative (TN)

Fig. 14. A Confusion Matrix

The Detection Accuracy Rate (DAR) refers to the total number of attacks records belonging to a particular class that are detected successfully. Parameters that used to measure the performance of the algorithm are Precision and Recall. *Precision* is the number of attack records detected correctly to the total number of records detected (TP + FP). High Precision means that the algorithm returned more relevant results than irrelevant for a particular class. *Recall* is the number of attack records detected correctly to the total number of records belonging to that class (TP + FN). High recall means that the algorithm returned most of the relevant results.

$$precision = \frac{TP}{TP + FP} \quad recall = \frac{TP}{TP + FN} \qquad \dots (7)$$

$$Detection\ Accuracy\ Rate\ (DAR) = \frac{TP + TN}{TP + TN + FP + FN} \qquad \dots . (8)$$

Results of feature selection

For R2L attacks, 21 relevant features are selected whereas for U2R attacks, 22 features are selected. Table 9 summarizes the results obtained by the feature selection algorithm.

Attack Type	**Features Selected**	**Number of features selected**
R2L	1,2,3,4,5,6,10,12,23,24,25,26, 27,28,29,32,34,36,38,39,41	21
U2R	1,2,3,4,5,6,10,11,12,14,15,17 18,19,23,24,29,31,32,33,34,36	22

Table 9. Features selected

Stage 1 Classifier Results

Overall detection accuracy rate of the stage 1 Naïve Bayes Classifier was found to be 71.2%. For R2L attack, the precision is found to be 52.2% and a very low recall of 10.5% indicating that 10.5% of the total R2L records were detected. The filter produces a very low precision of 2.6% for U2R attack records indicating that 97.4% of the records detected for this class were false positives. Recall for U2R records is found to be 25.5% (detects 51 out of 200 records). It was found that the classifier misclassifies 6489 out of the total 22544 test records. Of these misclassified records, 149 were U2R attack instances and 2466 were R2L attack instances. They were stored in u2r_misclassified.arff and r2l_misclassified.arff respectively so that they could be applied to the stage 2 of the classification. The results obtained after stage 1 of Testing are shown in Table 10.

Class	**TP Rate (%)**	**FP Rate (%)**	**Precision (%)**	**Recall (%)**
Normal	86.9	23.9	73.3	86.9
DoS	71.1	4.1	89.6	71.1
U2R	25.5	8.5	2.6	25.5
R2L	10.5	1.3	52.2	10.5
Probe	81.6	3.2	75.4	81.6
Weighted Average.	71.2	12.2	74.7	71.2

Table 10. Results of Stage 1 classifier

Stage 2 results

It can be seen that of the 2466 misclassified R2L records, 2086 were detected by the stage 2 classifier with an accuracy of 84.59%. Of the 149 misclassified U2R records, 143 were detected by the new algorithm with an accuracy of 95.97%. Out of the 6 record connections that were not classified, 4 of them belonged to the buffer_overflow attack, 1 belonged to the xterm attack and 1 belonged to the httptunnel attack. Stage 2 results are tabulated in Table 11 and Table 12.

Class	TP Rate (%)	FP Rate (%)	Precision (%)	Recall (%)
R2L	84.6	0	1	84.6
Others	0	15.4	0	0
Correctly Classified Instances -		2086	84.5904%	
Incorrectly Classified Instances-		380	15.4096%	
Classifier Accuracy -		84.59%		

Table 11. Results of Stage 2 R2L classifier

Class	TP Rate (%)	FP Rate (%)	Precision (%)	Recall (%)
U2R	95.97	0	1	95.97
Others 0	4.02	0	0	
Correctly Classified Instances -		143	95.97%	
Incorrectly Classified Instances -		6	4.02%	
Classifier Accuracy -		95.97%		

Table 12. Results of Stage 2 U2R Classifier

The proposed algorithm produced the lowest detection rate of 28% for snmpguess attack and 82.58% for snmpgetattack. This can be attributed to the fact that these attacks work on the UDP protocol, whereas all the attacks in the training set were based on the TCP protocol. As a result, the classifier was not trained properly to detect UDP based attacks and can be concluded as one of the flaws in the dataset construction. The overall detection accuracy rate of R2L and U2R attacks is calculated with the following formula:

$$overall\ detection\ accuracy = \frac{(no.of\ records\ detected\ in\ stage\ 1 + no.of\ records\ detected\ in\ stage\ 2)}{2} \quad \ldots\ldots (9)$$

The overall detection accuracy rate of the proposed approach for R2L attacks is obtained to be 86.2% and for U2R attacks it is obtained to be 97%. This shows that the proposed two-stage classifier gains a significant improvement in the detection rate of the attacks from 10.5% for R2L attacks to 86.2% and from 25.5% for U2R attacks to a substantial 97%. Comparing the results with those obtained by Ingre & Yadav in [27], it can be seen that two-stage Naïve Bayes classifier provides better results in classifying instances of the NSL-KDD dataset as compared to Artificial Neural Networks. The proposed approach significantly improves the detection rate of R2L and U2R attacks as shown in Figure 15.

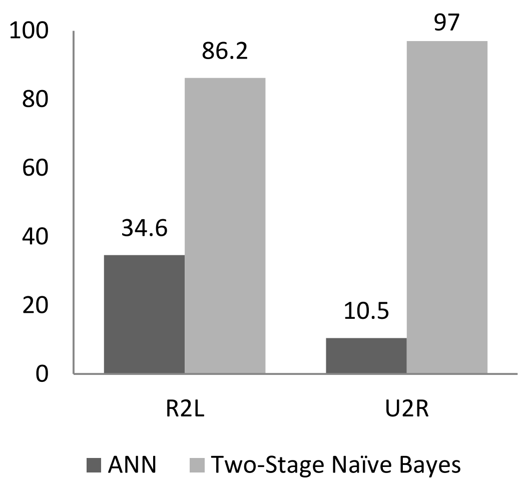


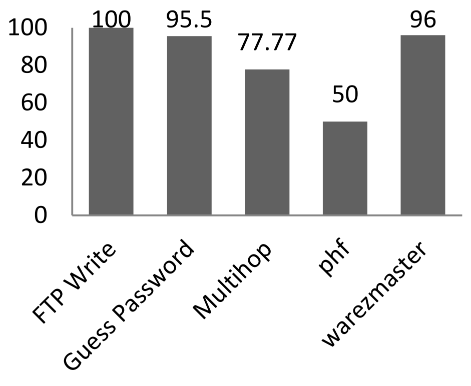
Fig 15. Comparison of the proposed approach with ANN used in [27]

Analysis of the detection rates of attacks

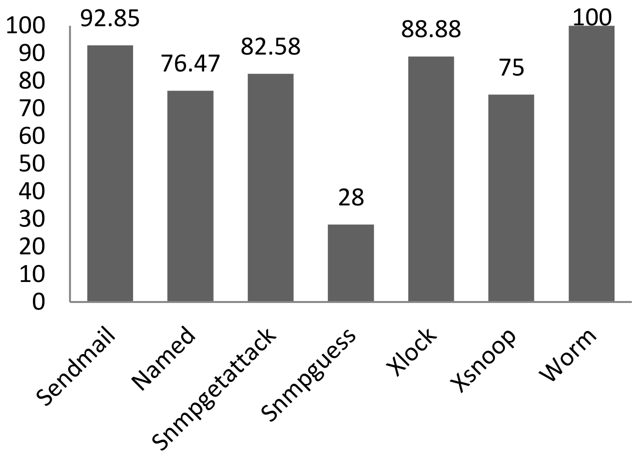
Table 13 shows the detection rates of different attacks.

Attack Type	Number of detected records	Total Number of attack records	Detection Rate (%)
R2L Attacks			
FTP Write	3	3	100
Guess Password	1176	1231	95.5
Imap	0	1	0
Multihop	14	18	77.77
Phf	1	2	50
Warezmaster	907	944	96
Sendmail	13	14	92.85
Named	13	17	76.47
Snmpgetattack	147	178	82.58
Snmpguess	93	331	28
Xlock	8	9	88.88
Xsnoop	3	4	75
Worm	2	2	100
U2R Attacks			
Buffer Overflow	16	20	80
Loadmodule	2	2	100
Perl	1	1	100
Rootkit	11	11	100
Httptunnel	132	133	99.24
Ps	15	15	100
Sqlattack	2	2	100
Xterm	12	13	92.3

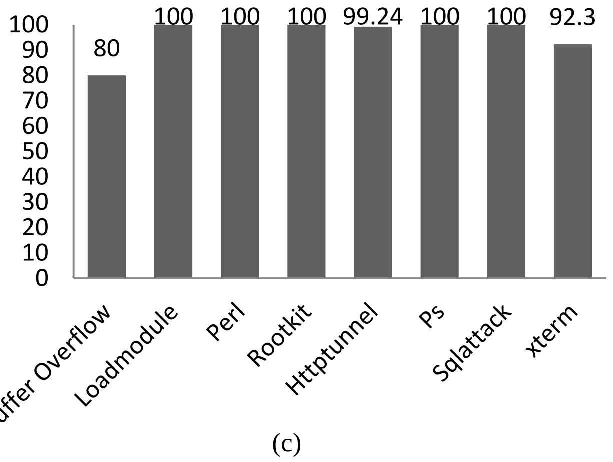
Table 13. Number of records detected for each attack type



(b)



(a)



(c)

Fig. 16. Detection rates of (a) Known R2L attacks (b) Novel R2L attacks (c) U2R attacks

Comparison of Naïve Bayes with different classifiers

Table 14 gives a comparison of different machine learners on the NSL-KDD dataset.

Classifier	Overall Accuracy (%)	R2L Detection Rate (%)	U2R Detection Rate (%)
Simple NB	71.21	10.5	25.5
C4.5 Decision Tree	75.25	6.5	4.5
Random Forest	77.89	6.9	2
Random Tree	75.85	14.7	9.5
NB Tree	77.58	8.8	9.5
Bayes Network	73.34	22.9	21
SOM	75.49	-	-
ANN	79.9	34.6	10.5
SVM	64.3	-	-

Table 14. Comparison of different machine learners on NSL-KDD dataset

The Simple Naïve Bayes classifier provides a reasonable accuracy compared to other machine learners. A Bayes Network gives a better detection rate with 22.9% R2L attacks detected and 21% U2R attacks detected successfully. It is bettered only by Artificial Neural Network based backpropagation approach as proposed in [27]. In comparison to all machine learners, SVM has the lowest performance on the NSL-KDD dataset with an overall accuracy of 64.3%. It also takes a huge time in training the dataset making it a bad choice for anomaly based IDS.

As a result Simple NB classifier is a good choice for first stage of anomaly detection which validates my claims of using it in the first stage. Now, since the output of the first stage is given as an input to the second stage, comparison of machine learners in stage 2 is shown in Table 15.

Classifier	**R2L records detected (out of 2466)**	**Detection Rate (%)**	**U2R records detected (out of 149)**	**Detection Rate (%)**
Proposed Approach	2086	84.59	143	96
C4.5	45	1.82	7	4.7
Random Forest	307	12.44	2	1.3
Random Tree	12	0.486	6	4
Simple K-Means	876	36	149	100
NB Tree	2	0.0811	6	4
Bayes Network	321	13.017	6	4
SVM [9]	64.3% (Overall detection rate)			
SOM [11]	75.49% (Overall detection rate)			

Table 15. Comparison of different machine learners in stage 2 classification

It can be seen that the misclassified instances of the Naïve Bayes classifier in the first stage are classified correctly only by the NB classifier in the second stage. NB classifier performs significantly over the other machine learners with Simple K-means clustering giving the second best results with a detection rate of 36% for R2L attack and 100% for U2R attacks.

Comparison of different feature selection methods

Table 16 and Table 17 gives the features selected by most commonly used feature selection methods i.e Correlation based Feature Selection (CFS), Information Gain, Gain Ratio and ChiSqauredAttribute evaluator.

Feature Selection Technique	**Features selected (in the order of ranking)**	**Number of features selected**
CFSSubsetEval + GeneticSearch	1,6,10,11,22,24	6
CFSSubsetEval + BestFirst	10,11,22,37	4
CFSSubsetEval + GreedyStepwise	10,11,22,37	4
InfoGainAttributeEval + Ranker	5,3,6,10,23,24,37,33,22,1,36,32, 35,12,39,4,30,29,34,38,40	21
GainRatioAttributeEval + Ranker	11,22,10,9,5,1,6,23,18,37,3,24, 12,36,33,39,29,30,26,32,40	21
ChiSquaredAttributeEval + Ranker	5,6,10,3,1,22,37,36,33,11,32, 23,24,39,40,35,12,34,26,41,4	21
ConsistencySubsetEval + GreedyStepwise	1,3,5,6,11,12,18,21,33,35,37	11
Proposed approach	1,2,3,4,5,6,10,12,23,24,25,26,27,28, 29,32,34,36,38,39,41	21

Table 16. Features selected by different techniques for R2L attacks

Feature Selection Technique	**Features selected (in the order of ranking)**	**Number of features selected**
CFSSubsetEval + GeneticSearch	9,14,17,18,32	5
CFSSubsetEval + BestFirst	14,17,18,32	4
CFSSubsetEval + GreedyStepwise	14,17,18,32	4
InfoGainAttributeEval + Ranker	3,14,1,10,17,13,32,24,33,6, 5,23,26,16,12,41,4,18,38,35	20
GainRatioAttributeEval + Ranker	14,18,17,9,16,13,10,1,32,5, 6,33,36,23,24	15
ChiSquaredAttributeEval + Ranker	10,14,17,13,1,16,18,3,5,41, 32,9,33,6,24	15
OneRAttributeEval + Ranker	41,15,13,14,16,11,17,19, 12,10,40,5,2,4,6,9,7	17
Proposed approach	1,2,3,4,5,6,10,11,12,14,15,17,18,19.23, 24,29,31,32,33,34,36	22

Table 17. Features selected by different techniques for U2R attacks

Table 18 gives a comparison of the using the feature selected by different techniques in stage 2 using the NB classifier.

Feature Selection Technique	**R2L records detected (Out of 2466)**	**Detection Rate (%)**	**U2R records detected (Out of 149)**	**Detection Rate (%)**
Proposed Approach using Genetic Algorithm	2086	86.2	143	95
CFSSubsetEval + GeneticSearch	698	283	10	6.7114
CFSSubsetEval + BestFirst	54	2.1898	10	6.7114
CFSSubsetEval + GreedyStepwise	54	2.1898	10	6.7114
InfoGainAttributeEval + Ranker	1323	53.64	2	1.342
GainRatioAttributeEval + Ranker	1396	56.60	1	0.6711
ChiSquaredAttributeEval + Ranker	1260	51.09	7	4.698
ConsistercySubsetEval + GreedyStepwise	846	34.30	11	7.38

Table 18. NB classifier output in stage 2 using features selected by different techniques

The proposed feature selection technique using genetic algorithm performs significantly better with GainRatioAttribute evaluator giving second best results for R2L attacks (56.6%). The Gain Ratio attribute evaluator gives ranks to features.

Trends in feature selection

The GainRatioAttributeEval calculates the information gain ratio (GR) for each feature and the ranker method ranks the features in the order of relevance.

The order of relevance is as specified

(11,22,10,9,5,1,6,23,18,37,3,24,12,36,33,39,29,30,26,32,40) amounting to 21 selected features.

However, considering all 41 features in order of relevance

(11,22,10,9,5,1,6,23,18,37,3,24,12,36,33,39,29,30,26,32,40,38,4,25,14,41,35,17,2,34,31,8,28,7,27,21,16,13,19,20,15)

Evaluating the trend in the number of features selected in the order of ranks to the detection rate using the NB classifier.

No. of features selected (Rank wise)	**R2L records detected (out of 2466)**	**Detection Rate (%)**
4	54	2.18
7	333	13.50
8	2185	88.6
9	2185	88.60
10	2184	88.6
14	1931	78.30
16	1261	51.13
18	1356	54.98
21	1396	56.60
24	1597	64.76
27	1647	66.78
29	1634	66.26
33	1776	72.01
37	1779	72.14
41	1779	72.14

Table 19. Number of records detected by NB classifier wrt features selected for R2L attacks

Analysing the trends in the number of features selected to the detection rate of R2L attacks, it can be seen that the detection rate is maximum (88.6%) when 8 features are selected i.e features 11,22,10,9,5,1,6,23. The detection rate climbs from a miserable 13.5% when 7 features are selected to a maximum of 88.6% with the addition of feature no. 23. Thereafter, with the addition of features in the order of relevance, the detection rate decreases steadily. When all the features are selected and the 2nd stage Naïve Bayes Classifier is tested, the detection rate is 72.14%. Hence, we conclude that when the classifier is trained to detect only R2L attacks in the 2nd stage, 72.14% of the misclassified records are detected correctly when all features are selected.

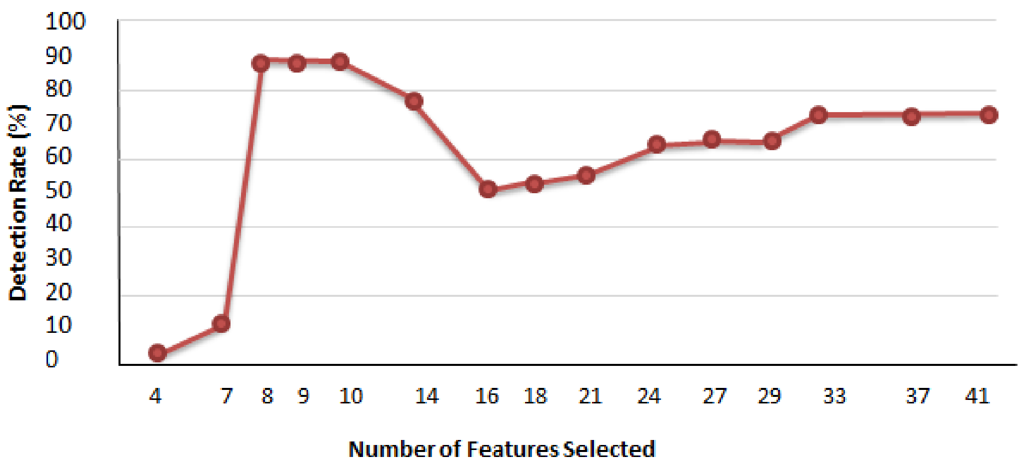


Fig 17. Number of features selected by Gain Ratio vs Detection rate of R2L using NB classifier

The most important feature for R2L attack is found to be feature no. 23. i.e. count which advocates the conclusions obtained in [9].

The GainRatioAttributeEval + Ranker search feature selection technique is found to be better than the proposed algorithm for finding relevant features for R2L attacks. The accuracy of the proposed algorithm is 86.2% as compared to this technique which is 88.6%. The accuracy is 2.4% lower than GainRatioAttributeEval.

Table 20 shows the number of U2R records detected correctly using a NB classifier in stage 2 using different selected features in the order of ranks.

No. of Feature Selected	U2R records detected (out of 149)	Detection Rate (%)
4	9	6.04
7	8	5.36
10	9	6.0403
11	37	24.83
12	119	79.86
13	74	49.66
14	135	90.60
15	140	93.95
16	8	5.36
17	8	5.36
22	1	0.6711
25	2	1.34
30	1	0.6711
41	1	0.6711

Table 20. Number of records detected by NB classifier wrt features selected for U2R attacks

The detection rate achieves maxima (93.95%) when 15 features are selected. i.e 14,18,17,9,16,13,10,1,32,5,6,33,36,23,24. Thereafter, with the addition of features in the order of relevance, the detection rate dips to 5.36% when 17 features are selected and 0.6711% when all 41 features are selected.

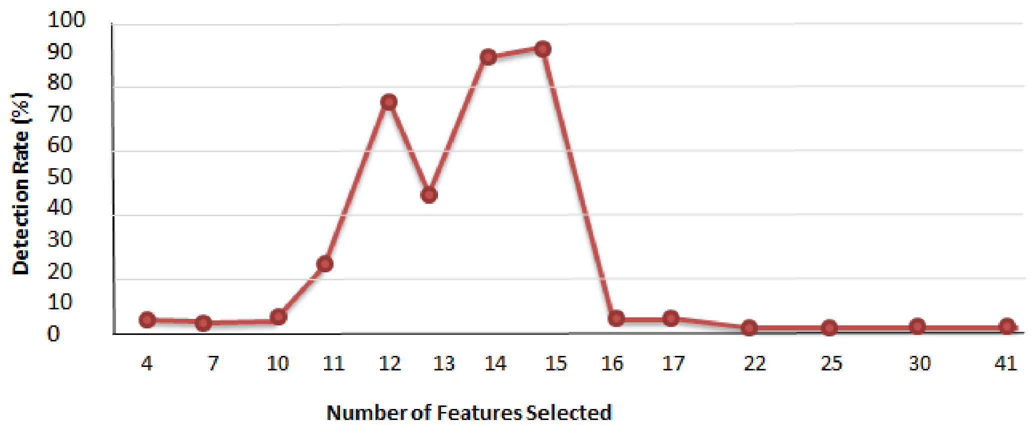


Fig. 18 Number of features selected by Gain Ratio vs Detection rate of U2R using NB classifier

The proposed approach for feature selection performs better in this case with the detection rate of 95%, almost 1.05% higher than GainRatioAttributeEval + Ranker algorithm. It detects 3 more U2R records correctly than GainRatioAttributeEval.

In general, after the analysis of the results, it was concluded that a two-stage or a multi-stage classifier improves the detection rate as compared to a single stage classifier. Usage of feature selection at different stages even enhances the detection rate further. The proposed feature selection algorithm gives the best results for U2R attacks with the detection rate reaching 95% in the second stage. However, its detection rate is lower than the GainRatioAttribute evaluator which classifies 88% of the records correctly.

Summary

As seen in Table 9, the number of features selected for R2L and U2R attacks are 21 and 22 respectively. This shows that approximately 50% of the features are relevant and hence the speed up caused due to this increases by almost 95%. As a result, improved training times will reduce the complexity of execution. To determine whether the detection rates have improved, study Tables 10, 11 and 12. The number of attack records detected for each attack type is given in Table 13. Analysis of the table shows that Snmpguess and snmpgetattack are difficult to detect since the classifier is not trained properly to detect UDP attacks. The protocol(feature no. 2 in Table 1) is biased towards TCP attacks, and hence the classifier produces biased results towards TCP based malicious records. The detection rate of U2R attacks is high with only 4 buffer overflow records not classified properly. Comparison of different classifiers is done in Table 14 and Table 15, showing the difference in variations of others with respect to Naïve Bayes.

Finally the filter feature selection technique, GainRatioAttribute eval along with a ranker search algorithm was compared with the proposed algorithm. Thorough analysis of GainRatioAttribute eval shows that the optimal subset varies according to number of features selected. The detection rate of this FS method reaches 88.6% when 8 features are selected for R2L attacks and 93.95% when 15 features are selected for U2R attacks.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

In this dissertation, a novel two-stage Naïve Bayes classifier was proposed to enhance the detection rates of R2L and U2R attacks. As discussed in Chapter 7, it is seen that the proposed method is better than most other existing techniques, improving the detection rate significantly. The below report summarizes my work, my research findings and discusses the future scope of this undertaking.

Summary

1. Feature selection is the proposed solution to improve detection rates of R2L and U2R attacks. Meta-heuristic based evolutionary computing algorithm is used as the feature selection technique.
2. Fitness function in GA is based on entropy based weight calculation.
3. Feature selection is run over 300 generations to find 21 relevant features for R2L attacks and 22 relevant features for U2R attacks.
4. Two-stage Naïve Bayes classifier is tested on the NSL-KDD dataset. NSL-KDD dataset was used since it removes the redundant records in the KDD99 dataset so that classifiers do not produce biased results towards the more frequent records.
5. The second stage classifier is run in parallel to detect both attacks, thereby increasing speed of execution.
6. The proposed method was compared with existing techniques, feature selection techniques were compared and even Gain Ratio feature selection technique is discussed in detail.

Summary of the results

1. Overall detection rate of the proposed algorithm is 86.2% for R2L attacks and 97% for U2R attacks. This is compared with the performance of backpropagation based neural networks in [27] and it is seen that the proposed method significantly improves results.
2. Snmpguess R2L attack was the most difficult to detect with 28% detection rate. This is because the classifier was not trained properly to detect UDP protocol packets since the training set consisted of all TCP based attacks.

3. Simple NB classifier has lower detection rates than C4.5 decision tree, SOM and NB tree, in a single detection stage. However, it is faster in achieving the results than the rest.
4. However, when two-stage NB classifier is used, its performance gets upgraded drastically along with achieving those output faster than the others.
5. NB Tree has a better performance than simple NB but lower speed.
6. GA based feature selection takes a comparable time with respect to CFSSubsetEval + GeneticSearch, however it is slower than Information Gain, Gain Ratio and CFSSubsetEval + bestfirst or greedy.
7. Using GA based feature selection gives the best accuracy with GainRatioAttributeEval + Ranker algorithm coming second best with an accuracy of 56%.
8. The performance of a classifier depends upon the optimal feature subset. 'Optimal' does not mean having the least number of features. As seen in Figure 16 and Figure 17, Gain Ratio attribute evaluator achieves an optimal result when 8 features are selected for R2L attacks and when 15 features are selected for U2R attacks.
9. The optimal feature subset varies for each feature selection algorithm. As a result, it is difficult to determine whether a particular algorithm is better than the other. Using a feature selection technique will depend upon cost complexity, usage and the environmental settings.
10. It was found that feature no. 23 i.e count (number of records to the server) is the most important for R2L attacks.
11. Since Naïve Bayes is faster than most other machine learners, its usage in the proposed system can serve as a prototype for organizations to enhance security.

Future Scope

This algorithm was tested with the NSL-KDD dataset which is not a real life representation of a network since it consists of spurious background traffic. Analysing this algorithm with a more realistic attack traffic like the ISCX 12' dataset can help in determining the proposed approach's worth. Real life deployment of anomaly detectors is still a big problem. Deploying a two stage classifier will really help in analysing real time detection parameters thereby help organizations to find the most reliant cost effective anomaly detector. The proposed method can be used to improve detection rates of DoS attacks as per the need.

REFERENCES

[1] Hung-Jen Liao; Chun-Hung Richard Lin et.al, “Intrusion Detection System: A comprehensive review”, *Journal of Network and Computer Applications* 36 (2013), pg 16-24.

[2] Monowar H. Bhuyan, D.K Bhattacharyya; J.K. Kalita, “Network Anomaly Detection: Methods, Systems and Tools”, *IEEE Communications Surveys & Tutorials*, Vol 16, No 1, First Quarter 2014

[3] DARPA dataset evaluation, MIT Lincoln Laboratories, 1999. Online: http://www.ll.mit.edu/ideval/docs/

[4] Carlos A. Catania, Carlos Garcia Garino, “Automatic network intrusion detection: Current techniques and open issues”, *Journal of Computers and Electrical Engineering*, Volume 38 Issue 5, Sept 2012, pg 1062-1072.

[5] Rebecca Bace and Peter Mell, NIST Special Publication on Intrusion Detection Systems, National Institute of Standards and Technology, pg 1-51.

[6] Nicholas Pappas, Network IDS & IPS Deployment Strategies, SANS Institute, April 2, 2008, pg 1-64.

[7] Zheng Zhang, Jun Li, C. N. Manikopoilos, Jay Jorgenson, Josh Ucles, “HIDE: a hierarchical network intrusion detection system using statistical preprocessing and neural network classification”, *IEEE Workshop on Information Assurance and Security, 2001.*

[8] Guiling Zhang, Yongzhen Ke, Zhichao Li, Mingjie E, “Improvements of Payload-based Intrusion Detection Models by using Noise against Fuzzy SVM”, *Journal of Networks*, Vol 6, No 2(2011), 330-340, Feb 2011.

[9] Hesham Altwaijry, Saeed Algarny, “Bayesian based intrusion detection system”, *Journal of King Saud University – Computer and Information Sciences (2012),* 24, pg 1-6

[10] Hesham Altwaijry, Saeed Algarny, “Multi-Layer Bayesian Based Intrusion Detection System”, *Proceedings of the World Congress on Engineering and Computer Science 2011*, Vol II, WCECS 2011, Oct 19-21, 2011, San Francisco, USA..

[11] Cheng Xiang, Png Chin Yong, Lim Swee Meng, “Design of multiple-level hybrid classifier for intrusion detection system using Bayesian clustering and decision trees”, *Pattern Recognition Letters* 29(2008), 918-924.

[12] Hui Lu, Jinhua Xu, “Three-level Hybrid Intrusion Detection System”, *International Conference on Information Engineering and Computer Science* (ICIECS 2009), pg 1-4.

[13] Kok-Chin Khor, Choo-Yee Ting, Sommuk-Phon Amnuaisuk, “Comparing Single and Multiple Bayesian Classifiers Approaches for Network Intrusion Detection”, *International Conference on Computer Engineering and Applications*, 2010,325-329.

[14] Yuteng Guo, Beizhan Wang Et.al, "Feature Selection Based on Rough Set and Modified Genetic Algorithm for Intrusion Detection", *5th International Conference on Computer Science & Education*, Hefei, China, August 24-27th ,2010, pg 1441-1446.

[15] Hua Zhou, Xiangru Meng, Li Zhang, "Application of Support Vector Machine and Genetic Algorithm to Network Intrusion Detection", *Wireless Communications, Networking and Mobile Computing 2007*, WiCom 2007, 21-25 Sept. 2007, pg 2267-2269.

[16] Megha Agarwal, "Performance Analysis of Different Feature Selection Methods in Intrusion Detection", *International Journal of Scientific & Technology Research* Volume 2, Issue 6, June 2013.

[17] Datta H.Deshmukh, Tushar Ghorpade, Puja Padiya, "Intrusion Detection System by Improved Preprocessing Methods and Naïve Bayes Classifier using NSL-KDD 99 Dataset", *2014 International Conference on Electronics and Communication Systems (ICECS*), 13-14th Feb 2014, pg 1-7

[18] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," *Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2009.

[19] J. McHugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by Lincoln laboratory", *ACM Transactions on Information and System Security*, Vol. 3, no. 4, pp. 262-294, 2000.

[20] Richard Lippman, Joshua W. Haines, David J. Fried, Jonathan Korba, Kumar Das, "The 1999 DARPA off-line intrusion detection evaluation", *International Journal of Computer and Telecommunications Networking-Special issue on recent advances in intrusion detection systems*, Volume 34 Issue 4, Oct. 2000, pages 579-595.

[21] Laheeb M. Ibrahim, Dujan T. Basheer, Mahmood S. Mahmood, "A Comparison Study for Intrusion Database(KDD99, NSL-KDD) Based on Self-Organization Map (SOM) Artificial Neural Network", *Journal of Engineering Science and Technology*, Vol. 8, No. 1(2013) 107-119.

[22] B.Senthilnayaki, K.Venkatalakshmi, A.Kannan, "An Intelligent Intrusion Detection System Using Genetic based Feature Selection and Modified J48 Decision Tree Classifier", *2013 Fifth International Conference on Advanced Computing (ICoAC)*, 18-20th Dec 2013, pg 1-7.

[23] Salah Eddine Benaicha, Lalia Saoudi Et.al, "Intrusion Detection System Using Genetic Algorithm", *Science and Information Conference 2014*, August 27-29th 2014, London, UK, pg-564-568.

[24] Dheeraj Pal, Amrita Parashar, "Improved Genetic Algorithm for Intrusion Detection System", 2014 *Internationl Conference on Computational Intelligence and Communication Networks (CICN)*, 14-16th Nov. 2014, pg 835-839.

[25] NSL-KDD Dataset [Online]. Available: http://nsl.cs.unb.ca/NSL-KDD/

[26] Chang-Hwan Lee, Fernando Guitierrez, Dejing Dou, “Calculating Feature Weights in Naïve Bayes with Kullback-Leibler Measure”, *IEEE International Conference on Data Mining*, 2011, pg 1146-1151.

[27] Bhupendra Ingre, Anamika Yadav, “Performance analysis of NSL-KDD dataset using ANN”, International Conference on Signal Processing and Communication Engineering Systems (SPACES), 2-3rd Jan 2015, pg 92-96.

[28] Svetlana S. Akesnova, WEKA Explorer Tutorial, 2004.

[29] KDD cup winner results. [online] http://cseweb.ucsd.edu/~elkan/clresults.html

[30] Rupali Datti, Bhupendra Verma, “Feature Reduction for Intrusion Detection Using Linear Discriminant Analysis”, International Journal on Computer Science and Engineering, Vol 2. No5, 2010, pg 1072-1078.

[31] Maheshkumar Sabhani, Gursel Serpen, “KDD Feature Set Complaint heuristic rules for R2L attack Detection”, Security and Management, 2003, pp 310-316.

[32] Olympia Roeva, Stefka Fidanova, Marcin Paprzycki, “Influence of the population size on the genetic algorithm performace in case of cultivation process modelling”, 2013 Federated conference on computer science and Information systems, pp-371-376.

[33] Rafeeq Ur Rehman, “Intrusion detection system using snort”, Prentice Hall PTR, 2003.

[34] V. Chandola, A. Banerjee, and V. Kumar, “Anomaly Detection:A Survey,” University of Minnesota, Tech. Rep., 2007.

[35] L. Khan, M. Awad, and B. Thuraisingham, “A New Intrusion Detection System Using Support Vector Machines and Hierarchical Clustering,” *The VLDB Journal*, vol. 16, no. 4, pp. 507–521, October 2007.

[36] KDD Cup 99’ dataset [online]:http://kdd.ics.uci.edu/databases/kddcup99/task.html

[37] Mark A. Hall, “Correlation-based feature selection for machine learning”, University of Waikato, Hamilton, New Zealand (1999).

[38] Mya Thidar Myo Win, and Kyaw Thet Khaing, “Detection and Classification of Attacks in Unauthorized Accesses”, *International Conference on Advances in Engineering and Technology (ICAET'2014)*, March 29-30, 2014 Singapore.

[39] Tom Carter, “An introduction to information theory and entropy”, Santa Fe, Aug 2014.

[40] WEKA machine learning tool. Available [online]: http://www.cs.waikato.ac.nz/ml/weka/