ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION SCIENCE

CONSTRUCTING PREDICTIVE MODEL FOR NETWORK INTRUSION DETECTION

BY:-

TIGABU DAGNE AKAL

JUNE 2012

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION SCIENCE

CONSTRUCTING PREDICTIVE MODEL FOR NETWORK INTRUSION DETECTION

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of Master of Science in Information Science

BY:-

TIGABU DAGNE AKAL

JUNE 2012

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

SCHOOL OF INFORMATION SCIENCE

CONSTRUCTING PREDICTIVE MODEL FOR NETWORK INTRUSION DETECTION

BY:-

TIGABU DAGNE AKAL

Name and signature of Members of the examining board:

Name	**Title**	**signature**	**Date**
MAHIDER ALEMAYHU	Chairperson	_____	June 13/2012
GASHAW KEBEDE	Advisor	_____	June 13/2012
WORKSHET LAMENEW	Examiner	_____	June 13/2012

DEDICATION

This thesis is dedicated to my father (Ato Dagne Akal) and my mother (W/o Kassaye Bayhe). They are always a great role model for me because of their devotion and who always encourages me to make the best effort to realize my dreams, and their love to our family. Their spirit will be with me forever.

ACKNOWLEDGEMENT

First and foremost extraordinary thanks go for my Almighty God and His Mother Saint-Merry.

It is with immense gratitude that I acknowledge the support and help of my advisor, Dr. Gashaw Kebede. I have been amazingly fortunate to have an advisor who gave me the freedom to explore on my own and at the same time the guidance to recover when my steps faltered. He taught me how to question thoughts and express ideas. His patience, advises and support helped me overcome challenging situations and finish this thesis, plus his comments added great value for my future career.

It gives me great pleasure in acknowledging the support and help of Dr. Million Meshesha who is one of the best teachers that I have had in my life. He sets high standards for his students and he encourages and guides them to meet those standards. He taught me more than four courses during my postgraduate study. He has given constructive ideas and comments about my thesis work which is related to one of the course, Data and Web Mining, that he has introduced me.

I would like to thank all lecturers, staff members and classmates of School of Information Science for their contribution in the special arena for the success of my study.

Many friends have helped me stay rational through these two years. I greatly value their friendship and deeply appreciate their belief in me. I have to give a special mention for my best friend Sewale Belachew. I have no word to express my feeling for his endless help to pull off my dream for the past eight years. He is unrevealed person for my success. Also, I would like to thank my friends: Adane L., Belaynaw A., Biniyam C., Getaneh B., Meazashwork A., Tariku A., Tsegaye B., and Zewdie M. Their support and care helped me to overcome setbacks and stay focused on my postgraduate study.

Most importantly, none of these would have been possible without the love and patience of my family. My father and my mother, to whom this thesis is dedicated to, have been a constant source of love, concern, support and strength all the past years. In addition, I would like to express my heart-felt gratitude to my sisters: Mastewal T., Netsanet A., Selam A. and Yikerta L., and to my brothers: Abate T., Admasu T., Mulugeta D., Tadele A., and Temesgen D. who have aided and encouraged me throughout this study.

Finally, I share the credit of my work for the rest of all my brothers, sisters, friends and ICT staff members of University of Gondar that I have not mentioned their name here.

Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ARFF: Attribute Relation File Format

CFS: Correlation Feature Selection

CRISP-DM: Cross Industry Standard Process for Data Mining

CSV: Comma Separated Values

DARPA: Defense Advanced research Project Agency

DM: Data Mining

DOS: Denial of Services

FP: False Positive

Http: hyper text transfer protocol

ICMP: Internet control message protocol

IDPS: Intrusion Detection Prevention Systems

IDS: Intrusion Detection System

IGR: Information Gain Ratio

IPS: Intrusion Prevention Systems

KD: Knowledge Discovery

KDD: Knowledge Discovery in Database

MIT: Massachusetts Institute of Technology

NBA: Network Behavior Analysis

NIDS: Network intrusion detection system

NSL: Network Simulation Language

num_failed_logins: number of failed logins

R2L: Remote to Local

SEMMA: Sample, Explore, Modify, Model, and Assess

SF: Source Flag

TCP: Transmission Control Protocol

TP: True Positive

U2R: User to Remote

UDP: User Datagram protocol

VPN: Virtual Private Network

WEKA: Waikato Environment for Knowledge Analysis

ABSTRACT

While advances in computer and communications technology have made the network ubiquitous, they have also rendered networked systems vulnerable to malicious attacks devised from a distance. These attacks or intrusions start with attackers infiltrating a network through a vulnerable host and then launching further attacks on the local network or Intranet. Nowadays, system administrators and network professionals can attempt to prevent such attacks by developing intrusion detection tools and systems using data mining technology.

In this study, the experiments were conducted following the Knowledge Discovery in Database process model. The Knowledge Discovery in Database process model starts from selection of the datasets. The dataset used in this study has been taken from Massachusetts Institute of Technology Lincoln laboratory. After taking the data, it has been preprocessed. The major preprocessing activities include fill in missed values, remove outliers; resolve inconsistencies, integration of data that contains both labeled and unlabeled datasets, dimensionality reduction, size reduction and data transformation activity like discretization tasks were done for this study.

A total of 21,533 intrusion records are used for training the models. For validating the performance of the selected model a separate 3,397 records are used as a testing set. For building a predictive model for intrusion detection J48 decision tree and the Naïve Bayes algorithms have been tasted as a classification approach for both with and without feature selection approaches.

The model that was created using 10-fold cross validation using the J48 decision tree algorithm with the default parameter values showed the best classification accuracy. The model has a prediction accuracy of 96.11% on the training datasets and 93.2% on the test dataset to classify the new instances as normal, DOS, U2R, R2L and probe classes. The findings of this study have shown that the data mining methods generates interesting rules that are crucial for intrusion detection and prevention in the networking industry. Future research directions are forwarded to come up an applicable system in the area of the study.

CHAPTER ONE

INTRODUCTION

1.1. Background

As network-based computer system plays increasingly vital roles in modern society. They have become the targets of cyber criminals. The security of a computer system is compromised when an intrusion takes place. An intrusion was defined as "any set of actions that attempt to compromise the integrity, confidentiality or availability of a resource" (Heady et.al, 1990).

Lee et al (1999) defined intrusion as the act or attempted act of using a computer system or computer resources without the requisite privileges, causing willful or incidental damage. Intrusion Detection Systems (IDS) are computer programs that attempt to perform intrusion detection by comparing observable behavior against suspicious patterns, preferably in real-time.

IDSs are systems that attempt to identify intrusions or abuses of computer systems by either authorized users or external perpetrators (Mukherjee et al., 1994). Some IDSs monitor a single computer, while others monitor several computers connected by a network. IDSs detect intrusions by analyzing information about user activity from sources such as audit records, log files, system tables, and network traffic summaries. Intrusion detection is a new, modern approach for providing a sense of security in existing computers and data networks, while allowing them to operate in their current "open" mode (Mukherjee et al., 1994).

IDSs have been developed and used at several institutions. Some example of IDSs are National Security Agency's Multics Intrusion Detection and Alerting System (MIDAS), ATandT's Computer Watch (Dowell and Ramstedt, 1990), SRI International's Intrusion Detection Expert System (IDES) (Lunt,1990), Next-Generation Intrusion-Detection System (NIDES) (Anderson, 1994), UC Sanat Barbara's State Transition Analysis Tool for UNIX

(USTAT) (Ilgun, 1993; Ilgun et al.,1995), Los Alamos National Laboratory's (LANL's) Network Anomaly detection and Intrusion Reporter (NADIR) (Hochberg, 1993), UC Davis' Network Security Monitor (NSM) (Heberlein et al.,1990) and Distributed Intrusion Detection System (DIDS) (Snapp, 1991).

Intrusion prevention techniques such as user authentication (e.g. using password or biometrics), avoiding programming errors, and information protection (e.g., encryption) have been used to protect computer systems as a first line of defense (Lee et al., 1999).

Intrusion detection is needed as a wall to protect computer systems. The elements central to intrusion detection are: resources to be protected in target system, i.e., user accounts, file systems, system kernels, etc; models that characterize the "normal" or "legitimate" behavior of these resources; techniques that compare the actual system activities with the established models, and identify those that are "abnormal" or "intrusive" (Lee et al., 1999).

According to Mukherjee et al (1994) the goal of intrusion detection is to identify, preferably in real time, unauthorized use, misuse, and abuse of computer systems by both system insiders and external penetrators.

Generally, an intrusion would cause loss of integrity, confidentiality, denial of resources, or unauthorized use of resources. According to Eric et al (2002), some specific examples of intrusions that concern system administrators include:

- ✓ Unauthorized modifications of system files so as to facilitate illegal access to either system or user information;
- ✓ Unauthorized access or modification of user files or information;
- ✓ Unauthorized modifications of tables or other system information in network components (e.g. modifications of router tables in an internet to deny use of the network);
- ✓ Unauthorized use of computing resources (perhaps through the creation of unauthorized accounts or perhaps through the unauthorized use of existing accounts).

The most widely used and commercially available IDSs are signature-based systems (William, 1995). A signature based system matches features observed from the audit stream to a set of signatures handcrafted by experts and stored in a signature database. Signature-based methods have some inherent limitations. The most significant is that a signature-based method is designed to only detect attacks for which it contains a signature in the database. In addition to the expense in time and human expertise of manually encoding a signature for each and every known attack, the signature-based methods therefore cannot detect unknown attacks since there is no signature in the database for them. It is these unknown attacks that are typically the most dangerous because the system is completely vulnerable to them (William, 1995).

Data mining (DM) -based methods are another paradigm for building intrusion detection systems. The main advantage of these methods is that they leverage the generalization ability of data mining methods and in order to detect new and unknown attacks. A data mining-based IDS uses machine learning and data mining algorithms on a large set of system audit data to build detection modes. These models have been proven to be very effective (Lee et al., 1999). These algorithms are generally classified as either misuse detection or anomaly detection. Misuse detection algorithms learn how to classify normal and attack data from a set of training data which contains both labeled attack and normal data (Lee et al., 1999). Anomaly detection algorithms learn a model of normal activity by training on a set of normal data. Anomaly detection algorithms then classify as an attack activity that diverges from this normal pattern based on the assumption that attacks have much different patterns than normal activity. In this way new unknown attacks can be detected (Hershkop et al., 2007).

According to Christos and Aikaterini (2004) a DM intrusion detection system (IDS) inspects all inbound and outbound network activity and identifies suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system. According to Christos and Aikaterini (2004) data mining has been known to aid the process of Intrusion Detection and the ways in which the various techniques have been applied to enhance the security of the network

Generating patterns and knowledge is vital for IDSs to differentiate standard behaviors from strange behavior by examining the dataset which is a list of tasks created by the operating system that are registered into a file in historical sorted order (Dewan and Mohammad, 2010).

According to Pachghare et al (2011) IDS can be implemented using unsupervised, supervised and semi-supervised machine learning algorithms. Unsupervised learning uses unlabeled data. This method can detect the intrusions that have not been previously learned. Examples of unsupervised learning for intrusion detection include K-means-based approaches and self-organizing map (SOM)-based approaches. In supervised learning for intrusion detection, the labeled data is needed for training. These are mainly neural network (NN)-based approaches, and support vector machine (SVM)-based approaches for IDS. The third method is semi-supervised learning in which both the labeled and unlabeled data is used for training.

1.2. Statement of the Problem

Intrusion detection problem is becoming a challenging task due to the proliferation of heterogeneous computer networks since the increased connectivity of computer systems gives greater access to outsiders and makes it easier for intruders to avoid identification (Helali, 2010). Hence, there is a need of effective and efficient system which allows protecting the network from the intruders. To develop such kind of system there is a need to use methods like feature selection which is a growing field of interest about selecting proper features from many features. This is because it is expensive to carry out the entire process and degrades the classification performance of data mining algorithms. Therefore, feature selection approaches reduce the complexity of the overall process by allowing the data mining system to focus on what are really important features.

Many researchers proposed different models for network intrusion detection system (NIDS). Adamu (2010) has tried to study a machine learning IDS that investigated the application of cost sensitive learning by applying decision tree algorithm. He did not compare the result with other predictive model techniques like neural network, Naïve Bayes and other techniques.

Zewdie (2011) proposed an optimal feature selection for Network Intrusion Detection using indirect cost sensitive feature selection approach. It is a DM approach system that tried to investigate jointly cost sensitive learning and feature selection to advance the classification performance of algorithms that incorporate cost. In his study, Information Gain Ratio (IGR) and Correlation Feature Selection (CFS) are investigated for ranking and selecting features using the proposed cost sensitive approach. Zewide has tried to investigate decision tree classification algorithms that used indirect cost sensitive feature ranking and selection algorithms. Zewdie used in his study only those records which are labeled. . He did not consider those records which are not labeled.

Both Adamu(2010) and Zewdie (2011) conducted the NIDS on a supervised approach. As described by Pachghare et al (2011) traditional intrusion detection algorithm is based on supervised learning and non-supervised learning. These two algorithms have some limitations; the supervised learning process cannot use a lot of unlabeled data while non-supervised learning often results in high false alarm rate.

Pachghare et al (2011) has evaluated the performance of the supervised intrusion detection model using labeled data. They concluded that labeling the training data for real-world applications is difficult, expensive, or time consuming, as it requires the effort of human sometimes with specific domain experience and training. There are implicit costs associated with obtaining these labels from domain experts, such as limited time and resources. This is especially true for applications that involve learning with large number of class labels and sometimes with similarities among them. Finally, Pachghare et al (2011) recommended Semi-supervised intrusion detection system as future direction.

This study attempted to construct a semi-supervised model by considering both labeled and unlabeled records and compared the performance gap of different classification technique. At the same time, the study has compared the result with feature selection algorithms that incorporate costs by applying CfsSubsetEval as attribute evaluator and Best First as a search method that were recommended as future research direction by Zewdie (2011).

Therefore, this research intends to get answers for the following research questions.

- ✓ Which Data Mining algorithm can be more suitable for the purpose of predicting Network Intrusions?
- ✓ To what degree can the NIDS correctly classify intrusions, and can the system correctly classify intrusion to such a degree that it can be trusted to respond actively to them?
- ✓ What is the pattern that describes whether given networks signal is a normal packet or an intrusion?
- ✓ How to design an IDS model which is based on features selection?

1.3. Objective of the Study

1.3.1. General objective

The general objective of this study is constructing a predictive model using a Semi-Supervised approach for intrusion detection system that will enhance the network security system.

1.3.2. Specific objectives

- ✓ To review previously proposed related works and there by identify different techniques and algorithms that have been used in the area of data mining for feature selection and model construction approaches,
- ✓ To study different classification of anomalies detection techniques,
- ✓ To construct an IDS model using a semi-supervised approach,
- ✓ To compare different classification DM algorithms with and without feature selection approaches for designing a network intrusion detection model,
- ✓ To apply different DM algorithms and to select the better classification algorithm for designing a prediction model for NIDS.
- ✓ To validate the designed intrusion detection model and describe its performance

1.4. Scope and Limitation of the Study

This research is basically the extension of the thesis work of Adamu (2010) and Zewdie (2011) in the area of NIDS. On top of their work those features which were not addressed by both of them has been addressed. Namely, the result of the J48 decision tree algorithm

compared with other predictive model techniques in developing an IDS model. For feature selection CfsSubsetEval as attribute evaluator and Best First as a search method are used in this study.

This research addressed the Semi-supervised modeling of intrusion detection system that considers both labeled and unlabeled records which were indicated as a future research direction by Pachghare et al (2011). Because it is not easy to classify network packets either attack or normal that always needs domain experts for applying only supervised modeling. At the same time labeling the class of network packets consumed resources.

Due to the datasets were taken from the Massachusetts Institute of Technology (MIT) Lincoln lab, this research did not include data from network security organizations in Ethiopia. So, further research needs to be conduct including data from these organizations. Because of time and financial limit this research also focused mainly on how to effectively detect attacks, not to prevent them. The IDS model constructed in this thesis just notify for the administrators after detecting an attack and administrators have to manually take proper actions.

1.5. Methodology of the Study

The methodologies that are used in conducting this research are described as follows.

1.5.1. Literature Review

The researchers reviewed different related literatures (books, journal articles, Conference proceeding papers, and the Internet) in order to have detailed understanding on the present research. As a continuation of the previous attempts by Adamu (2010) and Zewdie (2011), different techniques and tools which are relevant for the current research analyzed, modified and adopted for this study. The detailed reviewed literatures of this study are presented under chapter two of this study

1.5.2. System Design and Data Preprocessing

Data processing, a critical initial step in data mining work, is often used to improve the quality of training data set. To do so data cleaning and preparation is the core task of data

mining which is dependent on software chosen and algorithms used (Mahbod et al., 2009). The IDS models in this study are developed on full training Network Simulation Language-Knowledge discovery in Database (NSL-KDD) dataset using a powerful machine learning and data mining WEKA tool.

The data mining model that used in this study is the KDD process. The KDD process refers to the whole process of changing low level data into high level knowledge which is automated discovery of patterns and relationships in large databases and data mining is one of the core steps in the KDD process. The goal KDD and DM is to find interesting patterns and/or models that exist in databases but are hidden among the volumes of data (Fayyad et al., 1996). The KDD process as described by Fayyad et al (1996) consists of five major phases. Data were collected then using appropriate algorithms then mined patterns will be modeled. Figure 1.1 showed the KDD process model that used in this thesis.

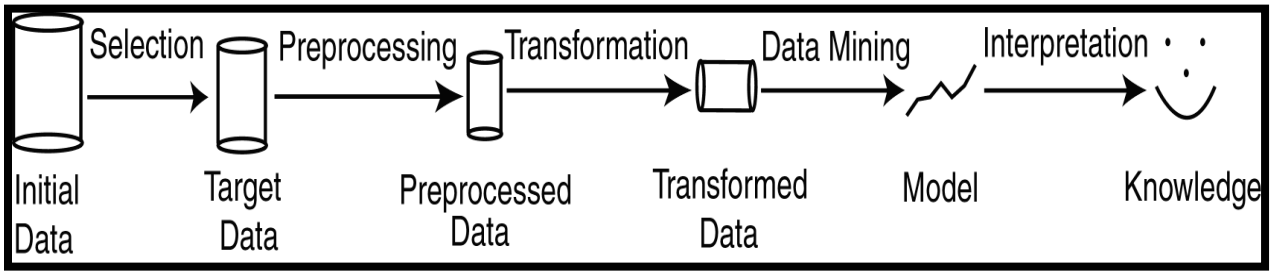

Figure 1.1.: An overview of the steps that compose the KDD process (Fayyad et al., 1996).

1.5.2.1. Initial Data Selection

NSL_KDD dataset (Helali, 2010) most widely used and publicly available for IDS is used for the experiment purpose. The KDD (Knowledge discovery in Database) Cup 1999 Intrusion detection contest data (KDD cup 99 Intrusion detection data sets) has been used in this study. This data was prepared by the 1998 DARPA (Defense Advanced research Project Agency) Intrusion Detection Evaluation program by MIT Lincoln Labs (MIT Lincoln Laboratory). .

1.5.2.2. Data Preprocessing

The data preprocessing step in this study includes basic operations, such as removing noise or outliers if appropriate, collecting the necessary information to model or account for

noise, deciding on strategies for handling missing data fields, and accounting for time sequence information and known changes, as well as deciding database management system issues, such as data types, schema, and mapping of missing and unknown values. Also, since a predictor can exploit only certain data features, it is important to detect which data preprocessing works best (Meera et al., 2003). For this study preprocessing of NSL-KDD dataset contains the following processes:

- ✓ Assigning attack names to one of the five classes Normal, Probe, DOS (Denial of Service), U2R (User to Root) and R2L (Remote to Local). To identify and label each attack different literatures are consulted and Microsoft Excel helps to filter and name easily using fill handle.
- ✓ There are records which don't have attributes and removed from the dataset and there is also a mismatch in the KDD 99 winner cost matrix and the confusion matrix; as a result arrangements are made to match the cost matrix and confusion matrix.
- ✓ The NSL-KDD dataset is available in text format; so to be read for Waikato Environment for Knowledge Analysis (WEKA) tool it has to be changed into ARFF format. For WEKA Data can be imported from a file in various formats: CSV, C4.5, binary (Chang et al., 2005).

1.5.2.3. Data Transformation

The data transformation step includes finding useful features to represent the data, depending on the goal of the task, and using dimensionality reduction or transformation methods to reduce the effective number of variables under consideration.

1.5.2.4. Choosing Data mining tasks

In this step the DM methods used for the thesis are decided. DM methods have been successfully applied for solving classification problems in many applications (Pradeep, 2005). In DM, algorithms (learners) try to automatically filter the knowledge from example data (datasets).This knowledge can be used to make predictions about original data in the future and to provide insight into the nature of the target concept(s). According to Pradeep (2005) the example data typically consists of a number of input patterns or examples to be

learned. DM systems typically attempt to discover regularities and relationships between features and classes in learning or training phase.

For analyzing the data and classification of network attacks from a network environment, the three machine learning algorithms (Eibe and Witten, 2005), the J48 decision tree classifier, Naïve Bayes Classifier and simple k-means clustering are used in this thesis.

1.5.2.4.1. Decision Tree

Decision tree is a predictive modeling technique most often used for classification in DM. The Classification algorithm is inductively learned to construct a model from the pre-classified dataset. Each data item is defined by values of the attributes. Classification may be viewed as mapping from a set of attributes to a particular class. The Decision tree classifies the given data item using the values of its attributes. The decision tree is initially constructed from a set of pre-classified data. The main approach is to select the attributes, which best divides the data items into their classes (Kruegel and Toth, 2003).

In this study the J48 decision tree algorithms used. It is an implementation of the C4.5 decision tree learner. This implementation produces decision tree models. It recursively splits a dataset according to tests on attribute values in order to separate the possible predictions. A decision-tree model is built by analyzing the training data and the model is used to classify the trained data.

The node of the J48 decision trees evaluates the existence and the significance of every individual feature. Considering a set A of case objects, J48 initially grows a tree and uses divide-and-conquer algorithm as follows: (i) if all the cases in A belong to the same class or if the set is a small one, the tree is leaf labeled with the most frequent occurring class in A. (ii) or, a test is selected based on a single attribute with two or more outcomes. This test is made the root of the tree with each branch as one outcome of the test. Further the same procedure is applied recursively for each subset.

1.5.2.4.2. Naive Bayes

The other supervised approach used in this thesis is the Naïve Bayes classifier which is based on probabilistic model for assigning the most likely class to given instance. Probabilistic model (approach) in classification field allows (model or looks for) the estimation of conditional probability of classes given instance, p(C/A1…, AN) where C∈ {C1…CM} the classes and Ai, i=1...N, a set of features describing dataset examples (Shekhar et al., 2007)). Given a valued example, the most appropriate class to be assigned to is the class with the upper a posterior probability,

Argmaxc p(C=c/A1=a1…, AN=aN)……. (1)

Bayesian approach splits a posterior distribution into a priori distribution and likelihood,

Argmaxc p(C=c/A1=a1…, AN=aN) = Argmaxc α p (A1=a1…
AN=aN/C=c) p(C=c)…….. (2)

Where α is normalization factor to ensure that sums of conditional probabilities over class labels are equal to 1. The distribution of features given class label is more complex to estimate. Its estimation is exponential in attribute number and requires a complete training dataset with sufficient examples for each class. Such problem can be avoided, assumed the independence of features of given class, and likelihood estimation uses the following formula.

P (A1=a1… AN=aN /C=c) =Πi p (Ai=ai /C=c)…… (3)

Depending on the precise nature of the probability model, Naive Bayes classifiers can be trained very efficiently in a supervised learning mode for this study.

1.5.2.4.3. K-means Clustering

In this study for semi-supervised modeling, the researcher used k-means clustering. The k-means clustering algorithm is used for clustering those unlabeled records into their appropriate classes. After clustering a classification techniques are applied. In K-means

clustering, assignment of the data points to clusters is depending upon the distance between cluster centroids.

1.5.2.5. Interpretation/evaluation

It is the final step in the selected KDD process model for this thesis. It includes two basic subcomponents: (a) interpretation of mined patterns (potentially leading to a repeat of earlier steps), and (b) consolidating discovered knowledge, which can included incorporating the knowledge in a performance system.

1.5.3. Architecture of the study

Supervised intrusion detection approaches use only labeled data for training. To label the data however are often difficult, expensive, or time consuming as they require the efforts of experienced domain experts. Semi-supervised learning addresses this problem by using large amount of unlabeled data, together with the labeled data, to build better classifiers. Semi-supervised learning requires less human effort.

The architecture used for this thesis showed in figure 1.2. This architecture proposed by Pachghare et al. (2011) for the semi-supervised approach for intrusion detection system. As showed in figure 1.2, labeled data used for training the system as supervised approach. After training, the system test using unlabeled data. The tested data will add to the training data so as to implement semi- supervised approach.

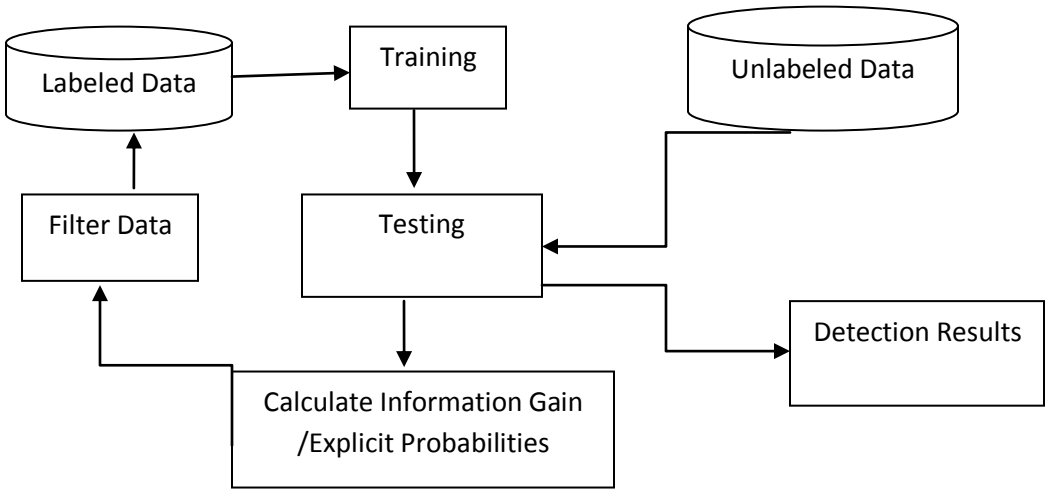

Figure 1.2: Architecture proposed for Semi-supervised IDS (Pachghare et al., 2011)

1.5.4. Implementation tool

The research has been conducted in preparing the data according to the requirements of the Tanagra (Eric, 2011) and WEKA data mining tools which are powerful, user friendly and freely available for noncommercial purpose. Tcptrace utility software used as the packet filtering tool in order to extract information about packets from TCP connections and to construct new features (Srinivasu et al., 2009).

1.5.4. Testing Procedure

The selected data has been preprocessed and using the data mining software package the network intrusion detection model has been constructed. Finally the constructed model validated by feeding real life records into the data mining experimenter package.

1.6. Significance of the study

Network security is an increasing industry as more and more of the corporate workspace is converted to digital media. Because companies and home users keep sensitive information on their computers, there is a great need to protect that information from those who would exploit it. One way to help keep attackers is by using IDS, which are designed to locate and notify systems administrators about the presence of malicious traffic (Sterry, 2004).

Detecting intrusions allows administrators to identify areas where their defenses need improvement, such as by identifying a previously unknown vulnerability, a system that was not properly patched, or a user that needs further education against social engineering attacks (Sterry, 2004). The research is applicable where; a continuous monitoring of many parameters is conducted to detect malicious applications.

This research will enhance the effectiveness and efficiency of Network Intrusion Detection System by proposing efficient feature selection and model constructing techniques.

This research will contribute its part for the country, Ethiopia. Because installation and securing the network infrastructure in Ethiopia is not well developed. In addition to this, the finding of this research will add knowledge for those professionals who want to advance their area of study in network security and IDS.

1.7. Organization of the Thesis

This thesis is structured into six chapters. The first chapter discusses background to the study problem area, statement of the problem, objective of the study, scope of the study, research methodology and application of the study.

The second chapter discusses about DM and knowledge discovery technology, DM Processes, DM models, application of DM in general and in particular in the area of intrusion detection and Intrusion detection and prevention principles.

The third chapter deals with understanding the data and dataset preparation, Data transformation and feature selection and evaluation metrics.

The fourth chapter provides a comprehensive discussion about the experimentation part of this thesis. This includes the supervised and semi-supervised approach.

The last chapter presents the conclusions and recommendations.

CHAPTER TWO

LITERATURE REVIEW

The researcher has reviewed different related literatures (books, journal articles, Conference proceeding papers, and the Internet) in order to have detailed understanding on the present research.

2.1. Data Mining and Knowledge Discovery

According to Fayyad et al (1996) the traditional method of turning data into knowledge relies on manual analysis and interpretation. For example, in the health-care industry, it is common for specialists to analyze current trends and changes in health-care data on a quarterly basis. The specialists then provide a report detailing the analysis to the sponsoring health-care organization; the report is then used as the basis for future decision making and planning for health-care management. For these (and many other) applications, such manual probing of a dataset is slow, expensive, and highly subjective. When the scale of data manipulation, exploration, and inference grows beyond human capacities, people look to computer technology to automate the bookkeeping (Nanda, 2010). The problem of knowledge extraction from large databases involves many steps, ranging from data manipulation and retrieval to fundamental mathematical and statistical inference, search, and reasoning. Researchers and practitioners interested in these problems have been meeting since the first KDD Workshop in 1989 (Fayyad et al., 1996). To bridge the gap of analyzing large volume of data and extracting valuable information and knowledge for decision making using new computerization technologies, DM and KDD has emerged since recent years.

Different scholars have provided different definitions for DM. The term “DM” is used most by statisticians, database researchers, and more recently by the management Information System and business communities (Fayyad et al., 1996). According to Nanda (2010), DM is a logical process that is used to search through large amounts of information in order to find important data. The goal of this technique is to find patterns that were previously

unknown. Once you have found these patterns, you can use them to solve a number of problems. DM refers to extracting or "mining" knowledge from large amount of data (Devedzic, 2000). According to Devedzic, DM is a process of discovering interesting knowledge from large amounts of data stored either, in database, data warehouse, or other information repositories. Devedzic has provided other alternative names for data mining: Knowledge mining, Knowledge extraction, Data/pattern analysis, Data Archaeology, Data dredging.

Different scholars have given their own definition for KDD. Fayyad et al (1996) used the term "KDD" to refer to the overall process of discovering useful knowledge from data and DM is a particular step in this process—application of specific algorithms for extracting patterns (models) from data. The additional steps in the KDD process, such as data preparation, data selection, data cleaning, incorporation of appropriate prior knowledge, and proper interpretation of the results of mining ensure that useful knowledge is derived from the data. According to Fayyad et al (1996) KDD has evolved, and continues to evolve, from the intersection of research in such fields as databases, machine learning, pattern recognition, statistics, artificial intelligence and reasoning with uncertainty, knowledge acquisition for expert systems, data visualization, machine discovery, scientific discovery, information retrieval, and high-performance computing. KDD software systems incorporate theories, algorithms, and methods from all of these fields. According to Nanda (2010) DM (sometimes called data or knowledge discovery in database) is the process of analyzing data from different perspectives and summarizing it into useful information - information that can be used to increase revenue, cuts costs, or both. Nanda (2010) used KDD and DM interchangeably. Data mining is a powerful tool because it can provide you with relevant information that you can use to your own advantage. When you have the right knowledge, all you will need to do is apply it in the right manner, and you will be able to benefit. It is relatively easy to get information these days. But it is not so easy to get relevant information that can help you achieve a desired goal (Nanda, 2010).

Although, the two terms KDD and DM are heavily used interchangeably, they refer to two related yet slightly different concepts (Roshan, 2011). KDD is the overall process of

extracting knowledge from data while Data Mining is a step inside the KDD process, which deals with identifying patterns in data. In other words, Data Mining is only the application of a specific algorithm based on the overall goal of the KDD process.

Data mining techniques must be fully integrated with a data warehouse as well as flexible interactive business analysis tools (Cheng, 2000). The raw data from the warehouse is preprocessed to remove noise and also to impart domain knowledge on the data. The knowledge discovery stage then extracts the knowledge which must then be post processed to facilitate human understanding. Post processing usually takes the form of representing the discovered knowledge in a user friendly display. Figure 2.1 illustrates architecture for advanced analysis in a large data warehouse.

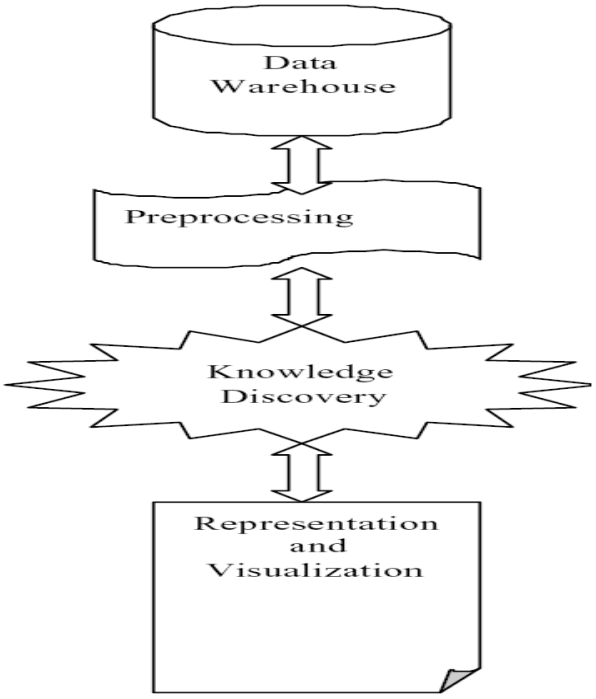

Fig 2.1: Knowledge Discovery in Databases: an Information Retrieval Perspective (Cheng, 2000).

2.2. Data Mining Models

Authors tend to use the terms process model, life cycle and methodology to refer to the same thing. This has led to some confusion in the field of DM (Adem and Julio, 2009).

- ✓ A process model is the set of tasks to be performed to develop a particular element, as well as the elements that are produced in each task (outputs) and the elements that are necessary to do a task (inputs) (Cheng, 2000). The goal of a process model is to make the process repeatable, manageable and measurable (to be able to get metrics).
- ✓ Methodology can be defined as the instance of a process model that lists tasks, inputs and outputs and specifies how to do the tasks (Cheng, 2000). Tasks are performed using techniques that stipulate how they should be done. After selecting a technique to do the specified tasks, tools can be used to improve task performance.
- ✓ Finally, the life cycle determines the order in which each activity is to be done. A life cycle model is the description of the different ways of developing a project.

2.2.1. The KDD Process Model

Fayyad et al (1996) presented their (necessarily subjective) perspective of a unifying process centric framework for KDD. The goal is to provide an overview of the variety of activities in this multidisciplinary field and how they fit together. KDD Process defined by Fayyad et al (1996) as: The nontrivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data. The term pattern goes beyond its traditional sense to include models or structure in data. In this definition, data comprises a set of facts (e.g., cases in a database), and pattern is an expression in some language describing a subset of the data (or a model applicable to that subset). The term process implies there are many steps involving data preparation; search for patterns, knowledge evaluation, and refinement all repeated in multiple iterations. The process is assumed to be nontrivial in that it goes beyond computing closed-form quantities; that is, it must involve search for structure, models, patterns, or parameters. The discovered patterns should be valid for new data with some degree of certainty. Fayyad et al (1996) also want patterns to be novel (at least to the system and preferably to the user) and potentially useful for the user or task. Finally, the patterns should be understandable if not immediately, then after some post processing.

As showed in figure 2.2 the DM process consists of five steps (Fayyad et al., 1996):

- ✓ **data selection** – having two subcomponents: (a) developing an understanding of the application domain and (b) creating a target dataset from the universe of available data;
- ✓ **preprocessing** – including data cleaning (such as dealing with missing data or errors) and deciding on methods for modeling information, accounting for noise, or dealing with change over time;
- ✓ **transformation** – using methods such as dimensionality reduction to reduce data complexity by reducing the effective number of variables under consideration;
- ✓ **data mining** – having three subcomponents: (a) choosing the data mining task (e.g., classification, clustering, summarization), (b) choosing the algorithms to be used in searching for patterns, (c) and the actual search for patterns (applying the algorithms);
- ✓ **interpretation/evaluation** – having two subcomponents: (a) interpretation of mined patterns (potentially leading to a repeat of earlier steps), and (b) consolidating discovered knowledge, which can include summarization and reporting as well as incorporating the knowledge in a performance system.

Fayyad et al (1996) emphasize both the interactive and iterative nature of KDD, that humans make many decisions and that the various steps and methods within them are repeated frequently as knowledge is being refined – thus our contention above that KDD is really about knowledge construction rather than discovery.

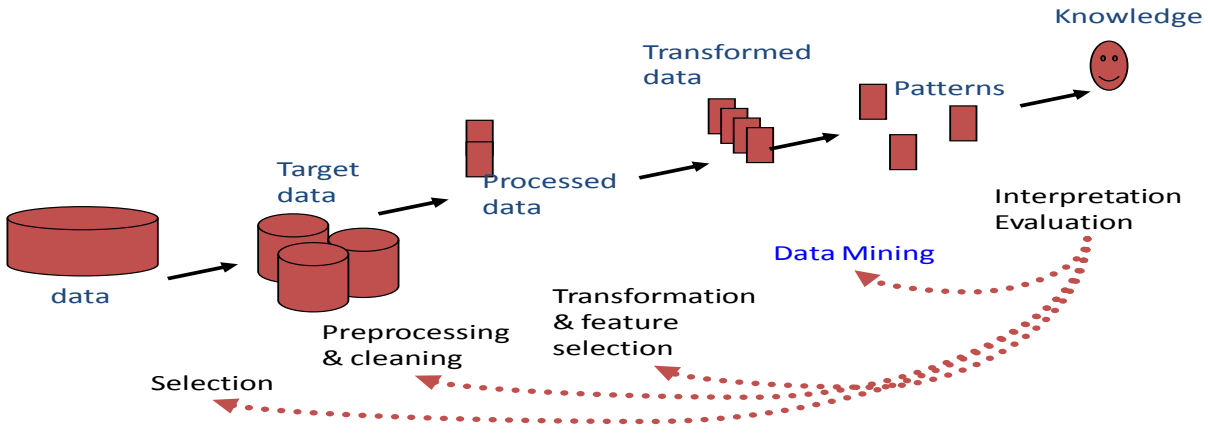

Figure 2.2: KDD process: "From Knowledge Discovery to Data Mining" (Fayyad et al., 1996).

Data mining is a step in the KDD process consisting of an enumeration of patterns (or models) over the data, subject to some acceptable computational-efficiency limitations. Since the patterns enumerable over any finite dataset are potentially infinite, and because the enumeration of patterns involves some form of search in a large space, computational constraints place severe limits on the subspace that can be explored by a data mining algorithm(Fayyad et al., 1996).

2.2.2. The CRISP- DM Process

Cross Industry Standard Process for Data Mining (CRISP-DM) is the most used methodology for developing DM projects (KdNuggets, 2012). Analyzing the problems of DM and KD projects, a group of prominent enterprises (Teradata, SPSS – ISL, Daimler-Chrysler and OHRA) developing DM projects, proposed a reference guide to develop DM and KD projects. This guide is called CRISP-DM (Chapman et al., 2003). CRISP-DM is vendor-independent so it can be used with any DM tool and it can be applied to solve any DM problem. CRISP-DM defines the phases to be carried out in a DM project. CRISP-DM also defines for each phase the tasks and the deliverables for each task. CRISP-DM is divided into six phases (see Figure 2.3).

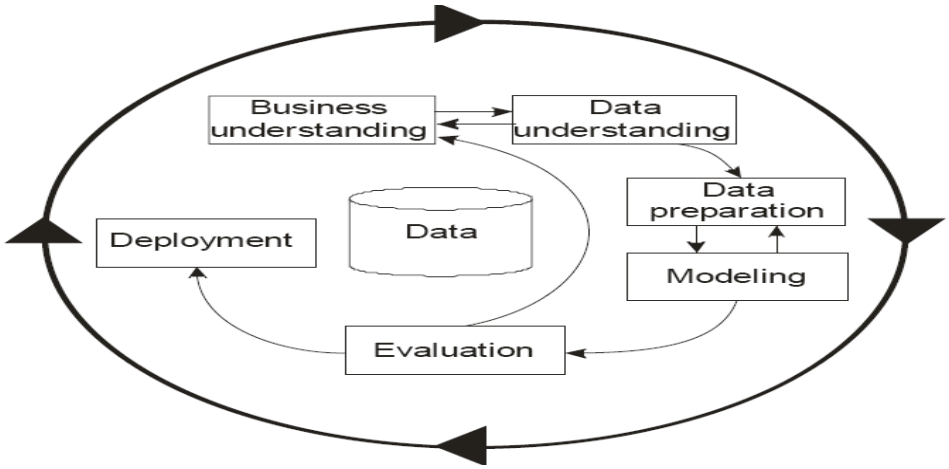

Figure 2.3: CRISP-DM process model (Chapman et al., 2003)

The detailed description of each phases of CRISP-DM model showed in table 2.1.

Business understanding	Data understanding	Data preparation	Modeling	Evaluation	Deployment
Determine business objectives	Collect initial data	Select data	Select modeling techniques	Evaluate results	Plan deployment
Assess situation	Describe data	Clean data	Generate test design	Review process	Plan monitoring & maintenance
Determine DM objectives	Explore data	Construct data	Build model	Determine next steps	Produce final report
Produce project plan	Verify data quality	Integrate data	Assess model		Review project
		Format data			

Table 2.1: CRISP-DM phases and tasks

- ✓ **Business understanding**: This phase focuses on understanding the project objectives and requirements from a business perspective, then converting this knowledge into a DM problem definition and a preliminary plan designed to achieve the objectives.
- ✓ **Data understanding**: The data understanding phase starts with an initial data collection and proceeds with activities in order to get familiar with the data, to identify data quality problems, to discover first insights into the data or to detect interesting subsets to form hypotheses for hidden information.
- ✓ **Data preparation**: The data preparation phase covers all the activities required to construct the final dataset from the initial raw data. Data preparation tasks are likely to be performed repeatedly and not in any prescribed order.
- ✓ **Modeling**: In this phase, various modeling techniques are selected and applied and their parameters are calibrated to optimal values. Typically, there are several techniques for the same DM problem type. Some techniques have specific requirements on the form of data. Therefore, it is often necessary to step back to the data preparation phase.
- ✓ **Evaluation**: What are, from a data analysis perspective, seemingly high quality models will have been built by this stage. Before proceeding to final model deployment, it is important to evaluate the model more thoroughly and review the steps taken to build

it to be certain that it properly achieves the business objectives. At the end of this phase, a decision should be reached on how to use of the DM results.

- ✓ **Deployment**: Model construction is generally not the end of the project. Even if the purpose of the model is to increase knowledge of the data, the knowledge gained will need to be organized and presented in a way that the customer can use it.

2.2.3. Comparison of Different Data Mining Models

In the early 1990s, when the KDD process term was first coined (Piatetsky and Frawley, 1991), there was a rush to develop DM algorithms that were capable of solving all problems of searching for knowledge in data. The KDD process (Piatetsky, 1994; Fayyad et al., 1996) has a process model component because it establishes all the steps to be taken to develop a DM project, but it is not a methodology because its definition does not set out how to do each of the proposed tasks. It is also a life cycle. The 5 A's (Martínez , 2003) is a process model that proposes the tasks that should be performed to develop a DM project and was one of CRISP-DM forerunners. Therefore, they share the same philosophy: 5 A's proposes the tasks but does not suggest how they should be performed. Its life cycle is similar to the one proposed in CRISP-DM.

A people-focused DM proposal is presented in (Brachman and Anand, 1996): Human-Centered Approach to DM. This proposal describes the processes to be enacted to carry out a DM project, considering people's involvement in each process and taking into account that the target user is the data engineer. Sample, Explore, Modify, Model, and Assess (SEMMA) (SAS, 2012) is the methodology that SAS proposed for developing DM products. Although it is a methodology, it is based on the technical part of the project only. Like the above approaches, SEMMA also sets out a waterfall life cycle, as the project is developed right through to the end.

The two models by Cabena et al (1998) and Anand et al (1998) are based on KDD with few changes and have similar features. Like the KDD process, Two Crows (Two Crows, 1999) is a process model and waterfall life cycle. At no point does it set out how to do the established DM project development tasks. CRISP-DM states which tasks have to be carried

out to successfully complete a DM project (Chapman et al., 2003). It is therefore a process model. It is also a waterfall life cycle. CRISP-DM also has a methodological component, as it gives recommendations on how to do some tasks. Even so these recommendations are confined to proposing other tasks and give no guidance about how to do them. Figure 2.4 shows a diagram of how the different DM and KD process models and methodologies have evolved.

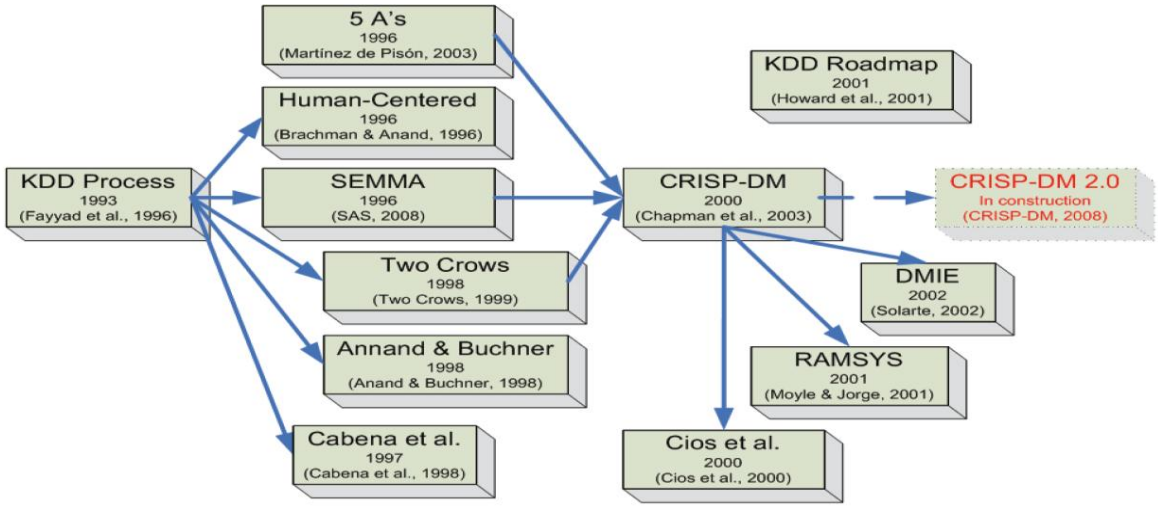

Figure 2.4: Evolution of DM and KD process models and methodologies (Adem and Julio, 2009)

It is clear from Figure 2.4 that CRISP-DM is the standard model. It borrowed ideas from the most important pre-2000 models and is the groundwork for many later proposals. The CRISP-DM 2.0 Special Interest Group (SIG) was set up with the aim of upgrading the CRISP-DM model to a new version better suited to the changes that have taken place in the business arena since the current version was formulated. This group is working on the new methodology (CRISP-DM, 2012). The firms that developed CRISP-DM 1.0 have been joined by other institutions that intend to input their expertise in the field to develop CRISP-DM 2.0. Changes such as adding new phases, renaming existing phases and/or eliminating the odd phase are being considered for the new version of the methodology.

Cios and Kurgan (2000) model was first proposed in 2000. This model adapted the CRISP-DM model to the needs of the academic research community, providing a more general, research-oriented description of the steps.

The KDD Roadmap (Howard et al., 2001) is a DM methodology used in the DM Witness Miner tool (Lanner, 2008). This methodology describes the available processes and algorithms and incorporates experience derived from successfully completed commercial projects. The focus is on the decisions to be made and the options available at each stage to achieve the best results for a given task.

The RAMSYS (Rapid collaborative data Mining System) methodology is described in (Howard et al., 2001) as a methodology for developing DM and KD projects where several geographically diverse groups work remotely and collaboratively to solve the same problem. This methodology is based on CRISP-DM and maintains the same phases and generic tasks.

DMIE or Data Mining for Industrial Engineering (Solarte, 2002) is a methodology because it specifies how to do the tasks to develop a DM project in the field of industrial engineering. It is an instance of CRISP-DM, which makes it a methodology, and it shares CRISP-DM's associated life cycle.

In summary different DM models have different steps for conducting a given study. Table 2.2 compares the phases into which the DM and KD process is decomposed according some of the above scheme. As showed in table 2.2, most of the scheme cover all the tasks in CRISP-DM, although they do not all decompose the KDD process into the same phases or attaches the same importance to the same tasks. However, some steps described above are omitted the study by (Yang and Wu, 2006), like 5 A's and DMIE, propose additional phases not covered by CRISP-DM that are potentially very useful in KD and DM projects. 5 A's proposes the "Automate" phase. This phase entails more than just using the model. It focuses on generating a tool to help non-experts in the area to perform DM and KD tasks. On the other hand, DMIE proposes the "On-going support" phase. It is very important to take this phase into account, as DM and KD projects require a support and maintenance phase. This maintenance ranges from creating and maintaining backups of the data used in the project to the regular reconstruction of DM models. The reason is that the behavior of the DM models may change as new data emerge, and they may not work properly.

Similarly, if other tools have been used to implement the DM models, the created programs may need maintenance, e.g. to upgrade the behavior of the user application models.

Model	Fayyad et al.	Cabena et al.	Anand & Buchner	CRISP-DM	Cios et al.
No of steps	9	5	8	6	6
Steps	Developing and Understanding of the Application Domain	Business Objectives Determination	Human Resource Identification	Business Understanding	Understanding the Data
			Problem Specification		
	Creating a Target Data Set	Data Preparation	Data Prospecting	Data Understanding	Understanding the Data
	Data Cleaning and Pre-processing		Domain Knowledge Elicitation		
	Data Reduction and Projection		Methodology Identification	Data Preparation	Preparation of the data
	Choosing the DM Task		Data Pre-processing		
	Choosing the DM Algorithm				
	DM	DM	Pattern Discovery	Modeling	DM
	Interpreting Mined Patterns	Domain Knowledge Elicitation	Knowledge Post-processing	Evaluation	Evaluation of the Discovered Knowledge
	Consolidating Discovered Knowledge	Assimilation of Knowledge		Deployment	Using the Discovered Knowledge

Table 2.2: Comparison of DM and KD process models and methodologies (Yang and Wu, 2006)

2.3. Data Mining Tasks

The tasks of data mining can be modeled as either Predictive or Descriptive in nature (Dunham, 2003). A Predictive model makes a prediction about values of data using known results found from different data while the Descriptive model identifies patterns or relationships in data. Unlike the predictive model, a descriptive model serves as a way to explore the properties of the data examined, not to predict new properties. Predictive model data mining tasks include classification, prediction, and regression. The Descriptive task encompasses methods such as Clustering, Association Rules, and Sequence analysis.

2.3.1. Descriptive Model

Descriptive data mining is normally used to generate frequency, cross tabulation and correlation. Descriptive method can be defined to discover interesting regularities in the data, to uncover patterns and find interesting subgroups in the bulk of data (Han et al., 2000). In education, studies Thair (2009) used Descriptive to determine the demographic influence on particular factors. Summarization maps data into subsets with associated simple descriptions (Dunham, 2003). Basic statistics such as Mean, Standard Deviation, Variance, Mode and Median can be used as Summarization approach.

2.3.1.1. Clustering

Clustering is a data mining technique where data points are clustered together based on their feature values and a similarity metric. Frank (1994) breaks clustering techniques into five areas: hierarchical, statistical, exemplar, distance, and conceptual clustering, each of which has different ways of determining cluster membership and representation. Berkhin (2002) presented an excellent survey of specific methods for techniques in most of these areas in.

In clustering, a set of data items is partitioned into a set of classes such that items with similar characteristics are grouped together. Clustering is best used for finding groups of items that are similar. For example, given a data set of customers, subgroups of customers that have a similar buying behavior can be identified. Clustering is a unsupervised learning process. Clustering is the process of grouping a set of physical or abstract objects into classes of similar objects (Han et al., 2000), so that objects within the same cluster must be similar to some extent, also they should be dissimilar to those objects in other clusters. In classification which record belongs which class is predefined, while in clustering there is no predefined classes. In clustering, objects are grouped together based on their similarities. Similarities between objects are defined by similarity functions, usually similarities are quantitatively specified as distance or other measures by corresponding domain experts.

Most clustering applications are used in market segmentation. By clustering their customers into different groups, business organizations can provide different personalized services to different group of markets. For example, based on the expense, deposit and

draw patterns of the customers, a bank can clustering the market into different groups of people. For different groups of market, the bank can provide different kinds of loans for houses or cars with different budget plans. In this case the bank can provide a better service, and also make sure that all the loans can be reclaimed. A comprehensive survey of current clustering techniques and algorithms is available in (Berkhin, 2002).

Clustering provides some significant advantages over the classification techniques, in that it does not require the use of a labeled data set for training. For example, Eskin et al (2002) have applied fixed-width and k-nearest neighbor clustering techniques to connection logs looking for outliers, which represent anomalies in the network traffic. Marin et al (2001) also use a similar approach utilizing learning vector quantization (LVQ), which is designed to find the Bayes Optimal boundary between classes, using k-means clustering to determine initial vector positioning. Unfortunately, this approach will overlook intensive malicious traffic, such as heavy probing and denial of service attacks, which should form their own clusters

K-means clustering: is a data mining/machine learning algorithm used to cluster observations into groups of related observations without any prior knowledge of those relationships (Berkhin, 2002). The k-means algorithm is one of the simplest clustering techniques and it is commonly used in medical imaging, biometrics and related fields. The k-means algorithm is an evolutionary algorithm that gains its name from its method of operation. The algorithm clusters observations into k groups, where k is provided as an input parameter. It then assigns each observation to clusters based upon the observation's proximity to the mean of the cluster. The cluster's mean is then recomputed and the process begins again. Here's how the algorithm works:

I. The algorithm arbitrarily selects k points as the initial cluster centers ("means").
II. Each point in the dataset is assigned to the closed cluster, based upon the Euclidean distance between each point and each cluster center.
III. Each cluster center is recomputed as the average of the points in that cluster.
IV. Steps II and III repeat until the clusters converge. Convergence may be defined differently depending upon the implementation, but it normally means that either

no observations change clusters when steps II and III are repeated or that the changes do not make a material difference in the definition of the clusters.

2.3.1.2. Association Rules

Associations or Link Analysis are used to discover relationships between attributes and items such as the presence of one pattern implies the presence of another pattern. i.e. to what extent one item is related to another in terms of cause-and-effect. This is common in establishing a form of statistical relationships among different interdependent variables of a model. These relations may be associations between attributes within the same data item like ('Out of the shoppers who bought milk, 64% also purchased bread') or associations between different data items like ('Every time a certain stock drops 5%, it causes a resultant 13% in another stock between 2 and 6 weeks later').

Association Rules is a popular technique for market basket analysis because all possible combinations of potentially interesting product groupings can be explored (Roiger et al., 2003).

2.3.1.3. Sequence Analysis

The investigation of relationships between items over a period of time is also often referred to as Sequence Analysis (Han et al., 2000). Sequence Analysis is used to determine sequential patterns in data (Dunham, 2003). The patterns in the dataset are based on time sequence of actions, and they are similar to association data, however the relationship is based on time. In Market Basket analysis, the items are to be purchased at the same time, on the other hand, for Sequence Analysis the items are purchased over time in some order.

2.3.2. Predictive Model

The goal of the predictive models is to construct a model by using the results of the known data and is to predict the results of unknown data sets by using the constructed model. For instance a bank might have the necessary data about the loans given in the previous terms. In this data, independent variables are the characteristics of the loan granted clients and the dependent variable is whether the loan is paid back or not (Berson et al., 2000). The

model constructed by this data is used in the prediction of whether the loan will be paid back by client in the next loan applications.

2.3.2.1. Classification and regression

Classification and regression are two data analyzing methods which determine important data classes or may construct models which can predict future data trends. The classification model predicts the categorical values; the regression is used in the prediction of values showing continuity. For instance while the classification model is constructed to categorize whether the bank loan applications are safe or risky, the regression model may be constructed to predict the spending of clients buying computer products whose income and occupation are given (Chaudhuri, 1998; Berson et al., 2000).

In the classification models the following techniques are mainly used (Berson et al., 2000): Decision Trees, Artificial Neural Networks and Navie-Bayes.

2.3.2.1.1 Decision Trees

Decision trees technique is commonly used in data mining because their construction is cheap, their interpretation is easy, their easy integration with database systems and their good reliability (Berson et al., 2000).

Decision trees are trees that classify instances by sorting them based on feature values (Thair, 2009). Each node in a decision tree represents a feature in an instance to be classified, and each branch represents a value that the node can assume. Instances are classified starting at the root node and sorted based on their feature values

2.3.2.1.2. Neural networks

Artificial neural network models have been studied for many years in the hope of achieving human-like performance in several fields such as speech and image understanding (Thair, 2009). The networks are composed of many nonlinear computational elements operating in parallel and arranged in patterns reminiscent of biological neural networks. Computational elements or nodes are connected in several layers (input, hidden and output) via weights that are typically adapted during the training phase to achieve high

performance. Instead of performing a set of instructions sequentially as in a Von Neumann computer, neural network models explore simultaneously many hypotheses using parallel networks composed of many computational elements connected by links with variable weights.

The back-propagation algorithm is an extension of the least mean square (LMS) algorithm that can be used to train multi-layer networks. Both LMS and back-propagation are approximate steepest descent algorithms that minimize squared error. The only difference between them is in the way in which the gradient is calculated. The back-propagation algorithm uses the chain rule in order to compute the derivatives of the squared error with respect to the weights and biases in the hidden layers. It is called back-propagation because the derivatives are computed first at the last layer of the network, and then propagated backward through the network, using the chain rule, to compute the derivatives in the hidden layers. For a multi-layer network, the output of one layer becomes the input of the following layer.

2.3.2.1.3. Bayesian Network (BN)

A BN is a graphical model for probability relationships among a set of variables features as shown in figure 2.5. The BN structure S is a directed acyclic graph (DAG) and the nodes in S are in one-to-one correspondence with the features X. The arcs represent casual influences among the features while the lack of possible arcs in S encodes conditional independencies. Moreover, a feature (node) is conditionally independent from its non-descendants given its parents (X1 is conditionally independent from X2 given X3 if P (X1|X2, X3) =P (X1|X3) for all possible values of X1, X2, X3)

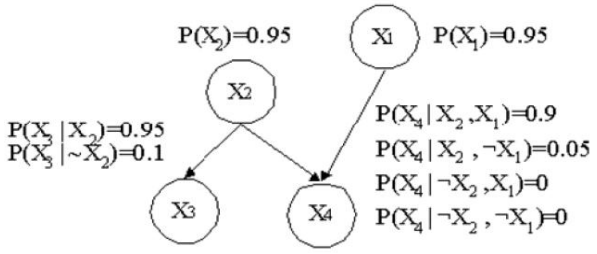

Figure 2.5: The structure of Bayes network (Thair, 2009)

Typically, the task of learning a BN can be divided into two subtasks: initially, the learning of the DAG structure of the network, and then the determination of its parameters. Probabilistic parameters are encoded into a set of tables, one for each variable, in the form of local conditional distributions of a variable given its parents. Given the independences encoded into the network, the joint distribution can be reconstructed by simply multiplying these tables. Within the general framework of inducing Bayesian networks, there are two scenarios: known structure and unknown structure.

In the first scenario, the structure of the network is given (e.g. by an expert) and assumed to be correct. Once the network structure is fixed, learning the parameters in the Conditional Probability Tables (CPT) is usually solved by estimating a locally exponential number of parameters from the data provided (Jensen, 1996). Each node in the network has an associated CPT that describes the conditional probability distribution of that node given the different values of its parents. In spite of the remarkable power of BNs, they have an inherent limitation. This is the computational difficulty of exploring a previously unknown network. Given a problem described by n features, the number of possible structure hypotheses is more than exponential in n. If the structure is unknown, one approach is to introduce a scoring function (or a score) that evaluates the "fitness" of networks with respect to the training data, and then to search for the best network according to this score. Several researchers have shown experimentally that the selection of a single good hypothesis using greedy search often yields accurate predictions (Cheng et al., 2002).

The most interesting feature of BNs, compared to decision trees or neural networks, is most certainly the possibility of taking into account prior information about a given problem, in terms of structural relationships among its features. This prior expertise, or domain knowledge, about the structure of a Bayesian network can take the following forms (Thair, 2009):

I. Declaring that a node is a root node, i.e., it has no parents.
II. Declaring that a node is a leaf node, i.e., it has no children.
III. Declaring that a node is a direct cause or direct effect of another node.

IV. Declaring that a node is not directly connected to another node.
V. Declaring that two nodes are independent, given a condition-set.
VI. Providing partial nodes ordering, that is, declare that node appears earlier than another node in the ordering.
VII. Providing a complete node ordering.

A problem of BN classifiers is that they are not suitable for datasets with many features (Cheng et al., 2002). The reason for this is that trying to construct a very large network is simply not feasible in terms of time and space. A final problem is that before the induction, the numerical features need to be discredited in most cases.

2.4. Application of Data Mining

Databases today can range in size into the terabytes — more than 1,000,000,000,000 bytes of data (Two Crows, 2005). Within these masses of data lies hidden information of strategic importance. But when there are so many trees, how do you draw meaningful conclusions about the forest? According to Two Crows (2005) the newest answer is data mining, which is being used both to increase revenues and to reduce costs. The potential returns are enormous. Innovative organizations worldwide are already using data mining to locate and appeal to higher-value customers, to reconfigure their product offerings to increase sales, and to minimize losses due to error or fraud.

Many organizations are using data mining to help manage all phases of the customer life cycle, including acquiring new customers, increasing revenue from existing customers, and retaining good customers (Two Crows, 2005). By determining characteristics of good customers (profiling), a company can target prospects with similar characteristics. By profiling customers who have bought a particular product it can focus attention on similar customers who have not bought that product (cross-selling). By profiling customers who have left, a company can act to retain customers who are at risk for leaving (reducing churn or attrition), because it is usually far less expensive to retain a customer than acquire a new one.

DM offers value across a broad spectrum of industries. Telecommunications and credit card companies are two of the leaders in applying data mining to detect fraudulent use of their

services (Nanda, 2010). Insurance companies and stock exchanges are also interested in applying this technology to reduce fraud. Medical applications are another fruitful area: data mining can be used to predict the effectiveness of surgical procedures, medical tests or medications. Companies active in the financial markets use data mining to determine market and industry characteristics as well as to predict individual company and stock performance. Retailers are making more use of data mining to decide which products to stock in particular stores (and even how to place them within a store), as well as to assess the effectiveness of promotions and coupons. Pharmaceutical firms are mining large databases of chemical compounds and of genetic material to discover substances that might be candidates for development as agents for the treatments of disease.

Due to the exponential growth of data, especially in areas such as business, KDD has become a very important process to convert this large wealth of data in to business intelligence, as manual extraction of patterns has become seemingly impossible in the past few decades (Roshan, 2011). For example, it is currently been used for various applications such as social network analysis, fraud detection, science, investment, manufacturing, telecommunications, data cleaning, sports, information retrieval and largely for marketing.

KDD is usually used to answer questions like what are the main products that might help to obtain high profit next year in Wal-Mart. This process has several steps. It starts with developing an understanding of the application domain and the goal and then creating a target dataset. This is followed by cleaning, preprocessing, reduction and projection of data. Next step is using Data Mining (explained below) to identify pattern. Finally, discovered knowledge is consolidates by visualizing and/or interpreting.

A large degree of the current interest in KDD is the result of the media interest surrounding successful KDD applications (Fayyad et al., 1996). Several well documented examples of successful systems can rightly be referred to as KDD applications and have been deployed in operational use on large-scale real-world problems in science and in business.

In science, one of the primary application areas is astronomy. Here, a notable success was achieved by SKICAT, a system used by astronomers to perform image analysis,

classification, and cataloging of sky objects from sky-survey images (Fayyad et al., 1996). In its first application, the system was used to process the 3 terabytes (10^12 bytes) of image data resulting from the Second Palomar Observatory Sky Survey, where it is estimated that on the order of 10^9 sky objects are detectable. SKICAT can outperform humans and traditional computational techniques in classifying faint sky objects (Fayyad et al., 1996).

As discussed by Fayyad et al (1996) in business, main KDD application areas includes marketing, finance (especially investment), fraud detection, and manufacturing, telecommunications, and Internet agents.

Marketing: In marketing, the primary application is database marketing systems, which analyze customer databases to identify different customer groups and forecast their behavior. Business Week (Berry, 1994) estimated that over half of all retailers are using or planning to use database marketing, and those who do use it have good results; for example, American Express reports a 10- to 15- percent increase in credit-card use. Another notable marketing application is market-basket analysis (Agrawal et al. 1996) systems, which find patterns such as, "If customer bought X, he/she is also likely to buy Y and Z." Such patterns are valuable to retailers.

Investment: Numerous companies use data mining for investment, but most do not describe their systems. One exception is LBS Capital Management. Its system uses expert systems, neural nets, and genetic algorithms to manage portfolios totaling $600 million; since its start in 1993, the system has outperformed the broad stock market (Agrawal et al. 1996).

Fraud detection: HNC Falcon and Nestor PRISM systems are used for monitoring credit card fraud, watching over millions of accounts. The FAIS system (Mannila et al., 1995), from the U.S. Treasury Financial Crimes Enforcement Network, is used to identify financial transactions that might indicate money laundering activity.

Manufacturing: The CASSIOPEE troubleshooting system, developed as part of a joint venture between General Electric and SNECMA, was applied by three major European airlines to diagnose and predict problems for the Boeing 737. To derive families of faults,

clustering methods are used. CASSIOPEE received the European first prize for innovative applications (Manago and Auriol, 1996).

Telecommunications: The telecommunications alarm-sequence analyzer (TASA) was built in cooperation with a manufacturer of telecommunications equipment and three telephone networks (Mannila et al., 1995). The system uses a novel framework for locating frequently occurring alarm episodes from the alarm stream and presenting them as rules. Large sets of discovered rules can be explored with flexible information- retrieval tools supporting interactivity and iteration. In this way, TASA offers pruning, grouping, and ordering tools to refine the results of a basic brute-force search for rules.

Intrusion Detections: Here are a few specific things that DM might contribute to an intrusion detection project (Eric et al., 2002):

- ✓ Remove normal activity from alarm data to allow analysts to focus on real attacks
- ✓ Identify false alarm generators and "bad" sensor signatures
- ✓ Find anomalous activity that uncovers a real attack
- ✓ Identify long, ongoing patterns (different IP address, same activity)

2.5. Intrusion detection and prevention principles

Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices (Karen and Peter, 2007). Incidents have many causes, such as malware (e.g., worms, spyware), attackers gaining unauthorized access to systems from the Internet, and authorized users of systems who misuse their privileges or attempt to gain additional privileges for which they are not authorized. Although many incidents are malicious in nature, many others are not; for example, a person might mistype the address of a computer and accidentally attempt to connect to a different system without authorization.

According to Karen and Peter (2007) an IDS is software that automates the intrusion detection process. An intrusion prevention system (IPS) is software that has all the capabilities of an intrusion detection system and can also attempt to stop possible incidents.

2.5.1. Uses of Intrusion Detection Prevention Systems (IDPSs) Technologies

As discussed by Carl (2003) IDPSs are primarily focused on identifying possible incidents. For example, an IDPS could detect when an attacker has successfully compromised a system by exploiting vulnerability in the system. The IDPS could then report the incident to security administrators, who could quickly initiate incident response actions to minimize the damage caused by the incident. The IDPS could also log information that could be used by the incident handlers.

As discussed by Karen and Peter (2007) many IDPSs can also be configured to recognize violations of security policies. For example, some IDPSs can be configured with firewall rule set like settings, allowing them to identify network traffic that violates the organization's security or acceptable use policies. Also, some IDPSs can monitor file transfers and identify ones that might be suspicious, such as copying a large database onto a user's laptop.

Many IDPSs can also identify reconnaissance activity, which may indicate that an attack is imminent (Christos and Aikaterini, 2004). For example, some attack tools and forms of malware, particularly worms, perform reconnaissance activities such as host and port scans to identify targets for subsequent attacks. An IDPS might be able to block reconnaissance and notify security administrators, who can take actions if needed to alter other security controls to prevent related incidents. Because reconnaissance activity is so frequent on the Internet, reconnaissance detection is often performed primarily on protected internal networks.

In addition to identifying incidents and supporting incident response efforts, organizations have found other uses for IDPSs, including the following (Karen and Peter, 2007):

- ✓ **Identifying security policy problems.** An IDPS can provide some degree of quality control for security policy implementation, such as duplicating firewall rule-sets and

alerting when it sees network traffic that should have been blocked by the firewall but was not because of a firewall configuration error.

- ✓ **Documenting the existing threat to an organization.** IDPSs log information about the threats that they detect. Understanding the frequency and characteristics of attacks against an organization's computing resources is helpful in identifying the appropriate security measures for protecting the resources. The information can also be used to educate management about the threats that the organization faces.
- ✓ **Deterring individuals from violating security policies.** If individuals are aware that their actions are being monitored by IDPS technologies for security policy violations, they may be less likely to commit such violations because of the risk of detection.

Because of the increasing dependence on information systems and the prevalence and potential impact of intrusions against those systems, IDPSs have become a necessary addition to the security infrastructure of nearly every organization.

2.5.2. Key Functions of IDPS Technologies

There are many types of IDPS technologies, which are differentiated primarily by the types of events that they can recognize and the methodologies that they use to identify incidents. In addition to monitoring and analyzing events to identify undesirable activity, all types of IDPS technologies typically perform the following functions (Crothers, 2002):

- ✓ **Recording information related to observed events.** Information is usually recorded locally, and might also be sent to separate systems such as centralized logging servers, security information and event management (SIEM) solutions, and enterprise management systems.
- ✓ **Notifying security administrators of important observed events.** This notification, known as an alert, occurs through any of several methods, including the following: e-mails, pages, messages on the IDPS user interface, Simple Network Management Protocol (SNMP) traps, and user-defined programs and scripts. A notification message typically includes only basic information regarding an event; administrators need to access the IDPS for additional information.

- ✓ **Producing reports.** Reports summarize the monitored events or provide details on particular events of interest.

Some IDPSs are also able to change their security profile when a new threat is detected. For example, an IDPS might be able to collect more detailed information for a particular session after malicious activity is detected within that session. An IDPS might also alter the settings for when certain alerts are triggered or what priority should be assigned to subsequent alerts after a particular threat is detected.

IPS technologies are differentiated from IDS technologies by one characteristic: IPS technologies can respond to a detected threat by attempting to prevent it from succeeding. They use several response techniques, which can be divided into the following groups (Karen and Peter, 2007):

- ✓ **The IPS stops the attack itself.** Examples of how this could be done are as follows:
  - ➢ Terminate the network connection or user session that is being used for the attack
  - ➢ Block access to the target (or possibly other likely targets) from the offending user account, IP address, or other attacker attribute
  - ➢ Block all access to the targeted host, service, application, or other resource.
- ✓ **The IPS changes the security environment.** The IPS could change the configuration of other security controls to disrupt an attack. Common examples are reconfiguring a network device (e.g., firewall, router, switch) to block access from the attacker or to the target, and altering a host-based firewall on a target to block incoming attacks. Some IPSs can even cause patches to be applied to a host if the IPS detects that the host has vulnerabilities.
- ✓ **The IPS changes the attack's content.** Some IPS technologies can remove or replace malicious portions of an attack to make it benign. A simple example is an IPS removing an infected file attachment from an e-mail and then permitting the cleaned email to reach its recipient. A more complex example is an IPS that acts as a proxy and normalizes incoming requests, which means that the proxy repackages the payloads of the requests, discarding header information. This might cause certain attacks to be discarded as part of the normalization process.

As discussed by Karen and Peter (2007) another common attribute of IDPS technologies is that they cannot provide completely accurate detection. When an IDPS incorrectly identifies benign activity as being malicious, a false positive has occurred. When an IDPS fails to identify malicious activity, a false negative has occurred. It is not possible to eliminate all false positives and negatives; in most cases, reducing the occurrences of one increases the occurrences of the other. Many organizations choose to decrease false negatives at the cost of increasing false positives, which means that more malicious events are detected but more analysis resources are needed to differentiate false positives from true malicious events. Altering the configuration of an IDPS to improve its detection accuracy is known as tuning.

Most IDPS technologies also offer features that compensate for the use of common evasion techniques (Crothers, 2002). Evasion is modifying the format or timing of malicious activity so that its appearance changes but its effect is the same. Attackers use evasion techniques to try to prevent IDPS technologies from detecting their attacks. For example, an attacker could encode text characters in a particular way, knowing that the target understands the encoding and hoping that any monitoring IDPSs do not. Most IDPS technologies can overcome common evasion techniques by duplicating special processing performed by the targets. If the IDPS can "see" the activity in the same way that the target would, then evasion techniques will generally be unsuccessful at hiding attacks.

2.5.3. Common Detection Methodologies

Most IDPSs use multiple detection methodologies, either separately or integrated, to provide more broad and accurate detection. The primary classes of detection methodologies are as follows (Karen and Peter, 2007):

- **Signature-based**, which compares known threat signatures to observed events to identify incidents. This is very effective at detecting known threats but largely ineffective at detecting unknown threats and many variants on known threats. Signature-based detection cannot track and understand the state of complex communications, so it cannot detect most attacks that comprise multiple events.

- **Anomaly-based detection**, which compares definitions of what activity, is considered normal against observed events to identify significant deviations. This method uses profiles that are developed by monitoring the characteristics of typical activity over a period of time. The IDPS then compares the characteristics of current activity to thresholds related to the profile. Anomaly-based detection methods can be very effective at detecting previously unknown threats. Common problems with anomaly-based detection are inadvertently including malicious activity within a profile, establishing profiles that are not sufficiently complex to reflect real-world computing activity, and generating many false positives.
- **Stateful protocol analysis**, which compares predetermined profiles of generally accepted definitions of benign protocol activity for each protocol state against observed events to identify deviations. Unlike anomaly-based detection, which uses host or network-specific profiles, Stateful protocol analysis relies on vendor-developed universal profiles that specify how particular protocols should and should not be used. It is capable of understanding and tracking the state of protocols that have a notion of state, which allows it to detect many attacks that other methods cannot. Problems with Stateful protocol analysis include that it is often very difficult or impossible to develop completely accurate models of protocols, it is very resource-intensive, and it cannot detect attacks that do not violate the characteristics of generally acceptable protocol behavior.

2.5.4. Types of IDPS Technologies

There are many types of IDPS technologies. For the purposes of this document, they are divided into the following four groups based on the type of events that they monitor and the ways in which they are deployed (Karen and Peter, 2007):

- ✓ **Network-Based**, which monitors network traffic for particular network segments or devices and analyzes the network and application protocol activity to identify suspicious activity. It can identify many different types of events of interest. It is most commonly deployed at a boundary between networks, such as in proximity to border firewalls or routers, virtual private network (VPN) servers, remote access servers, and wireless networks.

- ✓ **Wireless**, which monitors wireless network traffic and analyzes its wireless networking protocols to identify suspicious activity involving the protocols themselves. It cannot identify suspicious activity in the application or higher-layer network protocols (e.g., TCP, UDP) that the wireless network traffic is transferring. It is most commonly deployed within range of an organization's wireless network to monitor it, but can also be deployed to locations where unauthorized wireless networking could be occurring.
- ✓ **Network Behavior Analysis (NBA)**, which examines network traffic to identify threats that generate unusual traffic flows, such as distributed denial of service (DDoS) attacks, certain forms of malware (e.g., worms, backdoors), and policy violations (e.g., a client system providing network services to other systems). NBA systems are most often deployed to monitor flows on an organization's internal networks, and are also sometimes deployed where they can monitor flows between an organization's networks and external networks (e.g., the Internet, business partners' networks).
- ✓ **Host-Based,** which monitors the characteristics of a single host and the events occurring within that host for suspicious activity. Examples of the types of characteristics a host-based IDPS might monitor are network traffic (only for that host), system logs, running processes, application activity, file access and modification, and system and application configuration changes. Host-based IDPSs are the most commonly deployed on critical hosts such as publicly accessible servers and servers containing sensitive information.

Some forms of IDPS are more mature than others because they have been in use much longer. Network-based IDPS and some forms of host-based IDPS have been commercially available for over ten years. Network behavior analysis software is a somewhat newer form of IDPS that evolved in part from products created primarily to detect DDOS attacks, and in part from products developed to monitor traffic flows on internal networks. Wireless technologies are a relatively new type of IDPS, developed in response to the popularity of wireless local area networks (WLAN) and the growing threats against WLANs and WLAN clients.

2.5.5. Approaches of IDS

Currently there are two basic approaches to intrusion detection (Anita et al., 2003):

The first approach, called anomaly detection, is to define and characterize correct static form and/or acceptable dynamic behavior of the system, and then to detect wrongful changes or wrongful behavior. It relies on being able to define desired form or behavior of the system and then to distinguish between that and undesired or anomalous behavior. The boundary between acceptable and anomalous form of stored code and data is precisely definable. One bit of difference indicates a problem. The boundary between acceptable and anomalous behavior is much more difficult to define.

The second approach, called misuse detection, involves characterizing known ways to penetrate a system. Each one is usually described as a pattern. The misuse detection system monitors for explicit patterns. The pattern may be a static bit string, for example a specific virus bit string insertion. Alternatively, the pattern may describe a suspect set or sequence of actions. Patterns take a variety of forms as will be illustrated later.

According to Chang et al (2005) intrusion detection systems have been built to explore both approaches – anomaly detection and misuse detection – for the past 15 to 20 years. In some cases, the two kinds of detection are combined in a complementary way in a single system. There is a consensus in the community that both approaches continue to have value.

According to Chang et al (2005) view, no fundamentally different alternative approach has been introduced in the past decade. However, new forms of pattern specifications for misuse detection have been invented. The techniques for single systems have been adapted and scaled to address intrusion in distributed systems and in networks. Efficiency and system control have improved. User interfaces have improved, especially those for specifying new misuse patterns and for interaction with the system security administrator.

2.6. Cost sensitive feature selection and Intrusion Detection

A very important but often neglected facet of intrusion detection is its cost-effectiveness, or cost-benefit trade-off. An educated decision to deploy a security mechanism such as IDS is

often motivated by the needs of security risk management (Denning, 1999). The objective of IDS is therefore to provide protection to the information assets that are at risk and have value to an organization. An IDS needs to be cost-effective in that it should be cost no more than the expected level of loss from intrusions. This requires that IDS consider the trade-off among cost factors, which at the minimum should include development cost, the cost of damage caused by an intrusion, the cost of manual or automatic response to an intrusion, and the operational cost, which measures constraints on time and computing resources. For example, an intrusion which has a higher response cost than damage cost should usually not be acted upon beyond simple logging.

Currently these cost factors are, for the most part, ignored as unwanted complexities in the development process of IDSs (Amir et al., 2011). This is caused by the fact that achieving a reasonable degree of technical effectiveness is already a challenging task, given the complexities of today's network environments and the manual effort of knowledge-engineering approaches (e.g., encoding expert rules). Some IDSs do try to minimize operational cost. For example, the Bro (1998) scripting language for specifying intrusion detection rules does not support for-loops because iteration through a large number of connections is considered time consuming.

It is possible to develop a data mining framework for building intrusion detection models in an effort to automate the process of IDS development and lower its development cost. The framework uses data mining algorithms to compute activity patterns and extract predictive features, and then applies machine learning algorithms to generate detection rules (Bro, 1998).

2.6.1. Feature selection

Feature selection is a process that selects a subset of original features. The optimality of a feature subset is measured by an evaluation criterion. As the dimensionality of a domain expands, the number of features N increases. Finding an optimal feature subset is usually intractable (Kohavi and John, 1997) and many problems related to feature selection have been shown to be NP-hard (Blum and Rivest, 1992). A typical feature selection process consists of four basic steps (shown in Figure 2.6), namely, subset generation, subset

evaluation, stopping criterion, and result validation (Dash et al., 1997). Subset generation is a search procedure (Liu and Motoda, 1998) that produces candidate feature subsets for evaluation based on a certain search strategy. Each candidate subset is evaluated and compared with the previous best one according to a certain evaluation criterion. If the new subset turns out to be better, it replaces the previous best subset.

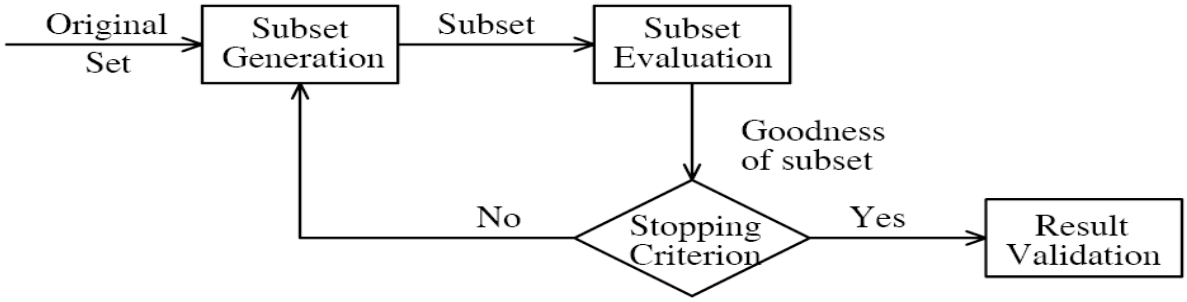

Figure 2.6.: Four key steps of feature selection (Dash et al., 1997)

The process of subset generation and evaluation is repeated until a given stopping criterion is satisfied. Then the selected best subset usually needs to be validated by prior knowledge or different tests via synthetic and/or real-world data sets. Feature selection can be found in many areas of data mining such as classification, clustering, association rules, regression. For example, feature selection is called subset or variable selection in Statistics (Miller, 2002).

2.6.1.1 General Procedure of Feature Selection

This section will be discussed in detail the four key steps as shown in Figure 2.6 of Section 2.6.1

2.6.1.1.1. Subset generation

Subset generation is essentially a process of heuristic search, with each state in the search space specifying a candidate subset for evaluation. The nature of this process is determined by two basic issues. First, one must decide the search starting point (or points) which in turn influences the search direction. Search may start with an empty set and successively add features (i.e., forward), or start with a full set and successively remove features (i.e., backward), or start with both ends and add and remove features simultaneously (i.e., bi-directional). Search may also start with a randomly selected subset in order to avoid being

trapped into local optima (Doak, 1992). Second, one must decide a search strategy. For a data set with N features, there exist 2N candidate subsets. This search space is exponentially prohibitive for exhaustive search with even a moderate N. Therefore, different strategies have been explored: complete, sequential, and random search.

Complete search

It guarantees to find the optimal result according to the evaluation criterion used. Exhaustive search is complete (i.e., no optimal subset is missed). However, complete search does not necessarily means that it must be exhaustive. Different heuristic functions can be used to reduce the search space without jeopardizing the chances of finding the optimal result. Hence, although the order of the search space is O (2N), a smaller number of subsets are evaluated. Some examples are branch and bound, and beam search (Doak, 1992).

Sequential search

It gives up completeness and thus risks losing optimal subsets. There are many variations to the greedy hill-climbing approach, such as sequential forward selection, sequential backward elimination, and bi-directional selection (Liu and Motoda, 1998). All these approaches add or remove features one at a time. Another alternative is to add (or remove) p features in one step and remove (or add) q features in the next step (p > q) (Doak, 1992). Algorithms with sequential search are simple to implement and fast in producing results as the order of the search space is usually O (N^2) or less.

Random search

It starts with a randomly selected subset and proceeds in two different ways. One is to follow sequential search, which injects randomness into the above classical sequential approaches. Examples are random-start hill-climbing and simulated annealing (Doak, 1992). The other is to generate the next subset in a completely random manner (i.e., a current subset does not grow or shrink from any previous subset following a deterministic rule), also known as the Las Vegas algorithm. For all these approaches, the use of randomness helps to escape local optima in the search space, and optimality of the selected subset depends on the resources available.

2.6.1.1.2. Subset evaluation

Each newly generated subset needs to be evaluated by an evaluation criterion. The goodness of a subset is always determined by a certain criterion (i.e., an optimal subset selected using one criterion may not be optimal according to another criterion). Evaluation criteria can be broadly categorized into two groups based on their dependency on mining algorithms that will finally be applied on the selected feature subset. The two groups of evaluation criteria are discussed below:

2.6.1.1.2.1. Independent criteria

Typically, an independent criterion is used in algorithms of the filter model. It tries to evaluate the goodness of a feature or feature subset by exploiting the intrinsic characteristics of the training data without involving any mining algorithm. Some popular independent criteria are distance measures, information measures, dependency measures, and consistency measures (Liu and Motoda, 1998)

Distance measures are also known as divergence, or discrimination measures. For a two-class problem, a feature X is preferred to another feature Y if X induces a greater difference between the two-class conditional probabilities than Y, because we try to find the feature that can separate the two classes as far as possible. X and Y are indistinguishable if the difference is zero.

Information measures typically determine the information gain from a feature. The information gain from a feature X is defined as the difference between the prior uncertainty and expected posterior uncertainty using X. Feature X is preferred to feature Y if the information gain from X is greater than that from Y.

Dependency measures are also known as correlation measures or similarity measures. They measure the ability to predict the value of one variable from the value of another. In feature selection for classification, it is possible to look for how strongly a feature is associated with the class. A feature X is preferred to another feature Y if the association between feature X and class C is higher than the association between Y and C. In feature selection for clustering, the association between two random features measures the similarity between the two.

Consistency measures are characteristically different from the above measures because of their heavy reliance on the class information and the use of the Min-Features bias in

selecting a subset of features. These measures attempt to find a minimum number of features that separate classes as consistently as the full set of features can. An inconsistency is defined as two instances having the same feature values but different class labels.

2.6.1.1.2.2. **Dependency criteria**

A dependency criterion used in the wrapper model requires a predetermined mining algorithm in feature selection and uses the performance of the mining algorithm applied on the selected subset to determine which features are selected. It usually gives superior performance as it finds features better suited to the predetermined mining algorithm, but it also tends to be more computationally expensive, and may not be suitable for other mining algorithms (Blum and Rivest, 1992). For example, in a task of classification, predictive accuracy is widely used as the primary measure. It can be used as a dependent criterion for feature selection. As features are selected by the classifier that later on uses these selected features in predicting the class labels of unseen instances, accuracy is normally high, but it is computationally rather costly to estimate accuracy for every feature subset.

In a task of clustering, the wrapper model of feature selection tries to evaluate the goodness of a feature subset by the quality of the clusters resulted from applying the clustering algorithm on the selected subset. There exist a number of heuristic criteria for estimating the quality of clustering results, such as cluster compactness, scatter and maximum likelihood. Recent work on developing dependent criteria in feature selection for clustering can be found in (Dy and Brodley, 2000)

2.6.1.1.3. Stopping criteria

A stopping criterion determines when the feature selection process should stop. Some frequently used stopping criteria are: (a) the search completes; (b) some given bound is reached, where a bound can be a specified number (minimum number of features or maximum number of iterations); (c) subsequent addition (or deletion) of any feature does not produce a better subset; and (d) a sufficiently good subset is selected (e.g., a subset may be sufficiently good if its classification error rate is less than the allowable error rate for a given task).

2.6.1.1.4. Result validation

A straightforward way for result validation is to directly measure the result using prior knowledge about the data. If we know the relevant features beforehand as in the case of synthetic data, we can compare this known set of features with the selected features. Knowledge on the irrelevant or redundant features can also help. We do not expect them to be selected. In real-world applications, however, we usually do not have such prior knowledge. Hence, we have to rely on some indirect methods by monitoring the change of mining performance with the change of features. For example, if we use classification error rate as a performance indicator for a mining task, for a selected feature subset, we can simply conduct the "before-and-after" experiment to compare the error rate of the classifier learned on the full set of features and that learned on the selected subset (Liu and Motoda, 1998).

2.6.3. Feature ranking and filtering criteria

The rapid developments in computer science and engineering have led to expediency and efficiency in capturing huge accumulations of data. The new challenge is to transform the enormous of data into useful knowledge for practical applications (Anirut and Nualsawat , 2011).

An earlier general task in data mining is to extract outstanding features for the prediction. This function can be broken into two groups—feature extraction or feature transformation, and feature selection (Mukkamala and Sung, 2003). Feature extraction (for example, principal component analysis, singular-value decomposition, manifold learning, and factor analysis) refers to the process of creating a new set of combined features (which are combinations of the original features). On the other hand, feature selection is different from feature extraction because it does not produce new variables. Feature selection also known as variable selection, feature reduction, attribute selection or variable subset selection, is a widely used dimensionality reduction technique, which has been the focus of much research in machine learning and data mining and found applications in text classification, web mining, and so on. It allows for faster model building by reducing the number of features, and also helps remove irrelevant, redundant and noisy features. This allows for building simpler and more comprehensible classification models with classification

performance. Hence, selecting relevant attributes are a critical issue for competitive classifiers and for data reduction. In the meantime, feature weighting is a variant of feature selection. It involves assigning a real-valued weight to catch feature. The weight associated with a feature measures its relevance or significance in the classification task (John et al., 1994). Feature selection algorithms typically fall into two categories; Feature Ranking and Subset Selection. Feature Ranking ranks the features by a metric and eliminates all features that do not achieve an adequate score (selecting only important features). Subset selection searches the set of possible features for the optimal subset. Feature Ranking methods are based on statistics, information theory, or on some function of classifier's outputs (Duch et al., 2003). In statistics, the most popular form of feature selection is stepwise regression. It is a greedy algorithm that adds the best feature (or deletes the worst feature) at each round. The main control issue is deciding when to stop the algorithm. In machine learning, this is typically done by cross validation (Ron, 1995).

The feature selection experiments can be conducted using filter-based approaches on the training data. The objective of this experiment is to find out which filter can achieve the best performance for intrusion detection data, and to suggest a good feature subset that contains relevant features with relative order of importance for baseline reference. Hence, the features can be ranked according to their relevance values, in order to determine their orders of importance. These filtering criteria are briefly discussed as follows (Chi-Ho et al., 2007):

- ✓ **Information gain (IG)**, which is also known as mutual information, measures the expected reduction in entropy of class before and after observing features. Larger difference indicates that the selected feature is more important to contain the class discriminatory information. IG is measured as

$$InfoGain(S, F) = Entropy(S) - \sum_{v \in V(F)} \frac{|S_v|}{|S|} \cdot Entropy(S_v),$$

Where S is the pattern set, |S| is the number of samples in S, v is value of feature F, and Sv is the subset of S where feature F has value v. The entropy of class before observing features is defined as

$$Entropy(S) = \sum_{c \in C} -\frac{|S_c|}{|S|} \cdot \log_2 \frac{|S_c|}{|S|},$$ where C is the class set and Sc is the subset of S belonging to class c. IG is the fastest and simplest ranking method, however, the drawback is that it flavors the features with many number of values.

Gain ratio (GR) normalizes the IG by dividing it by the entropy of S with respect to feature F, in order to discourage the selection of features with many uniformly distributed values. GR is measured as:

GainRatio(S, F) = InfoGain(S, F)/SplitInfo(S, F),

$$SplitInfo(S, F) = \sum_{i=1}^{n} -\frac{|S_i|}{|S|} \cdot \log_2 \frac{|S_i|}{|S|},$$

- ✓ **Chi-square (CS)** measures the well-known _2 statistics of each individual feature with respect to the classes. The features are ranked by the descending order of their _2 values, in which large _2 values obtained by the features reveal their strong correlation with the classes. The _2 of a feature F is measured as:

$$\chi^2(F) = \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{(A_{ij} - E_{ij})^2}{E_{ij}},$$

$$E_{ij} = \frac{R_i \cdot C_j}{|S|},$$ Where m is the number of intervals discretized from the numerical values of F, k is the number of classes, Aij is the number of samples in the ith interval with jth class, and Eij is the expected occurrence of Aij in which Ri is the number of samples in the ith interval and Cj is the number of samples in the jth class. These information-theoretical and statistical criteria have been empirically proved to be effective to select relevant features from some high-dimensional real-world domains; however, the major limitation of their applications is that they assume the independence property of features. As the interaction of relevant features is not taken into account, filtering on highly correlated features might degrade performance of classifiers.

2.7. Related Work in intrusion detection

Intrusion detection concept was introduced by James Anderson (1980) in 1980 defined an intrusion attempt or threat to be potential possibility of a deliberate unauthorized attempt to access information, manipulate or render a system unreliable or unusable.

Denning (1987) have developed a model for monitoring audit record for abnormal activities in the system. Sequential rules are used to capture a user's behavior over time. These rules are used to store patterns of user's activities which deviate significantly from those specified rules in the database. According to Denning (1987) high quality sequential patterns are automatically generated using inductive generalization and the lower quality patterns are eliminated.

Frank (1994) notes that clustering is an effective way to find hidden patterns in data that human might otherwise miss. Clustering is useful in intrusion detection as malicious activity should cluster together, separate from non-malicious activity.

As addressed by Frank (1994) a classification based IDS attempts to classify all traffic as either normal or malicious in some manner. The primary difficulty in this approach is how accurately the system can learn what these patterns are. This ultimately affects the accuracy of the system both in terms of whether non-hostile activity is flagged (false positive) and whether malicious activity will be missed (false negative). Some classification techniques are binary (they classify data into one of two classes), while others are n-ary (they classify data into one of an arbitrary number of classes). Some of the techniques have also been investigated as being used as a filtering mechanism to limit the amount of data that successive classification systems need to evaluate (Frank, 1994; Barbar´a et al., 2001).

Jake et al (1998) applied neural networks to detect intrusions. Neural network can be used to learn a print (user behavior) and identify each user. If it does not match then the system administrator can be alerted. A back propagation neural network called NNID was trained for this process.

Lane (2000) tested his host-based IDS in part of an instance-based learner, a type of exemplar clustering approach. It performed comparably to his Hidden Markov Model on

the same data. While not all clustering techniques are applicable to the intrusion detection domain, the wealth of techniques that Berkhin (2002) presented easily leaves the impression that there is a great deal of potential for further research in the application of clustering techniques to network intrusion detection.

Dewan and Mohammad (2010) presents an alert classification to reduce false positives in IDS using improved self adaptive Bayesian algorithm (ISABA). It is applied to the security domain of anomaly based network intrusion detection.

Sathyabama et al (2011) used clustering techniques to group user's behavior together depending on their similarity and to detect different behaviors and specified as outliers.

Amir et al (2011) formalized SOM to classify IDS alerts to reduce false positive alerts. Alert filtering and cluster merging algorithms are used to improve the accuracy of the system.SOM is used to find correlations between alerts.

Alan et al (2002) has developed NIDS using classifying self organizing maps for data clustering. MLP neural network is an efficient way of creating uniform, grouped input for detection when a dynamic number of inputs are present.

An ensemble approach Srinivas et al (2004) helps to indirectly combine the synergistic and complementary features of the different learning paradigms without any complex hybridization. The ensemble approach outperforms both SVMs MARs and ANNs. SVMs outperform MARs and ANN in respect of Scalability, training time, running time and prediction accuracy. Shilendra et al (2011) focused on the dimensionality reduction using feature selection. The Rough set support vector machine (RSSVM) approach deploy Johnson's (2002) and genetic algorithm of rough set theory to find the reduce sets and sent to SVM to identify any type of new behavior either normal or attack one.

Taeshik and Jong (2007) proposed an enhanced SVM approach framework for detecting and classifying the novel attacks in network traffic. The overall framework consist of an enhanced SVM- based anomaly detection engine and its supplement components such as packet profiling using SOFM, packet filtering using PTF, field selection using Genetic Algorithm and packet flow-based data preprocessing. SOFM clustering was used for normal profiling. The SVM approach provides false positive rate similar to that of real NIDSs. Sadiq

(2011) genetic algorithm can be effectively used for formulation of decision rules in intrusion detection through the attacks which are more common can be detected more accurately.

Norouzian and Merati (2011) defined Multi- Layer Perceptron (MLP) for implementing and designing the system to detect the attacks and classifying them into six groups with two hidden layers of neurons in the neural networks. Host based intrusion detection is used to trace system calls. Normal and intrusive behavior are collected through system call and analysis is done through DM and fuzzy technique.

To the knowledge of the researcher there are only two attempts in our country that have been done so far towards the application of DM in the intrusion detection. Adamu (2010) had tried to study a machine learning intrusion detection System that investigate the application of cost sensitive learning using data mining approach to network intrusion detection. Adamu's proposed learning approach for network intrusion detection performed using cost sensitive learning techniques by testing decision tree algorithm on labeled records. The other research which is undertaken by Zewdie (2010), tried to develop a predicative model for network intrusion detection using information gain value for feature selection. He proposed an optimal feature selection for Network Intrusion Detection using indirect cost sensitive feature selection approach using decision tree as classification technique.

Both Adamu and Zewdie have tried to test data mining application on intrusion detection using the supervised classification techniques of the J48 decision tree algorithm. Both of them after building the model they did not validate the model with real life data. Hence this study has a great contribution in applying DM technology for the purpose of enhancing security in the networking environment. It is contributed in the areas considering both labeled and unlabeled records which solved the problem of both supervised and unsupervised approaches. The IDS model constructed in this thesis notify for the administrators after detecting an attack and administrators will take proper actions.

CHAPTER THREE

DATASET PREPARATION

This chapter showed that how data is prepared for the purpose of the experiment.

As showed in chapter one, under the methodology section of this thesis, the DM process model selected is KDD which starts from selection of data. Here there was no need to understanding the business because the researcher has taken a readymade dataset from the MIT Lincoln lab.

3.1. Initial Data Selection

Before generating a pattern the data should be ready in the way that is easily understandable. The intrusion detection dataset is based on the KDD 99. The KDD 99 intrusion detection datasets are based on the 1998 DARPA initiative, which provides designers of IDS with a benchmark on which to evaluate different methodologies (KDD Cup, 2011). Raw data was initially collected regarding intrusion detection attacks and related issues from the MIT data set and KDD 99 intrusion detection datasets, which are based on DARPA 98 dataset, provide labeled and unlabeled data for researchers working in the field of intrusion detection. Originally 311,027 dataset were collected.

The training and test dataset of NSL-KDD similar to KDD99 consists both quantitative and qualitative 41 features and is labeled as either normal or an attack, with exactly one specific attack type and the simulated attacks fall in one of the following five main classes for intrusion analysis, namely, one normal class and four main intrusion classes: Probe, DOS, U2R, and R2L (Lee et al., 1999).

I. **Normal connections (Normal)** are produced by pretending daily user behavior such as downloading files, and visiting web pages.
II. **Denial of service (DOS):** In this type of attack an attacker makes some computing or memory resources too busy or too full to handle legitimate requests, or denies

legitimate users access to a machine. Examples are Apache2, Back, Land, Mailbomb, SYN Flood, Ping of death, Process table, Smurf, Teardrop.

III. **Remote to user (R2L):** In this type of attack an attacker who does not has an account on a remote machine sends packets to that machine over a network and exploits some vulnerability to gain local access as a user of that machine. Examples are Dictionary, FTP_write, Guest, Imap, Named, Phf, Sendmail, Xlock .

IV. **User to root (U2R):** In this type of attacks an attacker starts out with access to a normal user account on the system and is able to exploit system vulnerabilities to gain root access to the system. Examples are Eject, Loadmodule, Ps, Xterm, Perl, Fdformat.

V. **Probing:** In this type of attacks an attacker scans a network of computers to gather information or find known vulnerabilities. An attacker with a map of machines and services that are available on a network can use this information to look for exploits. Examples are Ipsweep, Mscan, Saint, Satan, Nmap.

The description of each feature is shown in the appendix A:

The researcher used NSL-KDD intrusion detection dataset which is available at KDD (2012). Initially the researcher has taken the Kddcup.data-10-perecnt which contains total of 311,027 records which includes both labeled and unlabeled records. From these records around 55 % of them are unlabeled and the remaining 45 % are labeled.

For Semi-supervised modeling, the first 21,699 records are taken. After preprocessing for this thesis 21,533 records are used. From these records 55 % of them are unlabeled and the remaining 45 % are labeled.

The distribution of labeled and unlabeled records for semi-supervised modeling of this thesis was depending on the initial distribution of records in the Kddcup.data-10-perecnt that explained previously in this section.

The distributions of the initial data which include both labeled and unlabeled are shown in Table 3.1:

Dataset Label	Number of Records Collected
Normal	4,757
Attacks	5,016
Unlabeled	11,926
Total	21,699

Table 3.1: Distribution of the initial dataset

The distribution of attacks before preprocessing is summarized in Table 3.2:

Dataset Label	Number of Records Collected
DOS	3,598
Probe	665
R2L	725
U2R	28
Total	5,016

Table 3.2: Summary of the distribution of attacks before preprocessing

Irrelevant data and redundant data are removed before the actual data mining process. Percentage generation of data include the number of data from each class proportional to its size. After removing irrelevant and unnecessary data only 21,533 datasets are used for conducting this study.

After preprocessing the number of datasets distributed for each dataset label is summarized as follows; this includes 9,662 labeled and 11,871 are unlabeled which is shown in Table 3.3.

Dataset Label	Number of Records Collected
Normal	4,716
Attacks	4,946
Unlabeled	11,871
Total	21, 533

Table 3.3: Distribution of collected records with respect to dataset label

3.2. Data Cleaning and preprocessing

Before data is fed into a DM algorithm, it must be collected, inspected, cleaned and selected. Since even the best predictor will fail on bad data, data quality and preparation is crucial. Also, since a predictor can exploit only certain data features, it is important to detect which data preprocessing works best (Meera et al., 2003). For this study preprocessing of NSL-KDD dataset contains the following processes:

- ✓ Assigning attack names to one of the five classes NORMAL, PROBE, DOS (Denial of Service), U2R (User to Root) and R2L (Remote to Local). To identify and label each attacks different literatures are consulted and Microsoft Excel helps to filter and name easily using fill handle.
- ✓ There are records which don't have attributes and these are removed from the dataset. There is also a mismatch in the KDD 99 winner cost matrix and the confusion matrix; as a result arrangements are made to match the cost matrix and confusion matrix.
- ✓ The NSL-KDD dataset is available in text format; so to be read by WEKA tool it has to be changed into ARFF format. Also for WEKA Data can be imported from a file in various formats: CSV, C4.5, binary (Chang et al., 2005). The WEKA format of attributes and data declaration is presented in appendix B.

3.3. Data Transformation and Feature Selection

Feature selection is the process of removing features from the data set that are irrelevant with respect to the task that is to be performed (Selvakani and Rajiesh , 2007). Feature selection can be extremely useful in reducing the dimensionality of the data to be processed by the classifier, reducing execution time and improving predictive accuracy (inclusion of irrelevant features can introduce noise into the data, thus obscuring relevant features). It is worth noting that even though some machine learning algorithms perform some degree of feature selection themselves (such as classification trees); feature space reduction can be useful even for these algorithms. Reducing the dimensionality of the data reduces the size of the hypothesis space and thus results in faster execution time.

Effective feature (attribute) selection from intrusion detection datasets is one of the important research challenges for constructing high performance IDS. Irrelevant and redundant attributes of intrusion detection dataset may lead to complex intrusion detection model as well as reduce detection accuracy. This problem has been studied during the early work of Selvakan and Rajiesh (2007) on KDD99 benchmark intrusion detection dataset, where 41 attributes were constructed for each network connection.

Feature selection methods are commonly used for cost insensitive learning and found to be very helpful because the elimination of useless features enhances the accuracy of detection while speeding up the computation, thus improving the overall performance of IDS (Selvakani and Rajiesh, 2007). The attribute selection in KDD99 dataset has been widely used as a standard method for network based intrusion detection learning, and it was found that all 41 attributes of KDD99 dataset are not the best ones for intrusion detection learning. Therefore Selvakani and Rajiesh (2007) suggested that the performance of IDS may be further improved by studying attribute selection methods

3.4. Evaluation Metrics

A cost matrix (C) is defined by associating classes as labels for the rows and columns of a square matrix; in the current context for the NSL-KDD dataset, there are five classes,{NORMAL, PROBE, DOS, U2R, R2L},and therefore the matrix has dimensions of 5×5. An entry at row i and column j, C (i, j), represents the non-negative cost of misclassifying a pattern belonging to class i into class j. Cost matrix values employed for the KDD 99 dataset are defined (Charles, 2001). These values were also used for evaluating results of the KDD99 computation. The magnitude of these values was directly proportional to the impact on the computing platform under attack if a test record was placed in a wrong category. A confusion matrix (CM) is similarly defined in that row and column 5×5 matrix for the KDD 99 dataset. An entry at row i and column j, CM (i, j), represents the number of misclassified patterns, which originally belong to class i yet mistakenly identified as a member of class j.

The form of the cost matrix C will depend on the actual application. In general, it is reasonable to choose the diagonal entries equal to zero, i.e. C (i, j) = 0 for i = j, since correct classification normally incurs no cost as shown in table 3.4. In addition the size of the cost matrix should be the same as that of the confusion matrix.

	Observed vs. Predicted				
Cost(i,j)	**normal**	**probe**	**DOS**	**U2R**	**R2L**
normal	0	2	2	2	2
Probe	2	0	2	2	2
DOS	2	2	0	2	2
U2R	2	2	2	0	2
R2L	2	2	2	2	0

Table 3.4: The 5X5 cost matrix used for the KDD 1999 winner result (Charles, 2001).

3.4.1. Standard Metrics to Evaluate Intrusion

To evaluate the approach, the four standard metrics of true positive, true negative, false positive and false negative developed for network intrusions, have been used. Table 3.5 shows these standard metrics.

Confusion metrics (standard metrics)		Predicted connection label	
		Normal	Intrusions (Attacks)
Actual Connection	Normal	True Negative (TN)	False Alarm (FP)
	Intrusions (attacks)	False Negative (FN)	Correctly detected Attacks(TP)

Table 3.5: Standard metrics for evaluations of Intrusions (attacks) (Charles, 2001)

The representation of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) are defined as follows:

- ✓ True Positive (TP): The number of malicious records that are correctly identified.

- ✓ True Negative (TN): The number of legitimate record that correctly classified.
- ✓ False Positive (FP): The number of records that are incorrectly identified as attacks however in fact they are legitimate activities.
- ✓ False Negative (FN): The number of records that are incorrectly classified as legitimate activities however in fact they are malicious.

3.4.2. Performance Measure

General performance of intrusion detection systems is measured in terms of numbers of selected features and the classification accuracies of the machine learning algorithms giving the best classification results. As discussed by Farhan et al (2010) there are different techniques used for performance measuring of the IDS. Good IDS require high detection rate, low false alarm rate and lower average misclassification cost (Farhan et al., 2010). Thus during developing IDS; overall classification rate (OCA), detection rate (DR), false Positive rate (FPR), average misclassification cost (AMC), Error rate, and training and testing time are considered.

3.4.2.1. Error Rate

The error rate, which is only an estimate of the true error rate and is expressed to be a good estimate, if the number of test data is large and representative of the population, is defined as (Mrutyunjaya, 2009):

Error Rate= [(Total Test samples- Total Correctly Classified Samples)*100%] / Total Test Samples ……. (3.1)

3.4.2.2. Accuracy

Overall Classification accuracy (OCA) is the most essential measure of the performance of a classifier. It determines the proportion of correctly classified examples in relation to the total number of examples of the test set i.e. the ratio of true positives and true negatives to the total number of examples. From the confusion matrix, we can say that accuracy is the percentage of correctly classified instances over the total number of instances in total test dataset, namely the situation TP and TN, thus accuracy can be defined as follows (Farhan et al., 2010):

Accuracy = [(TP+TN)*100%]/TP+TN+FP+FN ……. (3.2)

3.4.2.3. Detection Accuracy

Detection accuracy (rate) refers to the proportion of attack detected among all attack data, namely, the situation of TP, thus detection rate is defined as follows (Farhan et al., 2010):

Detection Accuracy = (TP*100%)/ (TP+FN) …… (3.3)

3.4.2.4. False Positive rate

False positive rate, also known as False Alarm Rate (FAR) measures the number of misclassified positive instances in relative to the total number of misclassified instances. It can be expressed also as the proportion that normal data is falsely detected as attack behavior, namely, the situation of FP. Thus false alarm rate is defined as follows (Farhan et al., 2010):

False Positive Rate= (FP*100%)/ (FP+TN) ….. (3.4)

3.4.2.5. Precision and Recall

Recall and precision are two widely used metrics employed in applications where successful detection of one of the classes is considered more significant than detection of the other classes (Ferri et al., 2002).

Precision

Precision is the number of class members classified correctly over the total number of instances classified as class members. Technically can be expressed as the attack has been occurred and the IDS detect correctly (Ferri et al., 2002).

Precision= (TP*100%)/ (TP+FP)……. (3.5)

Recall

Also called True Positive Rate (TPR), Recall measures the number of correctly classified examples relative to the total number of positive examples. In other words the number of class members classified correctly over the total number of class members (Ferri et al., 2002).

Recall= (TP*100%)/ (TP+FN) …

CHAPTER FOUR

EXPERIMENTATION

This chapter describes experimental study of the algorithms and procedures, which are described in the previous chapters. In this research both labeled and unlabeled records are used. The dataset is in a spreadsheet (Excel) format save a lot of time in preprocessing. Rows are records of connection; columns are attributes of records (duration, protocol type, service, src_bytes, dst_bytes, etc) and each cell should include one value only (Yeung and Ding, 2003). Data mining software tools used are WEKA (version 3.7.5) and Tanagra (Version 1.4.42).

4.1. Experimentation Design

All experiments are performed in a computer with the configurations Intel(R) Core(TM) 2 CPU 2.16GHz, 4 GB RAM, and the operating system platform is Microsoft Windows 7. Weka and Tanagra are collections of machine learning algorithms for data mining tasks that contain facilities for data preprocessing, classification, regression, clustering, association rules, and visualization. Microsoft Excel is used to filter records for manual labeling of the five classes from the NSL-KDD dataset. The following are the steps used in this thesis experimentation:

I. In the beginning, in order to build the experiment, the researcher selected the data mining software and records used in the experiment.
II. The selected records are changed from text format into Microsoft excel format. The Microsoft excel portion of the records contain both labeled and unlabeled records.
III. To come up with cleaned datasets preprocessing tasks are undertaken for underling missing values, outliers and other issues.
IV. Train the classifier using WEKA data mining software. The records are included both labeled and unlabeled that is a semi-supervised modeling approach is followed. The researcher used clustering over the classes to cluster evaluation for

semi-supervised modeling. Here, the researcher made the number of clusters into five by using the simple k-means clustering techniques. The number of clusters set to five because the main intention is to train the classifier that will classify unlabeled records into five classes by referring the labeled records. Using visualizes the clusters method the researcher saved the clusters into ARFF format. Then the ARFF format opened in the Microsoft excel format. Then the researcher assigned the classes as Normal, Probe, DOS, U2R and R2L based on the clusters in the trained classifier.

V. At this step open the ARRF file using WEKA tool. Before running classification techniques different filtering techniques are done depending on the objective of the study. For feature selection there is a filtering technique which is called Attribute Selection under supervised approach or under select attribute menu. Evaluate the trained classifier using all attributes or some selected attributes by excluding unimportant features to achieve Feature ranking and selection of relevant features. After selecting either all features or some selected features develop the classifier model for different predictive modeling techniques. Here, there were a number of experiments done by changing different test options and classifier techniques. The performance comparison between different experimentation was evaluated and discussed.

VI. From previous step, those training models which scores better classification accuracy has selected for this study.

VII. Lastly, the selected model for this study is tested by previously unseen records which were unlabeled that enable to determine the performance of the selected model.

4.2. Semi-Supervised Modeling

Supervised intrusion detection approaches use only labeled data for training. To label the data however are often difficult, expensive, or time consuming as they require the efforts of experienced human attention. Meanwhile unlabeled data may be relatively easy to collect, but there has been few ways to use them. In this study semi-supervised learning addressed

this problem by using large amount of unlabeled data, together with the labeled data, to build better classifiers.

Semi-supervised learning (SSL) addressed in this thesis by allowing the model to integrate part or all of the available unlabeled data in its supervised learning (Pachghare et al., 2011). The goal is to maximize the learning performance of the model through such newly-labeled examples while minimizing the work required of human effort.

In semi-supervised learning the training data (observations, measurements, etc.) are accompanied by both labels and unlabeled records. Label records indicating the class of the observations. Those unlabeled records show that the class is empty. New data is classified based on the training set. Classification is one of the categories under Semi-supervised learning.

In a classification task of machine learning each instance of a dataset is assigned into a particular class. A classification based IDS attempts to classify all traffic as either normal or malicious. The challenge in this case is to minimize the number of false positives (classification of normal traffic as malicious) and false negatives (classification of malicious traffic as normal). In this research, for the classification modeling experiments, the decision tree (in particular the J48 algorithm) and the Naïve Bayes methods are selected.

As it has been discussed in this study under section 3.2, initially 311,027 records which included both labeled and unlabeled are taken. For this study after preprocessing and other tasks 21,533 records are selected. From these records 9,662 (45%) are labeled and 11,817(55%) of them are unlabeled. These records are used for next experimentation under semi-supervised modeling. Before applying the classification learning techniques, however, applying a clustering technique will help to identify the class for those unlabeled records.

4.2.1. Training the classifier

To train the classifier the researcher applied the clustering technique of K-Means algorithm. In the k-Means clustering algorithm the default value for number of clusters is 2. For this study the value changed into 5 because the researcher want to train a classifier

which will classify those unlabeled records into 5 classes by refereeing to those labeled records. The cluster mode used in this experiment is "classes to clusters evaluation".

Before making a semi-supervised experiment, an experimentation is done that will help to train a classifier which cluster both labeled and unlabeled records into 5 clusters. Using the K-Means clustering; the researchers build a model that is going to cluster every unlabeled record into specific cluster.

The researcher made the clusters into 5 because the classes of the intrusion are known classes which are 5. These classes are Normal, Probe, U2R, R2L and DOS. Form the total given records 45% of the records are labeled by these classes. The semi-supervised modeling will classify those remaining 55% unlabeled records into their corresponding classes based on the given classes. After the clustering applied the distribution of the clustered instances showed in table 4.1. The percentage distribution of the clustered instances from table 4.1: Cluster 0=7888 (37%), Cluster 1= 7939 (36%), Cluster 2= 1226 (6%), Cluster 3= 1138 (5%), and Cluster 4=3342 (16%). The Class attribute are considered as classes. There are 9217(42.80%) records which are not correctly clustered.

Clustered as	Cluster0	Cluster1	Cluster2	Cluster3	Cluster4	Sum
Normal	7853	4373	779	707	2875	16587
DOS	0	3566	0	0	0	3566
U2R	4	0	10	6	0	20
Probe	0	0	232	420	0	652
R2L	31	0	205	5	467	708
Sum	7888	7939	1226	1338	3342	21533

Table 4.1: Classes to cluster for Semi-supervised Approach

Cluster 0 <-- Normal, Cluster 1 <-- DOS, Cluster 2 <-- U2R, Cluster 3 <-- Probe, and Cluster 4 <-- R2L

For the classification modeling experiments, the decision tree (in particular the J48 algorithm) and the Naïve Bayes simple algorithms will explored in the in semi – supervised modeling.

4.2.2. J48 decision tree modeling

As described earlier, the J48 algorithm is used for building the decision tree model. The training of the decision tree classification models of the experimentation is done by employing the 10-fold cross validation and the percentage split classification models.

J48 algorithm contains some parameters that can be changed to further improve classification accuracy. Initially the classification model is built with the default parameter values of the J48 algorithm. Table 4.2 summarizes the default parameters with their values for the J48 decision tree algorithm.

parameter	Description	Default Value
ConfidenecFactor	The confidence factor used for pruning (smaller values incur more pruning)	0.25
minNumObj	The minimum number of instances per leaf	2
Unpruned	Weather pruning is performed	False
Subtreeraising	Weather sub tree information is hidden or expanded	True

Table 4.2: Some of the J48 algorithm parameters and their default values

As described before, the J48 algorithm is used for building the decision tree model. The training of the decision tree classification models of the experimentation is done by employing the 10-fold cross validation and the percentage split classification modes.

4.2.2.1. Experimentation I:

The first experimentation is performed with the default parameters. The default 10-fold cross validation test option is employed for training the classification model. Using these default parameters the classification model is developed with a J48 decision tree having 608 numbers of leaves and 675 tree size.

The decision tree used more than 17 of the total 41 variables for generating the tree. Some of these are: logged_in, dst_bytes, protocol_type, diff_srv_rate, srv_diff_host_rate, src_bytes, falg, service, dst_host_srv, and num_compromised. The decision tree has also shown that

the protocol_type variable is the most determining one. Table 4.3 depicts the resulting confusion matrix of this model.

Total number of instances (training sets)	Correctly classified Instances	Incorrectly Classified Instances
21,533	20,696 (96.11 %)	837 (3.89%)

Table 4.3: Semi-supervised classification accuracy using J48 algorithm parameters with 10-fold cross validation

As shown in the resulting confusion matrix, the J48 learning algorithm scored an accuracy of 96.11%. This result shows that out of the total training datasets 21,533 (96.11%). records are correctly classified, while 837 (3.89%) of the records are incorrectly classified. Resulting of confusion is presented in Appendix C.

As described before, the size of the tree and the number of leaves produced from this training was 675 and 608 respectively. This seems that it is difficult to traverse through all the nodes of the tree in order to come out with valid rule sets. Therefore, to make ease the process of generating rule sets or to make it more understandable, the researcher attempted to modify the default values of the parameters so as to minimize the size of the tree and number of leaves. With this objective, the minNumObj (minimum number of instances in a leaf) parameter was tried with 0.25, 0.20, and 0.1.

With minNumObj of 20 a number of leaves produced are 485 and the size of the tree is 492. The accuracy is almost similar with minNumObj value of 0.25. From the total records 21,533 (95.42%) are correctly classified and 878 (4.58%) are incorrectly classified instances.

With minNumObj of 10 a number of leaves produced are 347 and the size of the tree is 374. The accuracy is almost similar with minNumObj value of 20. From the total 21,533 records 95.14% are correctly classified. This experiment has shown an improvement in the number of leaves and tree size. The size of the tree is lowered from 492 to 374 and the number of the leaves decreased to 485 from 347.

In general, though the tree size and the number of the leaves decreased, the accuracy of the J48 decision tree algorithm in this experiment is somewhat poorer than the first experiment with the default parameter value of the minNumObj.

4.2.2.2. Experimentation II:

The second experiment is performed, by changing the default testing option (the 10-fold cross validation). In this learning scheme a percentage split is used to partition the dataset into training and testing data. The purpose of using this parameter was to assess the performance of the learning scheme by increasing the proportion of testing dataset if it could achieve better classification accuracy than the first experimentation. Even if different experiments were tested using percentage splitting mechanisms the one with 75 % training and 25 % testing has scored better classification accuracy. The result of this learning scheme is summarized and presented in Table 4.4.

Total number of instances (testing sets)	Correctly classified Instances	Incorrectly Classified Instances
5,383	5,165 (95.95%)	218(4.05%)

Table 4.4: classification accuracy using J48 algorithm parameters with percentage-split set to 75%

The size of the tree and the number of leaves produced from this training were 675 and 608 respectively. In this experiment out of the 21,533 total records 16,150 (75%) of the records are used for training purpose while 5,383 (25%) of the records are used for testing purpose. As we can see from the confusion matrix of the model developed with this proportion, out of the 5,383 testing records 95.95% of them are correctly classified and 218(4.05%) records are incorrectly classified. Resulting confusion matrix is presented in Appendix C.

The above experiments showed that when the training data decreases the performance of the algorithm for predicting the newly coming instances also decreases. Though this experiment is conducted by varying the value of the training and the testing datasets, the accuracy of the algorithm for predicting new instances in their respective class could not be

improve. This shows that the previous experiment conducted with default 10-fold cross validation is better than the percentage splitting mechanism.

Therefore, from the above two experiments using decision tree, the first experiment using 10 fold-cross validations has been chosen due to its better overall classification accuracy

4.2.3. Naive Bayes modeling using all features

The other classification data mining technique employed for the classification sub phase of this research is the Naïve Bayes. To build the Naïve Bayes model, WEKA software package is used and it employs the Naïve Bayes Simple algorithm in developing the model. The 10-fold cross validation, which is set by default, and the percentage split with 75-25 for training and testing the model test options are employed.

As discussed by Pachghare et al (2011) Naive Bayes is one of the Probabilistic learning algorithm that Calculate explicit probabilities for hypothesis. In Naïve Bayes learning algorithm pprior knowledge and observed data can be combined. It is also an incremental learning algorithm that each training example can incrementally increase/decrease the probability that a hypothesis is correct. It predicts multiple hypotheses, weighted by their probabilities.

4.2.3.1. Experimentation I:

The first experiment of the Naive Bayes model building is performed using the Naive Bayes Simple algorithm with the default10-fold cross validation test option. Table 4.5 showed the resulting confusion matrix of the model developed using the Naive Bayes Simple algorithm with the default 10-fold cross validation test option.

Total number of instances (training sets)	Correctly classified Instances	Incorrectly Classified Instances
21,533	20,418 (94.82 %)	1,115(5.18 %)

Table 4.5: Classification accuracy using Naïve Bayes algorithm parameters with their default values

As shown in the above confusion matrix the Naïve Bayes Simple Algorithm scored an accuracy of 94.82%. This means out of the total 21,533 records 20,418 (94.82 %) of the records are correctly classified, while 1,115(5.18 %) of the records are misclassified. The snapshot confusion matrix taken from the tool is for training phase of using Naïve Bayes algorithm with default values in Appendix D.

The output of the above experiment shows that the model developed with Naïve Bayes Simple Algorithm is poor in the accuracy of classifying new intrusions to the corresponding class, compared with the decision tree model that is developed in experiment 4.2.2.1.

4.2.3.2. Experimentation II:

Another experiment of the Naïve Bayes model building for all features is conducted using the Naïve Bayes simple algorithm with 75 % training records and 25% testing records of percentage split test option. Even if different experiments were tested using percentage splitting mechanisms the one with 75 % training and 25 % testing has scored better classification accuracy. Resulting confusion matrix and accuracy for both the training and testing phase of using Naïve Bayes algorithm with Percentage Split is shown in Appendix D.

As shown in table 4.6 that resulted from the model developed by the Naïve Bayes Simple Algorithm with the 75% training and 25% testing percentage split, the model scored an accuracy of 94.67%.This shows that from the total 5383 test data, 5,096 (94.67%)of the records are correctly classified, while 287(5.33%)of them are misclassified.

Total number of instances (testing sets)	Correctly classified Instances	Incorrectly Classified Instances
5,383	5,096 (94.67 %)	287(5.33%)

Table 4.6: classification accuracy using Naïve Bayes with percentage-split set to 75%

In summary from the above two experiments which are conducted using the Naïve Bayes simple algorithm by using all features the algorithm with the default 10-fold cross validation test option resulted a better classification model with a better classification accuracy than the second one conducted with 75 % training and 25% testing and percentage split test option.

4.2.4. Naive Bayes modeling with feature selection

For both the experimentation of Naïve Bayes modeling with cost sensitive feature selection from a total of 41 features (attributes) 11 of them are selected. These are: protocol_type, service, flag, src_bytes, num_compromised, is_host_login, count, srv_count, diff_srv_rate, dst_host_same_src_port_rate and class. In both of the following experimentations the attribute evaluator used for cost sensitive feature selection classification model is CfsSubsetEval and the search method is Best first search.

4.2.4.1. Experimentation I:

The first experimentation of feature selection with cost sensitive modeling is performed with the default parameters. The default 10-fold cross validation test option is employed for training the classification model. The cost assigned at this experiment is 0.1. Resulting confusion matrix and accuracy for the training phase using Naïve Bayes algorithm with default values is shown in Appendix E.

As shown in table 4.7 out of the total training datasets 21,533 (94.02%). records are correctly classified, while 1,287 (5.98%) of the records are incorrectly classified.

Total number of instances (training sets)	Correctly classified Instances	Incorrectly Classified Instances
21,533	20,246 (94.02%)	1,287 (5.98%)

Table 4.7: classification accuracy using Naïve Bayes simple algorithm parameters with 10 fold cross validation using cost sensitive feature selection

To build the above cost sensitive feature selection model the total Cost incurred is 1,287 and the average classification cost is 0.0598.

4.2.4.2. Experimentation II:

The second experiment is conducted by making the percentage split parameter set to 75, which is to mean 75% for training and 25% for testing. The classification accuracy resulted from this experiment is shown in table 4.8.

(testing sets)	Correctly classified Instances	Incorrectly Classified Instances
5,383	5,061(94.02 %)	322 (5.98%)

Table 4.8: classification accuracy using Naïve Bayes simple algorithm parameters with percentage-split set to 75% using cost sensitive feature selection

In this experiment out of the 21533 total records 16150 (75%) of the records are used for training purpose while 5383 (25%) of the records are used for testing purpose. As we can see from the confusion matrix of the model developed with this proportion, out of the 5383 testing records 5,061(94.02 %) of them are correctly classified and 322(5.98%) records are incorrectly classified. Using the cost sensitive feature selection model the total Cost incurred is 322 and the average classification cost is 0.0598. Resulting confusion matrix and accuracy using Naïve Bayes algorithm with Percentage Split is shown in appendix E.

4.2.5. Comparison of Semi-Supervised Approaches: J48 decision tree and Naive Bayes model

Comparing different classification techniques and selecting the best model for predicting the network intrusions is one of the aims of this study. Accordingly the decision trees particularly the J48 algorithm and the Naïve Bayes classification approaches were used for conducting experiments. Summary of experimental result for the two classification algorithms is presented in table 4.9 below:

		Accuracy			
		Input ares all Features		**With feature Selection**	
Classifier/ Model	**Test Mode**	Correctly Classified	Incorrectly Classified	Correctly Classified	Incorrectly Classified
Naïve Bayes: Semi-Supervised	10-fold cross validation and Other default values	94.82 %	5.18 %	94.02%	5.98 %
	Percentage Split	94.67%	5.33%	94.02%	5.98%
J48 : Semi-Supervised	10-fold cross validation and Other default values	96.11 %	3.89%		
	Percentage Split	95.95%	4.05%		

Table 4.9: Comparison of Semi-Supervised Approaches

The detailed classification accuracy and confusion matrix for each algorithm conducted in this thesis were presented in the Appendix C, D and E. For comparison of the selected models summarized experimental results has showed in the table 4.10 below.

Features used	**Classifiers**	**Accuracy**	**Classes**									
			Normal		**DOS**		**Probe**		**R2L**		**U2R**	
			TP	**FP**	**TP**	**FP**	**TP**	**FP**	**TP**	**FP**	**TP**	**FP**
17	**J48 with 10 fold cross validation**	96.11 %	0.95	0.01	1	0	0.97	0.01	0.98	0.02	0.53	0.01
17	**J48 with percentage split (set to 75%)**	95.95%	0.95	0.01	1	0	0.97	0.01	0.98	0.02	0.44	0.01
41	**Naïve Bayes with 10 fold cross validation**	94.82 %	0.92	0	1	0	0.87	0	0.92	0.02	0.93	0.03
41	**Naïve Bayes with percentage split (set to 75%)**	94.67%	0.93	0	1	0	0.85	0	0.91	0.02	0.91	0.03
11	**Naïve Bayes with 10 fold cross validation**	94.02%	0.92	0	1	0	1	0.03	0.92	0.02	0.35	0.02
11	**Naïve Bayes with percentage split (set to 75%)**	94.02%	0.93	0	1	0	1	0.03	0.91	0	0.35	0.03

Table 4.10: Comparison of the confusion matrix result for J48 and Naïve Bayes Algorithms

In this thesis J48 and Naïve Bayes algorithms performed different prediction accuracy. As showed in figure 4.1 from all six experiments, the J48 with 10-fold cross validation performed better classification accuracy in identifying intrusions either normal or attack (DOS, U2R, R2L and Probe).

The reason for the J48 decision tree performing better than Naïve Bayes is because of the linearity nature of the dataset. This means there is a comprehensible segregation point that can be defined by the algorithm to predict the class of a particular network intrusion

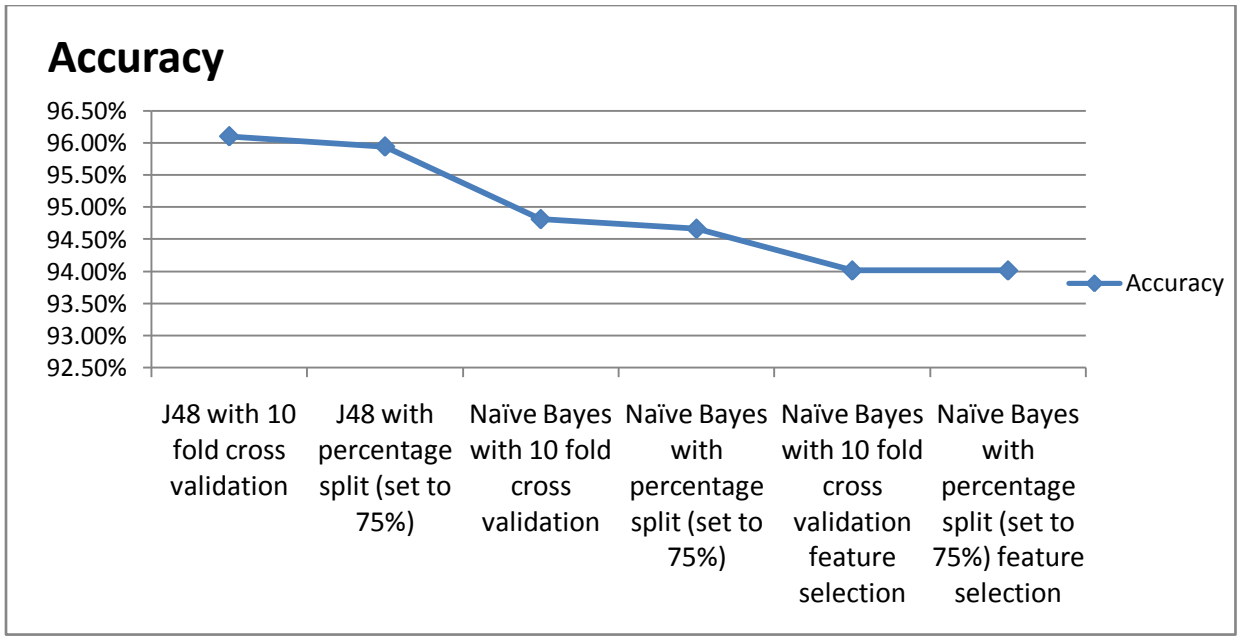

Figure 4.1: Comparison of Accuracy the J48 and Naïve Bayes Algorithms

The other reason for the Naïve Bayes, scoring a lower accuracy than the J48 decision tree is because class conditional independence assumption may not hold for some attributes, therefore loss of accuracy. In addition, in terms of ease to interpret and implement the J48 decision tree is more self-explanatory. It can handle large number of features and generates rules that can be converted to simple and easy to understand classification if-then-else rules. The average TP and FP rates for all experiments conducted in this study showed in table 4.11.

Algorithms	TP	FP
J48 with 10 fold cross validation	0.96	0.008
J48 with percentage split (set to 75%)	0.96	0.008
Naïve Bayes with 10 fold cross validation	0.95	0.004
Naïve Bayes with percentage split (set to 75%)	0.95	0.004
Naïve Bayes with 10 fold cross validation feature selection	0.94	0.006
Naïve Bayes with percentage split (set to 75%) feature selection	0.94	0.006

Table 4.11: Average TP and FP Rates

The detailed FP and TP rate for each algorithm is presented in the Appendix C, D and E.

For a good IDS TP rate should be high. As shown in the following figure 4.2, TP rate of the J48 algorithm is higher in most classes when it compared with other algorithms

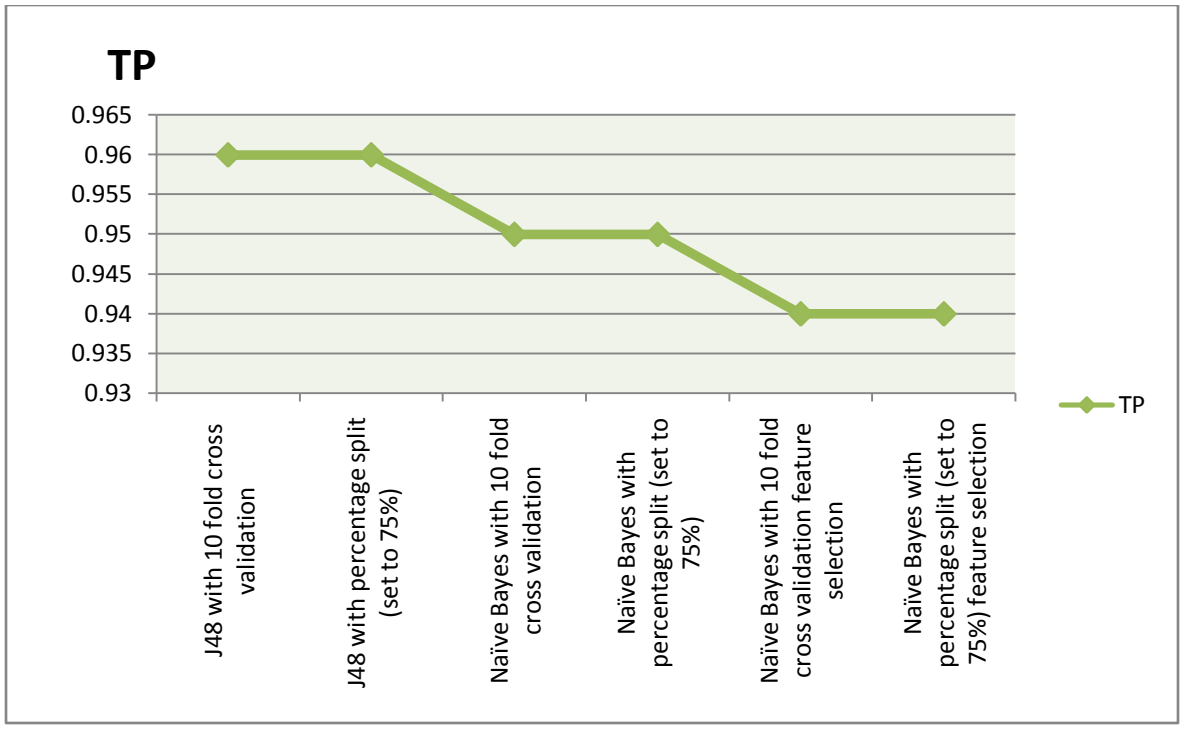

Figure 4.2: True Positive (TP) rate comparison of the J48 and Naïve Bayes Algorithms

For a good IDS FP rate should be low. As shown in figure 4.3 the FP rate of the J48 algorithm for both cases (10 fold-cross validation and percentage split mechanism) is higher when compared with the Navie Bayes algorithms. From all experiments the Naïve Bayes algorithms with all features has the lowest FP rate.

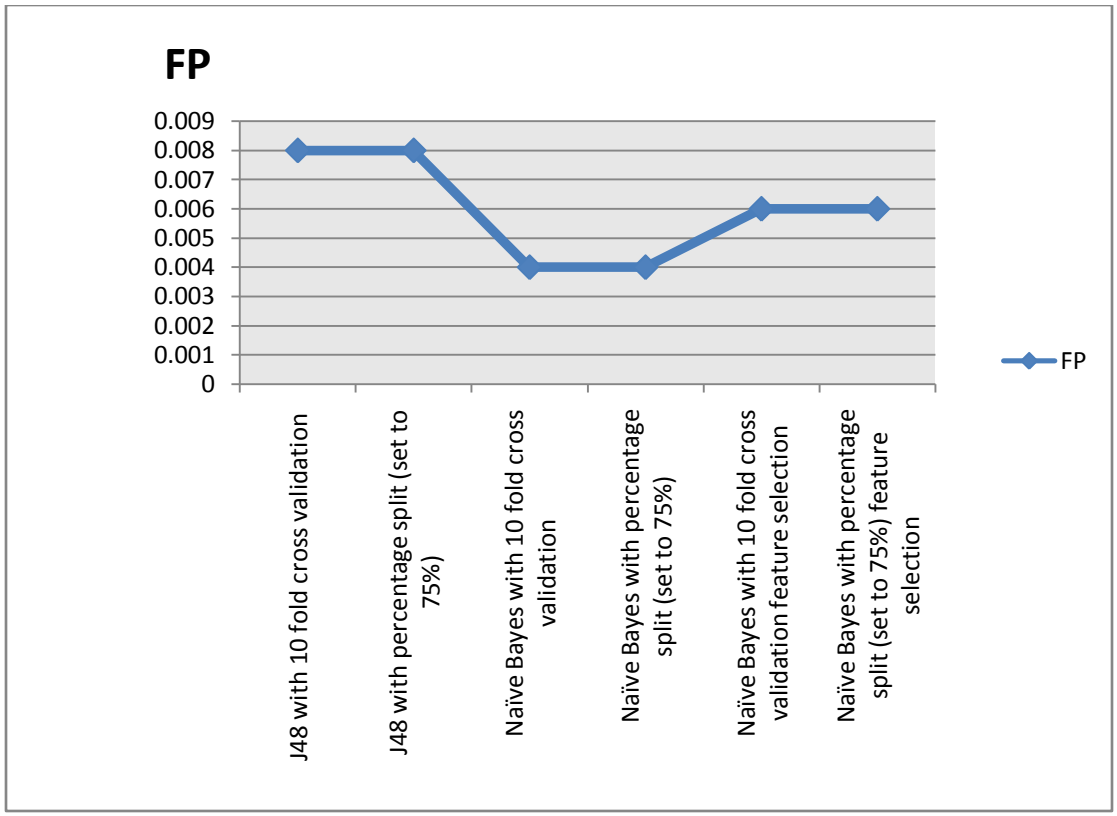

Figure 4.3: False Positive (FP) rate comparison of the J48 and Naïve Bayes Algorithms

The gap with the TP rate between the J48 decision tree and Naïve Bayes from figure 4.2 is 0.01. This means the TP rate of J48 decision tree is grater by 0.01. The gap with FP rate between the J48 decision tree and Naïve Bayes algorithm from figure 4.3 is 0.004. This means the FP rate of Naïve Bayes algorithm is lower by 0.004. So, the greater TP rate of J48 decision tree will leads it effective than Naïve Bayes algorithm.

In summary, from figure 4.1 and 4.3 it is clear that J48 algorithm with 10-fold cross validation Accuracy and TP rate is better than other algorithms. As a result, it is reasonable to conclude that the J48 algorithm is better than Naïve Bayes method for this study. Therefore, the model which is developed with the J48 decision tree with 10-fold cross validation classification techniques is considered as the selected working model for this thesis.

4.2. Evaluation of the Discovered Knowledge

In this section the selected models from those 6 experiments conducted in this study are evaluated. From all the experiments in this study, one model has achieved better classification performance

As discussed before from those experiments conducted in semi-supervised approach, the J48 decision tree algorithm with the 10-fold cross validation model gives a better classification accuracy of predicting newly arriving intrusions in their respective class category.

Some of the rules generated from the selected model are the following.

Rule 1: If protocol_type=tcp and rerror_rate='(-inf-0.1]' and logged_in = '(-inf-0.5]' and flag = SF and Duration = '(-inf-0.5]' and num_failed_logins ='(-inf-0.5]' and src_bytes = '(3-6.5]' then normal (the intrusion is normal traffic)

Rule 2: Ifprotocol_type=udpandsame_srv_rate='(-inf-0.005]'and dst_host_same_src_port_rate = '(-inf-0.965]' then probe (the attack type is probe)

Rule 3: If protocol_type=icmp and service = telnet or http or private or domain_u or smtp or finger or ftp or pop_3 or X11 or ftp_data then DOS (the attack type is DOS)

Rule 4: If protocol_type=icmp and service = ecr_i and src_bytes ='(27.5-38.5]'and dst_host_count = '(-inf-5.5]' then normal (the intrusion is normal traffic)

Rule 5: If protocol_type=tcp and rerror_rate='(-inf-0.1]' and logged_in = '(-inf-0.5]' and flag = SF and Duration = '(-inf-0.5]' and Duration = '(-inf-0.5]' and num_failed_logins ='(0.5- inf)' and dst_host_diff_srv_rate = '(-inf-0.005]' then R2L (the attack type is R2L)

Rule 6: If protocol_type=udp and same_srv_rate = '(0.995-inf)'and service = telnet or http then R2L (the attack type is R2L)

Rule 7: If protocol_type=tcp and rerror_rate='(-inf-0.1]' and logged_in = '(-inf-0.5]' and flag = SF and Duration = '(-inf-0.5]' and num_failed_logins ='(-inf-0.5]'and dst_bytes = '(36.5-41.5]'then U2R (the attack type is U2R)

Rule 8: If protocol_type=icmp and service = eco_i and dst_host_srv_count = '(-inf-1.5]' then U2R (the attack type is U2R)

Rule 9: If protocol_type=tcp and rerror_rate='(-inf-0.1]' and logged_in = '(-inf-0.5]' and flag = SF and Duration = '(-inf-0.5]' and num_failed_logins = '(-inf-0.5]' and src_bytes = '(-inf- 0.5]' then U2R (the attack type is U2R)

Rule 10: If protocol_type=udp and same_srv_rate = '(0.995-inf)'and service = domain_u and dst_host_count = '(-inf-51.5]' then normal (the intrusion is normal traffic)

Rule 11: If protocol_type=tcp and rerror_rate='(-inf-0.1]' and logged_in = '(-inf-0.5]' and flag = SF and Duration = '(- inf-0.5]' and num_failed_logins ='(- inf-0.5]'and dst_bytes = '(34.5-35.5]'then U2R (the attack type is U2R)

Rule 12: If protocol_type=icmp and service = ecr_i and src_bytes = '(-inf-27.5]' then DOS (the attack type is DOS)

Rule 13: If protocol_type=tcp and rerror_rate='(-inf-0.1]' and logged_in = '(-inf-0.5]' and flag = SF and Duration = '(- inf-0.5]'and Duration = '(-inf-0.5]' and num_failed_logins ='(inf- 0.5]' and src_bytes ='(6.5-11.5]' and dst_bytes= '(-inf-16]' then normal (the intrusion is normal traffic)

Rule 14: If protocol_type=tcp and rerror_rate='(-inf-0.1]' and logged_in = '(-inf-0.5]' and flag = SF and Duration= '(-inf-0.5]' and Duration = '(-inf-0.5]' and num_failed_logins ='(-inf-0.5]' and src_bytes = '(6.5-11.5]' and ' dst_bytes = '(16-34.5]'then R2L (the attack type is R2L)

Rule 15: If protocol_type=udp and same_srv_rate = '(0.005-0.19]' and count = '(-inf- 44.5]' then U2R (the attack type is U2R)

Rule 16: If protocol_type=tcp and rerror_rate='(-inf-0.1]' and logged_in = '(-inf-0.5]' and flag = SF and Duration = '(- inf-0.5]'and Duration = '(-inf-0.5]' and num_failed_logins ='(inf- 0.5]' and dst_bytes ='(35.5-36.5]'then normal (the intrusion is normal traffic)

Rule 17: If protocol_type=icmp and service = eco_i and dst_host_srv_count='(14.5-57.5]' then normal (the intrusion is normal traffic)

For the above generated sample rules, natural language version was prepared by the researcher and presented in appendix F. The natural language version of these rules can be easily understood by peoples.

As discussed in section 3.1 of Appendix A of this thesis, the attributes in KDD99 dataset can be divided into three groups. The first group of attributes is the basic features of network connection, which include the prototype, duration, service, number of bytes from source IP addresses (Src_bytes) or from destination IP addresses (dst_bytes), and some flags in TCP connections. In this study as shown the above generated rules, most of the attributes used for constructing the selected predictive model are from the basic features of network connection.

The second group of attributes in KDD99 is composed of the content features of network connections. From the second group of attributes in this study logged_in, num_failed_logins and num_compromised attributes are mostly used in the construction of rules. The third group is composed of the statistical features that are computed either by a time window or a window of certain kind of connections. The third groups of attributes are like rerror_rate, dst_host_srv_count and count are selected for generating rules in this study.

When checking the rules generated from the J48 decision tree shows that more than 17 attributes are used for generating the tree from among 41 variables. . Some of these are: logged_in, dst_bytes, protocol_type, diff_srv_rate, srv_diff_host_rate, src_bytes, falg, service, dst_host_srv, and num_compromise. From these generated attributes Protocol_type is the most determinant variable.

For checking the rules and other related issues the researcher has consulted Ethiopian Information Network Security Agency (INSA) experts. The domain experts agreed with the

most determinant attributes which are selected during the design of intrusion detection model in this study. The experts said that most of the time features like protocol_type, src_bytes, service, dst_host_same_src_port_rate, dst_host_srv_count, wrong_fragment, flag should have to be check in order to say a given network packet is either normal traffic or attack. As the domain experts said that most of the time the attack is DOS if the protocol_type is icmp (Internet Control Message protocol) and the only additional attribute is service with any value.

In the INSA firewall, the configured protocols are including Transmission Control Protocol (TCP), UDP (User Datagram Protocol), ICMP (Internet Control Message Protocol), IPSec (Internet Protocol Security), and Point to Point Transmission Protocol (PPTP) and Layer Two Transmission Protocol (L2TP). But on the dataset which is taken from the MIT Lincoln lab did not considered IPSec, PPTP and L2TP protocols.

From the above sample generated rules some of the rules are prevailing (that is known rules) and some of them are interesting rules (that is new rules). For deciding theses rules, the researcher consulted the domain expert from Ethiopian INSA. Also, the researcher referred the Cisco IOS firewall IDS (Cisco, 2012) and Microsoft Internet Security Exchange (ISA) Server (Microsoft, 2011) default security features. Rule 1, 3, 4, 7, 10, 12 and 15 are prevailing rules while the rest are interesting rules.

In INSA as the domain experts said there is no a data mining intrusion detection system. They have used firewalls as a network security tool for safe guarding their networks. Most of the protocols which are selected for the construction of the selected model for this study are considered in the INSA firewall router. Those protocols are enabled for the classification purpose of a given network traffic on the router which firewall configured.

The quality of IDS measures by its effectiveness, adaptability and extensibility features (Karen et al., 2007). As the domain experts from INSA said there is no one perfect firewall. As the domain experts said the software and hardware firewalls which are available in the market currently lucks effectiveness, adaptability and extensibility. So, DM based IDS which has effectiveness, adaptability and extensibility features will enhance the network security.

The selected model for this study is J48 decision tree algorithm with default value which scores the highest classification accuracy of 96.11%. This model is tested with 3,397 testing dataset and scored a prediction accuracy of 93.2%. Sample rules generated from the selected model's decision tree are presented in Appendix G.

The selected model for this study is validated by real life data. The real life data is with unlabeled classes. The prediction performance of this model is tested using a java code by using either Disk Operating System (DOS) or Simple Command line Interface (SCLI) on WEKA.

The command used for training purpose of the J48 algorithm is the following:

Java weka.classifiers.trees.J48 -C [ConfidenecFactor] -M 2 -t [PATH]

\Training_data_Name.arff -d [destination path]\Model_Name.model

>Java weka.classifiers.trees.j48 –C 0.25 -M 2 -t C:\Users\N.E.Tigabu\Desktop

\FinalDatasetsIDS.arff –d C: \Users\N.E. Tigabu\Desktop\selectedmodel.model

The command used for testing purpose of the J48 algorithm is the following:

Java weka.classifiers.trees.J48 –p [path]\Model_Name.model -T [destination path] \Testing_data_Name.arff

>Java weka.classifiers.trees.j48 –p:\Users\N.E.Tigabu\Desktop \selectedmodel.model -T C: \Users\N.E. Tigabu\Desktop\ real_life_data.arrf

The performance of the selected model is tested with the separately prepared 3,397 validating datasets as shown in the table 4.12.

Total number of instances (Validating tests)	Correctly classified Instances	Incorrectly Classified Instances
3,397	3,166 (93.2%)	231 (6.8%)

Table 4.12: Validating the selected model with real life

The performance of the model on the test set was 93.2% to classify the new instances as normal, DOS, U2R, R2L and probe classes. This result showed that out of the 3,397 testing datasets, the developed decision tree classification model predicted around 3,166 (93.2%) records correctly. The model has a prediction accuracy of 96.11% on the training datasets. There may different reasons why the performance of the model was lower during the testing phases. According to Shilendra and Preeti (2011) the errors committed by a classification model are generally divided into two types: training errors and generalization errors. Training error is the number of misclassification error committed on training records, whereas generalization error is the expected error of the model on previously unseen records. Here, the training error which is the FP rate was high for the training phase of the model in this study. The lower performance registered also due to lack of representative Samples in the model training.

The researcher has encountered different challenges during conducting this study. The first challenge was related to dataset. Each record of the dataset obtained from the MIT Lincoln lab, which includes both labeled and unlabeled records has 41 features. Most of the features were new and require long time to understand them. The second challenge was related to understanding the domain area. The researcher has tried to consult domain experts in the area of intrusion detection but most of the experts were not used the system in Ethiopia; rather they used firewall technologies for protecting attacks.

CHAPTER FIVE

CONCLUSIONS AND RECOMMENDATIONS

5.1. Conclusions

Computer networks are dynamic, growing, and continually evolving. As complexity grows, it becomes harder to effectively communicate to human decision-makers the results of methods and metrics for monitoring networks, classifying traffic, and identifying malicious or abnormal events. Network administrators and security analysts require tools that help them understand reason for, and make decisions about the information their analytic systems produce.

Because of the dynamic change of the technology and number of hackers and crackers in networking industry are in an increasing manner there should be a means to minimize or remove such challenges. And DM is one of the technologies that is used for intrusion detection and prevention

In this study, attempts have been made to use DM technology with the aim of detecting and predicting intrusions in the networking industry. The KDD process model designed by Fayyad et al. (1996) has been followed during undertaking the experimentation and discussion. The dataset used in this study has been taken from MIT Lincoln lab. After taking the data, it has been preprocessed and prepared in a format suitable for the DM tasks.

This study proposed the semi supervised approach for IDS. The proposed model will offer the advantage of considering those unlabeled records. In this case filling only the top few most confident data points and make it empty the class of rest records. Semi-supervised learning is more suitable for intrusion detection because they require a small quantity of labeled data while still taking advantage of the large quantities of unlabeled data.

The number of features required is another major concern in processing the dataset as well in this study. According to Kok-Chin et al (2009) as the number of features increase, the relationships among the features as well as the relationships between features and classes

will become very complex. High computational cost will inevitably be needed in processing such complex relationships. It is thus necessary to undergo a feature selection stage to obtain an optimal feature set with less number of features but able to provide high intrusion detection accuracies.

Both J48 decision tree algorithm and the Naïve Bayes simple algorithm have been tested as a classification approach for building a predictive model to intrusion detection. By changing the training test options and the default parameter values of these algorithms, different models have been created. The model created using 10-fold cross validation using the J48 decision tree algorithm with the default parameter values showed the best prediction accuracy. The model has a prediction accuracy of 96.11% on the training datasets. This model is then validated with 3,397 testing dataset and scored a prediction accuracy of 93.2% for classifying new instances of datasets as normal, DOS, U2R, R2L and probe classes.

In summary, the results from this study can contribute towards in improving the networking security. The study has shown that it is promising to identify those network intrusions either normal or attacks (DOS, U2R, Probe and R2L) and put forward tangible mechanisms for detecting and preventing them using the appropriate Data mining approaches.

5.2. Recommendations

The result of the study has shown that the J48 decision tree algorithm with cross-validation test mode and other default values is appropriate in the area of intrusion detection. Hence, based on the findings of this study, the following are recommended as future research directions:

- ✓ The Network Intrusion predictive model, which is developed in this study, generated various patterns and rules. To use this model effectively in the real world Network Security environment, designing a knowledge base system which will add adaptability and extensibility features of the IDS and connect to DM model is one of the future research directions.
- ✓ Constructing an IDS which will have both high intrusion detection (that is true positive) rate and low false alarm (that is false positive) rate.
- ✓ To use the selected models there is a need to visualize the patterns, as visualization methods enhance network intrusion detection and anomaly detection. Information visualization techniques help network administrators and security analysts to quickly recognize patterns and anomalies; visually integrate heterogeneous data sources; and provide context for critical events. Information visualization and visual analytics hold great promise for making the information accessible, usable, and actionable by taking advantage of the human perceptual abilities. Visualization methods are also employed in the classification of network traffic and its analysis. So, designing and integrating computer network visualization and visual analytics with the predictive intrusion detection model is one of the future research directions.
- ✓ This study is conducted on the dataset taken from the MIT Lincoln lab. Future research should be conducted on real life dataset from organizations that have their own network by combining the problem domain and the domain expert on the study process.
- ✓ This study was carried out using clustering technique of simple K-means and classification algorithms such as J48 decision tree and Naïve Bayes algorithms. So further investigation needs to be done using other classification algorithms such as Neural Networks and Support Vector Machine plus using association rule discovery.

REFERENCES

Adamu T., "Computer Network Intrusion Detection: Machine Learning Approach," M.Sc Thesis, School of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2010

Adem K. and Julio P., "Data Mining and Knowledge Discovery in Real Life Applications", In-Teh is Croatian branch of I-Tech Education and Publishing KG, Vienna, Austria, February 2009

Agrawal S., Mannila N., Srikant Y., Toivonen M. and Verkamo L., "Fast Discovery of Association Rules, In Advances in Knowledge Discovery and Data Mining", American Association for Artificial Intelligence Press, PP.307-328, , California ,1996

Alan B., Chandrika P., Rasheda S. and Boleslaw S., "Network-Based Intrusion Detection Using Neural Networks", in Proceedings of the Intelligent Engineering Systems Through Artificial Neural Networks, St.Louis, Vol. 12, PP. 579-584, ASME Press, New York, 2002

Amir A., Ahmad H. and Hadi B., "A New System for Clustering and Classification of Intrusion Detection System Alerts Using SOM", International Journal of Computer Science and Security, Vol. 4, No. 6, PP. 589-597, 2011

Anand S., Patrick A., Hughes J., and Bell D., "Data Mining Methodology for Cross-sales", Knowledge Based Systems Journal., Vol.10, PP.449–461, 1998

Anderson D., "Netx Generation Intrusion Detection Expert System, (NIDES)," Software Design, product specification and version description document, project 3131, SRI International, 1994

Anderson J., "Computer Security Threat Monitoring and Surveillance", Technical Report, Washington, 1980

Anirut S. and Nualsawat H., "Euclidean-based Feature Selection for Network Intrusion Detection", International Conference on Machine Learning and Computing, IACSIT Press, Vol.3, Singapore, 2011

Barbar´a D., Wu N., and Jajodia S., “Detecting novel network intrusions using Bayes estimators”, In Proc. of the First SIAM Int. Conf. on Data Mining (SDM 2001), Chicago. Society for Industrial and Applied Mathematics (SIAM), 2001

Berkhin P., “Survey of clustering data mining techniques”, Tech. rep., Accrue Software, San Jose, CA, 2002

Berry M., “Database Marketing”, Business Week, Vol.5, PP. 56-62, California, 1994

Berson A., Smith S. and Thearling K., “Building Data Mining Applications for CRM”, McGraw-Hill Professional Publishing, New York, USA, 2000

Blum A. and Rivest L., “Training 3-node neural networks is NP-complete”, Neural Networks, Issue 5, PP. 117 – 127, 1992

Brachman R. and Anand T., “The process of knowledge discovery in databases”, PP. 37–57, 1996

Bro V., “System for detecting network intruders in real-time”, In Proceedings of the 7th USENIX Security Symposium, San Antonio, TX, 1998

Cabena P., Hadjinian P., Stadler R., Verhees J., and Zanasi A., “Discovering Data Mining: From Concepts to Implementation”, Prentice Hall, 1998

Chang L., Arnold P. and Prajwal M, “Exploiting Efficient Data Mining Techniques to Enhance Intrusion Detection Systems”, Department of Computer Science, Virginia Polytechnic Institute and State University, IEEE, PP. 512-517, 2005

Chapman P., Clinton J., Kerber R., Khabaza T., Reinartz T., Shearer C., and Wirth R., “CRISPDM 1.0 step-by-step data mining guide”, Technical report, CRISP-DM, 2003

Carl F., “Intrusion Detection and Prevention”, McGraw-Hill, Osborne Media, 2003

Cisco, “Configuring Cisco IOS Firewall Intrusion Detection System”, Cisco Security Guide, USA, 2012

Charles E., “The foundations of cost-sensitive learning”. In Proceedings of the Seventeenth International Joint Conference of Artificial Intelligence, Morgan Kaufmann, Seattle, Washington, PP. 973–978, 2001

Crothers M., “Implementing Intrusion Detection Systems”, a Hands-On Guide for Securing the Network, USA, 2002

Chaudhuri S. “Data Mining and Database Systems: Where is the Intersection?” IEEE Bulletin of the Technical Committee on Data Engineering, Vol. 21, No.1, PP. 4-8, 1998

Cheng J., Greiner R., Kelly J., Bell D., and Liu W., “Learning Bayesian networks from data: An information-theory based approach”, Artificial Intelligence 137, PP. 43–90, 2002

Cheng S. “KNOWLEDGE DISCOVERY IN DATABASES: AN INFORMATION RETRIEVAL PERSPECTIVE”, Malaysian Journal of Computer Science, Vol. 13 .No. 2, pp. 54-63, Malaysia,December 2000.

Chi-Ho T., Sam K. and Hanli W., “Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection”, City University of Hong Kong, Pattern Recognition, Issue.40, PP. 2373 – 2391, 2007

Christos D. and Aikaterini M., “DDoS attacks and defense mechanisms: classification and state-of-the-art of Computer Networks”, the International Journal of Computer and Telecommunications Networking, Vol. 44, No.5 , PP. 643 - 666, 2004.

Cios K. and Kurgan L., “Trends in Data Mining and Knowledge Discovery”, Springer-Verlag, London, PP. 1–26, 2000

Dash M. and Liu H., “Feature selection for classification”, Intelligent Data Analysis: An International Journal, PP. 131–156, 1997

Denning D., “An Intrusion Detection Model”, Transactions on Software Engineering, IEEE Communication Magazine, SE-13, PP.222-232, 1987

Denning D., “Information Warfare and Security”, Addison Wesley, ACM Press, USA, 1999

Dewan M. and Mohammad Z., "Anomaly Network Intrusion Detection Based on Improved Self Adaptive Bayesian Algorithm", Journal of Computers, Vol.5, No.1, January, 2010.

Dewan M. and Mohammed Z., "Anomaly Network Intrusion Detection Based on Improved Self Adaptive Bayesian Algorithm", Journal of Computers, Vol.5, PP.23-31, Jan 2010

Doak J., "An evaluation of feature selection methods and their application to computer security", Technical report, Davis CA: University of California, Department of Computer Science, 1992.

Dowell C. and Ramstedt P., "The computer watch Data reduction Tool", Proc., 13th Natioanl Computer Security Conference, Washington, D.C., pp.99-108, October 1990

Duch T., Winiarski J., and Kachel A., "Feature Ranking Selection and Discretization", Int. Conf. on Artificial Neural Networks (ICANN) and Int. Conf. on Neural Information Processing (ICONIP), Istanbul, PP. 251-254, 2003

Dunham H., "Data mining introductory and advanced topics", Upper Saddle River, NJ: Pearson Education, Inc, 2003

Dy J. and Brodley C., "Feature subset selection and order identification for unsupervised learning", In Proceedings of the Seventeenth International Conference on Machine Learning, PP. 247–254, 2000

Eibe F. and Witten I., "Data Mining–Practical Machine Learning Tools and Techniques," 2nd Edition, Elsevier, 2005

Eric B., Alan D. and Christiansen W., "Data Mining for Network Intrusion Detection: How to Get Started?" The MITRE Corporation, 2002

Eric, "Data Mining Software Package"

http://eric.univ-lyon2.fr/~ricco/tanagra/en/tanagra.html, Accessed date: October 8, 2011].

Eskin E., Arnold A., Preraua M., Portnoy L., and Stolfo S., "A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data", Data Mining for Security Applications, Kluwer Academic Publishers, Boston, 2002

Farhan A., Zulkhairi M., Dahalin and Shaidah J., "Distributed and Cooperative Hierarchical Intrusion Detection on MANETs", International Journal of Computer Applications, Vol. 12, No.5, PP. 33-40, 2010

Fayyad U., Piatetsky G., and Smyth P.," The KDD process for Extracting Useful Knowledge from Volumes of Data", Communications of the ACM, Vol. 39, PP. 27-34, 1996.

Ferri C., Flach P. and Henrandez-Orallo J., "Learning Decision Trees Using Area under the ROC Curve", Proceedings of the 19th International Conference on Machine Learning, Morgan Kaufmann, PP. 139-146, 2002

Frank J., "Artificial intelligence and intrusion detection: Current and future directions", In Proc. of the 17th National Computer Security Conference, Baltimore, MD. National Institute of Standards and Technology (NIST), 1994

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

Heberlein L., Dias G., Levitt K., Mukherjiee B., Wood J., and Wolber D., "A Network Security Monitor," Proc., 1990 IEEE Symposium on Research in Security and Privacy, Oakland, CA, pp.196-304, 1990

Helali M., "Data Mining Based Network Intrusion Detection System", A Survey in Novel Algorithms and Techniques in Telecommunications and Networking, PP. 501-505, 2010

Hershkop S., Apap F., Eli G., Tania D., Eskin E., Stolfo S., "A data mining approach to host based intrusion detection. Technical reports, CUCS Technical Report, 2007

Hochberg J., "NADIR: An Automated System for Detecting Network Intrusion and Misuse," Computers and Security, Vol.12, no.3, PP.235-248, 1993

Howard C., Debuse J.,and Rayward-Smith V, "Building the KDD Roadmap: A Methodology for Knowledge Discovery, chapter of Industrial Knowledge Management", PP. 179–196, Springer-Verlag, 2001

Ilgun K, Kemmerer R., and Porras P., "State Transition Analysis: A Rule-Based Intrusion Detection Approach," IEEE Transactions on Software Engineering, Vol.21, No.3, pp. 181-199, 1995.

Ilgun K., "USTAT: A Real-Time Intrusion Detection System for UNIX," Proc., IEEE Symposium on Research in Security and Privacy, Oakland, CA, pp. 16-28, 1993

Jake R., Meng L., Miikkulainen R., "Intrusion Detection with Neural Networks", Advances in Neural Information Processing System 10, Cambridge, MA, MIT Press, 1998

Jensen F., "An Introduction to Bayesian Networks", Springer, 1996

John R., Langley K. and Pfleger G., "Irrelevant features and the subset selection problem", International Conference on Machine Learning, Morgan Kaufman, San Francisco, PP. 121-129, 1994

Johnson R., "Applied Multivariate Statistical Analysis", Prentice Hall, PP. 356-395, 2002

Karen S. and Peter M., "Guide to Intrusion Detection and Prevention Systems", National Institute of Standards and Technology, Department of Commerce, USA, 2007

KDD Cup Intrusion detection dataset Competition
http://kdd.ics.uci.edu/databases/kddcup99/kddcup9 9.html), accessed date January 17, 2012

KdNuggets, "Data Mining Methodology",
http://www.kdnuggets.com/polls/2007/data_mining_methodology.htm,
Accessed date February 03, 2012

Kohavi R. and John G., "Wrappers for feature subset selection", Artificial Intelligence, PP. 273–324, 1997

Kok-Chin K., Choo-Yee T. and Somnuk-Phon A., "From Feature Selection to Building of Bayesian Classifiers: A Network Intrusion Detection Perspective", American Journal of Applied Sciences, Vol.11, No.6, PP. 1948-1959, 2009

Kruegel C. and Toth T., "Using Decision Tree to Improve Signature Based Intrusion Detection", 6thSymposium on Recent Advances in Intrusion Detection (REID), Lecture Notes in Computer Science, Spring Verlag, USA, September 2003.

Lane T., "Machine Learning Techniques for the computer security domain of anomaly detection", PhD thesis, Purdue Univ., West Lafayette, 2000

Lanner G., "Witness Miner On-line Help System", http://www.witnessminer.com, accessed date January 27, 2012.

Lee W. , Stolfo S., and Mok K., "A Data Mining Framework for Building Intrusion Detection Model", In Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, PP. 120-132, 1999.

Liu H. and Motoda H., "Feature Selection for Knowledge Discovery and Data Mining", Boston: Kluwer Academic Publishers, 1998

Lunt T., "IDES: A progress Report," Proc., 6th Annual Computer Security Applications Conference, Tucson, AZ, 1990

Mahbod T., Ebrahim B., Wei L., and Ali A., "A detailed analysis of the KDD CPU 99 Dataset", proceedings of 2009 of the IEEE Symposium on computational Intelligence in Security and Defense Applications, National Research Council, PP.1-6, Canada, 2009

Manago K., and Auriol S., "Mining for OR.ORMS Today (Special Issue on Data Mining)", American Association for Artificial Intelligence Press, PP. 28-32, California, 1996

Mannila, Toivoenn and Verkamo, "Discovering Frequent Episodes in Sequences", In proceedings of the First International Conference on Knowledge Discovery and Data Mining", American Association for Artificial Intelligence Press, PP. 201-215, 1995

Marin J., Ragsdale D. and Surdu J., “A hybrid approach to profile creation and intrusion detection”, In Proc. of DARPA Information Survivability Conference and Exposition, Anaheim, CA. IEEE Computer Society, 2001

Martínez d., “Optimización Mediante Técnicas de Minería de Datos Del Ciclo de Recocido de Una Línea de Galvanizado, PhD thesis, Universidad de LaRioja, 2003

Meera G., Gandhi and Srivatsa S, “Adaptive Machine Learning Algorithm (AMLA) Using J48 Classifier for an NIDS Environment”, Advances in Computational Sciences and Technology, Vol. 3, PP. 291–304, 2010

Microsoft, “Configuring ISA Hardware and Software on windows NT/2000/Win-7”, Microsoft Internet Security Exchange (ISA) Server 2010, USA, 20111

Miller A., “Subset Selection in Regression”, Chapman and Hall/CRC, 2 edn, 2002

Mrutyunjaya P., “Evaluating Machine Learning Algorithms for Detecting Network Intrusions”, International Journal of Recent Trends in Engineering, Vol. 1, No 1, PP. 472-477, 2009

Mukherjee B., Todd L., and Karl N., “Network Intrusion Detection,” IEEE, Vol.12, No.4, PP. 132-143, 1994

Mukkamala H. and Sung A., “Comparative study of techniques for intrusion detection”, Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence (ICTAI’03), 2003

Nanda A, “Data Mining and Knowledge Discovery in Database: An AI perspective”, Proceedings of national Seminar on Future Trends in Data Mining, 2010

Norouzian R. and Merati S., “Classifying Attacks in a Network Intrusion Detection System Based on Artificial Neural Networks”, in the Proceedings of 13th International Conference on Advanced Communication Technology(ICACT), 2011.

Pachghare V., Vaibhav K., Parag K., “Performance Analysis of Supervised Intrusion Detection System”, IJCA Special Issue on Network Security and Cryptography, NSC, 2011

Piatetsky S. and Frawley W., “Knowledge Discovery in Databases”, AAAI/ MIT Press, MA, 1991

PiatetskyG., “An overview of knowledge discovery in databases: Recent progress and challenges Rough Se ts, Fuzzy Sets and Knowledge Discovery”, PP.1–11, 1994

Pradeep S. “Comparing the Effectiveness of Machine Learning Algorithms for Defect Prediction“, International Journal of Information Technology and Knowledge Management, Vol.2, No.2, PP.481-483, 2005

Pressman R., “Software Engineering: A Practitioner’s Approach”, McGraw-Hill, New York, 2005

Ron K., “A study of cross-validation and bootstrap for accuracy estimation and model selection”, Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, PP. 1137-1143, 1995

Roshan R., “Difference between KDD and Data mining”, http://www.differencebetween.com/difference-between-kdd-and-vs-data-mining accessed date October 29, 2011

Sadiq A., “Rule-Based Network Intrusion Detection Using Genetic Algorithm”, International Journal of Computer Applications, No.8, 2011

SAS Institute, “SEMMA data mining methodology”, http://www.sas.com, accessed date January 29, 2012

Sathyabama S., Irfan A. and Saravanan A., “Network Intrusion Detection Using Clustering: A Data Mining Approach”, International Journal of Computer Application, Vol. 30, No. 4, Sep-2011

Selvakani S. and Rajiesh R., “Genetic Algorithm for framing rules for intrusion detection”, IJCSNS International Journal of Computer Science and network Security, Vol.7, No.11, 2007

Shekhar R., Vir V. and Kiran S. "K-Means+ID3: A Novel Method for Supervised Anomaly Detection by Cascading K-Means Clustering and ID3 Decision Tree Learning Methods," IEEE Transactions on Knowledge and Data Engineering, Vol.19, No.3, PP. 345-354, 2007

Shilendra K. and Preeti J., "Effective Anomaly Based Intrusion Detection Using Rough Set Theory and Support Vector Machine", Vol. 18, No. 3, 2011

Snapp S., Brentano J., Dias G., Goan T., Mukherjiee B., Teal D, and Mansur D., "Distributed Intrusion Detection System- Motivation, Architecture, and An early Prototype", Proceedings of the 14th National Computer Security Conference, Washington, D.C., PP. 167-176, 1991.

Solarte J., "A proposed data mining methodology and its application to industrial engineering", M.Sc thesis, University of Tennessee, Knoxville, 2002

Srinivas M., Andrew H. and Abraham A., "Intrusion Detection Using an Ensemble of Intelligent Paradigms", Journal of Network and Computer Applications, PP.1-15, 2004

Srinivasu P., Avadhani P., Vishal K., Prudhvi R., "Approaches and Data Processing Techniques for Intrusion Detection Systems", Vol. 9, No. 12, pp. 181-186, 2009

Sterry B., "Data Mining Methods for Network Intrusion Detection," University of California, 2004

Taeshik S. and Jong S., "A Hybrid Machine Learning Approach to Network Anomaly Detection", Information Sciences, USENIX Association, Vol. 177, No.18, PP. 3799-3821, 2007

Thair P., "Survey of Classification Techniques in Data Mining", Proceedings of the International Multi-Conference of Engineers and Computer Scientists, Hong Kong, 2009.

Two Crows Corp, "Introduction to Data Mining and Knowledge Discovery", 3rd edn, USA, 1999

Two Crows Corporation, "Introduction to Data Mining and Knowledge Discover", 3rd edition, MD 20854(USA), 2005

William W., “Fast effective rule induction”, In Internal conference on Machine learning, IEEE, PP. 115-123, 1995

Yang Q., and Wu X., “10 Challenging Problems in Data Mining Research”, International Journal of Information Technology and Decision Making, World Scientific Publishing Company , Vol. 5, No. 4, PP.597–604., 2006

Yeung D. and Ding Y., “Host-Based Intrusion Detection Using Dynamic and Static Behavioral Models”, Journal of Pattern Recognition, Vol. 36, PP. 229-243, 2003

Zewdie M., “Optimal feature selection for Network Intrusion Detection: A Data Mining Approch,” M.Sc thesis, School of Information Science, Addis Ababa University, Addis Ababa, Ethiopia, 2011.

APPENDIXES

Appendix A: Description of features

#	Features	Description	Type
	Basic features of network connection (source IP addresses or from destination IP addresses)		
1	Duration	Duration of the connection.	Cont.
2	Protocol type	Connection protocol (e.g. tcp, udp)	Disc.
3	Service	Destination service (e.g. telnet,ftp)	Disc.
4	Flag	Status flag of the connection	Disc.
5	Source bytes	Bytes sent from source to destination	Cont.
6	Destination bytes	Bytes sent from destination to source	Cont.
7	Land	1 if connection is from/to the same host/port; 0 otherwise	Disc.
8	Wrong_fragment	number of wrong fragments	Cont.
9	urgent	number of urgent packets	Cont.
content features of network connections suggested by domain knowledge			
10	hot	number of "hot" indicators	Cont.
11	failed logins	number of failed logins	Cont.
12	logged in	1 if successfully logged in; 0 otherwise	Disc.
13	# compromised	number of "compromised'' condition	Cont.
14	root shell	1 if root shell is obtained; 0 otherwise	Cont.
15	su attempted	1 if "su root'' command attempted; 0 otherwise	Cont.
16	# root	number of "root'' accesses	Cont.
17	# file creations	number of file creation operations	Cont
18	# shells	number of shell prompts	Cont.
19	# access files	number of operations on access control files	Cont.
20	# outbound cmds	number of outbound commands in an ftp session	Cont.
21	is hot login	1 if the login belongs to the "hot'' list; 0 otherwise	Disc.
22	is guest login	1 if the login is a "guest'' login; 0 otherwise	Disc.
Statistical traffic features computed using a two-second			
23	Count	number of connections to the same host as the current connection in the past two seconds	Cont.
24	srv count	number of connections to the same service as the current connection in the past two seconds	Cont.
25	serror rate	% of connections that have “SYN'' errors	Cont.
26	srv serror rate	% of connections that have “SYN'' errors	Cont.
27	rerror rate	% of connections that have “REJ'' errors	Cont.
28	srv rerror rate	% of connections that have “REJ'' errors	Cont.

29	same srv rate	% of connections to the same service	Cont.
30	diff srv rate	% of connections to different services	Cont.
31	srv diff host rate	% of connections to different hosts	Cont.
32	dst host count	count of connections having the same destination host	Cont.
33	dst host srv count	count of connections having the same destination host and using the same service	Cont.
34	dst host same srv rate	% of connections having the same destination host and using the same service	Cont.
35	dst host diff srv rate	% of different services on the current host	Cont.
36	dst host same src port rate	% of connections to the current host having the same src port	Cont.
37	dst host srv diff host rate	% of connections to the same service coming from different hosts	Cont.
38	dst host serror rate	% of connections to the current host that have an S0 error	Cont.
39	dst host srv serror rate	% of connections to the current host and specified service that have an S0 error	Cont.
40	dst host rerror rate	% of connections to the current host that have an RST error	Cont.
41	dst host srv rerror rate	% of connections to the current host and specified service that have an RST error	Cont.

Appendix B: Attribute's Header and Data Declaration

@relation 'NSL-KDD'

@attribute 'duration' real

@attribute 'protocol_type' {'tcp','udp', 'icmp'}

@attribute 'service' {'aol', 'auth', 'bgp', 'courier', 'csnet_ns', 'ctf', 'daytime', 'discard', 'domain', 'domain_u', 'echo', 'eco_i', 'ecr_i', 'efs', 'exec', 'finger', 'ftp', 'ftp_data', 'gopher', 'harvest', 'hostnames', 'http', 'http_2784', 'http_443', 'http_8001', 'imap4', 'IRC', 'iso_tsap', 'klogin', 'kshell', 'ldap', 'link', 'login', 'mtp', 'name', 'netbios_dgm', 'netbios_ns', 'netbios_ssn', 'netstat', 'nnsp', 'nntp','ntp_u', 'other', 'pm_dump', 'pop_2', 'pop_3', 'printer', 'private', 'red_i', 'remote_job', 'rje', 'shell','smtp', 'sql_net', 'ssh', 'sunrpc', 'supdup', 'systat', 'telnet', 'tftp_u', 'tim_i', 'time', 'urh_i', 'urp_i','uucp', 'uucp_path', 'vmnet', 'whois', 'X11', 'Z39_50', 'icmp'}

@attribute 'flag' { 'OTH', 'REJ', 'RSTO', 'RSTOS0', 'RSTR', 'S0', 'S1', 'S2', 'S3', 'SF', 'SH' }

@attribute 'src_bytes' real

@attribute 'dst_bytes' real

@attribute 'land' {'0', '1'}

@attribute 'wrong- fragment' real

@attribute 'urgent' real

@attribute 'hot' real

@attribute 'num_failed_logins' real

@attribute 'logged_in' {'0', '1'}

@attribute 'num_compromised' real

@attribute 'root_shell' real

@attribute 'su_attempted' real

@attribute 'num_root' real

@attribute 'num_file_creations' real

@attribute 'num_shells' real

@attribute 'num_access_files' real

@attribute 'num_outbound_cmds' real

@attribute 'is_host_login' {'0', '1'}

@attribute 'is_guest_login' {'0', '1'}

@attribute 'count' real

@attribute 'srv_count' real

@attribute 'serror_rate' real

@attribute 'srv_serror_rate' real

@attribute 'rerror_rate' real

@attribute 'srv_rerror_rate' real

@attribute 'same_srv_rate' real

@attribute 'diff_srv_rate' real

@attribute 'srv_diff_host_rate' real

@attribute 'dst_host_count' real

@attribute 'dst_host_srv_count' real

@attribute 'dst_host_same_srv_rate' real

@attribute 'dst_host_diff_srv_rate' real

@attribute 'dst_host_same_src_port_rate' real

@attribute 'dst_host_srv_diff_host_rate' real

@attribute 'dst_host_serror_rate' real

@attribute 'dst_host_srv_serror_rate' real

@attribute 'dst_host_rerror_rate' real

@attribute 'dst_host_srv_rerror_rate' real

@attribute 'class' {'normal.', 'Probe', 'DOS', 'U2R', 'R2L'}

Sample data declaration for supervised learning

@data

0,tcp,http,SF,429,9556,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,5,0,0,0,0,1,0,0,255,255,1,0,0,0,0,0,0,0,normal

0,tcp,http,SF,326,2229,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,15,15,0,0,0,0,1,0,0,255,255,1,0,0,0,0,0,0,0,normal

0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0,0,0,0,1,0,0,255,255,1,0,1,0,0,0,0,0,DOS

0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0,0,0,0,1,0,0,255,255,1,0,1,0,0,0,0,0,DOS

0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0,0,0,0,1,0,0,255,254,1,0.01,0,0,0,0,0,0,R2L

0,udp,other,SF,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,1,0,0,0,0,0.2,1,0,255,1,0,0.44,1,0,0,0,0,0,Probe

0,tcp,other,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,1,1,0,0,255,14,0.05,0.02,0,0,0,0,0.05,1,U2R

Sample data declaration for Semi-Supervised learning

@data

0,udp,private,SF,105,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,3,0,0,0,0,1,0,0,255,255,1,0,0,0,0,0,0,0,normal

0,udp,private,SF,105,105,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,1,0,0,255,255,1,0,0,0,0,0,0,0,R2L

0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,4,0,0,0,0,1,0,0,255,255,1,0,0.01,0,0,0,0,0, ?

0,tcp,telnet,REJ,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4,9,0,0,1,0.89,0.25,1,0.89,175,84,0.33,0.03,0.01,0.02,0,0,0.89,0
.69,Probe

1,tcp,pop_3,RSTO,0,36,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,9,0,0,1,1,0.5,1,1,255,41,0.16,0.02,0,0,0,0,0.6,1,?

0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0,0,0,0,1,0,0,255,255,1,0,1,0,0,0,0,0,DOS

0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,0,0,0,0,1,0,0,255,255,1,0,1,0,0,0,0,0, ?

Appendix C: Resulting confusion matrix and accuracy using J48 algorithm

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.95	0.01	0.985	0.96	0.965	0.992	Normal
	0.98	0.02	0.878	0.98	0.927	0.987	R2L
	1	0	0.999	1	1	1	DOS
	0.97	0.01	0.93	0.97	0.949	0.996	Probe
	0.53	0.01	0.639	0.53	0.58	0.963	U2R
Weighted Average	0.96	0.008	0.962	0.96	0.961	0.994	

Table C.1: Detailed Accuracy by Class using Semi-supervised J48 algorithm parameters with their default values- 10 fold cross validation

Classified as	Normal	R2L	DOS	U2R	Probe	Sum
Normal	7990	363	4	91	6	8454
R2L	13	3167	1	40	4	3225
DOS	2	0	7933	0	0	7935
Probe	1	0	0	42	1300	1343
U2R	105	77	0	306	88	576
Sum	8111	3607	7938	476	1398	21533

Table C.2: Confusion Matrix using J48 algorithm parameters with their default values-10 fold cross validation

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	class
	0.95	0.01	0.981	0.95	0.962	0.991	Normal
	0.98	0.02	0.881	0.98	0.929	0.988	R2L
	1	0.00	9.998	1	0.999	0.999	DOS
	0.44	0.00	0692	0.44	0.538	0.926	U2R
	0.97	0.01	0.89	0.97	0.93	0.997	Probe
Weighted Average	0.96	0.008	0.959	0.96	0.958	0.992	

Table C.3: Detailed Accuracy by Class using J48 algorithm parameters with percentage-split set to 75%

Classified as	Normal	R2L	DOS	U2R	Probe	Sum
Normal	1975	90	3	14	9	2091
R2L	6	789	0	5	2	802
DOS	0	0	2006	0	0	2006
U2R	32	17	1	63	30	143
Probe	0	0	0	9	332	341
Sum	2013	896	2010	91	373	5383

Table C.4: Confusion Matrix using J48 algorithm parameters with percentage-split set to 75%

Appendix D: Confusion matrix and accuracy using Naïve Bayes algorithm using all features

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.92	0.00	0.999	0.93	0.961	0.994	Normal
	0.92	0.02	0.893	0.92	0.906	0.991	R2L
	1	0	1	1	0.999	1	DOS
	0.93	0.03	0.432	0.93	0.59	0.985	U2R
	0.87	0.00	0.964	0.87	0.913	0.999	probe
Weighted Average	0.95	0.004	0.966	0.95	0.954	0.996	

Table D.1: Detailed Accuracy by Class Naïve Bayes algorithm with 10 Fold cross Validation.

Classified as	Normal	R2L	DOS	U2R	Probe	Sum
Normal	7828	353	0	267	6	8454
R2L	6	2699	0	249	4	2958
DOS	0	0	8191	11	0	8202
U2R	4	2	0	537	33	576
probe	0	0	0	180	1163	1343
Sum	7838	3054	7924	1244	1206	

Table D.2: Confusion Matrix by Naïve Bayes algorithm parameters with 10 Fold cross Validation

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	class
	0.93	0	0.999	0.93	0.963	0.994	Normal
	0.91	0.02	0.892	0.91	0.901	0.992	R2L
	1	0	1	1	1	1	DOS
	0.91	0.03	0.442	0.91	0.576	0.982	U2R
	0.85	0.00	0.935	0.85	0.889	0.999	Probe
Weighted Average	0.95	0.004	0.964	0.95	0.952	0.996	

Table D.3: Detailed Accuracy by Class using Naïve Bayes with percentage-split set to 75%

Classified as	Normal	R2L	DOS	U2R	Probe	Sum
Normal	1943	88	0	54	6	2091
R2L	1	730	0	70	1	802
DOS	0	0	2004	2	0	2006
U2R	0	0	0	130	13	143
Probe	10	0	0	42	289	341
Sum	1954	818	2004	298	309	5383

Table D.4 Confusion Matrix using Naïve Bayes with percentage-split set to 75%

Appendix E: Resulting confusion matrix and accuracy using Naïve Bayes algorithm with feature selection

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.92	0.00	0.995	0.926	0.959	0.99	Normal
	0.92	0.02	0.886	0.923	0.904	0.976	R2L
	1	0	1	0.995	0.997	1	DOS
	0.35	0.02	0.4	0.352	0.375	0.971	U2R
	1	0.03	0.706	0.999	0.827	0.997	Probe
WA	0.94	0.006	0.946	0.94	0.941	0.992	

Table E.1: Detailed Accuracy by Class using Naïve Bayes with their default values and using cost sensitive-feature selection

Classified as	Normal	R2L	DOS	U2R	Probe	Sum
Normal	7831	362	0	170	91	8454
R2L	25	2976	0	110	114	3225
DOS	2	15	7922	23	0	7962
U2R	15	5	0	203	353	576
Probe	0	0	0	2	1314	1316
Sum	7873	3358	7922	505	1872	21,533

Table E.2: Confusion Matrix by Naïve Bayes simple algorithm parameters with their default values using u cost sensitive feature selection

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	class
	0.93	0.00	0.995	0.93	0.96	0.991	Normal
	0.91	0.02	0.883	0.91	0.898	0.973	R2L
	1	0	1	1	0.998	1	DOS
	0.35	0.02	0.385	0.35	0.366	0.952	U2R
	1	0.03	0.716	1	0.835	0.996	Probe
WA	0.94	0.006	0.946	0.94	0.941	0.991	

Table E.3: Detailed Accuracy by Class using Naïve Bayes simple algorithm parameters with percentage-split set to 75%

Classified as	Normal	R2L	DOS	U2R	Probe	Sum
Normal	1941	89	0	44	17	2091
R2L	7	732	0	32	31	802
DOS	0	5	1997	4	0	2006
U2R	3	3	0	50	87	143
Probe	0	0	0	0	341	341
Sum	1951	829	1997	130	476	5383

Table E.4: Confusion Matrix using Naïve Bayes simple algorithm parameters with percentage-split set to 75%

Appendix F: Natural Language version for sample rules

Rule 1: This rule means for a given network packet with its attribute and values like protocol type is transmission control Protocol (TCP), rerror_rate which means the percentage of connections that have "REJ" errors is less than or equal to 0.1 percent, logged_in which means either successfully logged or not in to the network is less than or equal to 0.5 seconds, the status of the flag is source flag (SF), the length (number of seconds) of the connection which means duration is less than or equal to 0.5 seconds ,the number of failed logins (num_failed_logins)is less than or equal to 0.5 logins and number of data bytes from source to destination which is the src_bytes is greater than 3 bytes and less than or equal to 6.5 bytes then the packet is normal network traffic.

Rule 2: This rule means for a given network packet with its attribute values like its protocol type is User Datagram protocol (UDP), percentage of connections to the same services which means same_srv_rate is less than or equal to 0.005 percent and percentage rate of connections to the current hosts having the same source port (dst_host_same_src_port_rate) is less than or equal to 0.965 percent then the network traffic is an attack which is Probe.

Rule 3: This rule means for a given network packet with its attributes like protocol type is internet control message protocol (ICMP), the network services on the destination (service) may be telnet or http or private or domain_u or smtp or finger or ftp or pop_3 or X11 or ftp_data or domain_u then the network traffic is an attack which is DOS.

Rule 4: This rule means for a given network packet with its attribute values like protocol type is ICMP, the network services on the destination is ecr_i, number of data bytes from source to destination which is the src_bytes is greater than 27.5 and less than or equal to 38.5 bytes and count of connections having the same destination host (dst_host_count) is less than or equal to 5.5 then the network traffic is normal.

Rule 5: This rule means for a given network packet if its attribute values like protocol type is TCP, rerror_rate which means the percentage of connections that have "REJ" errors is less than or equal to 0.1 percent, logged_in which means either successfully logged or not in to the network is less than or equal to 0.5 seconds, the status of the flag is SF, the length (number of seconds) of the connection which means duration is less than or equal to 0.5 seconds, num_failed_logins is less than or equal to 0.5 logins and the percentage of different services on the current host which is dst_host_diff_srv_rate is less than or equal to 0.005 percent then the network traffic is an attack which is R2L.

Rule 6: This rule means for a given network packet if its attribute values like protocol type is UDP, percentage of connections to the same services which means same_srv_rate is greater than 0.995 percent and the network services on the destination is either telnet or hyper text transfer protocol (http) then the network traffic is an attack which is R2L.

Rule 7: This rule means for a given network packet if its attribute values like protocol type is TCP, rerror_rate which means the percentage of connections that have "REJ" errors is less than or equal to 0.1 percent, logged_in which means either successfully logged or not in to the network is less than or equal to 0.5 seconds, the status of the flag is SF, the length (number of seconds) of the connection which means duration is less than or equal to 0.5 seconds, num_failed_logins is less than or equal to 0.5 logins and number of data bytes from destination to source which is dst_bytes greater than 36.5 and less than or equal to 41.5 bytes then the network traffic is an attack which is U2R.

Rule 8: This rule means for a given network packet if its attribute values like protocol type is ICMP, the network services on the destination is eco_i and count of connections having the same destination host and using the same service (dst_host_srv_count) is less than or equal to 1.5 then the network traffic is an attack which is U2R.

Rule 9: This rule means for a given network packet if its attribute values like protocol type is TCP, rerror_rate which means the percentage of connections that have “REJ” errors is less than or equal to 0.1 percent, logged_in which means either successfully logged or not in to the network is less than or equal to 0.5 seconds, the status of the flag is SF, the length (number of seconds) of the connection which means duration is less than or equal to 0.5 seconds, num_failed_logins is less than or equal to 0.5 logins and number of data bytes from source to destination which is the src_bytes is less than or equal to 3 bytes then the network traffic is an attack which is U2R

Rule 10: This rule means for a given network packet if its attribute values like protocol type is UDP, percentage of connections to the same services which means same_srv_rate is greater than 0.995 percent, the network services on the destination is domain_u and count of connections having the same destination host and using the same service (dst_host_srv_count) is less than or equal to 51.5 then the network traffic normal.

Rule 11: This rule means for a given network packet if its attribute values like protocol type is TCP, rerror_rate which means the percentage of connections that have “REJ” errors is less than or equal to 0.1 percent, logged_in which means either successfully logged or not in to the network is less than or equal to 0.5 seconds, the status of the flag is SF, the length (number of seconds) of the connection which means duration is less than or equal to 0.5 seconds, num_failed_logins is less than or equal to 0.5 logins , and number of data bytes from destination to source which is dst_bytes greater than 34.5 and less than or equal to 35.5 bytes then the network traffic is an attack which is U2R.

Rule 12: This rule means for a given network packet if its attribute values like protocol type is ICMP, the network services on the destination is ecr_i and number of data bytes from source to destination which is the src_bytes is less than or equal to 27.5 bytes then the network traffic is an attack which is DOS.

Rule 13: This rule means for a given network packet if its attribute values like its protocol type is TCP, rerror_rate which means the percentage of connections that have “REJ”

errors is less than or equal to 0.1 percent, logged_in which means either successfully logged or not in to the network is less than or equal to 0.5 seconds, the status of the flag is source flag (SF), the length (number of seconds) of the connection which means duration is less than or equal to 0.5 seconds, num_failed_logins is less than or equal to 0.5 logins , number of data bytes from source to destination which is the src_bytes is greater than 6.5 bytes and less than or equal to 11.5 bytes and number of data bytes from destination to source which is dst_bytes is less than or equal to 16 bytes then the packet is normal network traffic.

Rule 14: This rule means for a given network packet if its attribute values like its protocol type is TCP, rerror_rate which means the percentage of connections that have "REJ" errors is less than or equal to 0.1 percent, logged_in which means either successfully logged or not in to the network is less than or equal to 0.5 seconds, the status of the flag is SF, the length (number of seconds) of the connection which means duration is less than or equal to 0.5 seconds, num_failed_logins is less than or equal to 0.5 logins , number of data bytes from source to destination which is the src_bytes is greater than 6.5 bytes and less than or equal to 11.5 bytes and number of data bytes from destination to source which is dst_bytes greater than 16 and less than or equal to 34.5 bytes then the network traffic is an attack which is R2L

Rule 15: This rule means for a given network packet if its attribute values like its protocol type is UDP, percentage of connections to the same services which means same_srv_rate is greater than 0.005 percent and less than or equal to 0.19 percent and number of connections to the same host as the current connections in the past two seconds which means count is less than or equal to 44.5 times then the network traffic is an attack which is U2R.

Rule 16: This rule means for a given network packet if its attribute values like its protocol type is TCP, rerror_rate which means the percentage of connections that have "REJ" errors is less than or equal to 0.1 percent, logged_in which means either successfully logged or not in to the network is less than or equal to 0.5 seconds, the status of the flag is SF, the length (number of seconds) of the connection which means duration

is less than or equal to 0.5 seconds, num_failed_logins is less than or equal to 0.5 logins , and number of data bytes from destination to source which is dst_bytes greater than 35.5 and less than or equal to 36.5 bytes then the network traffic is normal.

Rule 17: This rule means for a given network packet if its attribute values which are protocol type is ICMP, the network services on the destination is eco_i and count of connections having the same destination host and using the same service (dst_host_srv_count) is greater than 14.5 and less than or equal to 57.5 then the network traffic is normal.

Appendix G: A sample decision tree generated rule from the J48 decision tree learner for selected model

```
J48 pruned tree
------------------
 protocol_type = tcp
|   rerror_rate = '(-inf-0.1]'
|   |   logged_in = '(-inf-0.5]'
|   |   |   flag = SF
|   |   |   |   Duration = '(-inf-0.5]'
|   |   |   |   |   num_failed_logins = '(-inf-0.5]'
|   |   |   |   |   |   src_bytes = '(-inf-0.5]': U2R (21.0/6.0)
|   |   |   |   |   |   src_bytes = '(3-6.5]': normal (5.0)
|   |   |   |   |   |   src_bytes = '(6.5-11.5]'
|   |   |   |   |   |   |   dst_bytes = '(-inf-0.5]': normal (1.0)
|   |   |   |   |   |   |   dst_bytes = '(0.5-2]': normal (0.0)
|   |   |   |   |   |   |   dst_bytes = '(16-34.5]': R2L (1.0)
|   |   |   |   |   |   |   dst_bytes = '(34.5-35.5]': U2R (3.0)
|   |   |   |   |   |   |   dst_bytes = '(111-136.5]': normal (0.0)
|   |   |   |   |   |   |   dst_bytes = '(136.5-142.5]'
|   |   |   |   |   |   |   |   dst_host_srv_diff_host_rate = '(-inf-0.005]': normal (3.0)
|   |   |   |   |   |   |   dst_bytes = '(176.5-179.5]': normal (0.0)
|   |   |   |   |   |   |   dst_bytes = '(192.5-553]'
|   |   |   |   |   |   |   |   dst_host_srv_count = '(-inf-1.5]': U2R (2.0)
|   |   |   |   |   |   |   |   dst_host_srv_count = '(254.5-inf)': normal (0.0)
|   |   |   |   |   dst_host_srv_count = '(-inf-1.5]': U2R (1.0)
|   |   |   |   |   dst_host_srv_count = '(1.5-14.5]': probe (14.0/2.0)
|   |   |   |   |   dst_host_srv_count = '(14.5-57.5]': U2R (3.0/1.0)
|   |   |   |   |   dst_host_srv_count = '(60.5-64.5]'
|   |   |   |   |   service = telnet: probe (0.0)
|   |   |   |   |   service = http: probe (0.0)
|   |   |   |   |   service = private: probe (0.0)
protocol_type = udp
|   same_srv_rate = '(-inf-0.005]'
|   |   dst_host_same_src_port_rate = '(0.965-0.995]': U2R (10.0/1.0)
|   |   dst_host_same_src_port_rate = '(0.995-inf)': probe (123.0/33.0)
|   same_srv_rate = '(0.005-0.19]'
|   |    count = '(-inf-1.5]': U2R (0.0)
|   same_srv_rate = '(0.995-inf)'
|   |   service = telnet: R2L (0.0)
|   |   service = http: R2L (0.0)
|   |   service = private: R2L (2773.0/244.0)
|   |   service = ecr_i: R2L (0.0)
|   |   service = domain_u
```

```
|  |  |    dst_host_count = '(-inf-5.5]': normal (3.0/1.0)
|  |  |    dst_host_count = '(110.5-214.5]': R2L (25.0)
|  |  |    dst_host_count = '(214.5-254.5]'
|  |  |  |   dst_host_same_srv_rate = '(-inf-0.005]': R2L (0.0)
protocol_type = icmp
|   service = telnet: DOS (0.0)
|   service = http: DOS (0.0)
|   service = private: DOS (0.0)
|   service = ecr_i
|  |  src_bytes = '(-inf-0.5]': DOS (0.0)
|  |  src_bytes = '(27.5-38.5]'
|  |  |    dst_host_count = '(-inf-5.5]': normal (1.0)
|  |  |    dst_host_count = '(5.5-51.5]'
|  |  |  |   dst_host_srv_diff_host_rate = '(0.025-0.045]': normal (0.0)
|  |  |    dst_host_count = '(51.5-110.5]': DOS (11.0/2.0)
|  |  |    dst_host_count = '(110.5-214.5]': normal (3.0/1.0)
|  |  |    dst_host_count = '(214.5-254.5]': DOS (0.0)
|  |  src_bytes = '(47-65.5]': normal (1.0)
|  |  src_bytes = '(65.5-79.5]': DOS (0.0)
|  |  src_bytes = '(1033.5-2702.5]'
|  |  |   srv_diff_host_rate = '(-inf-0.005]': normal (2.0)
|  |  |   srv_diff_host_rate = '(0.48-0.515]': U2R (4.0/1.0)
|  |  |   srv_diff_host_rate = '(0.515-0.665]': normal (0.0)
|  |  |   srv_diff_host_rate = '(0.98-inf)': normal (0.0)
|  |  src_bytes = '(2702.5-215668.5]': DOS (0.0)
|  |  src_bytes = '(215668.5-inf)': DOS (0.0)
|   service = domain_u: DOS (0.0)
|   service = other: DOS (0.0)
|   service = smtp: DOS (0.0)
|   service = X11: DOS (0.0)
|   service = finger: DOS (0.0)
|   service = pop_3: DOS (0.0)
|   service = time: DOS (0.0)
|   service = eco_i
|  |    dst_host_srv_count = '(-inf-1.5]': U2R (7.0/3.0)
|  |    dst_host_srv_count = '(14.5-57.5]': normal (6.0/1.0))
|  |    dst_host_srv_count = '(254.5-inf)': U2R (0.0)
```

DECLARATION

I declare that the thesis is my original work and has not been presented for a degree in any other university.

_____

Tigabu Dagne Akal (CCNA, CCNP, MCITP Candidate)

Cisco Network Engineer

This thesis has been submitted for examination with my approval as university advisor.

_____

Gashaw Kebede (PhD)

Advisor

June 2012