THE REPUBLIC OF TURKEY
BAHCESEHIR UNIVERSITY

# INTELLIGENT ANOMALY DETECTION TECHNIQUES FOR DENIAL OF SERVICE ATTACKS

Master of Science Thesis

RAMAZAN KARADEMİR

İSTANBUL, 2015

THE REPUBLIC OF TURKEY
BAHCESEHIR UNIVERSITY

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE COMPUTER ENGINEERING

# INTELLIGENT ANOMALY DETECTION TECHNIQUES FOR DENIAL OF SERVICE ATTACKS

**Master of Science Thesis**

**RAMAZAN KARADEMİR**

**Supervisor: ASSOC. PROF. DR. VEHBİ ÇAĞRI GÜNGÖR**

**İSTANBUL, 2015**

**THE REPUBLIC OF TURKEY**
**BAHCESEHIR UNIVERSITY**

**GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES SCHOOL COMPUTER ENGINEERING**

Name of the thesis: Intelligent anomaly detection techniques for denial of service attacks
Name/Last Name of the Student: Ramazan Karademir
Date of the Defense of Thesis: 12-01-2015

The thesis has been approved by the Graduate School of Natural And Applied Sciences.

Assoc. Prof. Dr. Nafiz ARICA
Acting Director
Signature

I certify that this thesis meets all the requirements as a thesis for the degree of Master of Arts.

Assist. Prof. Dr. Tarkan AYDIN
Program Coordinator
Signature

This is to certify that we have read this thesis and we find it fully adequate in scope, quality and content, as a thesis for the degree of Master of Arts.

| Examining Committee Members | Signature |
|---|---|
| Assoc. Prof. Dr. Vehbi Çağrı GÜNGÖR | ---------------------------------- |
| Assoc. Prof. Dr. M. Alper TUNGA | ---------------------------------- |
| Assist. Prof. Dr. Selçuk BAKTIR | ---------------------------------- |

# DEDICATION

This thesis is dedicated to my wife Yasemin, thank you for your endless patience and support that push me to achieve my goals.

It is also dedicated to my lovely sons, Tufan and Eren: there were many times when this thesis took me away from precious moments with them.

To other family members, managers at work and great friends who have encouraged me throughout this work.

İstanbul, 2015 Ramazan Karademir

# ACKNOWLEDGEMENT

I am grateful to my thesis advisor, Assoc. Prof. Dr. Vehbi Çağrı Güngör, whose support and patience helped me throughout the study.

I wish to thank my committee members who were more than generous with their expertise and precious time.

A special thanks to Labris Network for providing me valuable dataset for this study.

İstanbul, 2015 Ramazan Karademir

# ABSTRACT

INTELLIGENT ANOMALY DETECTION TECHNIQUES FOR DENIAL OF SERVICE ATTACKS

Ramazan Karademir

Computer Engineering

Thesis Supervisor: Assoc. Prof. Dr. Vehbi Çağrı Güngör

January 2015, 59 of Main Text

With the increase of services provided over the internet, attacks to cease the availability of these services are increasing, diversifying and renewing every day. These types of attacks, which are called Denial of Service (DoS) attacks, constitute most of the attacks over the internet these days. When you think of the diversity of the services and commercial volumes of the services provided over the internet, any disruption of these services even in short durations, may cause inconvenience for the services, financial loss as well as prestige and loss of confidence for companies and institutions.

Most of the time it is very difficult to identify and detect denial of service attacks that targets to computer networks. The most important reason for this is that, the network traffic that is generated by denial of service attacks is almost identical with the network traffic that is generated by a real user. Here, the adversary is identified by only it's intend.

With this work, we aim to detect denial of service attacks quickly, in a right way and differentiate the real user from adversary with the lowest possible error. In order to achieve this aim we think that the use of different data mining techniques is suitable.

In this direction, the traffic of Ligtv.com.tr web sites, which has a millions of users from all over the world, is traced in live environment. In order to differentiate real user traffic and denial of service attack traffic, significant network traffic features are identified. Attack free network traffic is recorded to the database and normal user profile is created. Then, different distributed denial of service attacks are generated for this site and this traffic is also recorded to the database to construct attack profile. Finally normal profile and attack profile are merged and analyzed with data mining methods.

**Keywords**: Denial of Service Attacks, Anomaly Detection, Data Mining

# ÖZET

HİZMET ENGELLEME SALDIRILARI İÇİN AKILLI ANOMALİ YAKALAMA TEKNİKLERİ

Ramazan Karademir

Bilgisayar Mühendisliği

Tez Danışmanı: Doç. Dr. Vehbi Çağrı Güngör

Ocak 2015, 59 Sayfa

İnternet üzerinden verilen hizmetlerin her geçen gün artması ile birlikte bu hizmetlerin verilmesini engellemeye yönelik yapılan saldırılar da her geçen gün artmakta, çeşitlenmekte ve yenilenmektedir. Hizmet engelleme saldırıları olarak adlandırılan bu tür saldırılar, günümüzde en çok karşılaşılan saldırı türlerini oluşturmaktadır. İnternet üzerinden verilen hizmetlerin çeşitliliği ve ticari boyutu düşünüldüğünde, bu hizmetlerde yaşanacak kısa süreli kesintiler dahi çok önemli hizmetlerin aksamasına yol açabilir, şirketlerin ve kurumların maddi kayıplar yanında itibar ve güven kaybı yaşamasına da sebep olabilir.

Bilgisayar ağlarına yönelik hizmet engelleme saldırılarını tespit etmek çoğu zaman çok zordur. Bunun en önemli sebebi ise hizmet engelleme saldırılarının oluşturduğu veri trafiğinin gerçek bir kullanıcı veri trafiğinden farksız olabilmesidir. Burada saldırgan sadece niyetinden anlaşılabilir.

Bu çalışma ile hizmet engelleme saldırılarının kısa zamanda, doğru biçimde tespit edilebilmesi ve gerçek kullanıcı ile saldırganların en az hata ile ayırt edilebilmesi amaçlanmıştır. Bu amacı gerçekleştirebilmek için çeşitli veri madenciliği yöntemlerinin kullanılmasının uygun olacağı düşünülmüştür.

Bu doğrultuda dünyanın dört bir tarafından milyonlarca kullanıcısı olan Ligtv.com.tr web sitesi trafiği gerçek ortamda izlenmiştir. Normal kullanıcı trafiği ile saldırı trafiğini ayırt edebilmek için önemli olabilecek ağ trafiği özellikleri belirlenmiştir. Saldırı olmayan zamanların ağ trafiği veri tabanına kaydedilerek normal kullanıcı trafiği profili oluşturulmuştur. Daha sonra bu web sitesine çeşitli hizmet engelleme saldırıları yapılmış ve bu saldırı trafiğinin de ayrıca profili oluşturulmuştur. Son olarak normal kullanıcı trafiği ve saldırı trafiği birleştirilerek, veri madenciliği yöntemleri ile analiz edilmiş ve saldırı trafiğini normal trafikten en doğru biçimde ayıran yöntemler belirlenmiştir.

**Anahtar Kelimeler**: Hizmet Engelleme Saldırıları, Anomali Yakalama, Veri Madenciliği

## TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# ABBREVIATIONS

| | |
|---|---|
| DDoS | : Distributed Denial of Service Attack |
| DNS | : Domain Name System |
| DoS | : Denial of Service Attack |
| HTTP | : Hypertext Transfer Protocol |
| ICMP | : Internet Control Message Protocol |
| IDS | : Intrusion Detection System |
| IRC | : Internet Relay Chat |
| IP | : Internet Protocol |
| TCP | : Transmission Control Protocol |
| UDP | : User Datagram Protocol |
| URL | : Uniform Resource Locator |

# 1. INTRODUCTION

Today internet is indispensable part of modern society. Be it a company, institution or any type of organization all they have many services on internet. There are many big companies running businesses solely on the internet. However, billions of devices connected to the internet can also be used by malicious users to attack a target on the internet.

One popular category of attacks that can be used by malicious users is denial of service (DoS) attacks. A denial of service attack can be defined as an attempt that aims to bring down the availability of the services provided by the server so that legitimate users of these services are blocked or temporarily disrupted.

In a typical scenario, DoS attacks are done by flooding the target network with high volume of traffic thereby depleting the critical resources such as bandwidth, memory, and CPU time of the server. If these attacks are generated by many different resources from internet then they are called distributed denial of service (DDoS) attacks. DDoS attack can seriously affect a company's business. Any failure at preventing a DDoS attack can pose a huge financial loss, customer dissatisfaction as well as bad reputation for the company.

Recent history clearly shows how big this threat is. (Ries 2010) In 2000, 15 year-old Michael Calce launched first major DDoS attack for some of the most popular websites such as Yahoo, CNN, eBay and Amazon. In 2002 and 2007, 13 DNS root servers in the worldwide attacked with DDoS to bring down the internet as a whole. Again in April 2007, Estonia government is attacked with DDoS and the country was isolated from the rest of the world. In 2008, a hacker group called "Anonymous" launched first high profile DDoS attack targeting scientology.org. In 2009 Arbor Networks[1], a global company in network security reported that DDoS attacks had increased from 400

[1] http://www.arbornetworks.com/

megabits per second to 49 gigabits per second in last 7 years. Every year, the attack incidents and volumes continue to grow with unprecedentedly.

In computer security realm, intrusion detection systems (IDS) are considered an important defense system for network intrusions including DDoS attacks. These systems monitor network traffic and detect intrusions or anomalies which may belong to a malicious user. IDS systems in terms of analysis perspective generally divided into two categories: misuse detection IDS systems and anomaly detection IDS systems. Misuse detection systems also called signature based systems, uses pre-defined attack patterns as a signature in order to identify attack traffic. Therefore, misuse detection systems cannot detect zero day attacks. On the other hand, Anomaly detection systems constructs normal usage profiles of network traffic data and then tries to discover deviations from the normal profiles. As a result, anomaly detection systems can detect zero day attacks but also produce too many false alarms as a negative effect. The nature of the current complex and high speed networking environment makes the attack detection task very difficult. Another challenge, the current diversity of known attacks is very high and various new types of attacks are emerging quickly.

In this study, an intrusion detection system is constructed to detect DoS/DDoS attacks by using outlier detection approach with k-means clustering and Naïve Bayes classification algorithms. To test the effectiveness of the system Ligtv.com.tr[2] web site real traffic data is used. Ligtv.com.tr web site is a football news related platform mainly Turkish Super Football League, which has a millions of customer all around the world. After collecting normal data from Ligtv.com.tr web site, various DDoS attacks are generated for this web site and these attack traffic captured for further processing. Also, various data mining methods, feature selection methods, hybrid solutions and several machine learning algorithms are experimented by using Weka data mining tool (Hall et.al 2009) and results are measured in terms of training time, accuracy, detection rate, and false alarm rate. The distinctive part of this study includes the originality of real network data, real DDoS attacks and unique features constructed for DDoS attack detection.

[2] http://www.ligtv.com.tr/

# 2. LITERATURE SEARCH

In literature, there is a lot of research effort going on network security against cyber-attacks. Some of them designed for specifically to detect DoS/DDoS attacks and some of them designed as a general solution to detect all kinds of network intrusions including DoS/DDoS attacks.

Some researchers define the attack detection problem as a classification problem such as the network traffic is normal or attack and others define it as an anomaly detection problem. According to the problem definition, they use appropriate tools and methods for solving the problem such as data mining methods, statistical methods and machine learning methods.

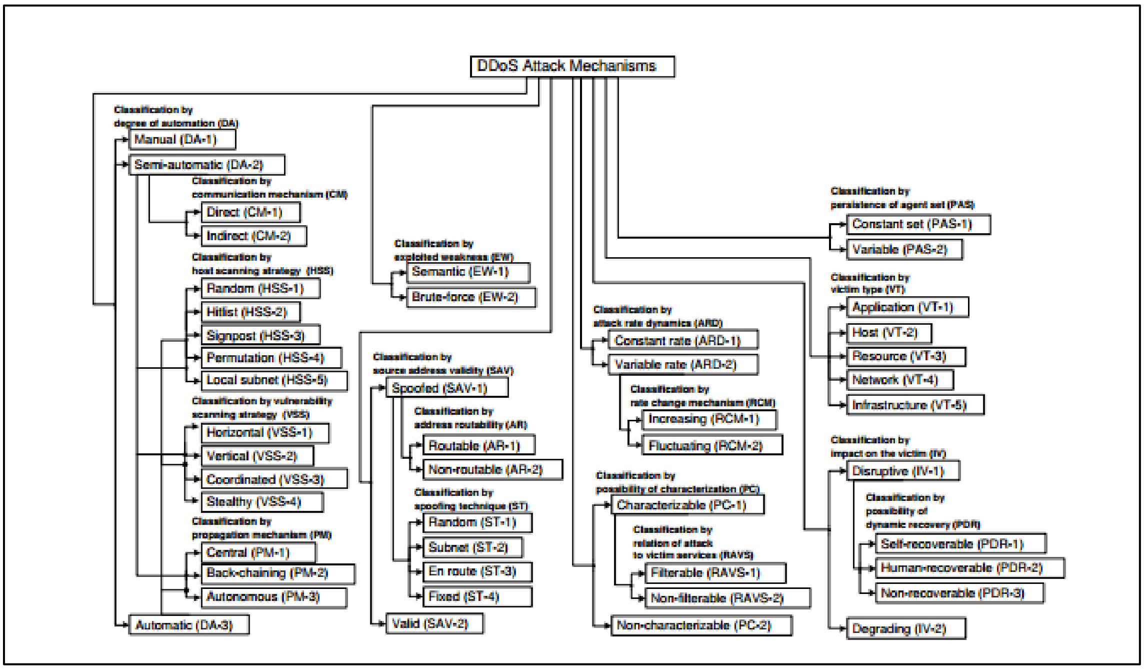
One of the earliest studies (Lee at.al. 1999) draws attention to the need of efficiency of data analysis in a real-time environment. The authors defined accurate yet efficient data mining process for network intrusion detection problem. The process employs frequent pattern mining for feature construction with a minimum data preprocessing. They start with intrinsic network traffic features such as service, src_host, dst_host, duration and flag. Then they extended initial connection records with statistical summaries of network activities within a time period. After then an iterative process is applied for different combinations of attributes and the best resulting performance of features are selected for model. In the end they constructed several statistical features based on two second time window parameter and a 100 connection window parameter.

Another research (Portnoy et.al. 2001) investigates intrusion detection with unlabeled data using clustering. The authors express that most often there is no purely labeled data available for intrusion detection. You can simulate intrusions, but in that case you are limited with the known attacks in your hand. Even in that case, it is very difficult to classify network data manually given that huge amount of network traffic data. The authors believe that unsupervised anomaly detection can overcome these problems and they made two important assumptions about data for their approach. The first assumption is that, since intrusions are very rare incidents, in a given dataset there should be very few intrusions compared to normal instances. The second assumption states that intrusions are inherently have different characteristics from normal instances. With the motivation from these two assumptions they clustered unlabeled data with simple distance metric and labelled small clusters as anomalies. The clustering algorithm uses a variant of single-linkage clustering, which starts with empty clusters, and passes through the dataset for assigning instances to the closest clusters only if the distance is less than some predefined constant otherwise a new cluster will be created for the instance. In the end they evaluated their approach with KDD CUP 99[3] data, which is publicly available intrusion attack dataset. Although the detection rate they found was between 40 percent-55 percent, which is poor, they found promising false alarm rates ranged from 1.3 percent to 2.3 percent.

Two researchers (Mirkovic and Reiher 2004) prepared taxonomy for other researchers which can be used as an introduction to the field. The taxonomy provides a common view of the DDoS attack and defense mechanisms and explains them by answering some important questions in the field such as what makes DDoS attacks possible? What are the options for performing DDoS attack? Why it is a difficult problem to be solved? Which attacks are handled by the current defense systems? What attacks still needs to be addressed? As a result they constructed DDoS attack taxonomy by using several criteria like automation degree used to perform attack, exploited weakness to service interruption, whether the source address is valid or not, whether the attack rate is constant or variable. Figure 2.1 presents all the classifications used in DDoS attack mechanism taxonomy.

[3] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

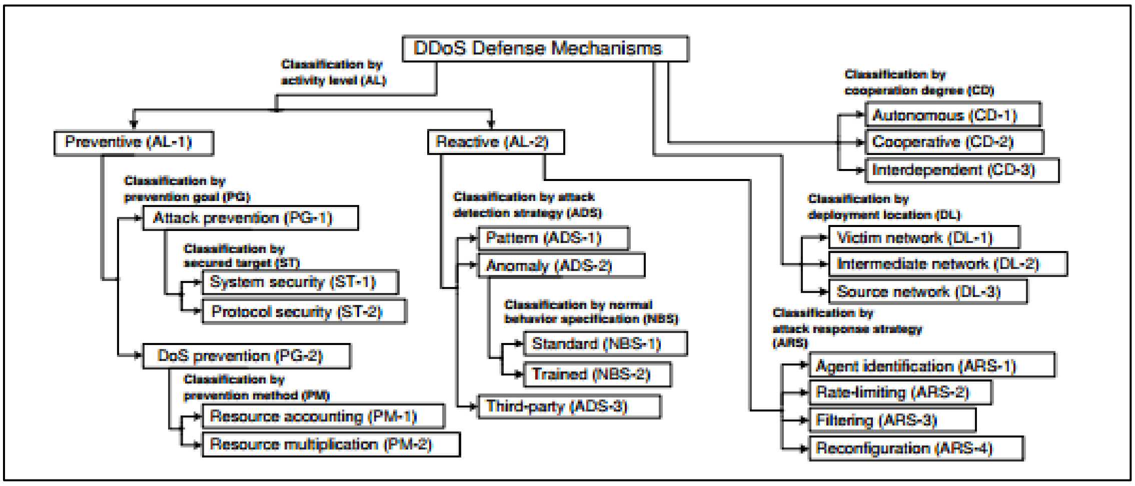
**Figure 2.1**: **Taxonomy of DDoS attack mechanisms**


DDoS Attack Mechanisms
Classification by degree of automation (DA)
Manual (DA-1)
Semi-automatic (DA-2)
Classification by communication mechanism (CM)
Direct (CM-1)
Indirect (CM-2)
Classification by host scanning strategy (HSS)
Random (HSS-1)
Hitlist (HSS-2)
Signpost (HSS-3)
Permutation (HSS-4)
Local subnet (HSS-5)
Classification by vulnerability scanning strategy (VSS)
Horizontal (VSS-1)
Vertical (VSS-2)
Coordinated (VSS-3)
Stealthy (VSS-4)
Classification by propagation mechanism (PM)
Central (PM-1)
Back-chaining (PM-2)
Autonomous (PM-3)
Automatic (DA-3)
Classification by exploited weakness (EW)
Semantic (EW-1)
Brute-force (EW-2)
Classification by source address validity (SAV)
Spoofed (SAV-1)
Classification by address routability (AR)
Routable (AR-1)
Non-routable (AR-2)
Classification by spoofing technique (ST)
Random (ST-1)
Subnet (ST-2)
En route (ST-3)
Fixed (ST-4)
Valid (SAV-2)
Classification by attack rate dynamics (ARD)
Constant rate (ARD-1)
Variable rate (ARD-2)
Classification by rate change mechanism (RCM)
Increasing (RCM-1)
Fluctuating (RCM-2)
Classification by possibility of characterization (PC)
Characterizable (PC-1)
Classification by relation of attack to victim services (RAVS)
Filterable (RAVS-1)
Non-filterable (RAVS-2)
Non-characterizable (PC-2)
Classification by persistence of agent set (PAS)
Constant set (PAS-1)
Variable (PAS-2)
Classification by victim type (VT)
Application (VT-1)
Host (VT-2)
Resource (VT-3)
Network (VT-4)
Infrastructure (VT-5)
Classification by impact on the victim (IV)
Disruptive (IV-1)
Classification by possibility of dynamic recovery (PDR)
Self-recoverable (PDR-1)
Human-recoverable (PDR-2)
Non-recoverable (PDR-3)
Degrading (IV-2)

*Source : Mirkovic, J., and Reiher, P., 2004. A taxonomy of DDoS attack and DDoS defense mechanisms. SIGCOMM Comput. Commun. Rev.34, 2, pp. 39-53.*

For DDoS defense mechanisms taxonomy, they looked for activity level of the defense mechanism whether it is preventive or reactive; they looked for cooperation degree and deployment location of the defense systems. Figure 2 presents all the classifications used in the DDoS defense taxonomy.

**Figure 2.2**: **Taxonomy of DDoS defense mechanisms**


DDoS Defense Mechanisms
Classification by activity level (AL)
Preventive (AL-1)
Classification by prevention goal (PG)
Attack prevention (PG-1)
Classification by secured target (ST)
System security (ST-1)
Protocol security (ST-2)
DoS prevention (PG-2)
Classification by prevention method (PM)
Resource accounting (PM-1)
Resource multiplication (PM-2)
Reactive (AL-2)
Classification by attack detection strategy (ADS)
Pattern (ADS-1)
Anomaly (ADS-2)
Classification by normal behavior specification (NBS)
Standard (NBS-1)
Trained (NBS-2)
Third-party (ADS-3)
Classification by cooperation degree (CD)
Autonomous (CD-1)
Cooperative (CD-2)
Interdependent (CD-3)
Classification by deployment location (DL)
Victim network (DL-1)
Intermediate network (DL-2)
Source network (DL-3)
Classification by attack response strategy (ARS)
Agent identification (ARS-1)
Rate-limiting (ARS-2)
Filtering (ARS-3)
Reconfiguration (ARS-4)

*Source : Mirkovic, J., and Reiher, P., 2004. A taxonomy of DDoS attack and DDoS defense mechanisms. SIGCOMM Comput. Commun. Rev.34, 2, pp. 39-53.*

Another paper (Kayacık et.al. 2005) investigated feature relevance analysis for the KDD CUP 99 Intrusion dataset. They applied information gain methodology to discriminate most relevant features and found that some features relevant with 98 percent of the training data while some other features totally unrelated. A related paper (Onut and Ghorbani, 2007) shows how to extracts features from network packets. They used DARPA dataset and identified important features based on basic TCP features and derived features which span multiple TCP connections.

Another work (Zhong et.al. 2007) used and compared four centroid based clustering algorithms to find network intrusions. These algorithms include k-means, Mixture-Of-Spherical Gaussians (MOSG), Self-Organizing Map (SOM), and Neural-Gas. They conducted empirical studies and compared these algorithms in terms of clustering quality and runtime performances. They used inter-class distances as well as cluster sizes in their detection process. In their paper they also proposed a self-labeling heuristic to detect and label clusters. The self-labeling process starts with finding the largest cluster with its centroid defined as $\mu_0$ and labeling it as normal. Then, remaining clusters are sorted in ascending order according to its distance to $\mu_0$. Then, data instances are sorted in ascending order again in each remaining cluster. Then, a given percentage of instances form each remaining cluster are marked with normal. Finally, they labeled the rest of all instances as attacks. They also used KDD CUP 99 data for the evaluation of algorithm performances and looked at false positive rates, attack detection rates and overall accuracy.

Another research (Bellaiche and Gregoire, 2009) on attack detection was done with entropy based approach. The authors proposed unusual handshake detection mechanism based on entropy measure for TCP connections. Another paper (Sperotto et.al. 2010) investigates advantages of IP-Flow based intrusion detection systems. According to observations, intrusion detection systems based on packet inspection that rely on header information requires too much resources in today's high speed networks. The authors also state that systems that based on payload inspection suffer from encrypted protocols. On the other hand, IP Flow based systems exhibit promising advantages compared to packet based systems. They require lower amount of data and processing power. After discussing the advantages and weakness of IP Flow based systems they give the definition of a flow as follow: “A flow is defined as a set of IP packets passing an observation point in the network during a certain time interval. All packets belonging to a particular flow have a set of common properties.” In their paper they also state that detection of brute force DoS attacks is often handled with flow based systems and gives examples of systems that employ this methodology such as TOPAS (Traffic flOw and Packet Analysis System).

Data preprocessing is a crucial step in performing attack detection whether it will be based on anomaly based detection or pattern matching. The success of the attack detection system largely depends on the dataset used in the system. Another paper (Davis and Clark 2011) states that preprocessing stage constitutes 50 percent of whole efforts in the knowledge discovery processes including network intrusion detection system development. For intrusion detection system development, data preprocessing steps converts network traffic data into collection of records where each record is represented by several attributes. These steps include dataset creation, data cleaning, integration, and feature construction. In their paper, the constructed features for grouped into three distinct categories. In the first group basic features are defined. Basic features are the features that can be extracted from network packet headers such as source port, source address, destination port destination address, flags, etc. Second group defines single connection derived features. Single connection derived features are constructed by using a single session for the connection duration. In the case of TCP connection the session ends with flags (e.g. FIN or RST). For other protocols such as ICMP and UDP a convenient way of representing a connection can be used like time windows. There are a lot of tools available that automatically produces these connection records like NetFlow[4]. Multiple connection derived features are calculated by using single derived features over multiple connections with a time window parameter or connection window parameter. The authors state that these types of features help differentiating between anomalous traffic and normal traffic.

Another paper (Muda et.al. 2011) proposed a hybrid intrusion detection system based on K-Means clustering and Naïve Bayes classification algorithms. They used the KDD

[4] http://en.wikipedia.org/wiki/NetFlow

CUP 99 dataset to evaluate their approach. In clustering, they used three as the number of cluster because they assumed that the dataset contains three different groups. After then they used classification for their clusters. They achieved 99.8 percent detection rate and 0.09 false alarm rate. Another work (Liu, 2011) investigated the effects of data normalization with support vector machine algorithm for intrusion detection. The normalization was done with min-max normalization approach and they achieved better performance in speed and accuracy. Another paper (Mohammad et. al., 2011) investigated data mining methods in Weka environment for intrusion detection. They combined anomaly detection with misuse detection to audit the network data. Another paper (Ashok et.al., 2011) investigated feature selection with hybrid intrusion detection system. They used information measure to reduce the features and evaluated the features with k-means clustering that is based on triangular area based SVM algorithm. The authors also used the KDD CUP 99 dataset in order to evaluate the system performance. Another research (Horng et.al. 2011) proposed a novel method for intrusion detection which uses hierarchical clustering ad support vector machines. They first used hierarchical clustering to reduce training dataset and obtaining higher quality instances for the support vector machine classification.

Another hybrid solution (Om and Kundu, 2012) for intrusion detection is done with k-means clustering and combining two classification algorithms namely K-nearest neighbor and Naïve Bayes. They tried to reduce false alarm rate and evaluated their performance by using KDD CUP 99 dataset. They achieved 98.18 percent detection rate and 0.830 percent false alarm rate with the system. Another work (Altwaijry, 2012) investigated Bayesian classification based intrusion detection on KDD cup dataset with using different subsets of data. They investigated the effects of training engine performance by combining only one attack type with normal data on each training phase.

Another remarkable paper (Beitollahi and Deconinck, 2012) suggested a novel scheme for detecting application layer DDoS attacks. They explained ConnectionScore technique as a scoring system which relies on history and statistical analysis. With this technique, they measure the scores of each connection and retake resources from low scored connections during attack. Another work (Mukherjee and Sharma, 2012) investigated intrusion detection performance of three different feature selection methods namely correlation-based feature selection, information gain and gain ratio. After selecting important features they applied the Naïve Bayes classifier to reduced datasets and evaluated performances.

Another research (Chen and Kim, 2013) combined principal component analysis (PCA) with decision tree and Naïve Bayes algorithm for adaptive intrusion detection. They first applied PCA to reduce data dimensionality and remove unimportant information from dataset then applied decision tree and Naïve Bayes classifiers to evaluate the performance on KDD CUP 99 dataset. Another paper (Wankhade et.al., 2013) investigated clustering analysis for intrusion detection that encompasses feature selection, filtering, divide and merge and clustering ensemble to achieve high detection accuracy and low false alarm rate. Another work (Kim et.al., 2014) proposed a novel hybrid intrusion detection method by integrating C4.5 decision tree algorithm with one-class SVM. The authors state that their method achieved better detection rates compared to conventional methods and training and testing time decreased 50 percent and 60 percent respectively.

# 3. DATA AND METHODOLGY

## 3.1 OVERVIEW OF DENIAL OF SERVICE ATTACKS

### 3.1.1 Defining Denial of Service Attacks

In literature data security is viewed from three aspects: integrity, confidentiality and availability. Integrity concerns with maintaining consistency, accuracy and trustworthiness of data. Confidentiality of data ensures that sensitive data should not be accessed by unauthorized person while allowing access to its legitimate users. Availability concerns with guaranteeing the availability of data when it is needed. Denial of service attack (DoS) targets the availability aspects of data security. In computing denial of service attack is defined as an explicit attempt by malicious users to make a resource on the network temporarily unavailable to its legitimate users. DoS attacks do not aim to break into systems or steal or change sensitive information. The only purpose for DoS attack is to interrupt the services provided by the server.

DoS attacks are generated by sending too much request to overwhelm the victim in a very short time. When these malicious requests come from many different hosts from the network then these attacks are called Distributed Denial of Service attacks (DDoS). It is also possible with a single or very few machines to simulate DDoS attacks by spoofing source IP addresses. However performing real DDoS attacks are becoming easier day by day. There are plenty of DDoS attack tools on the web. Table 3.1 shows examples of DDoS attack tools that.

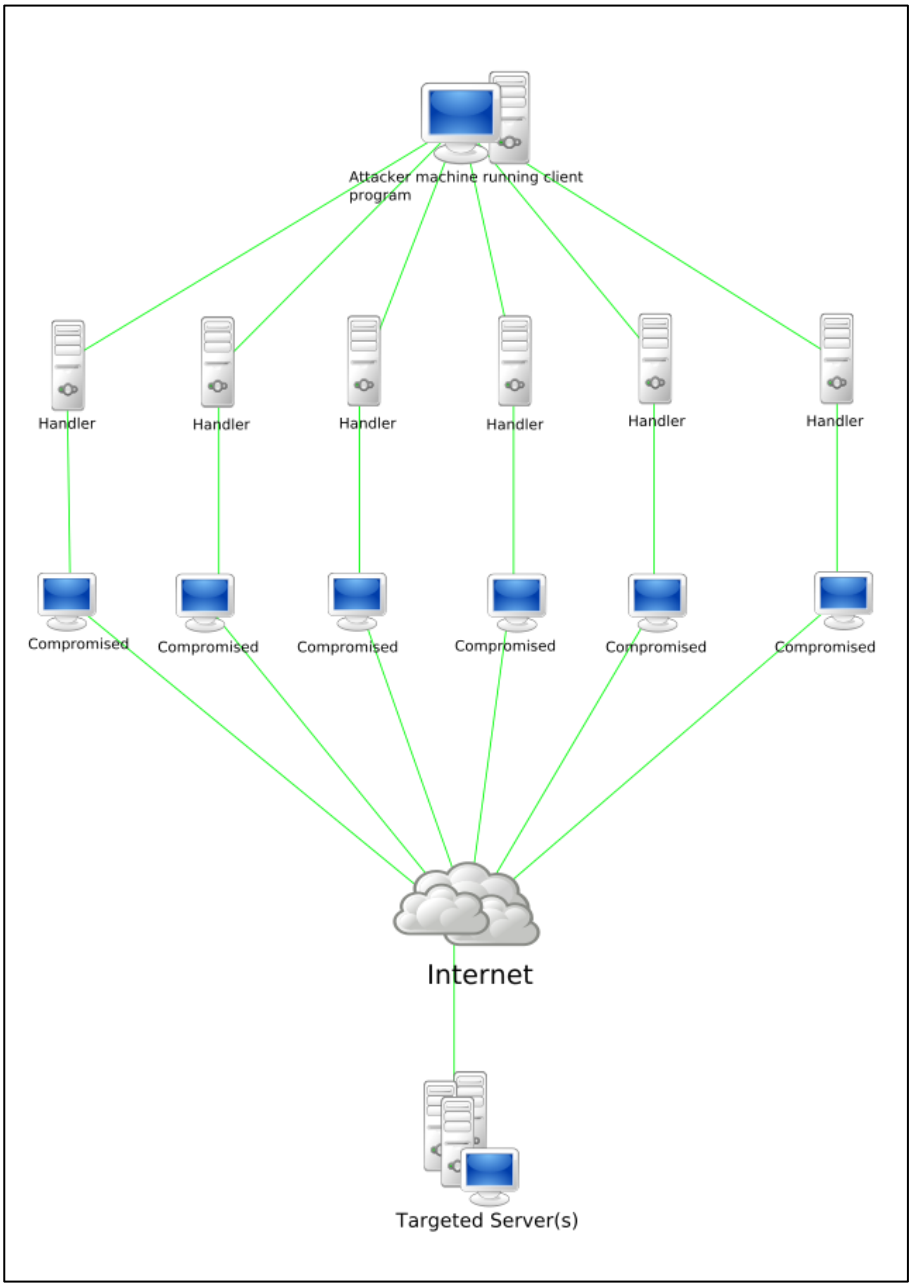
**Table 3.1: Examples of DDoS attack tools**

| | **Description and Characteristics** | **Example/ Reference** |
|---|---|---|
| Slowloris | Exploit HTTP Protocol Weakness | OWASP HTTP SlowPOST |
| DDoS Tool | Generate HTTP FLOOD using HTTP Tool, usually do not implement full function of a browser | Low Orbit Ion Canon (LOIC) |
| Botnet with DLL injection | Generate HTTP FLOOD using full function browser by DLL injection | Black DDoS |
| Launch Pad | Abuse hosted web services to generate large amount of web requests | Google+ |

*Source:* http://a-infosec.com/2013/11/11/layer-7-ddos-attack-a-web-architect-perspective/

With the help of these tools and many others on the web, usually an army of compromised internet hosts called “zombies” are employed. The attacker controls all of these zombies and generates a massive traffic to victim site. If the attacker succeeds, the victim’s all network resources become exhausted and victim becomes unavailable to its legitimate users. The architecture of DDoS attack can depicted as in figure 3.1.

**Figure 3.1: DDoS attack architecture.**



*Source*: http://en.wikipedia.org/wiki/Denial-of-service_attack

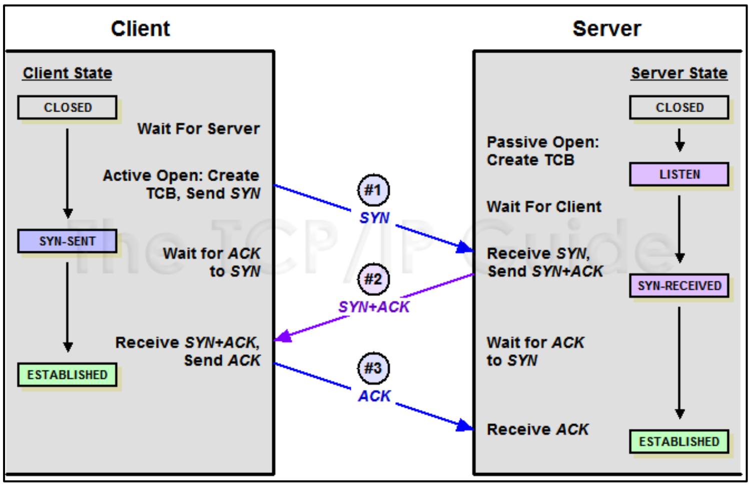
### 3.1.2 Denial of Service Attack Types

There are many types of DDoS attacks. Some of them exploit the weaknesses inherent in the protocol implementations and others simply use brute-force attacks that aim to deplete the resources of the victims. Also it should be kept in mind that attacks are very dynamic in nature. Every day a brand new attack or new variations of known attack comes out. We are going to define only the most common DDoS attack types to get some intuition about them.

### 3.1.2.1 SYN Flood

In SYN flooding the three way handshake mechanism of TCP/IP protocol implementation is exploited. According to TCP/IP protocol, connection establishment occurs after three way handshake mechanism. First the client sends a SYN packet to the server in order to establish a connection. Then, server responds with a SYN-ACK packet meaning that it accepts the connection request. Finally, the client responds with ACK packet to finish connection establishment. In this state, after a connection is established, client and server can exchange data. Attackers exploit this protocol by sending too much SYN packets to the server causing them to fill up its connection tables. The attacker does not responds to the servers SYN-ACK packets and server connection table fills up with half-open connections. When the connection table of the server fills up, then no other legitimate users can reach to the server. Normally a timeout mechanism from TCP clears the half open connections but the attacker also continues sending fake connection requests and keeping the server busy all the time.

**Figure 3.2: TCP three way handshake mechanism**



*Source:* http://www.tcpipguide.com/free/t_TCPConnectionEstablishment
ProcessTheThreeWayHandsh-3.htm

### 3.1.2.2 TCP ACK Flood

After establishing a TCP session between client and server, ACK packets are used to exchange information. In this attack, the server receives ACK flood with spoofed IP source addresses or a random sequence number at a very high packet rate. These bogus packets cause the server to check its connection table for corresponding session thereby resulting in performance degradation by exhausting system resources such as memory or CPU.

### 3.1.2.2 RST or FIN Flood

TCP RST or FIN flag is used to terminate a connection in a three or four way of handshake mechanism of TCP. In this attack, the server receives RST or FIN flood with spoofed IP source addresses or a random sequence number at a very high packet rate. These bogus packets that do not belong to any session cause the server to check its session table for corresponding session thereby resulting in performance degradation by exhausting system resources such as memory or CPU.

### 3.1.2.3 UDP Flooding

UDP flooding attacks are categorized as brute-force attacks. During a UDP flood, spoofed UDP packets are sent to a random port on the server. When the server receives a UDP packet, it looks for an application in the destination port. If there is no application listening on that port, the server generates an ICMP packet stating that destination is unreachable. If the UDP flood is strong enough to overwhelm the server then the network goes to congestion and performs poorly. When a server is under UDP attack the most common response is putting the server rebooting cycle until the attack ends.

#### 3.1.2.4 ICMP Flood

In this attack, spoofed ICMP packets are sent to the server at a very high packet rate with a large IP source address range. The server is overwhelmed and the attack consumes network resources and available bandwidth thereby causing too much disconnections.

#### 3.1.2.5 HTTP GET Flood

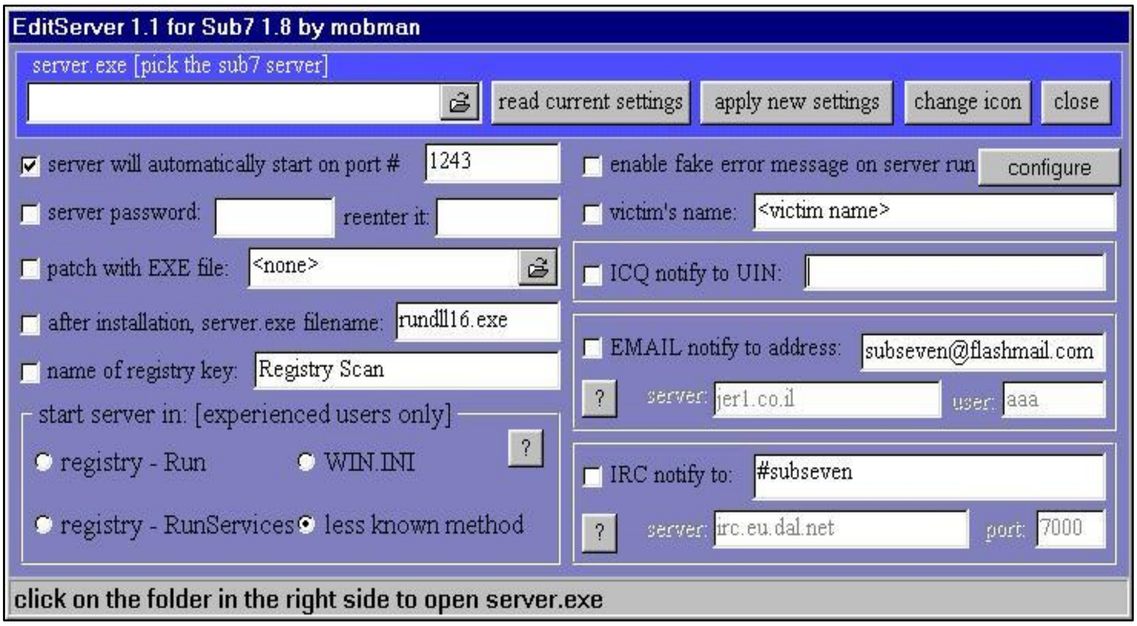
HTTP GET floods are the most common DDoS attacks. In HTTP GET attack scenario, attackers mimic the real users and send too many requests in order to overwhelm the victim server. Unlike other network level flooding attacks which uses spoofed IP addresses, these attacks uses real requests like real users. There are two common ways of conducting GET attacks. The most basic way is simply repeating the same request continuously. A bit more intelligent type of GET attack is done by recursive get requests. In this type, attacker parses the response and then recursively request each URL in random order. Detection of this type of attacks more difficult because the behavior looks like a real user.

### 3.1.3 Performing Denial of Service Attacks

There are many tools on the web in order to launch a powerful DDoS attack. Usually these tools allow a malicious user to employ massive amount of compromised machines in a coordinated way on the internet. In order to create such a zombie army, hackers install a software program called Trojan to the compromised hosts all over the internet. There are many ways to distribute these Trojans programs. One way is finding a security hole and manually installing the software. Other popular way is to write a free game and put the Trojan installer inside the game. Any person installing the game also installs the Trojan and becomes a zombie machine. Once installed, a Trojan must communicate its master with a convenient way that does not reveal its identity. This is usually achieved by using IRC protocol. The Trojan connects to a predefined public

IRC channel and waits commands from its master. Sub7[5] is a famous Trojan that operates using IRC channels. A screenshot from this program is shown in figure 3.3.

**Figure 3.3: A screenshot from Sub7.**



server.exe [pick the sub7 server]

☑ server will automatically start on port # 1243

□ server password: reenter it:

□ patch with EXE file: <none>

□ after installation, server.exe filename: rundll16.exe

□ name of registry key: Registry Scan

start server in: [experienced users only]

○ registry - Run ○ WIN.INI

○ registry - RunServices ● less known method

□ enable fake error message on server run

□ victim's name: <victim name>

□ ICQ notify to UIN:

□ EMAIL notify to address: subseven@flashmail.com

server: jer1.co.il user: aaa

□ IRC notify to: #subseven

server: irc.eu.dal.net port: 7000

click on the folder in the right side to open server.exe

*Source*: http://www.junglekey.com/wiki/definition.php?terme=Sub7

## 3.2 OVERVIEW OF INTRUSION DETECTION SYSTEMS

### 3.2.1 Defining Intrusion Detection System

The National Institute of Standards and Technology, which is an agency of U.S. Department of Commerce, defined Intrusion Detection (NIST 2007) as monitoring and analyzing the events in computer systems for signs of possible violations to the security and acceptable usage policies. After then, they defined intrusion detection system (IDS) as a software process that automates intrusion detection.

Denial of service attacks also considered intrusions and violations of security policies. Therefore, DoS attack detection can be done with intrusion detection systems. Actually, like Firewalls, IDSs are indispensable parts of security system. Unlike firewall, which

[5] http://en.wikipedia.org/wiki/Sub7

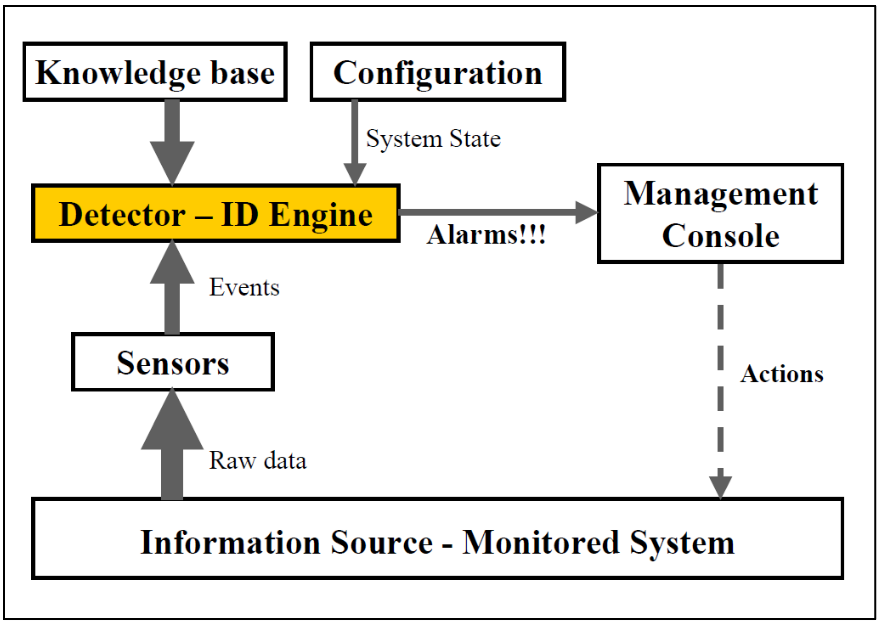
has static rules for applying security policies; IDSs are capable of detecting malicious traffic that seems legitimate to the firewall.

IDS systems can be specialized according to the needs for a particular environment. For example, in a wireless environment, IDS monitors wireless protocols, whereas Host-based IDSs monitor host activities such as applications running on the host, operating system calls and network activities of that host. For that reason, a robust solution necessitates combination of different types of IDSs. Mainly there are four types of IDS solutions: Network Based, Wireless, Network Behavior Analysis, and Host-Based.

### 3.2.2 Components of Intrusion Detection System

All types of intrusion detection systems share common component architecture. Figure 3.4 shows the basic architectural components.

**Figure 3.4: Basic IDS Architecture**



*Source*: http://www-users.cs.umn.edu/~kumar/Presentation/minds.ppt

Information source component is the monitored system against malicious activities. This can be a DNS server, a web application or a network router. Sensors are used for collecting log or activity data from the monitored system. This data can be network traffic data or operating system security logs. Detector, in other words, intrusion detection engine is responsible for processing sensor data and analyzing them for the signs of a malicious activity. If detector founds a malicious activity it informs the security operator by sending an alarm to the Management console. Management console is monitoring application that is used by a security operator. Configuration keeps the system state information that includes parameters related with the operation of the system. Knowledge base injects predefined attack signatures to attack engine for easier attack detection.

### 3.2.3 Analysis Types of Intrusion Detection System

IDS technologies use different methodologies to detect intrusions. These methodologies are usually grouped into three categories: Anomaly based, signature based and stateful protocol analysis based. Sometimes these methodologies are used together as a hybrid system in order to provide high detection capabilities.

#### 3.2.3.1 Anomaly Based Detection

Anomaly based intrusion detection compares activities that are considered as normal behavior, and finds activities that are significantly different from normal. In order to be able to identify anomalies, the system should have normal profiles of system usage. These normal profiles can be constructed monitoring the system characteristics over a period of time. These characteristics should represent the typical usage of the system such as the number concurrent connections in the past 2 seconds, the number of received packets in the past 100 connections. These statistical features construct the normal profiles of the system. An IDS system can identify any deviations from normal behavior that are above or below some thresholds. The profiling and anomaly detection for an intrusion detection system is done by employing data mining tools, machine learning methods or statistical methods.

The main advantage of anomaly based detection is that it can identify zero day attacks in other words brand new attacks that are never seen before. On the other hand, this capability brings its side effect as producing too many false alarms that is any new usage that are legal also considered as anomaly by the IDS. In addition to this, obtaining normal usage profiles that is attack free is also very difficult task.

### 3.2.3.2 Signature Based Detection

Signature based IDS systems are also called misuse based systems in literature. It uses predefined signatures that correspond to a pattern of malicious or attack behavior. For example, a telnet attempt with administrator account that is a violation of a company's security policies indicates an intrusion attempt. This can be easily defined as pattern rule for the detection engine.

The main advantage of signature based detection is that it detects any malicious behavior that is defined before and produces zero false alarm rates. However, any small variation from the attack signature can bypass the detection system. This drawback brings a burden for security administrators of keeping the system current with new attacks and variant of known attacks.

### 3.2.3.3 Stateful Protocol Analysis Based Detection

Stateful protocol analysis relies on vendor provided profiles that specify how the protocol should be used unlike anomaly detection which uses network specific usage profiles.

The main advantage of this analysis is that it can detect unexpected sequence of commands that can belong to a malicious user. On the other hand, the complexity of the analysis and the burden for keeping state for many concurrent sessions makes this methodology very resource intensive.

## 3.3 BUILDING INTRUSION DETECTION SYSTEMS FOR DENIAL OF SERVICE ATTACKS

There are two options for building intrusion detection system in terms of analysis time perspective namely, real-time analysis based intrusion detection and offline analysis based intrusion detection. In real-time analysis, IDS must operate in near real-time speeds for intrusion analysis therefore restricting its analysis capacity for complex attacks. On the other hand, offline analysis operates offline and can perform complex analysis to detect broad range of attacks. In this thesis offline analysis method were preferred, because we are researching a methodology to detect attacks with high accuracy and low false alarm rates. In advance, it is not know which methodology and which algorithm is best for achieving these objectives.

Another important thing for building offline intrusion detection system it is necessary to have network data which includes normal user traffic data and a broad range of different attacks in order to evaluate the performance of the system. There is a widely used public data on the internet called KDD CUP 99 data**.** This data has been prepared by MIT Lincoln Labs in order to help research evaluation in intrusion detection. However this data is rather old and some attacks in it are out of date now. In addition to this, as stated by a research paper (Tavallaee et.al. 2009)**,** there are some problems with this data such as redundant records which can lead an algorithm to a bias towards most frequent records. Therefore, in this thesis study, the network data were produced from Ligtv.com.tr which is a popular football news related web site from Turkey. This site is chosen because this site has a millions of visitors from all around the world and 24 hours a day is active. Also in this study, several DDoS attacks were also generated for the ligtv.com.tr web site and attack data is captured for further data processing steps.

After having the network traffic data which includes normal traffic and attack traffic the data were preprocessed into some features and created connection records, then labelled each connection records as “attack” or “normal” to have a ground truth for evaluations. For “attack” labelled connections records were further refined with the attack type. Having data ready in hand, some statistical features of the data were analyzed in order to understand it deeply. After then several data mining algorithms were applied and different clustering and classification methods were experimented to achieve best attack detection results as described in the following sections.

### 3.3.1 Normal Data Collection

As stated above, we have collected normal traffic data that is attack free from Ligtv.com.tr web site. By saying attack free, we believe that there were no DDoS attacks while collecting normal traffic data because this site was under protection for DDoS. For normal data collection we setup a data collection machine on the network and used TCPDump[6] program to capture network traffic. TCPDump is free software that can capture network packets and directs its output to a file. This program is scheduled for two consecutive days to collect network raw data for one hour duration between 01:00 am and 02:00 am. In both days, 13 million packets of 13 GB and 17 million packets of 16 GB of network data captured and named NormalSet1 and NormalSet2 respectively. The captured files are further processed with TShark[7] program which is free software to look inside and parse network packet captures, in order to get structured field information for network traffic such as ip source, ip destination, ip source port, ip destination port, and ip protocol etc. All the extracted fields and extract command can be found at appendix section A. After extracting fields, the traffic data is written to Microsoft SQL Server Database for data preprocessing steps.

### 3.3.2 Attack Data Collection

#### 3.3.2.1 Attack Generation

To generate various DDoS attacks hping[8] utility is used. This tool is free packet generator tool and used mainly by security experts to test networks. With this tool, SYN flood, IP Fragmentation, FIN Flood, RST flood, and SYN_RST Flood attacks are

[6] http://www.tcpdump.org/
[7] https://www.wireshark.org/docs/man-pages/tshark.html
[8] http://en.wikipedia.org/wiki/Hping

generated against ligtv.com.tr web site with using random ip source generation option in order to simulate distributed attacks between 01:05 am and 01:12 am. 3 GB of network traffic data which includes DDoS attacks and normal data are captured together. After then TCPDump and TShark programs are used to parse and extract traffic fields and the traffic data is written to database for further preprocessing steps. This set of data is called AtackSet1.

#### 3.3.2.2 Labris Network Attack Data

In addition to Ligtv.com.tr web site attack data, we have been provided a pure and richer attack data set in terms of attack diversity by Labris Networks[9]**,** an R&D company which specializes in network security solutions. They setup a lab environment to produce various DDoS attacks and generated the following DDoS attacks: syn_ack_ddos, icmp_ddos, rst_ack_ddos, rst_ddos, fin_ddos, ack_ddos, http_get, and syn_ddos. 16 GB of attack data which includes only attacks are parsed and written to database for further preprocessing steps. This set of data is called AtackSet2.

### 3.3.3 Statistical Properties of Datasets

After collecting datasets for normal traffic and attack traffic, some statistical features are analyzed. First, source ip country distributions are analyzed. It can be seen from the table 3.2 and figure 3.5 that, NormalSet1 and NormalSet2 shows nearly the same statistical values whereas attack traffic distributions are very different.

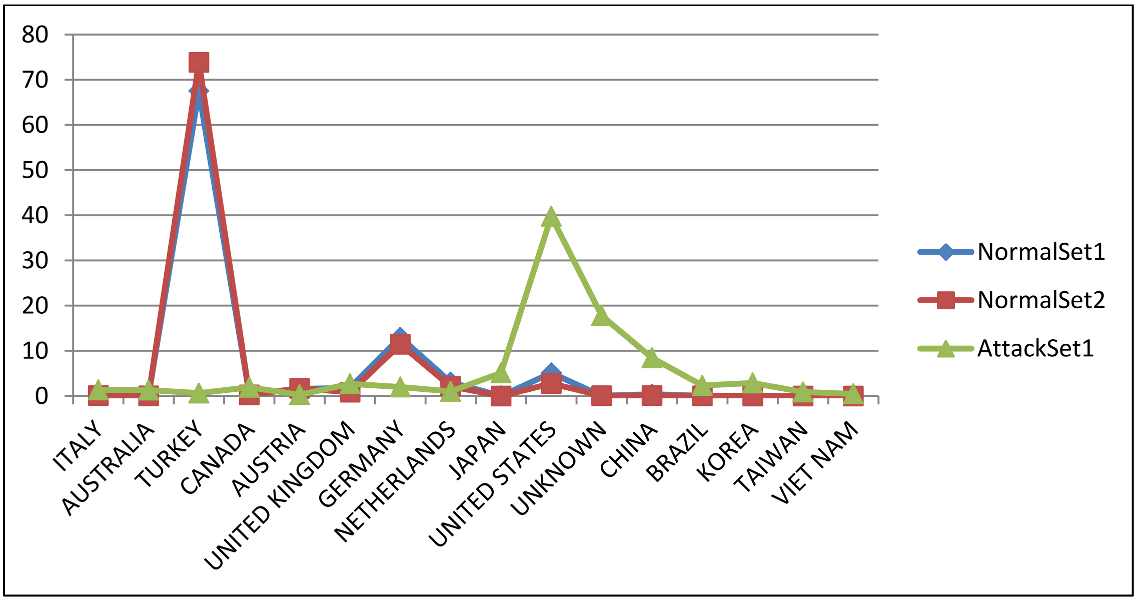
[9] http://labrisnetworks.com/

**Table 3.2: Results of attack traffic country distributions.**

| Country | NormalSet1 % | NormalSet2 % | AttackSet1 % |
|---|---:|---:|---:|
| ITALY | 0,14 | 0,12 | 1,34 |
| AUSTRALIA | 0,24 | 0,04 | 1,31 |
| TURKEY | 67,53 | 73,86 | 0,62 |
| CANADA | 0,41 | 0,23 | 1,92 |
| AUSTRIA | 1,56 | 1,71 | 0,27 |
| UNITED KINGDOM | 1,8 | 0,85 | 2,68 |
| GERMANY | 12,96 | 11,45 | 2 |
| NETHERLANDS | 3,03 | 2,12 | 1,04 |
| JAPAN | 0,01 | 0,01 | 5,11 |
| UNITED STATES | 5,09 | 2,78 | 39,79 |
| UNKNOWN | 0,05 | 0,07 | 17,81 |
| CHINA | 0,39 | 0,11 | 8,44 |
| BRAZIL | 0,05 | 0,02 | 2,34 |
| KOREA | 0,02 | 0,01 | 2,88 |
| TAIWAN | 0 | 0 | 0,89 |
| VIET NAM | 0,01 | 0,01 | 0,47 |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Figure 3.5: Attack traffic country distributions graphic.**



*Source*: *This figure has been prepared by Ramazan Karademir.*

For normal datasets there are 96 different countries, but for attack dataset there are 226 different countries. Turkey and Germany together are responsible for about 80 percent of normal traffic, but under attack traffic they are responsible only for about 3 percent of the traffic. Under attack, traffic from countries like Vietnam, Taiwan, Mexico, Brazil, China and United States shows a huge increase.

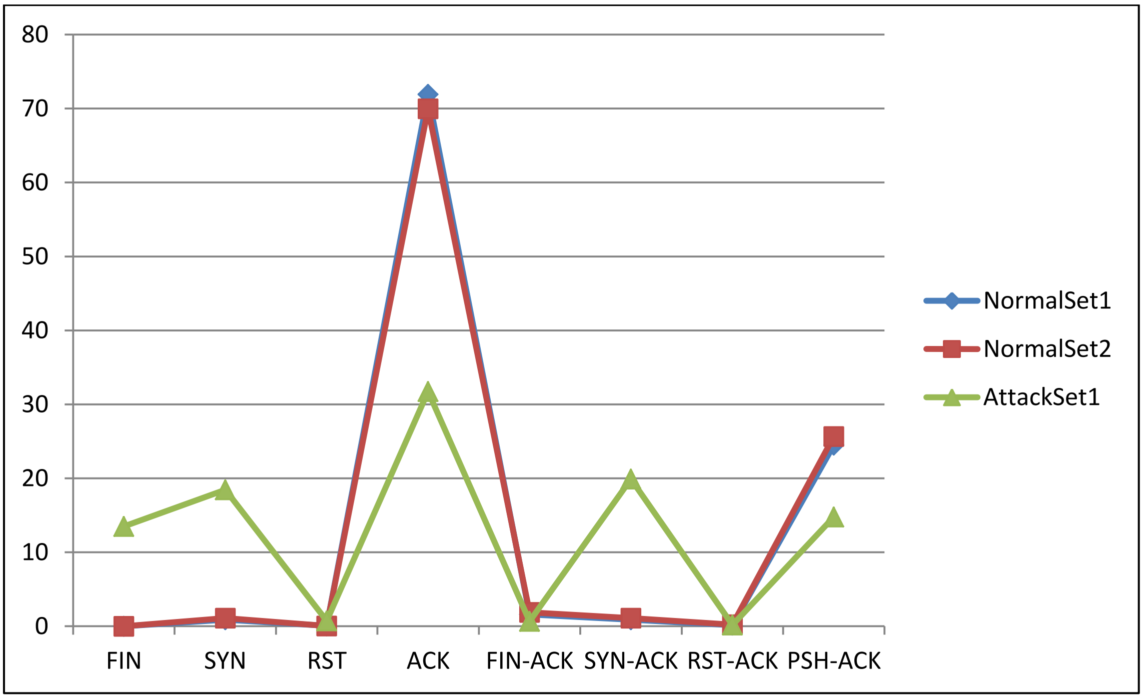
Then, TCP/IP flag distributions are analyzed. Table 3.3 and figure 3.6 show the analysis results. Under normal traffic, the flag distributions are nearly the same for NormalSet1 and NormalSet2. While under attack, TCP/IP flag distributions are changing radically. FIN and SYN flags are dominating under attack traffic.

**Table 3.3: TCP/IP flag distributions table**

| TCP/IP Flag | NormalSet1 % | NormalSet2 % | AttackSet1 % |
|---|---:|---:|---:|
| FIN | 0 | 0 | 13,5 |
| SYN | 0,88 | 1,1 | 18,46 |
| RST | 0,07 | 0,05 | 0,74 |
| ACK | 71,91 | 69,97 | 31,75 |
| FIN-ACK | 1,59 | 1,89 | 0,68 |
| SYN-ACK | 0,89 | 1,11 | 19,91 |
| RST-ACK | 0,15 | 0,21 | 0,18 |
| PSH-ACK | 24,5 | 25,67 | 14,78 |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Figure 3.6: TCP/IP flag distributions graphic.**



*Source*: *This figure has been prepared by Ramazan Karademir.*

### 3.3.4 Data Preprocessing and Feature Construction

One of the most important parts of this thesis is data preprocessing and feature construction. The success of any machine learning or data mining process heavily depends on the selected features. For that reason, several intrusion detection system features were studied mentioned in the literature search section and defined 41 features mostly similar to KDD Cup 99 features for detection of denial of service attacks. The features that are constructed from packet contents are omitted because in this thesis it is not aimed to detect semantic DDoS attacks.

The network packet data was transformed and summarized into connection records that have 41 features. All the feature names and descriptions can be found in appendix B. The selected 41 feature can be grouped into three categories as follows:

i. Basic features
ii. Time based features
iii. Connection based features

Basic features contain features that can be easily extracted from packet headers by counting some properties of packets for the connection. There are 23 basic features.

Time based features are calculated by using a 2 second time window parameter that is current connection and connections that are started within 2 seconds are considered. There are 9 time based features.

Connection based features are calculated by using a 200 connection window parameter that is current connection and past 200 connections. There are 9 connection based features.

The feature construction process that is transformation and summarization was done with custom written Microsoft SQL Server stored procedures. After data transformation raw network traffic data is transformed into connection records. The following table 3.4 shows connection counts and unique connection counts for each datasets after transformation.

**Table 3.4: Results of Dataset record counts.**

| Dataset | Record Count | Unique Record Count |
|---|---|---|
| NormalSet1 | 131.210 | 129.453 |
| NormalSet2 | 222.135 | 219.241 |
| AttackSet1 | 2.006.094 | 179.319 |
| AttackSet2 | 7.484.564 | 13.019 |

*Source*: *This table has been prepared by Ramazan Karademir.*

Finally, the labelling and attack type specification was done for all records by writing several queries and searching attack characteristics for AttackSet1 data. As it was stated before, only this dataset have mixed traffic as normal and attack traffic. NormalSet1 and NormalSet2 dataset records are labelled as normal records and AttackSet2 dataset records which contain only attack traffic are labelled as attack records. Within labelling process, an attribute for specifying the attack type is also added to records to be able to measure success of multiclass classification approaches.

## 3.4 ATTACK DETECTION WITH DATA MINING

### 3.4.1 Outlier Detection with K-Means Clustering

Outlier detection also known as anomaly detection is an important data mining technique to find abnormal behaviors that are significantly different from expected behaviors in large dataset. It can be applied to fraud detection, sensor/video network surveillance, intrusion detection as well as several other areas. Outlier detection is usually done with clustering techniques. Clustering can be defined as grouping objects into multiple groups so that objects in the same group are similar each other but different from other objects in other groups (Han et.al. 2012).

In this thesis, a customized outlier detection methodology by using K-Means clustering algorithm is implemented in Microsoft.NET C# language. The methodology works as follows. First, K-Means clustering algorithm is used to cluster normal dataset. After clustering, for each cluster the maximum Euclidean distance between cluster centroid, which is the mean of the all objects in the cluster, and its objects is selected. Then, each object's distance in attack dataset is compared with each cluster centroids and assigned to closest cluster. After assignment, the object is also checked against maximum distance for that cluster. If the new object's distance is greater than the maximum distance of that cluster then the object is marked as outlier.

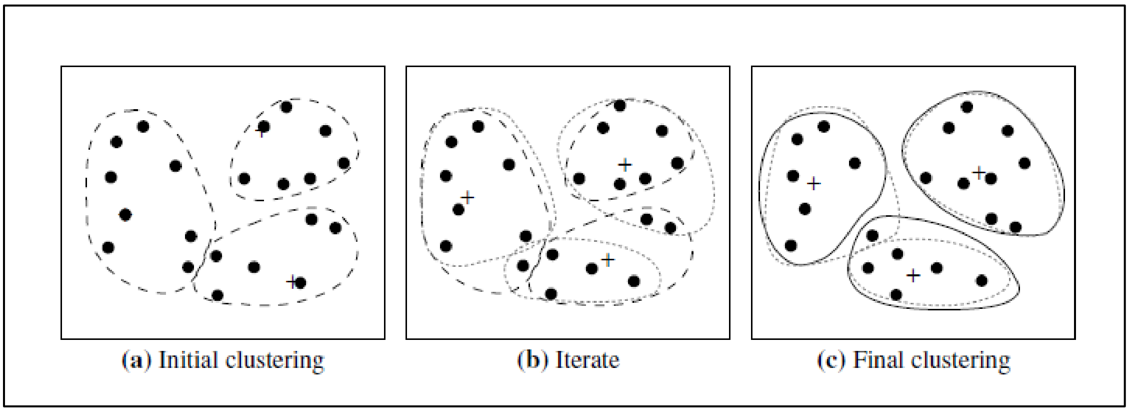
#### 3.4.1.1 K-Means Clustering Algorithm

Basically K-Means clustering algorithm works like this:

1. Randomly select *k* objects from dataset as initial centroids. *k* is the number of clusters which should be given as parameter.
2. Assign each object to the closest centroid based on Euclidean distance.
3. Recalculate the cluster means that is centroids.
4. Repeat step 2 and 3 until no change occurs that is no new assignment for objects in clusters.

Figure 3.7 shows the steps in graphic format. + represents the cluster centroids.

**Figure 3.7: K-Means clustering steps.**



*Source*: (Han et.al. 2012, pp.453).

Euclidean distance measures distance between two objects. Can be calculated like that: First define $i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ and j = $(x_{j1}, x_{j2}, \ldots, x_{jp})$ be as two objects described by p numeric attributes. Then, calculate the Euclidean distance by;

$$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{ip} - x_{jp})^2} \tag{3.1}$$

Each attribute has different scales in dataset. Without data normalization this yields biases against wider ranged attributes. Therefore, we have done data normalization for numeric attributes before calculating the distances. Normalization is done with min-max normalization to the scales [0.0, 1.0] with the given formula below;

$$v_i' = \frac{v_i - min_A}{max_A - min_A} \tag{3.2}$$

Where $min_A$ and $max_A$ are the minimum and maximum values of an attribute, A.

### 3.4.2 Classification Based Attack Detection

Attack detection systems can be built with classification based approaches if you have labeled data. As we stated in data preprocessing section we had labelled our datasets after a careful and intensive work. Therefore in this thesis we have also studied the classification based approaches for detection of attacks.

Classification can be defined as extracting models from existing observations in order to assign new observations to a set of categories that are learnt from existing observations. It is an example of supervised learning technique which requires a learning phase and a classification phase. In learning phase, a classification model is constructed by using training data. In classification phase, the learned classification model is used to predict class labels for new observations. Classification problems can be analyzed under two categories namely binary classification problem and multiclass classification problem. In binary classification, there are only two class labels such as “yes/no” or “normal/attack”, on the other hand, in multiclass classification there are more than two classes.

In this thesis, Naïve Bayes classification algorithm is implemented in Microsoft.NET C# language in order to detect attacks and evaluated its performance. Naïve Bayes classifier uses statistical probabilities in order to predict class labels. It is called Naïve due to its simple assumption that assumes an attribute is independent from other attributes in predicting class memberships.

#### 3.4.2.1 Naïve Bayes Algorithm

Before explaining Naïve Bayes classification algorithm in detail, understanding the Bayes’ Theorem terminology will be helpful. In Bayes’ theorem (Han et.al. 2012) a tuple $X$ is defined as “evidence” which has a set of $n$ attributes and hypothesis $H$ is defined as a probability that showing the “evidence” $X$ belongs to a class $C$. Then, Posterior probability $P(H|\boldsymbol{X})$ is formulated as the probability of “evidence” $X$ belongs to a class $C$. The theorem also defines the Prior Probabilities $P(H)$ of $H$, $P(\boldsymbol{X})$ of $X$, and posterior probabilities $P(\boldsymbol{X}|H)$ that is the probability of $X$ is a specific "evidence" based on $H$. Then, posterior probability is calculated with the given formula below;

$$P(H|\boldsymbol{X}) = \frac{P(H)P(\boldsymbol{X}|H)}{P(\boldsymbol{X})} \tag{3.3}$$

The Naïve Bayes classification algorithm works as follows:

1. Assume that there are $m$ classes, $C_1, C_2, \dots, C_m$ in a given training set $D$. The classifier predicts $X$ as belong to class $C_i$ only if

   $$P(C_i|\boldsymbol{X}) > P(C_j|\boldsymbol{X}) \text{ For } 1 \le j \ \le m, j \ne i. \tag{3.4}$$

   Therefore, we select the maximum $P(C_i|\boldsymbol{X})$.

2. In posterior probability formula the $P(\boldsymbol{X})$ is constant for all classes. Therefore it can be omitted. Only $P(C_i)P(\boldsymbol{X}|C_i)$ needs to be maximized.

3. Class prior probability $P(C_i)$ can be calculated by the formula given below:

   $$P(C_i) = \frac{|C_{i,D}|}{|D|} \tag{3.5}$$

   Where $|C_{i,D}|$ is the number of classes in $C_i$ in training set $D$.

4. The posterior probability for $\boldsymbol{X}$ can be calculated by the formula given below:

$$P(\boldsymbol{X}|C_i) = \prod_{k=1}^{n} P(x_k|C_i) \tag{3.6}$$

Where $x_k$ refers to value of attribute $A_k$ in tuple $\boldsymbol{X}$.

While computing $P(\boldsymbol{X}|C_i)$ we should follow different formulas for categorical and continuous attributes. For categorical attributes $P(x_k|C_i)$ is the number of tuples of class $C_i$ in $D$ having the value $x_k$ for $A_k$, divided by the number of tuples of class $C_i$ in $D$.

For continuous-valued attribute, it is assumed that the attribute values have Gaussian distribution with a mean $\mu$ and standard deviation $\sigma$ defined by

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.7}$$

So that the posterior probability can be calculated by the formula given below:

$$P(x_k|C_i) = g\left(x, \mu_{C_i}, \sigma_{C_i}\right) \tag{3.8}$$

### 3.4.3 Attack Detection with Weka Data Mining Tool

Weka (Hall et.al 2009) is free data mining tool that offers several machine learning algorithms for data analysis. It also offers utilities for data preprocessing, data visualization, and feature selection and performance evaluations.

In this thesis study, six classification algorithms from Weka were applied to our dataset in order to see performances of different algorithms and different approaches. The algorithms were evaluated with using binary classification and multiclass classification approaches. Then combinations of binary and multiclass classification approaches were analyzed together in two phase detection model. Then, hybrid methods that include clustering and classification approaches were analyzed. Then, the effect of data normalization is measured. Finally, feature selection methods were applied and selected features were also evaluated using the algorithms.

The following classification algorithms were evaluated from Weka: J48 Decision Tree, Naïve Bayes, Multilayer Perceptron, LibSVM, Random Forest and Random Tree. J48 decision tree classification is an implementation of the C4.5[10] algorithm in Weka. This algorithm generates a decision tree by using the concept of information entropy. The attribute with the highest information gain is selected for tree splitting criteria. Naïve Bayes classification is a probabilistic classifier based on Bayes' theorem which we described in the previous section. Multilayer Perceptron is a feed forward artificial neural network classification algorithm in Weka tool. It can also work on data that are not linearly separable. LibSVM[11] is a library for support vector machine (SVM) algorithm classification by. SVM tries to separate data by using the best hyper plane which provides the largest separation margin between classes. Random Forest based classification uses ensemble learning methods. This method uses bagging idea and random feature selection in order to construct decision trees. Random Tree constructs decision trees with randomly chosen attributes at each node.

[10] http://en.wikipedia.org/wiki/C4.5_algorithm

[11] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

# 4. FINDINGS

## 4.1 EVALUATION METHODS AND METRICS

In this thesis, several different data mining methodologies and machine learning algorithms employed to detect denial of service attacks. Accordingly different metrics are used to measure the performances. For intrusion detection systems the performance of the system is measured with accuracy, detection rate and false alarm rate. High accuracy, high detection rate and low false alarm rates are key performance indicators of any intrusion detection systems. In addition to this metrics we also evaluated clustering qualities and other classification measures like F1 measure and runtime performances of the algorithms.

For outlier detection approach k-means clustering algorithms were evaluated with different number of cluster parameter $k$ ranging from 2 up to 100 and looked for minimizing the sum of squared error (SSE) marginally. Because after some point increasing the $k$ will not provide a meaningful decrease in SSE. The SSE can be calculated by the following formula:

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i , x) \tag{4.1}$$

In SSE formula, $x$ is a data point in cluster $C_i$ and $m_i$ is the cluster centroid. Although clustering quality is not the primary concern for the attack detection problem, we think that a good quality clustering yields a better attack detection results.

To evaluate classification algorithms accuracy, detection rate, false alarm rate and F1 measures are calculated. These measures depend on the following key measures as described in the table 4.1 below:

**Table 4.1: Key measures for performance evaluation.**

| Key Measures | Description |
|---|---|
| True Positive (TP ) | Attack traffic and attack traffic is correctly identified. |
| False Positive (FP) | Normal traffic and but incorrectly identified as attack traffic. |
| True Negative (TN) | Normal traffic and correctly rejected as normal traffic. |
| False Negative (FN) | Attack traffic and incorrectly rejected as normal traffic. |

*Source*: *This table has been prepared by Ramazan Karademir.*

These key measures can also be summarized in confusion matrix format as shown in table 4.2.

**Table 4.2: Format of confusion matrix.**

| Actual class | Predicted class | |
|---|---|---|
| | attack | normal |
| attack | TP | FN |
| normal | FP | TN |

*Source*: *This table has been prepared by Ramazan Karademir.*

Now the formulas for the performance metrics for classifiers that are based on key measures can be seen on table 4.3.

**Table 4.3: Formulas for performance metrics.**

| Measure | Formula |
|---|---|
| Accuracy | $(TP + TN)/\ (TP + TN + FP + FN)$ |
| Detection Rate (precision) | $TP\ /\ (TP + FP)$ |
| False Alarm Rate | $FP\ /\ (FP + TN)$ |
| True Positive Rate (recall) | $TP\ /\ (TP + FN)$ |
| F1 Measure | $2 * TP\ /\ (2 * TP + FP + FN)$ |

*Source*: *This table has been prepared by Ramazan Karademir.*

Accuracy is the rate of correct classification that is attacks are classified as attacks and normal records are classified as normal.

Detection rate is the rate of attack detection success rate that is in what rate of the detected attacks are real attacks.

False alarm rate is the rate of incorrect attack classification rate. In other words, the rate of normal classes incorrectly identified as attack over the whole normal traffic.

True positive rate is the rate of correct attack classification rate. In other words, the rate of attack classes correctly identified as attack over the whole attack traffic.

F1 measure is the harmonic mean of precision (detection rate) and recall (true positive rate).

## 4.2 EVALUATIONS

### 4.2.1 Outlier Detection Evaluation

As it was stated in data preprocessing section four dataset were created in order to use in attack detection experiments. In outlier detection evaluation NormalSet1 dataset has been used which contains only normal traffic data to construct normal profiles with using k-means clustering algorithms. Normal clustering profiles are constructed with 11 different cluster parameters of $k$ and evaluations are done with attack datasets for each clustering. AttackSet1 and AttackSet2 datasets are used as test evaluation datasets. AttackSet1 dataset contains five different attack records as well as normal records. AttackSet2 dataset contains eight different attack records. The distribution of AttackSet1 and AttackSet2 dataset are shown on table 4.4 and table 4.5 respectively.

**Table 4.4: Distributions of AttackSet1 records.**

| **AttackSet1 Records** | |
|---|---|
| **Label** | **Count** |
| Normal | 31912 |
| FIN_attack | 12146 |
| FragmantedSet | 32836 |
| RST_Attack | 1184 |
| SYN_Attack | 66515 |
| SYN_RST | 34726 |
| **Total** | **179.319** |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table 4.5: Distributions of AttackSet2 records.**

| **AttackSet2 Records** | |
|---|---|
| **Label** | **Count** |
| syn_ack_ddos | 1208 |
| icmp_ddos | 38 |
| rst_ack_ddos | 2848 |
| rst_ddos | 1809 |
| fin_ddos | 21 |
| ack_ddos | 844 |
| http_get | 3073 |
| syn_ddos | 3188 |
| **Total** | **13.029** |

*Source*: *This table has been prepared by Ramazan Karademir.*

In testing phase, the Attack dataset records are normalized and assigned to closest clusters. During the assigning operation, the key measures that are true positives, false positives, true negatives and false negatives are calculated as described in section 3.4.1. With key measures in hand, the accuracy, detection rate, false alarm rate and F1 performance measures are calculated.

### 4.2.2 Classification Based Evaluations

For the classification based evaluations as it was stated earlier Naïve Bayes classification algorithm was implemented. To evaluate the algorithm performance in our datasets, the normal data sets was mixed with attack datasets and prepared combined datasets. First, NormalSet1 data was combined and mixed randomly with AttackSet1 and then two new separate dataset were created for training and testing phases. Training dataset holds 60 percent and testing dataset holds 40 percent of the combined dataset. Secondly, similar to the first approach, NormalSet1 was combined with AttackSet2 and an alternative datasets were created as training and testing datasets for evaluations of classification based algorithms.

### 4.2.3 Weka Data Mining Tool Evaluations

Also with Weka data mining tool the same combined training and test datasets were used to evaluate J48, Naïve Bayes, LibSVM, Multilayer Perceptron, Random Forest, and Random Tree classification algorithm performances.

Additionally, six different feature selection methods from Weka data mining tool were applied to datasets and resulting features were evaluated using classification algorithms. With feature selection, data sizes decreases and only important features are used in classification algorithms. Thereby, training time decreases and classification algorithms generalize better by eliminating over fitting to the data.

## 4.3 RESULTS

### 4.3.1 Outlier Detection Evaluation Results

After running K-Means outlier detection algorithm for AttackSet1 data to detect attacks the following performance results were obtained as shown on table 4.6.

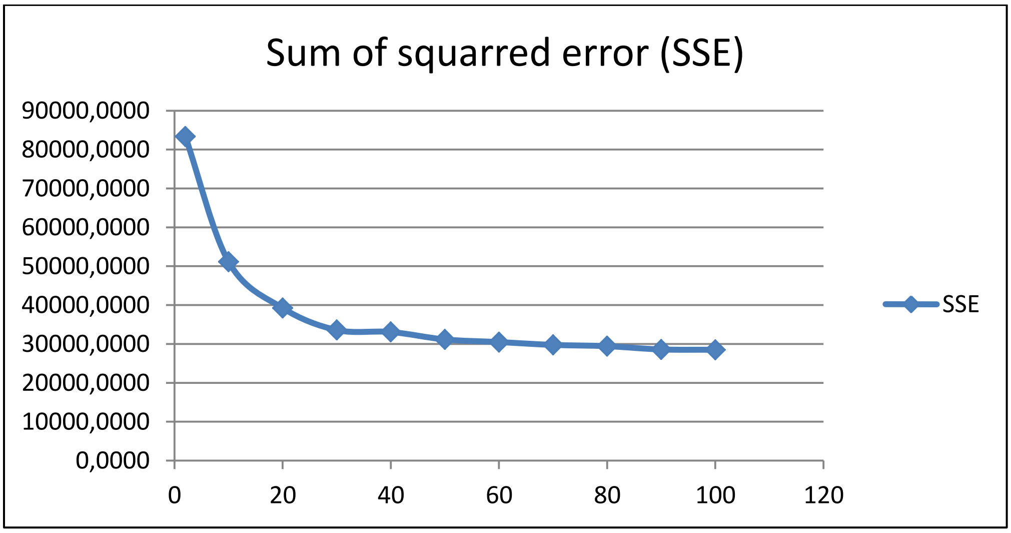
**Table 4.6: Outlier detection results for AttackSet1 dataset.**

| K | SSE | TP | FP | TN | FN | Accuracy Rate | Detect Rate | False Alarm Rate | F1 Measure |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 83365,56 | 146125 | 15144 | 16768 | 1282 | 90,84 | 90,609 | 47,456 | 94,679 |
| 10 | 51162,14 | 146059 | 15203 | 16709 | 1348 | 90,77 | 90,572 | 47,64 | 94,638 |
| 20 | 39231,95 | 146058 | 15206 | 16706 | 1349 | 90,768 | 90,571 | 47,65 | 94,637 |
| 30 | 33577,70 | 146123 | 15286 | 16626 | 1284 | 90,759 | 90,53 | 47,9 | 94,634 |
| 40 | 33084,56 | 146071 | 15207 | 16705 | 1336 | 90,775 | 90,571 | 47,653 | 94,641 |
| 50 | 31155,79 | 146153 | 15299 | 16613 | 1254 | 90,769 | 90,524 | 47,941 | 94,641 |
| 60 | 30483,24 | 147336 | 30510 | 1402 | 71 | 82,946 | 82,845 | 95,607 | 90,598 |
| 70 | 29755,86 | 146153 | 15299 | 16613 | 1254 | 90,769 | 90,524 | 47,941 | 94,641 |
| 80 | 29428,46 | 146158 | 15299 | 16613 | 1249 | 90,772 | 90,524 | 47,941 | 94,642 |
| 90 | 28585,59 | 146158 | 15299 | 16613 | 1249 | 90,772 | 90,524 | 47,941 | 94,642 |
| 100 | 28508,21 | 146153 | 15299 | 16613 | 1254 | 90,769 | 90,524 | 47,941 | 94,641 |

*Source: This table has been prepared by Ramazan Karademir.*

The K-Means outlier detection algorithm was experimented eleven times with different number of cluster parameters ranging from 2 to 100 in order to see the effects of different clustering’s to attack detection. As it can be seen from the table the attack detection rate and false alarm rate nearly the same for all different clustering’s. Although the sum of squared error (SSE) is decreasing as the number of cluster increases which can be seen on graphic 4.1., the attack detection performance is not changing in any direction for this dataset.

**Figure 4.1: The SSE graph for NormalSet1 dataset**



*Source: This figure has been prepared by Ramazan Karademir.*

The K-Means outlier detection algorithm also experimented with AttackSet2 from Labris Network which contains only attack data. The following performance results were obtained as shown on table 4.7.

**Table 4.7: Binary outlier detection results for AttackSet2 dataset.**

| K | SSE | TP | FP | TN | FN | Accuracy Rate | Detect Rate | False Alarm Rate | F1 Measure |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 83365,57 | 11689 | 0 | 0 | 1340 | 89,715 | 100 | NaN | 94,579 |
| 10 | 51162,14 | 11758 | 0 | 0 | 1271 | 90,245 | 100 | NaN | 94,872 |
| 20 | 39231,96 | 11751 | 0 | 0 | 1278 | 90,191 | 100 | NaN | 94,843 |
| 30 | 33577,7 | 11978 | 0 | 0 | 1051 | 91,933 | 100 | NaN | 95,797 |
| 40 | 33084,56 | 11743 | 0 | 0 | 1286 | 90,13 | 100 | NaN | 94,809 |
| 50 | 31155,8 | 12078 | 0 | 0 | 951 | 92,701 | 100 | NaN | 96,212 |
| 60 | 30483,24 | 13029 | 0 | 0 | 0 | 100 | 100 | NaN | 100 |
| 70 | 29755,86 | 12081 | 0 | 0 | 948 | 92,724 | 100 | NaN | 96,225 |
| 80 | 29428,46 | 12073 | 0 | 0 | 956 | 92,663 | 100 | NaN | 96,192 |
| 90 | 28585,59 | 12073 | 0 | 0 | 956 | 92,663 | 100 | NaN | 96,192 |
| 100 | 28508,22 | 12085 | 0 | 0 | 944 | 92,755 | 100 | NaN | 96,241 |

*Source*: *This table has been prepared by Ramazan Karademir.*

For AttackSet2 dataset the accuracy rate was increasing with the increase of number of cluster. Best result achieved with the number of 60 clusters. After then increasing the number of cluster affects the accuracy rate negatively. Since this dataset does not contain normal data, there is no false alarm rate.

### 4.3.2 Classification Based Evaluations Results

Naïve Bayes classification based attack detection performed with two different datasets. First binary classification is used then multiclass classification performed. Binary classification and multiclass classification performance results can be seen on the following table 4.8 and table 4.9 respectively.

**Table 4.8: Binary classification results of Naïve Bayes implementation.**

| Evaluation Data | # of Test Instances | TP | FP | TN | FN | Accuracy | Detection Rate | False Alarm Rate | F1 Measure |
|---|---|---|---|---|---|---|---|---|---|
| Labris Network | 56.954 | 5.192 | 26 | 51.726 | 10 | 99,937 | 99,502 | 0,05 | 99,655 |
| Ligtv.com.tr | 123.954 | 58.990 | 1.512 | 63.184 | 268 | 98,564 | 97,501 | 2,337 | 98,514 |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table 4.9: Multi class classification results of Naïve Bayes implementation.**

| Evaluation Data | # of Test Instances | TP | FP | TN | FN | Accuracy | Detection Rate | False Alarm Rate | F1 Measure |
|---|---|---|---|---|---|---|---|---|---|
| Labris Network | 56.954 | 1.996 | 3.186 | 51.749 | 23 | 94,366 | 38,518 | 5,8 | 55,437 |
| Ligtv.com.tr | 123.954 | 57.546 | 3.201 | 62.944 | 263 | 97,205 | 94,731 | 4,839 | 97,078 |

*Source*: *This table has been prepared by Ramazan Karademir.*

Confusion matrix details for each attack class can be found at appendix C.

Additionally, K-means clustering and Naïve Bayes classification methods were used together as a hybrid solution in order to detect attacks. First, the training data were clustered into two clusters and then each cluster was trained with Naïve Bayes algorithm separately. Similar to training approach, in evaluation step, test data first assigned to a cluster. After then, test data is evaluated with this cluster’s classification model. The following table 4.10 shows the results of hybrid approach.

**Table 4.10: Evaluation results of hybrid approach.**

| Hybrid Method | TP | FP | TN | FN | Accuracy | Detection Rate | False Alarm Rate | F1 Measure |
|---|---|---|---|---|---|---|---|---|
| KMeans + Naïve Bayes | 59.083 | 1.508 | 63.188 | 175 | 98,642 | 97,511 | 2,331 | 98,596 |

*Source*: *This table has been prepared by Ramazan Karademir.*

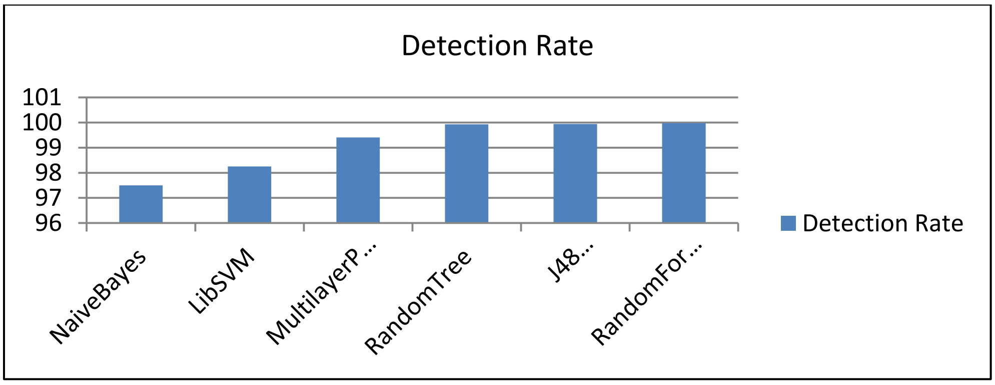
### 4.3.3 Weka Data Mining Tool Evaluations Results

With Weka data mining tool six classification algorithms were evaluated in terms of attack detection rate, false alarm rate and training time performances. After then, six feature selection methods applied and selected features were evaluated using the same classification algorithms. Finally, in order to measure the effects of feature selection methods the performances of the classification algorithms were compared. All the results obtained from Weka can be in tables in appendix D.

#### 4.3.3.1 Classification Algorithms Evaluations

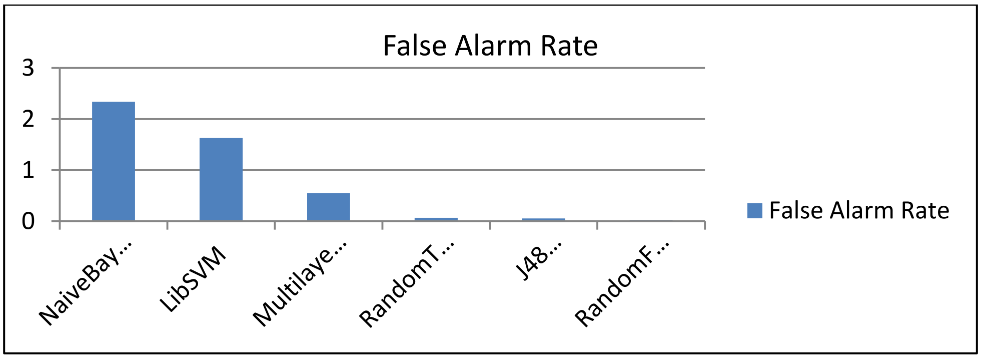
As figure 4.2 and 4.3 shows respectively, Random Forest algorithm achieved the best attack detection rate and false alarm rate whereas Naïve Bayes algorithm performed worst among others.

**Figure 4.2: Binary attack detection rates of algorithms.**



*Source*: *This figure has been prepared by Ramazan Karademir.*

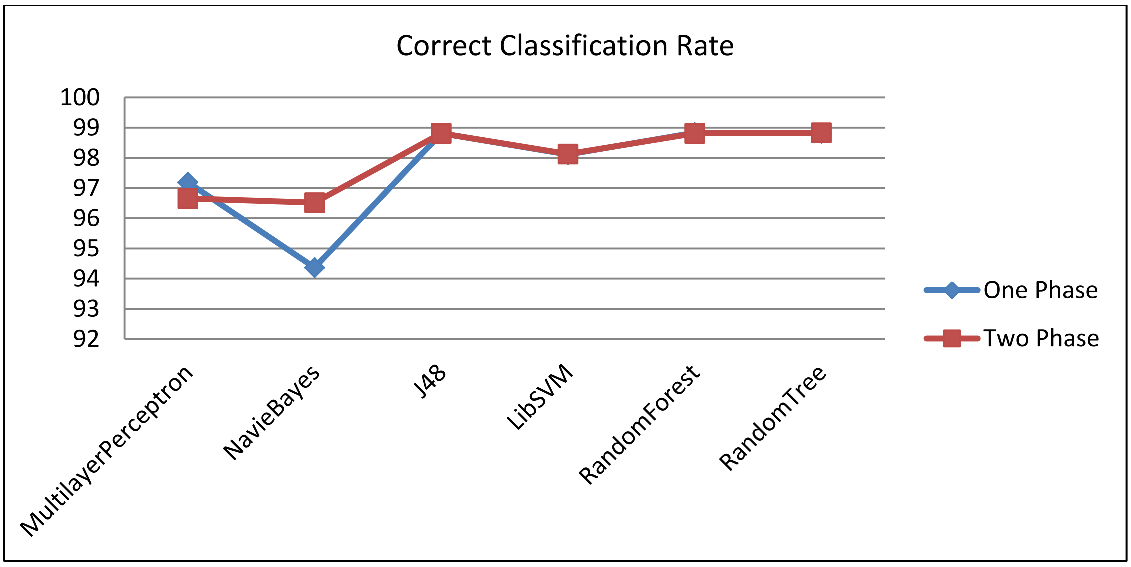
**Figure 4.3: Binary false alarm rates of algorithms.**



*Source*: *This figure has been prepared by Ramazan Karademir.*

In two phase attack detection approach, first phase uses binary classification, and if founds an attack, then second phase classifies the attack type. As figure 4.4 shows only Naïve Bayes algorithm performed better attack detection results compared to one phase approach. Other algorithms performed nearly the same as in one phase.

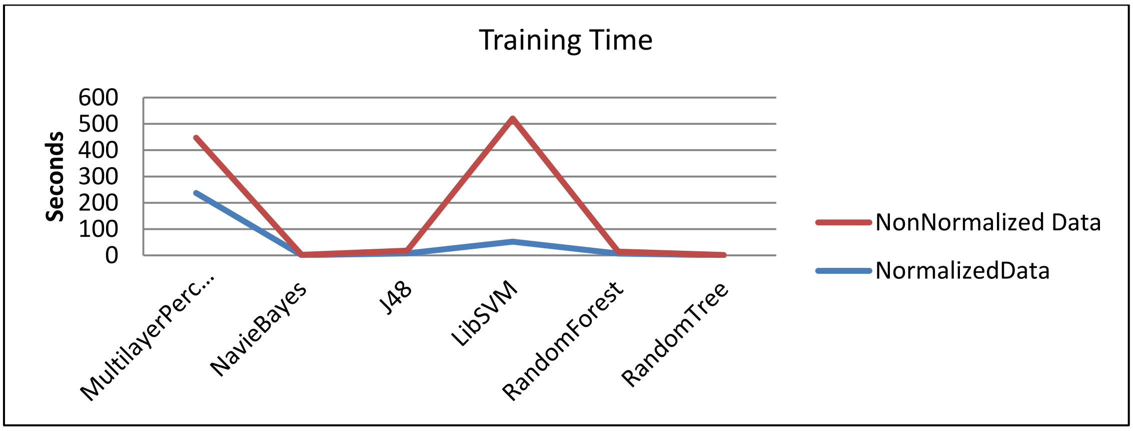
**Figure 4.4: Correct classification rates of two phase classification approach**



*Source*: *This figure has been prepared by Ramazan Karademir.*

In terms of training time, Multilayer perceptron and LibSVM took much more time compared to other algorithms. However, after data normalization, LibSVM performed much better as it can be seen in the figure 4.5.

**Figure 4.5: Training time performances of algorithms.**



*Source*: *This figure has been prepared by Ramazan Karademir.*

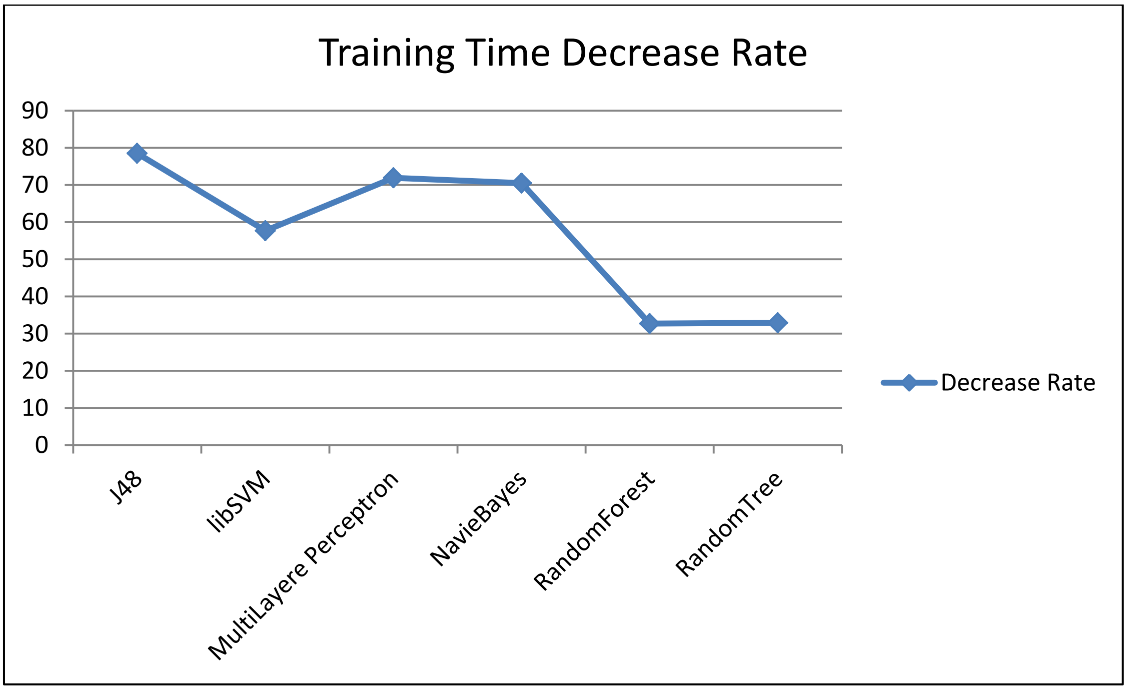
### 4.3.3.2 Feature Selection Evaluations

The following combinations of attribute evaluation and search methods used for feature selections from Weka:

i. BestFirst and CFS Subset Evaluator
ii. GeneticSearch and CFS Subset Evaluator
iii. Greedy Stepwise and CFS Subset Evaluator
iv. Attribute ranking and Chi-squared
v. Attribute ranking and Gain Ratio
vi. Attribute ranking and Info Gain

After the feature selection every feature subset evaluated with six classification algorithms in terms of training time and correct classification rate. The selected attributes and evaluation results can be seen tables on appendix E. As the figure 4.6 shows training times of algorithms decreased in the range between 32 and 78 percent.

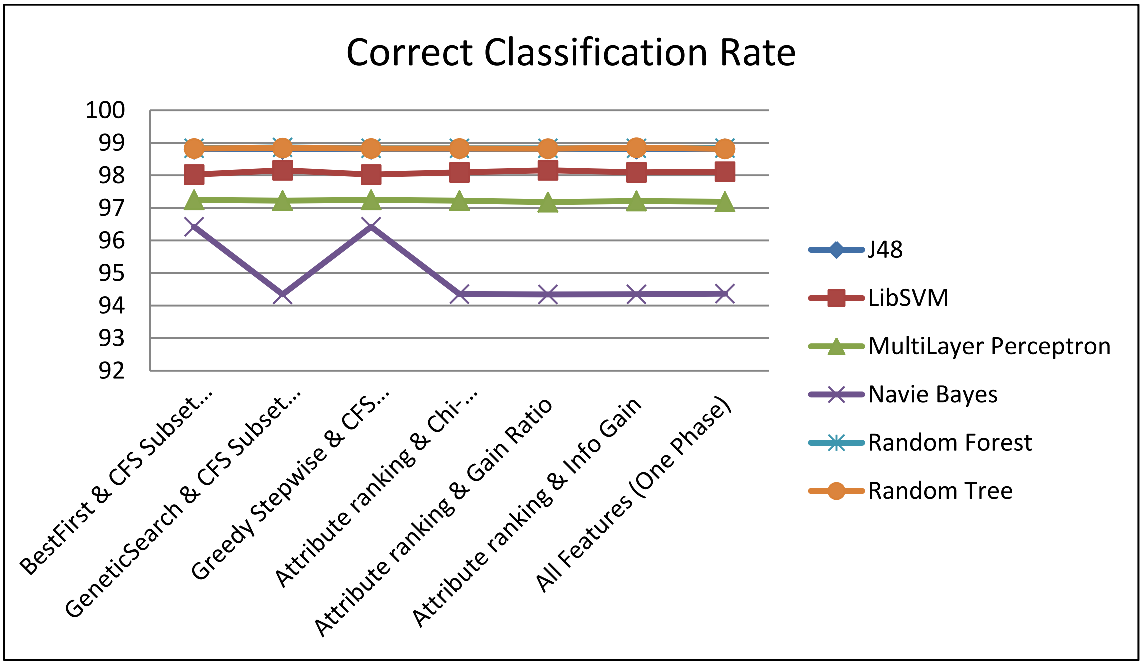
**Figure 4.6: Training time decrease rate after feature selection.**



*Source: This figure has been prepared by Ramazan Karademir.*

After feature selection only Naïve Bayes algorithm performed differently for each feature subset. Others performed nearly the same in terms of correct attack classification as shown in figure 4.7.

**Figure 4.7: Correct classification rate after feature selection.**



*Source*: *This figure has been prepared by Ramazan Karademir.*

# 5. CONCLUSIONS

In this thesis study an intrusion detection system for detecting denial of service attacks were designed and implemented. For detecting attacks data mining based K-Means outlier detection approach and Naïve Bayes classification approach were utilized separately and together as a hybrid solution. The approaches were trained and tested against real network data and real denial of service attacks. Real data was captured from Ligtv.com.tr web site and processed into connection based records which contains 41 statistical features. These features are unique to this study. Attacks were also generated and processed with labelling according to attack type into connection records. As an alternative to the real attack data, a lab environment pure attack data were also obtained and processed into connection records. After then, these records were evaluated for attack detection by using different data mining approaches.

K-Means clustering based outlier detection approach achieved 90 percent attack detection rate which seems comparable with other results from literature, whereas false alarm rates as high as 47 percent which is not good. Naïve Bayes based classification approach achieved 99 percent attack detection rate and 0.05 false alarm rate in binary classification for the lab data which is very good. But in real data, the detection rate was 98 percent and false alarm rate 2.33 which is not so well for a classification algorithm. On the other hand, it should not be missed that the real data may have incorrect labeling because of manual labelling process. Nevertheless, the evaluation results were promising. For multi class classification the algorithms performed higher false alarm rates due to close similarities between attack types. Hybrid solution which uses K-means clustering and Naïve Bayes classification achieved a slightly better result than using them separately.

In addition, the following classification algorithms were evaluated from Weka: J48 Decision Tree, Naïve Bayes, Multilayer Perceptron, LibSVM, Random Forest and Random Tree. J48 decision tree classification is an implementation of the C4.5 algorithm which generates a decision tree by using the concept of information entropy.

Naïve Bayes classification is a probabilistic classifier based on Bayes' theorem. Multilayer Perceptron is a feed forward artificial neural network classification algorithm in Weka tool. It can also work on data that are not linearly separable. LibSVM is a library for support vector machine (SVM) algorithm classification by. SVM tries to separate data by using the best hyper plane which provides the largest separation margin between classes. Random Forest based classification uses ensemble learning methods. This method uses bagging idea and random feature selection in order to construct decision trees. Random Tree constructs decision trees with randomly chosen attributes at each node. After the evaluations Random Forest performed the best attack detection rate and false alarm rate which is 99,973 percent and 0,025 percent respectively, where as Naïve Bayes performed the worst attack detection rate and false alarm rate which is 97,501 percent and 2,337 percent respectively. These results showed that ensemble methods achieved better attack detection results.

Finally, six different feature selection methods applied to our dataset namely: BestFirst, GeneticSearch, Greedy Stepwise with CSF Subset Evaluator, and Chi-squared, Gain Ratio, Info Gain with attribute ranking methods. After applying feature selection methods every feature subset evaluated with six classification algorithms in terms of training time and correct classification rate. Naïve Bayes classification performed 2 percent improvement in correct classification rate with BestFirst and Greedy Stepwise feature selection methods. Other algorithms performed nearly the same correct classification rates with less training. Overall, the results showed that with fewer features and less training time, at least the same detection and false alarm rates are achievable.

In the future, for outlier detection approach we will investigate the ways of lowering false alarm rates. For the dataset, we will add new attack types for DDoS as well as other attacks and try to extract new features to discriminate different attack types. Also, an online version of this intrusion detection system implementation will be planned.

# REFERENCES

## Books

Han J., Kamber M., and Pei J., 2011, *Data Mining: Concepts and Techniques,* 3rd edition, Morgan Kaufmann Publishers.

**Periodicals**

Altwaijry, H., 2013. Bayesian based intrusion detection system. *In IAENG Transactions on Engineering Technologies (pp. 29-44). Springer Netherlands.*

Ashok, R., Lakshmi, A. J., Rani, G. D. V., & Kumar, M. N., 2011. Optimized feature selection with k-means clustered triangle SVM for Intrusion Detection. *In Advanced Computing (ICoAC), 2011 Third International Conference on (pp. 23-27). IEEE.*

Beitollahi, H., & Deconinck, G., 2012. Tackling application-layer DDoS attacks. *Procedia Computer Science, 10, 432-441*.

Bellaiche, M., & Gregoire, J. C., 2009. SYN flooding attack detection based on entropy computing. *In Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE (pp. 1-6). IEEE.*

Chen, Z. G., & Kim, S. R., 2013. Combining principal component analysis, decision tree and naïve Bayesian algorithm for adaptive intrusion detection. *In Proceedings of the 2013 Research in Adaptive and Convergent Systems (pp. 312-316). ACM.*

Davis, J.J., and Clark, A.J., 2011, Data preprocessing for anomaly based network intrusion detection: A review. *Computers & Security, Volume 30, Issues 6–7,* pp. 353-375.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I.H., 2009; The WEKA Data Mining Software: An Update; *SIGKDD Explorations, Volume 11, Issue 1.*

Horng, S. J., Su, M. Y., Chen, Y. H., Kao, T. W., Chen, R. J., Lai, J. L., & Perkasa, C. D., 2011. A novel intrusion detection system based on hierarchical clustering and support vector machines. *Expert systems with Applications, 38(1), 306-313.*

Kayacik, H. G., Zincir-Heywood, A. N., & Heywood, M. I., 2005. Selecting features for intrusion detection: A feature relevance analysis on KDD 99 intrusion detection datasets. *In Proceedings of the third annual conference on privacy, security and trust.*

Kim, G., Lee, S., & Kim, S., 2014. A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. *Expert Systems with Applications, 41(4), 1690-1700.*

Lee W., Stolfo S.J., and Mok. K.W., 1999, Mining in a data-flow environment: experience in network intrusion detection. *In Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* (KDD '99), pp. 114-124.

Liu, Z., 2011. A method of svm with normalization in intrusion detection.*Procedia Environmental Sciences, 11, 256-262.*

Mirkovic, J., and Reiher, P., 2004. A taxonomy of DDoS attack and DDoS defense mechanisms. *SIGCOMM Comput. Commun. Rev.*34, 2, pp. 39-53.

Mohammad, M. N., Sulaiman, N., & Muhsin, O. A., 2011. A novel intrusion detection system by using intelligent data mining in weka environment.*Procedia Computer Science, 3, 1237-1242.*

Muda, Z.; Yassin, W.; Sulaiman, M.N.; Udzir, N.I., 2011,Intrusion detection based on K-Means clustering and Naïve Bayes classification, *Information Technology in Asia (CITA 11), 2011 7th International Conference on, vol.1, no.6,* pp.12-13.

Mukherjee, S., & Sharma, N., 2012. Intrusion detection using naive Bayes classifier with feature reduction. *Procedia Technology, 4, 119-128.*

Om, H., & Kundu, A., 2012. A hybrid system for reducing the false alarm rate of anomaly intrusion detection system. *In Recent Advances in Information Technology (RAIT), 2012 1st International Conference on (pp. 131-136). IEEE.*

Onut, I. V., & Ghorbani, A. A., 2007. A Feature Classification Scheme For Network Intrusion Detection. *IJ Network Security, 5(1), 1-15.*

Portnoy L., Eskin E., and Stolfo S., 2001, Intrusion detection with unlabeled data using clustering, *In Proceedings of ACM CSS Workshop on Data Mining Applied to Security (DMSA-2001),*pp. 5-8.

Sperotto, A.; Schaffrath, G.; Sadre, R.; Morariu, C.; Pras, A.; Stiller, B., 2010. An Overview of IP Flow-Based Intrusion Detection, *Communications Surveys & Tutorials, IEEE, vol.12, no.3,* pp.343-356.

Tavallaee, M.; Bagheri, E.; Wei Lu; Ghorbani, A.A., 2009, A Detailed Analysis of the KDD CUP 99 Data Set, *Computational Intelligence for Security and Defense Applications, 2009. CISDA 2009. IEEE Symposium on, vol.1, no.6*, pp.8-10.

Wankhade, K., Patka, S., & Thool, R., 2013. An efficient approach for Intrusion Detection using data mining methods. *In Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on(pp. 1615-1618). IEEE.*

Zhong S., Khoshgoftaar T.M., and Seliya N., 2007, Clustering Based Network Intrusion Detection, *Int. J. Rel. Qual. Saf. Eng.14*, pp. 169

## Other References

Kddcup, KDD Cup 1999 Data, **http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html** [accessed 01-03-2014]

Ries, B., 2010, Hackers' Most Destructive Attacks, http://www.thedailybeast.com/articles/2010/12/11/hackers-10-most-famous-attacks-worms-and-ddos-takedowns.html, [accessed 12-08-2014]

Wikipedia, NetFlow, http://en.wikipedia.org/wiki/NetFlow [accessed 10-07-2014]

NIST, National Institute of Standards and Technology, Guide to Intrusion Detection and Prevention Systems, 2007, http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf [accessed 12-04-2014]

# APPENDICES

## APPENDIX A: TShark Field Extract Command.

The following command is used from the Microsoft Windows command prompt window to extract fields from packet capture files and write to comma separated file for further processing.

```
tshark.exe -r D:\DDosParse\packetsXXX -T fields -e frame.number -e frame.time -e frame.time_delta -e frame.len -e frame.protocols -e eth.type -e ip.version -e ip.hdr_len -e ip.dsfield -e ip.len -e ip.id -e ip.flags -e ip.frag_offset -e ip.ttl -e ip.protocol -e ip.checksum -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport -e tcp.seq -e tcp.ack -e tcp.hdr_len -e tcp.flags -e tcp.window_size_value -e tcp.checksum -e tcp.options -e udp.srcport -e udp.dstport -e udp.length -e udp.checksum -e http.host -e http.request.method -e http.request.uri -e http.content_length -e http.content_length_header -e http.response.code -e http.response.phrase -e http.referer -E header=y -E separator=} -E quote=d -E occurrence=f > D:\DDoSParse\packetsXXX.csv
```

## APPENDIX B: Feature Name Descriptions.

### Table B.1: Descriptions of constructed features for basic category.

| ID | Feature | Description |
|---|---|---|
| 1 | duration numeric | Length (number of seconds) of the connection |
| 2 | tcp_proto_count numeric | # of packets that contain tcp protocol |
| 3 | http_proto_count numeric | # of packets that contain http protocol |
| 4 | data_proto_count numeric | # of packets that contain data protocol |
| 5 | ssl_proto_count numeric | # of packets that contain ssl protocol |
| 6 | other_proto_count numeric | # of packets that contain protocols other than the above |
| 7 | total_frame_len numeric | frame length |
| 8 | avg_frame_len numeric | average frame lentgh |
| 9 | protocol {tcp,xxx} | Type of the protocol, e.g., TCP, UDP, etc. |
| 10 | network_service {80,443,1433,445,290,139,0,135,21,22,23,9999} | Network service on the destination, e.g., http, telnet, etc. |
| 11 | src_bytes | Number of data bytes from source to destination |
| 12 | dst_bytes numeric | Number of data bytes from destination to source |
| 13 | flag_normal_open {1,0} | Connections that have normal open sequence flags (SYN, SYN-ACK etc.) |
| 14 | flag_normal_close {1,0} | Connections that have normal close sequence flags (FIN, FIN-ACK etc.) |
| 15 | flag_reset {1,0} | Connections that have RST flag set |
| 16 | src_packet_count numeric | Number of packets from source to destination |
| 17 | dst_packet_count numeric | Number of packets from destination to source |
| 18 | http_request_method_count numeric | Number of http request messages |
| 19 | http_response_OK_count numeric | Number of http response messages that have OK response |
| 20 | http_response_NOK_count numeric | Number of http response messages that have NOT OK response |
| 21 | http_referer_count numeric | Number of http request messages that have referrer |
| 22 | http_request_uri_count numeric | Number of distinct URIs |
| 23 | http_content_length numeric | length of content |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table B.2: Descriptions of constructed features for time based category.**

| ID | Feature | Description |
|---|---|---|
| 24 | tw_shConnectionCount numeric | Number of connections to the same host as the current  connection in the past 2 seconds. |
| 25 | tw_shSYNErrorRate numeric | % of connections that have “SYN” errors |
| 26 | tw_shResetRate numeric | % of connections that have “RST” errors |
| 27 | tw_shSameServiceRate numeric | % of connections to the same service |
| 28 | tw_shDiffServiceRate numeric | % of connections to different services |
| 29 | tw_ssConnectionCount numeric | Number of connections to the same service  as the current  connection in the past 2 seconds. |
| 30 | tw_ssSYNErrorRate numeric | % of connections that have “SYN” errors |
| 31 | tw_ssResetRate numeric | % of connections that have “RST” errors |
| 32 | tw_ssDiffHostRate numeric | % of connections to different hosts |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table B.3: Descriptions of constructed features for connection based category.**

| ID | Feature | Description |
|---|---|---|
| 33 | cw_shConnectionCount numeric | Number of connections to the same host as the current  connection in the past 200 connection |
| 34 | cw_shSYNErrorRate numeric | % of connections that have “SYN” errors |
| 35 | cw_shResetRate numeric | % of connections that have “RST” errors |
| 36 | cw_shSameServiceRate numeric | % of connections to the same service |
| 37 | cw_shDiffServiceRate numeric | % of connections to different services |
| 38 | cw_ssConnectionCount numeric | Number of connections to the same service  as the current  connection in the past 200 connection |
| 39 | cw_ssSYNErrorRate numeric | % of connections that have “SYN” errors |
| 40 | cw_ssResetRate numeric | % of connections that have “RST” errors |
| 41 | cw_ssDiffHostRate numeric | % of connections to different hosts |

*Source*: *This table has been prepared by Ramazan Karademir.*

## APPENDIX C: Confusion Matrix Details for Attack Classes

### Table C.1: Multilayer Perceptron multiclass classification confusion matrix.

| syn_ack ddos | icmp ddos | rst_ack ddos | rst ddos | fin ddos | ack ddos | http get | syn ddos | Normal | <-- classified as |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 0 | 0 | 0 | 0 | 1 | 0 | 485 | 0 | syn_ack_ddos |
| 7 | 0 | 0 | 0 | 0 | 2 | 0 | 9 | 0 | icmp_ddos |
| 0 | 0 | 1127 | 1 | 0 | 0 | 0 | 0 | 0 | rst_ack_ddos |
| 0 | 0 | 759 | 0 | 0 | 0 | 0 | 0 | 0 | rst_ddos |
| 0 | 0 | 0 | 0 | 0 | 2 | 0 | 4 | 0 | fin_ddos |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 327 | 0 | ack_ddos |
| 0 | 0 | 0 | 0 | 0 | 0 | 1194 | 0 | 0 | http_get |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 1279 | 0 | syn_ddos |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 51750 | Normal |

*Source*: *This table has been prepared by Ramazan Karademir.*

### Table C.2: Naïve Bayes multiclass classification confusion matrix.

| syn_ack ddos | icmp ddos | rst_ack ddos | rst ddos | fin ddos | ack ddos | http get | syn ddos | Normal | <-- classified as |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 153 | 0 | 0 | 0 | 327 | 0 | 1 | 5 | syn_ack_ddos |
| 0 | 17 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | icmp_ddos |
| 0 | 0 | 23 | 1105 | 0 | 0 | 0 | 0 | 0 | rst_ack_ddos |
| 0 | 0 | 8 | 751 | 0 | 0 | 0 | 0 | 0 | rst_ddos |
| 0 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | fin_ddos |
| 0 | 316 | 0 | 0 | 0 | 7 | 0 | 0 | 4 | ack_ddos |
| 0 | 0 | 0 | 0 | 0 | 0 | 1194 | 0 | 0 | http_get |
| 2 | 138 | 0 | 0 | 2 | 1124 | 0 | 2 | 14 | syn_ddos |
| 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 51749 | Normal |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table C.3: J48 multiclass classification confusion matrix.**

| syn_ack ddos | icmp ddos | rst_ack ddos | rst ddos | fin ddos | ack ddos | http get | syn ddos | Normal | <-- classified as |
|---|---|---|---|---|---|---|---|---|---|
| 263 | 0 | 0 | 0 | 0 | 27 | 0 | 198 | 0 | syn_ack_ddos |
| 0 | 17 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | icmp_ddos |
| 0 | 0 | 1028 | 100 | 0 | 0 | 0 | 0 | 0 | rst_ack_ddos |
| 0 | 0 | 203 | 556 | 0 | 0 | 0 | 0 | 0 | rst_ddos |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 5 | 0 | fin_ddos |
| 10 | 0 | 0 | 0 | 0 | 283 | 0 | 34 | 0 | ack_ddos |
| 0 | 0 | 0 | 0 | 0 | 0 | 1194 | 0 | 0 | http_get |
| 67 | 0 | 0 | 0 | 1 | 28 | 0 | 1186 | 0 | syn_ddos |
| 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 51749 | Normal |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table C.4: LibSVM multiclass classification confusion matrix.**

| syn_ack ddos | icmp ddos | rst_ack ddos | rst ddos | fin ddos | ack ddos | http get | syn ddos | Normal | <-- classified as |
|---|---|---|---|---|---|---|---|---|---|
| 127 | 0 | 14 | 0 | 0 | 67 | 0 | 278 | 2 | syn_ack_ddos |
| 0 | 12 | 0 | 0 | 1 | 0 | 0 | 0 | 5 | icmp_ddos |
| 3 | 0 | 1030 | 84 | 0 | 2 | 0 | 5 | 4 | rst_ack_ddos |
| 4 | 0 | 239 | 507 | 0 | 1 | 0 | 8 | 0 | rst_ddos |
| 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 4 | fin_ddos |
| 5 | 0 | 6 | 1 | 0 | 284 | 0 | 30 | 1 | ack_ddos |
| 0 | 0 | 0 | 0 | 0 | 0 | 949 | 0 | 245 | http_get |
| 4 | 0 | 10 | 0 | 0 | 48 | 0 | 1217 | 3 | syn_ddos |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 51752 | Normal |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table C.5: Random Forest multiclass classification confusion matrix.**

| syn_ack ddos | icmp ddos | rst_ack ddos | rst ddos | fin ddos | ack ddos | http get | syn ddos | Normal | <-- classified as |
|---|---|---|---|---|---|---|---|---|---|
| 285 | 0 | 0 | 0 | 0 | 29 | 0 | 174 | 0 | syn_ack_ddos |
| 0 | 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | icmp_ddos |
| 0 | 0 | 991 | 137 | 0 | 0 | 0 | 0 | 0 | rst_ack_ddos |
| 0 | 0 | 157 | 602 | 0 | 0 | 0 | 0 | 0 | rst_ddos |
| 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | fin_ddos |
| 11 | 0 | 0 | 0 | 0 | 281 | 0 | 35 | 0 | ack_ddos |
| 0 | 0 | 0 | 0 | 0 | 0 | 1194 | 0 | 0 | http_get |
| 89 | 0 | 0 | 0 | 0 | 31 | 0 | 1162 | 0 | syn_ddos |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 51751 | Normal |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table C.6: Random Tree multiclass classification confusion matrix.**

| syn_ack ddos | icmp ddos | rst_ack ddos | rst ddos | fin ddos | ack ddos | http get | syn ddos | Normal | <-- classified as |
|---|---|---|---|---|---|---|---|---|---|
| 282 | 0 | 0 | 0 | 0 | 30 | 0 | 176 | 0 | syn_ack_ddos |
| 0 | 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | icmp_ddos |
| 1 | 0 | 969 | 158 | 0 | 0 | 0 | 0 | 0 | rst_ack_ddos |
| 0 | 0 | 137 | 622 | 0 | 0 | 0 | 0 | 0 | rst_ddos |
| 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | fin_ddos |
| 14 | 0 | 0 | 0 | 0 | 280 | 0 | 33 | 0 | ack_ddos |
| 0 | 0 | 0 | 0 | 0 | 0 | 1194 | 0 | 0 | http_get |
| 94 | 0 | 0 | 0 | 0 | 30 | 0 | 1158 | 0 | syn_ddos |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 51752 | Normal |

*Source*: *This table has been prepared by Ramazan Karademir.*

## APPENDIX D: Weka Classification and Feature Selection Result

**Table D.1: Weka classification algorithms evaluations results.**

| Classification Method | TP | FP | TN | FN | Accuracy | Detection Rate | False Alarm Rate | F1 Measure |
|---|---|---|---|---|---|---|---|---|
| Naïve Bayes | 58.990 | 1.512 | 63.184 | 268 | 98,564 | 97,501 | 2,337 | 98,514 |
| LibSVM | 59.189 | 1.053 | 63.643 | 69 | 99,095 | 98,252 | 1,628 | 99,061 |
| Multilayer Perceptron | 59.164 | 353 | 64.343 | 94 | 99,639 | 99,407 | 0,546 | 99,624 |
| Random Tree | 59.217 | 43 | 64.653 | 41 | 99,932 | 99,927 | 0,066 | 99,929 |
| J48 Decision tree | 59.209 | 35 | 64.661 | 49 | 99,932 | 99,941 | 0,054 | 99,929 |
| Random Forest | 59.218 | 16 | 64.680 | 40 | 99,955 | 99,973 | 0,025 | 99,953 |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table D.2: Correct classification rates of two phase classification.**

| | Correct Classification Rate | |
|---|---|---|
| | One Phase | Two Phase |
| Multilayer Perceptron | 97,187 | 96,653 |
| Naïve Bayes | 94,366 | 96,518 |
| J48 | 98,81 | 98,81 |
| LibSVM | 98,113 | 98,128 |
| Random Forest | 98,834 | 98,811 |
| Random Tree | 98,818 | 98,834 |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table D.3: Data normalizations effects on training time for algorithms.**

| | Normalized Data | NonNormalized Data |
|---|---|---|
| Multilayer Perceptron | 2372,04 | 2101,55 |
| Naïve Bayes | 0,83 | 0,95 |
| J48 | 7,75 | 9,88 |
| LibSVM | 52,06 | 4681,67 |
| Random Forest | 6,71 | 7,61 |
| Random Tree | 0,36 | 0,76 |

*Source*: *This table has been prepared by Ramazan Karademir.*

**Table D.4: Selected features after feature selection methods evaluatons.**

| Feature Selection Method | Number of selected Features | Selected Features |
|---|---|---|
| BestFirst & CFS Subset Evaluator | 9 | network_service, dst_bytes, tw_shConnectionCount, tw_shSYNErrorRate, cw_shConnectionCount, cw_shResetRate, cw_shSameServiceRate, cw_ssSYNErrorRate, cw_ssResetRate |
| GeneticSearch & CFS Subset Evaluator | 17 | tcp_proto_count, avg_frame_len, protocol, flag_normal_open, flag_normal_close, http_response_OK_count, tw_shConnectionCount, tw_shResetRate, tw_shSameServiceRate, tw_ssSYNErrorRate, tw_ssResetRate, tw_ssDiffHostRate, cw_shConnectionCount, cw_shSYNErrorRate, cw_shResetRate, cw_ssResetRate, cw_ssDiffHostRate |
| Greedy Stepwise & CFS Subset Evaluator | 9 | network_service, dst_bytes, tw_shConnectionCount, tw_shSYNErrorRate, cw_shConnectionCount, cw_shResetRate, cw_shSameServiceRate, cw_ssSYNErrorRate, cw_ssResetRate |
| Attribute ranking & Chi-squared | 25 | tw_shConnectionCount, tw_ssConnectionCount, total_frame_len, src_bytes, tcp_proto_count, dst_bytes, cw_shResetRate, cw_ssResetRate, avg_frame_len, cw_shConnectionCount, tw_ssResetRate, cw_ssSYNErrorRate, cw_shSYNErrorRate, tw_shResetRate, network_service, cw_ssConnectionCount, tw_ssSYNErrorRate, http_content_length, tw_shSYNErrorRate, src_packet_count, protocol, cw_ssDiffHostRate, tw_ssDiffHostRate, dst_packet_count, duration |
| Attribute ranking & Gain Ratio | 25 | protocol, cw_shResetRate, cw_ssDiffHostRate, cw_ssResetRate, tw_ssDiffHostRate, tw_ssResetRate, network_service, tw_shResetRate, cw_ssSYNErrorRate, cw_shSYNErrorRate, cw_shConnectionCount, dst_bytes, flag_normal_open, tw_ssSYNErrorRate, tw_shSYNErrorRate, total_frame_len, src_bytes, tw_shConnectionCount, avg_frame_len, tcp_proto_count, flag_normal_close, flag_reset, http_content_length, duration, src_packet_count |
| Attribute ranking & Info Gain | 25 | tw_shConnectionCount, cw_shResetRate, total_frame_len, tw_ssConnectionCount, dst_bytes, cw_ssResetRate, cw_ssSYNErrorRate, avg_frame_len, src_bytes, cw_shConnectionCount, cw_shSYNErrorRate, tw_ssResetRate, tw_ssSYNErrorRate, cw_ssDiffHostRate, tw_shResetRate, tw_ssDiffHostRate, tw_shSYNErrorRate, tcp_proto_count, src_packet_count, network_service, cw_ssConnectionCount, dst_packet_count, http_content_length, duration, http_proto_count |

*Source*: *This table has been prepared by Ramazan Karademir.*