DDOS ATTACK DETECTION USING SIGNAL PROCESSING AND STATISTICAL APPROACHES

by

Derya Erhan

B.S., Department of Electrical and Electronics Engineering, Hacettepe University, 2003

M.S., Department of Electrical and Electronics Engineering, Boğaziçi University, 2007

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Graduate Program in Department of Electrical and Electronics Engineering
Boğaziçi University
2021

ABSTRACT

DDOS ATTACK DETECTION USING SIGNAL PROCESSING AND STATISTICAL APPROACHES

DDoS attacks cause a variety of changes in the properties of the attributes in the network traffic. Modeling these changes using signal processing and statistical approaches provides detection of these attacks. This thesis focuses on detecting DDoS attacks using time series analysis, sparse signal representation methods, and statistical modeling. We also investigate the effect of DDoS attacks on traffic features in a statistical manner. In addition, we propose two simple but effective network-based DDoS attack detection methods based on the statistical signal processing approach, using the advantage of statistical changes in traffic features.

We propose a novel DDoS detection framework using the Matching Pursuit algorithm to detect resource depletion type DDoS attacks. We use multiple characteristics of network traffic simultaneously to detect low-density DDoS attacks efficiently. The proposed method uses the dictionary produced from the parameters of the network traffic using the K-SVD algorithm. Dictionary generation using network traffic provides legitimate and attack traffic models and adds adaptability to the proposed method to network traffic. We also implement DDoS detection approaches that use Matching Pursuit and Wavelet techniques and compare them using two different data sets. Additionally, we offer a hybrid DDoS detection framework that combines these approaches with a decision-making mechanism using an artificial neural network. We evaluate the proposed methods with two different data sets. In the hybrid intrusion detection system with more than one attack, the detection performances of other approaches have decreased. In contrast, the proposed method achieves true-positive rates higher than 99% with a false positive rate lower than 0.7%.

ÖZET

SİNYAL İŞLEME VE İSTATİSTİKSEL YAKLAŞIMLARLA DDOS SALDIRI TESPİTİ

DDoS saldırıları, ağ trafiğindeki özniteliklerin özelliklerinde çeşitli değişikliklere neden olur. Bu değişikliklerin sinyal işleme ve istatistiksel yaklaşımlar kullanılarak modellenmesi bu saldırıların tespit edilmesini sağlar. Bu tez, zaman serisi analizi, seyrek sinyal temsil yöntemleri ve istatistiksel modelleme kullanarak DDoS saldırılarını tespit etmeye odaklanmaktadır. Ayrıca bu tezdeki çalışmalar, DDoS saldırılarının trafik özellikleri üzerindeki etkisini istatistiksel olarak incelemektedir. Trafik özelliklerindeki istatistiksel değişikliklerin avantajını kullanarak istatistiksel sinyal işleme yaklaşımına dayalı iki basit ama etkili ağ tabanlı DDoS saldırı tespit yöntemi öneriyoruz.

Ayrıca, kaynak tükenme tipi DDoS saldırılarını tespit etmek için Matching Pursuit algoritmasını kullanan yeni bir DDoS tespit çerçevesi öneriyoruz. Bu yöntemde düşük yoğunluklu DDoS saldırılarını verimli bir şekilde tespit etmek için ağ trafiğinin birden çok özniteliğini aynı anda kullanmaktayız. K-SVD algoritması kullanılarak ağ trafiğinin parametrelerinden üretilen sözlüğü kullanmaktayız. Bu yöntem önerdiğimiz DDoS tespiti yaklaşımına uyarlanabilir bir özellik sağlamaktadır. Ayrıca Matching Pursuit ve Wavelet tekniklerini kullanan yaklaşımları önerdiğimiz yöntemle karşılaştırmak için kendi veri setlerimizle uyguladık. Ek olarak, bu tezde, yöntemleri yapay sinir ağı kullanan bir karar verme mekanizmasıyla birleştiren hibrit bir DDoS algılama çerçevesi sunmaktayız. Önerilen yöntemleri iki farklı veri seti ile değerlendiriyoruz. Birden fazla saldırıya sahip hibrit saldırı tespit sisteminde diğer yöntemlerin tespit performansları düşmekte iken önerdiğimiz DDoS tespit yaklaşımı ile %99'dan büyük doğru kestirim oranı ve %0.7'den düşük yanlış alarm oranları elde ettik.

TABLE OF CONTENTS

ABSTRACT i
ÖZET ii
LIST OF FIGURES vi
LIST OF TABLES x
LIST OF SYMBOLS xii
LIST OF ACRONYMS/ABBREVIATIONS xiv
1. INTRODUCTION 1
1.1. Literature Review 2
1.2. Contributions 7
1.3. Outline of the Thesis 8
2. DDoS ATTACKS and INTRUSION DETECTION SYSTEMS 10
2.1. DDoS attacks 10
2.1.1. TCP flood Attacks 11
2.1.2. UDP flood attacks 12
2.1.3. ICMP flood attacks 13
2.2. Intrusion Detection Systems 14
2.2.1. Performance Evaluation Metrics 15
2.3. Chapter Summary 18
3. DATASETS and TRAFFIC FEATURES 19
3.1. Datasets 19
3.1.1. DETER Testbed 19
3.1.1.1. Background Traffic 20
3.1.1.2. Simulation Set Up 22
3.1.1.3. A comprehensive simulation with attack traffic 27
3.1.2. Boğaziçi University DDoS Dataset 28
3.1.3. The CAIDA Datasets 34
3.2. Traffic Features 35
3.2.1. Preprocessing 35

3.2.2. Feature Generation . . . 35
3.2.3. Normalization . . . 37
3.2.4. Features . . . 39
3.2.4.1. Packet Based Features . . . 39
3.2.4.2. Flow Based Features . . . 41
3.2.5. Feature Selection . . . 42
3.2.5.1. Chi-Squared Feature Selection . . . 42
3.2.5.2. Information Gain Feature Selection . . . 43
4. DETECTION USING AUTO-REGRESSIVE MODELING . . . 45
4.1. Abrupt change detection . . . 45
4.1.1. ICMP Flood Attack . . . 50
4.1.2. TCP SYN Flood Attacks . . . 52
4.1.3. UDP Flood Attacks . . . 54
4.2. Results and Conclusion . . . 55
5. DDoS DETECTION with SIGNAL REPRESENTATION METHODS . . . 57
5.1. Sparse Signal Representation . . . 57
5.1.1. Matching Pursuit . . . 59
5.1.2. Basic Matching Pursuit Algorithm . . . 60
5.1.2.1. K-SVD Dictionary Generation Algorithm . . . 61
5.2. Adaptive Matching Pursuit Based DDoS Detection . . . 63
5.2.1. Features and Feature Generation . . . 64
5.2.2. Dictionary Generation . . . 66
5.2.3. Alarm Generation . . . 67
5.3. Detection Using Matching Pursuit Mean Projection . . . 69
5.4. Detection Using Wavelet Transform . . . 72
5.5. Hybrid Detection Framework with Signal Representation Methods . . . 73
5.5.1. Decision Module . . . 74
5.5.2. Hybrid Detection Framework Training and Alarm Generation . . . 75
5.5.3. Hybrid Detection Method using Wavelet and MPMP . . . 76
5.6. Evaluation . . . 78
5.6.1. Algorithm Complexity Analysis . . . 78

5.6.2. Evaluation for Two Traffic Classes 79
5.6.3. Evaluation of Hybrid AMP Framework 81
5.7. Summary and Conclusion 84
6. DDoS DETECTION USING STATISTICAL METHODS 85
6.1. Statistical Properties of DDoS Attacks 85
6.1.1. Probability Distribution Fitting 86
6.1.2. Likelihood Function 86
6.1.3. Akaike Information Criterion 86
6.1.4. Bayesian Information Criterion 87
6.1.5. Probability Distribution Functions 89
6.1.5.1. Gaussian (Normal) Distribution 89
6.1.5.2. Generalized Extreme Value Distribution 90
6.1.5.3. Logistic Distribution 91
6.1.6. Binary Hypothesis Testing 91
6.1.7. Experimental Results and Discussion 92
6.2. DDoS Detection Using Statistical Modelling 94
6.2.1. Empirical Model Generation 96
6.2.2. Alarm Generation 98
6.3. Experimental Results and Discussion 101
6.4. Summary and Conclusion 102
7. CONCLUSION and FUTURE WORK 104
REFERENCES 107

LIST OF FIGURES

Figure 2.1. TCP SYN Flood attack network topology. 12

Figure 2.2. UDP Flood attack network topology. 13

Figure 2.3. ICMP Flood attack network topology. 14

Figure 3.1. NS file used to generate DETER testbed experiment. 23

Figure 3.2. Topology of experiments. 23

Figure 3.3. Attack-free traffic generated using all traffic generators in SEER. 24

Figure 3.4. Traffic features of experiment 1. Legitimate traffic includes www, FTP, DNS, ICMP, Harpoon, and SSH. 25

Figure 3.5. Traffic graph of DDoS simulation in DETER testbed. 28

Figure 3.6. Traffic feature vectors obtained from DETER testbed experiment. 29

Figure 3.7. Network Topology of BOUN DDoS Dataset traffic generation and recording. 30

Figure 4.1. Piece-wise stationary segments. 45

Figure 4.2. Traffic features obtained from ICMP flood attacks. 51

Figure 4.3. Traffic health function for one of the ICMP flood attack experiments in DETER testbed using information gain selected features and attack instances. 52

Figure 4.4. Traffic features obtained from TCP SYN flood attacks. 53

Figure 4.5. Traffic health function for TCP flood attacks using information gain selected features and attack instances. 53

Figure 4.6. Feature vectors that can used for manual selection of features. . . 54

Figure 4.7. Traffic health function for UDP flood attacks. 55

Figure 5.1. Block diagram of the AMP DDoS Detection approach. 64

Figure 5.2. Mean of the characteristic feature vectors for Attack and Normal samples in CAIDA'07 and CAIDA'08 dataset. 65

Figure 5.3. Block diagram of dictionary generation module. 67

Figure 5.4. AMP Alarm Generation pseudo-code. 68

Figure 5.5. Normalized histograms of abnormality vectors obtained using anomaly and misuse dictionaries for the CAIDA dataset in the AMP method. 69

Figure 5.6. Average MPMP values obtained for attack and normal traffic using 16 feature vectors. 70

Figure 5.7. Wavelet transformation with three decomposition levels. 73

Figure 5.8. Artificial Neural Network structure used as a decision module. The number of input layer changes depending on the detection method. The number of output layer changes depending on the number of traffic classes to be detected. 74

Figure 5.9. Block diagram of AMP-based Hybrid DDoS detection framework, MPMP, and Wavelet methods with decision module. 76

Figure 5.10. Training Hybrid DDoS detection framework based on the AMP method for multiple traffic classes. 77

Figure 5.11. Comparison of CID and TPR values for AMP, MPMP and Wavelet based hybrid DDoS detection framework using two traffic classes in BOUN UDP and TCP SYN flood datasets. 81

Figure 5.12. Comparison of CID and TPR values for AMP, MPMP and Wavelet based hybrid DDoS detection framework using two three classes in BOUN UDP and TCP SYN flood datasets. 82

Figure 6.1. Examples of probability distribution fits for attack and normal classes of (a) number of unique sources, (b) number of UDP packets per flow and (c) number of ICMP packets feature vectors in BOUN DDoS dataset. 89

Figure 6.2. Overall block diagram of DDoS detection using statistical modeling methodology. 95

Figure 6.3. Normalised histograms of four divergence measures calculated for attack and attack-free samples. These divergences are calculated using anomaly model generated by K-Means Clustering. 99

Figure 6.4. Normalised histograms of four divergence measures calculated for attack and attack-free samples. These divergences are calculated using misuse model generated by K-Means Clustering. 100

LIST OF TABLES

Table 1.1. DDoS detection with sparse signal representation in the literature. 3

Table 2.1. Explanations of simple per-class evaluation metrics. 16

Table 3.1. An example of the correlation coefficients between features. 27

Table 3.2. Information about attack instances in BOUN TCP SYN Flood attack dataset. 33

Table 3.3. Information about attack instances in BOUN UDP Flood attack dataset. 33

Table 3.4. Traffic Features and Descriptions. 38

Table 3.5. Information Gain and Chi-Squared Feature Selection for Different Feature Vectors in BOUN TCP Flood Dataset. 44

Table 4.1. Performance metrics obtained for ICMP UDP and TCP SYN flood attacks. 55

Table 5.1. Obtained mean MPMP for TCP Flood Attack. 71

Table 5.2. Detection performance of AMP, MPMP and Wavelet based approaches using CAIDA dataset. 79

Table 5.3. Detection performance of AMP, MPMP and Wavelet based approaches using BOUN TCP SYN flood dataset. 80

Table 5.4. Comparison of AMP, MPMP and Wavelet based DDoS detection for BOUN UDP flood dataset 80

Table 5.5. Comparison of AMP, MPMP and Wavelet based hybrid DDoS detection framework for two traffic classes using BOUN UDP and TCP SYN flood datasets. 82

Table 5.6. A Comparison of Hybrid Detection Framework based on AMP, MPMP and Wavelet using three traffic classes dataset. 83

Table 6.1. Probability Distributions Fit for Selected Features Normal and Attack Classes of BOUN DDoS Dataset. 88

Table 6.2. Accuracy obtained from likelihood ratio test for GEV, Gaussian and Logistic distribuitons for BOUN DDoS dataset. 93

Table 6.3. Accuracy obtained from likelihood ratio test for GEV, Gaussian and Logistic distribuitons for CAIDA datasets. 93

Table 6.4. Accuracy obtained from likelihood ratio test for Selected 3 distributions for CAIDA datasets. 94

Table 6.5. Evaluation of misuse-based DDoS attack detection. 101

Table 6.6. Evaluation of anomaly-based DDoS attack detection. 102

LIST OF SYMBOLS

A_{ij}	The observed frequency
$AR(p)$	An order p Auto-Regressive process
cov	Covariance
D_{JS}	Jensen and Shannon divergence
D_{KL}	Kullback Leibler divergence
D_M	Manhattan distance
e_{mse}	Mean squared error
E	Expected value
E_{ij}	The expected frequency of A_{ij}
E_W	Energy of Wavelet sub-bands
f_n	n_{th} traffic feature
$f(\varepsilon_1, ...\varepsilon_i)$	The joint probability density function
$\hat{f}$	Normalized feature
$g_\gamma(t)$	Real Gabor function
$Gain(F)$	Information gain of a feature F
k	Number of elements in feature vector
$I(x)$	Information of signal x
L	Likelihood ratio
m	The attributes in the feature vector
N	Time window size for AR process
$O(n)$	Algorithm complexity
$P_t(w)$	Empirical distribution for time window t
$\mathbb{R}$	Real numbers
$R(t)$	Learning window for abrupt change detection
$S(t)$	Test window for abrupt change detection
V	Victim node in DETER testbed topology
w	Time window for feature generation
y_i	Characteristic feature vector for i^{th} time window

α	Dictionary Atom
γ	Set of Gabor Dictionary Parameters
$\varepsilon_i(t)$	Residual error obtained using AR process for i^{th} time window
ε_t	iid sequence of $N(0, \sigma^2)$ random variables
η_L	Log Likelihood Ratio
μ_X	Mean of random variable X
ξ_t	Network health indicator for time t
σ_X	Standard Deviation
$\hat{\sigma_R}$	Covariance estimate
$\varphi(t)$	Abnormality Vector for Abrupt Change Detection
χ^2	Chi-Square Value
$\Psi_{m,n}$	A family of discrete wavelet functions.
ψ_i	The abnormality indicator value for the AMP method
$\|.\|_0$	The l^0 Norm

LIST OF ACRONYMS/ABBREVIATIONS

1D	One Dimensional
2D	Two Dimensional
Acc	Accuracy
ACK	Acknowledgement
AIC	Akaike information criterion
AMP	Adaptive Matching Pursuit
ANN	Artificial Neural Network
AR	Auto Regressive
AUC	Area under the ROC Curve
BIC	Bayesian Information Criterion
BOUN	Boğaziçi University
CAIDA	Cooperative Association for Internet Data Analysis
CID	Capability of Intrusion Detection
DBSCAN	Density Based Spatial Clustering of Applications with Noise
DDoS	Distributed Denial of Service Attack
DETER	Defense Technology Experimental Research
DNS	Domain Name System
DoS	Denial of Service Attack
DWT	Discrete wavelet Transform
EMIST	Evaluation Methods for Internet Security Technologies
FN	False negative
FNR	False Negative Rate
FP	False-positive
FPR	False Positive Rate
FTB-OMP	Flexible Tree Search based Orthogonal Matching Pursuit
FTP	File Transfer Protocol
GEV	Generalized Extreme Value
GLR	Generalized Likelihood Ratio

GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IP	Internet Protocol
IRC	Internet Relay Chat
ISI	Information Sciences Institute
JS	Jensen Shannon
KL	Kullback Leibler
LLR	Log Likelihood Ratio
MP	Matching Pursuit
MPMP	Matching Pursuit Mean Projection
PCA	Principal Component Analysis
ROC	Receiver Operating Characteristic
SEER	Security Experimentation Environment
SSH	Secure Shell
SYN	Synchronize
TB-OMP	Tree Based Search Orthogonal Matching Pursuit
TCP	Transmission Control Protocol
TLS	t Location Scale
TN	True negative
TNR	True Negative Rate
TPR	True Positive Rate
UCS	University of Southern California
UDP	User Datagram Protocol

1. INTRODUCTION

The exponential increase in the use of various applications over the internet led to a rise in security threats, such as Distributed Denial of Service (DDoS) attacks [1]. The DDoS attack aims to make an online service unavailable by consuming the target system's bandwidth, memory, or CPU. DDoS is a well-known problem in intrusion detection systems; therefore, there is a comprehensive prior art around the subject. However, DDoS attacks continue to be one of the biggest cyber threats affecting the financial, health, retail, gaming, and political sectors and resulting in financial loss [2,3].

In 2019 DDoS attack size increased by 273%, and 91% of the victims reported that the attack saturated their internet bandwidth. In April 2019, the most comprehensive network and application layer attacks were seen with 580 million packets per second (PPS) [2]. Another attack lasted for 13 days and generated 292,000 Requests Per Second (RPS). Additionally, DDoS attack indicators increased by 84% in the last quarter of 2019 [3]. It is reported that DDoS attacks increased by 55% between January 2020 and March 2021 [4].

In general, DDoS attacks are divided into two groups as bandwidth depletion attacks and resource depletion attacks [5]. Bandwidth depletion attacks deny the service of the target system by flooding the target network with an excessive amount of packets. Resource depletion attacks aim to consume computing resources of the target system using malformed packets that exploit the network protocols. This thesis examines resource depletion flood-type DDoS attacks.

Studies on computer networks have shown that network usage of end users follows a specific pattern. These patterns provide the possibility to model network traffic. The systems used to detect attacks directed to information systems are called intrusion detection systems (IDS). An IDS aims to detect intrusions against computer systems [6] using different approaches. According to the detection methodology, intrusion detection

methods are generally divided into two categories: anomaly and misuse detection [6,7]. Anomaly detection methods build models from the attack-free behavior of the computer system. This method uses the deviation of the system behavior from these models as detection criteria. On the contrary, misuse-based detection uses models or signatures of attacks and searches for computer network behavior similarities with the attack model. We perform both misuse, anomaly, and hybrid detection methods in this thesis.

IDSs are divided into two classes according to the network edge they detect attacks, namely host-based IDS and network-based IDS. Host-based IDS systems monitor the hosts' operating systems, while Network-based IDS monitors whole network activity. The proposed DDoS detection approaches in this thesis use network-based detection.

The scope of this thesis includes the detection of DDoS attacks using signal processing approaches. For this purpose, we first determine the possible features for DDoS as a result of early experiments using the DETER testbed [8] testbed. We select the most suitable features for attacks using the information gain and chi-square feature selection algorithms. We develop auto-regressive modeling, sparse signal representation approaches, and statistical methods for DDoS detection.

1.1. Literature Review

In this thesis, we first focus on handling the traffic features as time series and define linear trends in features. There are anomaly detection methods based on linear changes in time series in the literature [9], [10]. In [9] they generate models from multiple time series for anomaly detection. They used the Greedy-Split algorithm [11] to generate a model for one time series. A sequence of n points in a feature space is first approximated by a sequence of $n-1$ boxes, each enclosing a pair of adjacent points. Then pairs of adjacent boxes are merged by greedily selecting the pair that minimizes the increase in volume after merging. Similar approach is proposed in [10]. They build clusters from time series according to the slope, level, and shape of time series.

Table 1.1. DDoS detection with sparse signal representation in the literature.

Ref.	**Methods**	**Dataset**	**Abnormality Value**	**Remarks**
[12]	MP	MAWI, CAIDA	MPMP	Gabor Dictionary is used, DDoS Detection
[13]	MP, DWT	MAWI, CAIDA	MPMP, Energy of subbands	Gabor Dictionary is used, worm detection
[14]	MP, KSVD	DARPA , CAIDA, KDD	MPMP, Energy of Dictionary Elements	Dictionary is constructed with K-SVD
[15]	MP, DWT	MAWI, CAIDA	MPMP, Energy of subbands	Gabor Dictionary is used, DDoS Detection
[16]	MP	Simulated Dataset	MPMP	Gabor Dictionary is used, DDoS Detection
[17]	OMP	Synthetic Traffic Data, GEANT	Corresponding dictionary atoms	Anomalous and non-anomalous dictionary atoms are generated.
[18]	MP, KSVD	BOUN DDoS	Residual of MP obtained from different dictionaries.	Residuals obtained fromMPis used as detection scores
[19]	PCA, OMP, ANN	Synthetic Traffic Data, Janepese SIP real data.	Residual error	They create basis functions using SVD.

They used an algorithm called Gecko that divides a time series into clusters which are determined by the algorithm and requires no user input. Thottan et al. [20] suggested a statistical analysis method to detect abrupt changes based on AR(1) least squares. They chose three SNMP MIB variables, and variable level alarms were obtained using a change detection algorithm. It has been experimentally shown that changes in the statistics of traffic data can be used to detect faults similar to the ones in [20–22]. Their abrupt change detection methodology is suitable for detection with multiple traffic features. For that reason, we implement abrupt change detection in this thesis. We use network traffic feature vectors instead of SNMP MIB variables.

Although there are many methods proposed in the literature using time series analysis in the literature, the approaches using sparse signal representation methods are limited. Matching Pursuit is one of the sparse representation methods that is used in DDoS detection. MP algorithm-based DDoS detection is first implemented using the Matching Pursuit Mean Projection (MPMP) of the reconstructed traffic feature [12–15]. Renk et al. propose an attack detection framework that utilizes the MP algorithm to create profiles of attack and legitimate traffic in [16]. MPMP-based DDoS detection approach is compared with different signal representation methods (e.g., Discrete Wavelet Transform) in [23]. Also, DDoS detection using MP and Orthogonal Matching Pursuit (OMP) algorithms is proposed in [15]. The OMP algorithm, principal component analysis, robust principal component analysis, and backpropagation neural network methods are used for DDoS detection in [19].

Generating network anomaly characteristic models using an MP-based methodology is used in [17]. They generate anomalous and non-anomalous basis functions to construct a dictionary from labeled data using Discrete Cosine Transformation and Wavelet basis. They use synthetic traffic data, GEANT network backbone router traffic, byte counts recorded from an internet backbone network.

We proposed an MP-based DDoS detection approach in this thesis and published an Adaptive Matching Pursuit (AMP) method in [18]. Unlike other methods that use

the MP algorithm, the AMP approach uses multiple network traffic features. Also, the AMP approach generates alarms using residuals of the MP algorithm. Another novel property of the AMP approach is using a dictionary obtained from multiple network traffic features. Dictionary generation from traffic data provides the adaptation of AMP-based DDoS detection to network traffic.

We summarise methods, datasets, and remarks from literature using the MP algorithm and other sparse representation methods used for DDoS detection in Table 1.1. As we can see from the table, the MPMP and Wavelet methods are essentially utilized for DDoS detection. Because of this, to compare the proposed MP-based DDoS detection approaches, we re-implement MPMP and Wavelet-based methods.

We also develop statistical methodologies for DDoS detection. The statistical behavior of DDoS attacks are generally handled as entropy and information theory metrics of traffic features [24,25] in the literature. The Bayesian classification is another popular statistical model used in the classification of DDoS attacks [26].

Several statistical detection mechanisms have been developed to protect computer networks from DDoS attacks [27–33]. Upon those approaches, we focus on statistical detection methods similar to our methodology.

The first part of the statistical approach for DDoS detection focuses on the statistical representation of traffic features under DDoS attacks. The statistical behavior of DDoS attacks are generally handled as entropy and information theory metrics of traffic features [24, 25] in the literature. The Bayesian classification is another popular statistical model used in the classification of DDoS attacks [26].

Çakmakçı et al. propose a sequential, DDoS detection approach using a kernel-based learning algorithm, the Mahalanobis distance, and a chi-square test in [34]. They applied the kernel-based learning algorithm using the entropy features to detect input vectors and achieved 99.55% Accuracy.

Ahmed et al. propose a traffic classification framework using fingerprints generated from packet-level ad flow level features in [35]. They use a multi-modal probability distribution function in the generation of traffic fingerprints. They extended the proposed classification framework to detect DDoS attacks using statistical information obtained from flow-level traffic data. They achieved an accuracy of 97%, with a false positive rate of 2.5%.

Bouyeddou et al. used Kullback-Leibler distance to detect SYN flood, UDP flood, and Smurf attacks in citebouyeddou2020nonparametric. They proposed using Kullback-Leibler Divergence with exponentially smoothing to improve the detector sensitivity to minor anomalies. They tested their method with the DARPA99 dataset.

Bhunyan et al. propose an adaptive boosting-based learning model for detecting DDoS attacks using Kullback-Leibler, Jensen-Shannon divergence methods. They focus on both low-rate and high-rate DDoS attacks. An evaluation of their approach used the UmU testbed, MIT legitimate, and CAIDA DDoS datasets and obtained an accuracy of 98.09%, while the false-positive rate was 0.0213%.

Ateş et al. propose an anomaly detection method based on the probability distribution functions generated from header information of network packets in [36]. They employed a Greedy algorithm to calculate the divergence of the incoming traffic from the developed statistical model. They also used the Support Vector Machine classifier in the detection phase to reduce false alarm rates. They evaluated their approach with BOUN dataset [37] and MIT Darpa 2000 dataset.

In this thesis, we propose a simple but effective network-based DDoS attack detection method based on a statistical signal processing approach in [38]. We added Greedy Algorithm, Jensen-Shannon divergence, Mahalanobis distance, Manhattan distance metrics, and DBSCAN clustering algorithm and compared the different approaches obtained.

We employed K-means, Density-based spatial clustering of applications with noise (DBSCAN) clustering algorithm, Jensen and Shannon divergence, and Greedy algorithms to perform the proposed DDoS detection approach.

1.2. Contributions

This thesis includes understanding DDoS attacks, creating the necessary attributes for their detection, and proposing novel DDoS detection techniques. The first part of the thesis consists of the data sets created/analyzed in order to understand DDoS attacks. We produce the necessary features for detecting attacks. In addition, the thesis includes novel DDoS detection methods using signal processing, signal representation, and statistical methods. Some of the contributions of this thesis have been published in [18, 38–42].

The main contribution of this thesis is to obtain a better understanding of DDoS attacks and develop novel methodologies to detect them. We give the contributions of this thesis according to the proposed methodology.

Simulations using the DETER testbed are performed to understand the effect of attacks on network traffic. The features used to detect DDoS attacks are generated using these features. Sixteen features are developed from these simulations.

To handle the features generated from DETER testbed as time series, we use the abrupt change detection proposed [20] for the first time to detect DDoS attacks. In addition, we use the methodology using the DDoS features generated in early experiments.

A novel DDoS detection approach using the MP algorithm, the Adaptive Matching Pursuit (AMP) DDoS detection approach, is proposed. The AMP approach can perform anomaly and misuse detection separately. We combine anomaly detection, and misuse detection in the AMP method using a decision engine. We also com-

bine one-dimensional traffic attributes with the decision engine in Wavelet and MPMP methods. We re-implement DDoS detection methods using Wavelet and MPMP and compare them with the AMP approach. We use two different datasets for evaluating and comparing the methods. Both methods are evaluated with and without the decision engine to detect TCP and UDP flood attacks. We also combine TCP and UDP flood datasets to evaluate the performance of the methods in the detection of DDoS attacks in three traffic classes.

We show researchers of the DDoS detection area that the frequently used DDoS features can be represented with various probability distribution functions and well-known probabilistic models. The hypothesis testing performed using the log-likelihood ratio shows that Gaussian, GEV, and logistic distributions show similar performance. As far as we know, there is no other study in the literature in which DBSCAN and K-Means clustering algorithms are used with empirical statistical models. In addition, unlike most studies using empirical distributions in statistical DDoS detection methods, we use both misuse and anomaly detection methods comparatively. Also, in this thesis, in addition to frequently used Kullback Leibler and Jensen-Shannon divergences, the Greedy algorithm, and Manhattan Distance for calculating the deviation of empirical distributions with the generated models comparatively.

1.3. Outline of the Thesis

While preparing this thesis, we focused primarily on understanding DDoS attacks and finding attributes that can best express these attacks. Later, we introduced new methods to detect DDoS attacks. Chapter 2 includes a brief information about DDoS attacks and DDoS attack methodologies. This chapter also includes information about intrusion detection systems and IDS performance evaluation metrics.

Chapter 3 includes the process of generating network traffic features and gives brief information about the features we generate to detect DDoS attacks. This chapter also includes the information of publicly available CAIDA [43, 44] datasets, and the

BOUN DDoS [37] dataset generated within this thesis. We also mention the feature selection methodologies used in this thesis. This chapter also includes the simulation experiment using DETER testbed [45] to develop a methodology on feature generation and obtain the convenient features of network traffic that can better reflect the change of traffic under DDoS attack.

DDoS detection using Autoregressive Modelling proposed in [20] is used to detect DDoS attacks in Chapter 4. They used AR(1) modeling to detect network anomalies using SNMP MIB variables. We use the methodology to detect DDoS attacks using traffic features we generated in the previous chapter.

We propose a novel approach to detect DDoS using the Matching Pursuit algorithm and implemented Matching Pursuit and Wavelet-based approaches in Chapter 5. We perform both misuse and anomaly detection in this chapter. We also propose a novel hybrid detection framework that uses sparse signal representation-based DDoS detection approaches.

We examine statistical properties of traffic features under DDoS attacks and propose a novel DDoS detection method in Chapter 6. We use empirical distributions generated from network traffic to build statistical models. We then calculated the divergence between empirical distributions obtained from network traffic and the statistical models. We perform both anomaly and misuse type detection in this chapter. In the final chapter, we evaluate the approaches and methodologies used in this thesis, and we will mention the future research directions.

2. DDoS ATTACKS and INTRUSION DETECTION SYSTEMS

In this chapter, we discuss DDoS attacks, intrusion detection systems, and intrusion detection evaluation metrics. The first section gives brief information about flood-type DDoS attacks, which are the main focus of the detection approaches proposed in this thesis. The second section discusses the intrusion detection methodology. Also, this section describes the metrics used in the assessment of detection approaches.

2.1. DDoS attacks

A DDoS attack is a coordinated attack that aims to prevent legitimate users from benefiting from an online service. There are many ways to deny service, [46] such as flooding a network with unwanted packets preventing legitimate network traffic.

DDoS attacks are a combination of different steps [47] starting with the execute command send by an actual attacker to the master application. As a second step, the master application sends the execute command to slaves or the attacker computers. Then, attacker hosts start sending attack packets to the victim. In most cases, attacker computers use fake source IP addresses to make it challenging to locate the source of the packets.

DDoS attacks are divided into two groups: bandwidth depletion attacks and resource depletion attacks [48–52]. Bandwidth depletion attacks deny the service of the target system by filling the target network with unwanted packets. The resource depletion attacks aim to consume the computing resources of the target system. Resource depletion attacks, unlike bandwidth depletion attacks, a malformed packet that exploits the network protocols are used [50]. For every incoming packet, the system will consume computational resources such as memory and processing power. Target systems resources are locked at high levels while processing these packets, and the system

no longer serves legitimate users. In this thesis, we focus on resource depletion-type flood attacks. Flood attacks include TCP SYN flood attacks, ICMP flood attacks, and UDP flood attacks. Most DDoS attacks performed in the world are TCP SYN flood attacks [53]. We will explain flood-type DDoS attacks in detail in the next sections.

2.1.1. TCP flood Attacks

TCP flood attacks exploit a three-way handshake of TCP protocol. The TCP three-way handshake includes a series of messages between client and server before establishing a TCP connection. We can list these messages as follows:

- The client requests a connection by sending a synchronize (SYN) message to the server. The SYN packets have an SYN flag bit in the TCP header of 3the packet set to one.
- The server acknowledges (ACK) this request by sending SYN-ACK back to the client.
- The client responds with an ACK and the connection is established.

We can see from Figure 2.1 that the attacking computers send SYN packets to the victim server. The victim sends SYN-ACK to the spoofed IP addresses of the attacker computers and allocates some of its computing resources to hold a half-open TCP connection until timeout. The attacker client either has the fake spoofed source IP address or does not respond to the SYN-ACK packet. In both cases, the victim server never receives the ACK packet and keeps the TCP connection half-open until timeout. Although the resource reserved by the victim server for half-open connections is small, increasingly large numbers of half-open connections will deplete all the resources. As a result, the server cannot accept new connections from any source, denying legitimate users. These packets can cause a severe crash in some cases if there are no resources left for the operating system to function correctly.

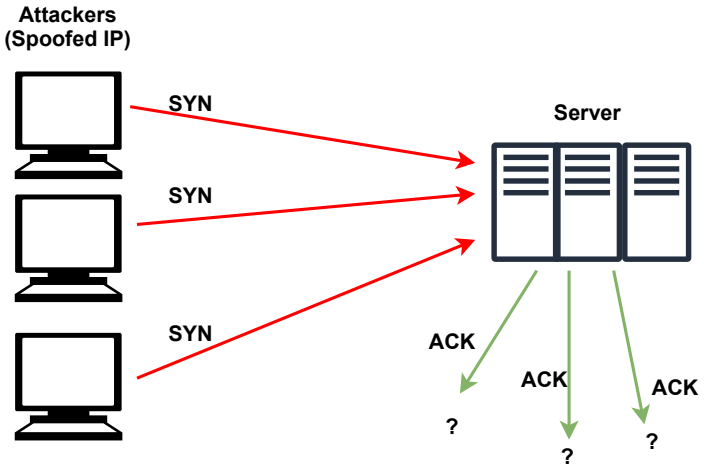

Figure 2.1. TCP SYN Flood attack network topology.

2.1.2. UDP flood attacks

Unlike TCP protocol, UDP protocol does not include a connection between client and server. Because of this, the UDP flood attack is not straightforward as the TCP flood. Instead of sending connection packets to the victim, attackers send a high number of UDP packets to random ports of the victim server. The victim server port tries to check the application listening to that port. Since the packets are sent to random ports, the victim cannot find the application that listens to the port mentioned above and sends an ICMP destination unreachable reply message to the attacker. As a result, receiving a large number of UDP packets forces the victim to send a high number of ICMP packets. The victim server becomes unreachable by the legitimate users with the increasingly large number of packets. The UDP flood attack network topology can be seen from Figure 2.2.

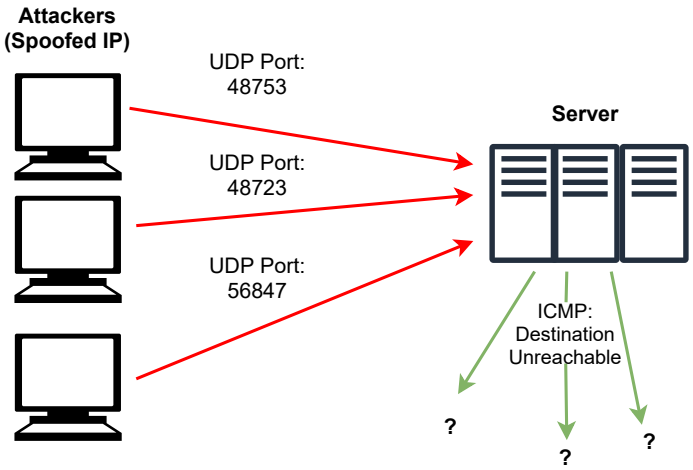

Figure 2.2. UDP Flood attack network topology.

Like the TCP SYN flood attacks, the attackers generally use spoofed fake source IP addresses, ensuring that excessive ICMP packets cannot reach the attacking computers. Most operating systems mitigate this part of the attack by limiting the rate at which ICMP responses are sent. This attack can also be mitigated by using firewalls to filter out unnecessary UDP ports in front of the victim server.

2.1.3. ICMP flood attacks

ICMP flood attacks, also known as ping flood attacks, aim to overwhelm the target system with ICMP echo-request packets. ICMP protocol is a transport layer protocol used for network diagnostics. Ping and traceroute are the most common two commands used for network diagnostics that are a part of ICMP protocol. ICMP protocol is consists of ICMP messages such as ICMP request, echo-request, and echo-reply. These messages require some of the computing and bandwidth resources of the target system. An increasingly large number of these messages causes the victim server or network to become unable to give service to legitimate users.

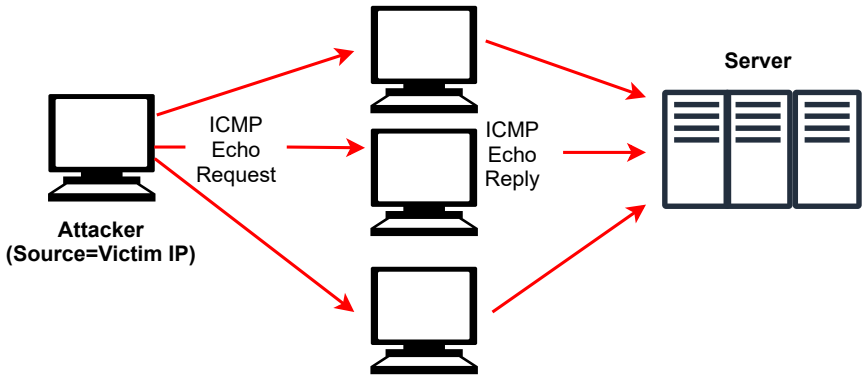

Figure 2.3. ICMP Flood attack network topology.

As seen in Figure 2.3 ICMP flood attacks start with a high number of ICMP echo request packets sent from attacker computers to the victim server. The victim server then tries to reply to all the echo-request packets with an ICMP echo-reply packet causes bandwidth congestion in both directions of the traffic flowing through the victim network. Also, processing so many ICMP messages deplete the victim server's resources, making it unable to serve legitimate users.

These attacks can be mitigated by disabling ICMP functionality at the victim router, network, or server-side. For this purpose, firewalls can block ICMP packets reaching the victim side of the network.

2.2. Intrusion Detection Systems

Intrusion detection is the process of detecting security incidents occurring in a computer system or computer network. Security incidents or intrusions are events that harm the availability, integrity, and privacy of computer systems. Intrusion detection systems are hardware or software application that monitors computer systems to find intrusions. They observe and analyze the events occurring in the computer systems or computer networks to detect malicious activity and generate alarms. Alarms are either monitored by a security professional or transmitted to a security device such as a firewall.

IDSs can be divided into three groups according to the source of information they use. These are network-based, host-based, and application-based IDS. The majority of commercial IDS perform network-based detection. They are located in various places in the network and can monitor an extensive network. Although it is easier to manage network-based IDS, monitoring and generating alarms in wider networks can be challenging.

Host-based IDS collect information from hosts of the computer system. They are usually software applications installed on clients of servers of the computer networks. Application-based IDS work like a subset of host-based IDS systems specialized to monitor specific applications. In this thesis, we focus on network-based intrusion detection. IDS are divided into two groups regarding their detection methodology. These are misuse detection and anomaly detection. The misuse detection method uses attack signatures or models to detect intrusions, while anomaly detection uses models of attack-free behavior of the computer system. Anomaly detection is searching for abnormal activity on the computer system and network. Most commercial IDSs use a misuse detection methodology. Hybrid detection uses a combination of misuse and anomaly type of detection. In this thesis, we use the misuse, anomaly, and hybrid detection approaches.

2.2.1. Performance Evaluation Metrics

In general, five different metrics, including true-positive rate (TP), false-positive rate (FP), Receiver Operating Characteristic (ROC) curve, Area Under ROC curve (AUC), and Accuracy (Acc), are calculated for evaluation detection of the methods.

These metrics are calculated using the number of correctly identified or misidentified samples. The true positive (TP) corresponds to attack samples classified as an attack, while true negative (TN) samples are attack-free samples classified as normal. Similarly, false-positive (FP) corresponds to attack-free samples falsely classified as an attack, while a false negative (FN) corresponds to attack data classified as normal.

Positive predictive value (PPV) is the ratio of the number of true positives to the sum of TP and FP. Similarly, the negative predictive value is the ratio between TN and the sum of TN and FN.

Table 2.1. Explanations of simple per-class evaluation metrics.

Metric Name	Explanation
False Negative Rate	Percentage of members of class X incorrectly classified as not belonging to class X.
False Positive Rate	Percentage of members of other classes incorrectly classified as belonging to class X.
True Negative Rate	Percentage of members of class X correctly classified as belonging to class X.
True Positive Rate	Percentage of members of other classes correctly classified as not belonging to class X.
True Negative Rate	Percentage of members of class X correctly classified as belonging to class X.
True Positive Rate	Percentage of members of other classes correctly classified as not belonging to class X.

The first step in evaluating the detection approaches is to find the true classes (the ground truth) of the applications by inspecting the network packets. The overall performance of detection approaches is usually evaluated according to their *accuracy*. The accuracy is defined as the percentage of correctly classified instances among all instances. This metric can be used to describe the whole system's success and can be used as a per-class success metric. Other per-class measures assess the success of a classifier for a given class. The evaluation parameters are explained in Table 2.1 and they are calculated as:

$$TPR = \frac{TP}{TP + FP}, \tag{2.1}$$

$$FPR = \frac{FP}{FP + TN}, \tag{2.2}$$

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}. \tag{2.3}$$

These metrics are incapable of finding the best operating point of the detection system. If the ROC curves of two IDS do not cross, it is not straightforward to compare the IDSs. It is not always appropriate to use the area under the ROC curve for comparison because it measures all possible operation points of an IDS. One can argue that difference should be based on the best operation point of each IDS because, in practice, an IDS is fine-tuned to a particular configuration. For this reason, we employ the Capability of Intrusion Detection (CID) metric [54] to find the best operating point. CID parameter considers all the fundamental aspects of evaluation metrics and subtle changes on these metrics, including TPR, FPR, positive predictive value, negative predictive value, and base rate. A higher CID value means that the IDS has a better capability of classifying input events accurately. We select the point in the ROC curve that gives the maximum CID value to compare detection performances. Operation points for the performance metrics in the results section are chosen according to the highest CID value.

Let X be the random variable representing the IDS input and Y the random variable representing the IDS output. The entropy of the input of the random variable X is defined as [55],

$$H(X) = -\sum_{x \in X} p(x) log(p(x)). \tag{2.4}$$

The mutual information [55] between random variables X and Y is defined as:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) log \frac{p(x,y)}{p(x)p(y)}. \tag{2.5}$$

Using equations (2.4), and (2.5) CID is defined as:

$$CID = \frac{I(X;Y)}{H(X)}. \tag{2.6}$$

The mutual information measures the reduction of uncertainty of the input by knowing the IDS output. Besides, this mutual information is normalized with the entropy of the input, $H(X)$. Thus, CID is the ratio of the uncertainty reduction of IDS input, given the IDS output with a value range in $[0, 1]$. The bigger the CID value, the better performance of the detection system.

2.3. Chapter Summary

This chapter gives information about DDoS attacks and intrusion detection systems, including performance evaluation metrics used to evaluate detection approaches. This thesis focuses on flood-type DDoS attacks, mainly UDP and TCP SYN flooding attacks. We proposed different DDoS detection approaches as misuse, anomaly, and hybrid detection methodology.

3. DATASETS and TRAFFIC FEATURES

In this chapter, we give information about the data sets and features used in this thesis. To understand DDoS attacks and determine the most appropriate features, we first benefit from experiments using the DETER testbed environment. We evaluate the proposed approaches by utilizing these features with other datasets.

There are publicly available data sets available for the detection of DDoS attacks. Among these datasets, we use CAIDA datasets, which are frequently used in the literature. We also create a new dataset to evaluate the performance of DDoS detection approaches. In this section, we give detailed information about these data sets.

We explain the DDoS simulation experiments we performed in DETER testbed. DETER testbed experiments help us decide which traffic features we will use in this thesis to detect flood-type DDoS attacks. In addition, in this chapter, we explained the experiments performed on the DETER testbed. We also describe the methodology and topology of these experiments. We decide the traffic features to be used for the detection of DDoS attacks. We discuss the traffic features, feature generation stages, and feature selection approaches in detail in this chapter.

3.1. Datasets

In this section, we give information about the data sets we used in the thesis. These datasets are the DETER test environment, the BOUN DDoS dataset, and the CAIDA datasets.

3.1.1. DETER Testbed

This section includes DDoS attacks created in DETER testbed in order to understand DDoS attack mechanisms and the effects of DDoS attacks on network fea-

tures. The results from these DDoS simulations help us decide on the DDoS detection methodology used in this thesis. This section also explains the details of the topology and the methods of flood attack realizations. Three types of flood attacks are realized and generated using the SEER (Security Experimentation Environment) graphical user interface provided with the DETER testbed.

One of the main challenges in detecting DDoS attacks is distinguishing the attack traffic from legitimate traffic. Network administrator's primary tool for monitoring network health is to use traffic graphs. However, these graphs cannot give enough information on the attack state of the network traffic alone. To reveal the existence of a DDoS attack and possibly to determine the type of attack, we need to utilize the variations on different traffic features. For example, a resource depletion type TCP SYN flood attack generates many TCP SYN packets. Hence, if we can model the number of TCP SYN packets of network traffic, we can detect the attack quickly. However, that increase in TCP SYN packet number can be caused by legitimate traffic to the FTP server after a potential update. Therefore we cannot just look at the rise in the number of SYN packets to claim the existence of an SYN flood attack. We need to understand the effects of these specific attacks on multiple features simultaneously.

3.1.1.1. Background Traffic. To obtain a realistic DDoS attack simulation, we need to have legitimate background traffic. The simplest form of background traffic generation is using packet trace replay [56]. Using multiple PCs to replay actual packet traces through high data transmission speeds can obtain high-speed background traffic. Since many detection systems need to be tested under realistic traffic conditions, packet trace replay seems suitable for our needs. However, replaying the same packet traces can result in the same statistical behavior looking into details of the packet in background traffic. Real background traffic instead has many different types of packet traces.

Another approach in background traffic generation is using application-specific traffic generators. These tools produce network traffic based on different applications such as FTP and HTTP. Examples for such tools are Surge [57], trafgen [58] and

PackMime [59]. By using various application-specific traffic generator tools, we can obtain more complicated background traffic.

There are also application-independent traffic generators that create traffic at the IP flow level. They make network traffic based on probabilistic distributions and stochastic processes for various traffic parameters such as inter-packet gap interval and packet size. A collection of these generators is developed by "Evaluation Methods for Internet Security Technologies" (EMIST) DDoS [60]. It includes configuring a broad mix of background traffic that consists of TCP traffic created using Harpoon [61]. We can also generate DNS traffic by setting up a server and periodically issuing requests from various locations in the topology. Also, we can create ICMP echo requests and reply traffic using the ping utility.

We use the SEER software tool developed by ISI (Information Science Institute) and provided by the DETER testbed environment. SEER tool contains a collection of background traffic generators, including Harpoon, with many options varying file size and node selection. SEER provides an easy-to-use user environment for controlling experiments and traffic generators. The following traffic generators in SEER software:

- Web: We can select the source and destination nodes and server nodes. Also, we can select the maximum and minimum thinking time in seconds. And also, we can specify the maximum/minimum file sizes. This tool is mostly used for background generation in this work.
- Dump Replay: This tool allows us to replay a captured traffic file at various nodes. We can also specify a multiplier value to set the intensity of traffic.
- Video Streaming: There are three files defined for video streaming traffic generation. We can also define the duration of the stream in seconds, maximum/minimum thinking time, and RTP/TCP protocol type.
- FTP: This tool has similar options with a web traffic generator.
- DNS: This tool allows us to control hosts and servers for DNS traffic. Also, we can choose the maximum/minimum thinking time and bad requests ratio.

- IRC: These traffic generators have controls of client nodes, server nodes, message sizes, message delay, connection duration, and connection time.
- Ping: We can control clients, servers, thinking time, and message size. Changing data size results in longer ping packets.
- Constant Bit Rate: This generator allows us to have constant background traffic during our simulations.
- SSH: This makes SSH connections from selected clients through selected servers. We can control connection duration, message delay, and reply sizes.
- Harpoon: This traffic generator uses the method proposed in [61].

In simulations, web, FTP, DNS, IRC, Ping, and SSH traffic generators are used. Harpoon and video streaming traffic generators dominate all background traffic due to their high data sizes. High data traffic results increase in the computational power feature extraction process; because of this reason, we didn't use these traffic generators.

3.1.1.2. Simulation Set Up. To create a network traffic simulation, we have to decide the network topology of the experiment in the DETER testbed environment. The topology is designed using a ".ns" file that specifies the nodes and the connection between nodes. The .ns file can be seen in Figure 3.1, and corresponding network topology can be seen in Figure 3.2.

Experiment topology includes nine nodes. Increasing node count will better realize network traffic and result in more resource usage in the testbed environment. The simulation environment provides us to use spoofed source IP addresses using multiple subnets. However, attack traffic coming from 9 nodes regarding source addresses of packets can create attacks from numerous resources behavior using 1024 IP addresses as an attack source. An increasing number of source IP addresses results in an increase in the number of flows at the listening node.

```
set ns [new Simulator]
source tb_compat.tcl
#Create the topology nodes
foreach node { v R1 R2 A11 A12 A13  A21 A22 A23  control } {

   #Create new node
   set $node [$ns node]

   #Define the OS image
   tb-set-node-os [set $node] Ubuntu1004-STD

   #Have SEER install itself and startup when the node is ready
   tb-set-node-startcmd [set $node] "sudo python /share/seer/v160/experiment-setup.py Basic"
}

#Create the topology links
set linkR1v [$ns duplex-link $v $R1 100Mb 3ms DropTail]
set linkR1A12 [$ns duplex-link $A12 $R1 100Mb 0ms DropTail]
set linkR1A13 [$ns duplex-link $A13 $R1 100Mb 0ms DropTail]
set linkR1A11 [$ns duplex-link $A11 $R1 100Mb 0ms DropTail]
set linkR2A21 [$ns duplex-link $R2 $A21 100Mb 0ms DropTail]
set linkR2A22 [$ns duplex-link $R2 $A22 100Mb 0ms DropTail]
set linkR2A23 [$ns duplex-link $R2 $A23 100Mb 0ms DropTail]
set linkR2R1 [$ns duplex-link $R2 $R1 100Mb 0ms DropTail]

$ns rtproto Static
$ns run
```

Figure 3.1. NS file used to generate DETER testbed experiment.

The interface of node n10 listens to its network ports, and all packets passing through the interfaces both in and out direction is captured (Figure 3.2). Although DETER testbed includes valuable tools for traffic generation and traffic flooding, attention is needed to achieve realistic background traffic. For this purpose, different combinations of nodes are used for background traffic generation.

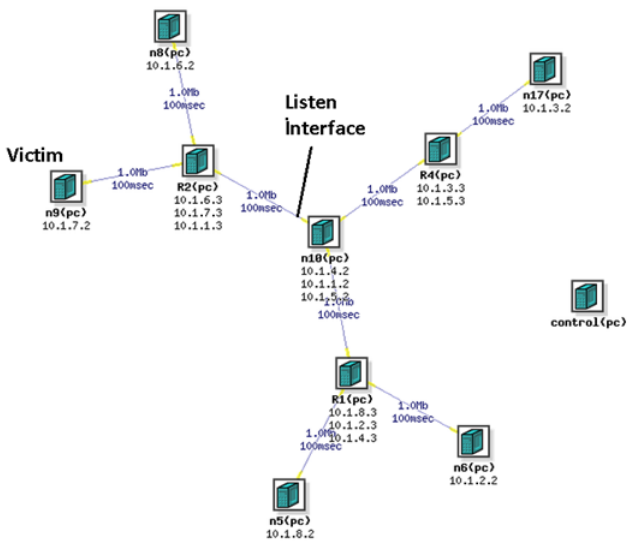

Figure 3.2. Topology of experiments.

There are various options for flood traffic generation in SEER GUI. Initially, we must build a network structure to catch all attack traffic while background legitimate traffic and attack traffic can be correctly captured. Also, there is a need to label attack packets and distinguish them to check the outcome of detection processes. For this purpose, background traffic is generated through node n8 in figure 3.2, and attack traffic is directed through node n9 in figure 3.2.

Traffic is captured from the victim router side, where the server is also connected. All nodes except n8 make traffic through node n8 and attack node n9. Node n9 is the victim node with IP address 10.1.7.2, and n8 is the server node with IP address 10.1.6.2. In general, attacks flow towards servers to interrupt or block requests of the server coming from legitimate hosts. However, in order to calculate the entropy of an attack, packets must be labeled. In order to distinguish attack packets from normal packets, victim and server nodes are separated. All packets flowing towards the victim's IP address are labeled as attack packets, while others are labeled as normal packets. Also, the attack label feature will be used to calculate information gain.

Figure 3.3. Attack-free traffic generated using all traffic generators in SEER.

There is a control node in Figure 3.2 that does not have any connection to other nodes. This is because the DETER testbed environment and SEER software tool requires a control node for connecting and controlling the experiment. The attacking and traffic nodes in the network topology are denoted as n_x. These nodes generate attack-free background traffic using FTP, DNS, ICMP, DNS, and Harpoon traffic generators.

The traffic generated using the SEER tool can be seen in Figure 3.3. The graph above the x-axis is the traffic going into node V, while the graph above the x-axis is the traffic going out from node V. The traffic in Figure 3.3 is monitored from the node V in the experiment topology. The first experiment that we create in the DETER testbed environment includes attack-free traffic generation with anomalous jump with video streaming.

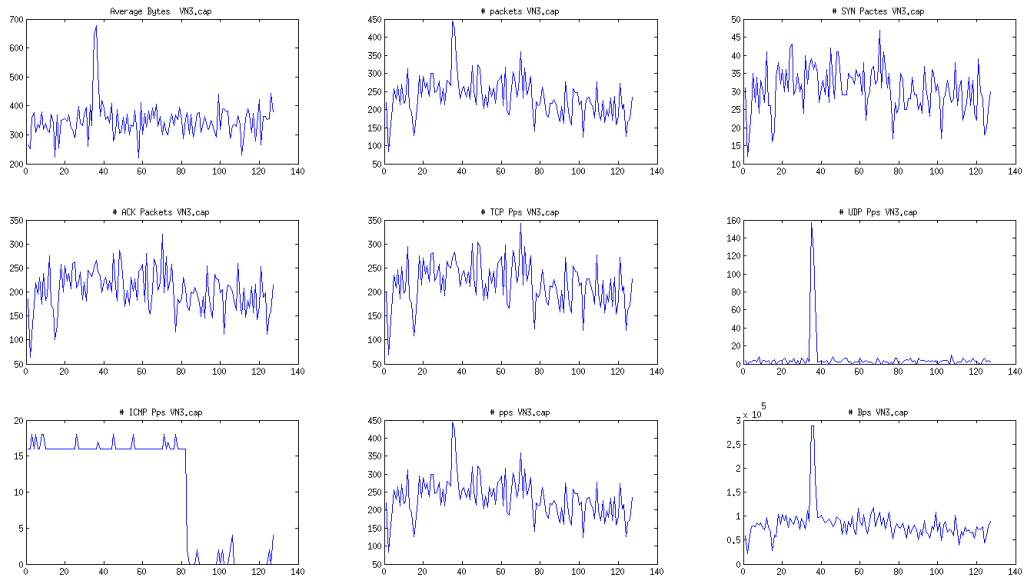

Figure 3.4. Traffic features of experiment 1. Legitimate traffic includes www, FTP, DNS, ICMP, Harpoon, and SSH.

We generated nine features from the data obtained in the first experiment. These features are generated in a time window manner and can be seen from Figure 3.4. We

will give the details of all the features used in the next section. It should be noted that these features are obtained to have a conjecture about network behavior under DDoS attacks.

Features obtained from network traffic are correlated with each other. For example, if the number of packets in a network increases, the total data flowing through the network will be increased. In Table 3.1 we can see cross-correlations of the features of traffic at Figure 3.4.

Correlation is a measure of statistical dependence that corresponds to any statistical relationship between two random variables. The degree of correlation between two random variables is measured by calculating the correlation coefficient. There are several ways to calculate the correlation coefficient. Pearson's correlation coefficient, which is frequently used, represents the linear correlation between two variables.

The Pearson product-moment correlation coefficient measures the linear correlation (dependence) between two variables X and Y, giving a value between $+1$ and -1 inclusive, where 1 is a total positive correlation, 0 is no, and -1 is a negative correlation. It is widely used in the sciences as a measure of the degree of linear dependence between two variables. DDoS attacks have increased or decreasing effects on features. We can use Pearson correlation coefficients to represent such behavior of features.

Pearson's correlation coefficient between two variables is defined as the covariance of the two variables divided by the product of their standard deviations and calculated as:

$$\rho_{XY} = \frac{Cov(XY)}{\sigma_X \sigma_Y} = \frac{E[(X - \mu x)(Y - \mu y)]}{\sigma_X \sigma_Y}, \tag{3.1}$$

where, *cov* is the covariance, σ_X is the standard deviation of X , μ_X is the mean of X, and E is the expectation.

Table 3.1. An example of the correlation coefficients between features.

Feature	Number of SYN packets	Average Packet Length	Number of packets	Number of TCP packets	Number of All packets	Number of ACK packets
Number of SYN	1.0000	-0.6111	0.9923	0.9923	0.7826	0.9667
Avg Packet Length	-0.6111	1.0000	-0.5939	-0.5937	-0.2679	-0.5685
Number of packets	0.9923	-0.5939	1.0000	1.0000	0.8447	0.9909
Number of TCP packets	0.9923	-0.5937	1.0000	1.0000	0.8447	0.9909
Number of All packets	0.7826	-0.2679	0.8447	0.8447	1.0000	0.8954
Number of UDP packets	0.9667	-0.5685	0.9909	0.9909	0.8954	1.0000

In our statistical approach, the network health function is obtained using a combination of abnormality indicators from the particular feature. The abnormality in traffic data is determined by detecting abrupt changes in their statistics. Since the statistical distribution of the features is significantly different, it is challenging to do joint processing of these variables. Therefore, the abrupt changes in each of the features are first obtained.

3.1.1.3. A comprehensive simulation with attack traffic. We generate various experiments using DETER testbed, but we do not include all the details of these experiments for the simplicity of this thesis. This simulation is used to understand the effects of SYN flood attacks on traffic features. The generated traffic includes 30 min of attack and normal traffic. We obtain complex test data that provides for six attack traffic of different rates. Figure 3.5 shows the traffic features obtained from this experiment.

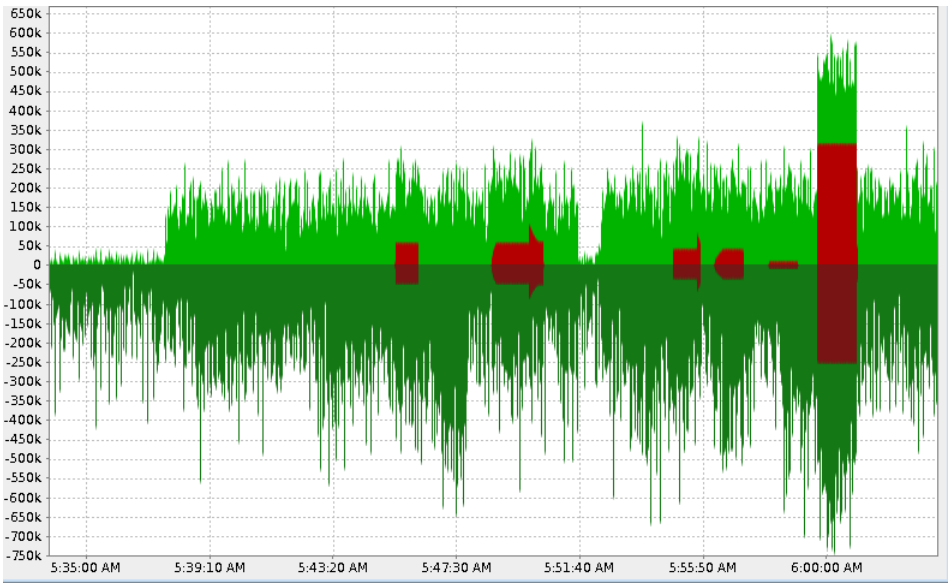

Figure 3.5. Traffic graph of DDoS simulation in DETER testbed.

In the final attack scenario (Figure 3.5), unlike previous attacks, the rate of SYN flood traffic dominates overall packets. As we can see from the figure, there are six instances of TCP SYN flood attacks with variable duration shapes and rates. In Figure 3.5 the traffic below the x-axis corresponds to outgoing traffic, while the traffic above the x-axis is the inward traffic.

Detection of these kinds of attacks is more straightforward. However first five attacks can disappear in normal traffic load. Network administrators cannot realize these kinds of attacks but may have hazardous effects on the victim system. Also, normal traffic differs over time, which makes overall traffic more random. The selected features for this attack simulations are shown in Figure 3.6.

3.1.2. Boğaziçi University DDoS Dataset

We generated resource depletion-type DDoS attacks on the campus network of Boğaziçi University and recorded the ongoing traffic from the backbone router's mirrored port. We generate TCP SYN and UDP flooding packets using Hping3 traffic generator software by flooding. This dataset includes attack-free user traffic and at-

tack traffic, suitable for evaluating network-based DDoS detection methods. Attacks are towards one victim server connected to the backbone router of the campus. Attack packets have randomly generated spoofed source IP addresses. We removed payloads of packets and anonymized the source IP addresses of legitimate users for the confidentiality of legitimate users.

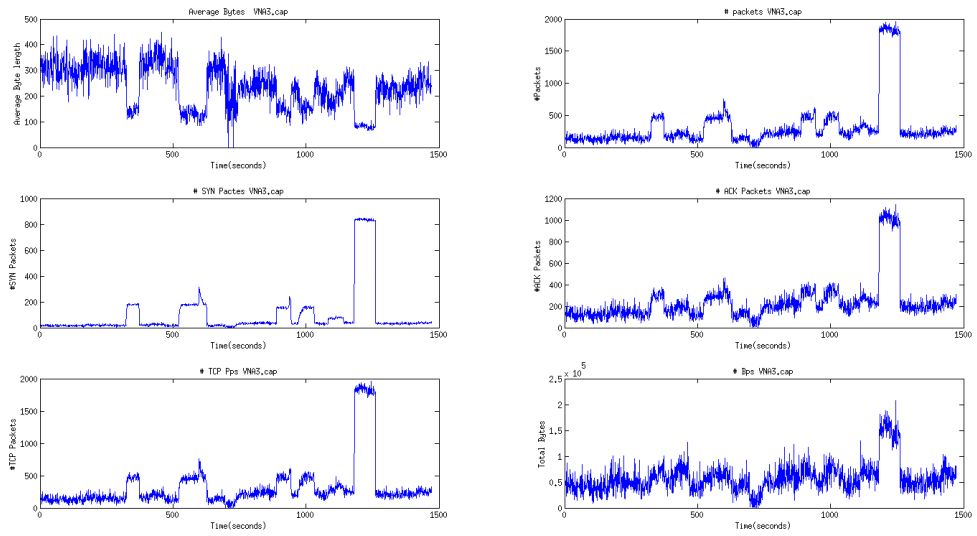

Figure 3.6. Traffic feature vectors obtained from DETER testbed experiment.

We can list the prominent advantages of the BOUN data set compared to other data sets as follows:

- Dataset provides aspects of DDoS attacks, including network-based probing of two-way legitimate user traffic mixed up with DDoS packets. Besides, it includes attacks of different intensities to help researchers train and evaluate their intrusion detection approaches for different attack densities.
- These datasets provide a general understanding of resource depletion type DDoS attacks collected from the backbone router of campus networks. These datasets are suitable for developing and evaluating network-based attack detection methodologies. Boğaziçi University DDoS (BOUN DDoS) Dataset has been already used in some academic publications [18, 62, 62–64, 64, 65].

- Unlike other publicly available DDoS datasets citemawilab,caida2007,caida2008, the BOUN dataset includes legitimate background internet traffic mixed with DDoS attack traffic. In addition, the BOUN datasets provide more direct stimulation and analysis because of small file sizes and fewer packets compared to other datasets [43, 44, 66].
- Attack and legitimate traffic packets can easily be separated from each other using destination IP addresses of packets. Attack-free packets in the datasets can be used for traffic analysis, or combined methods with another attack dataset can be evaluated [67].
- Datasets are given in comma-separated file format, including header information of packets to help researchers easily import datasets in different research software platforms.

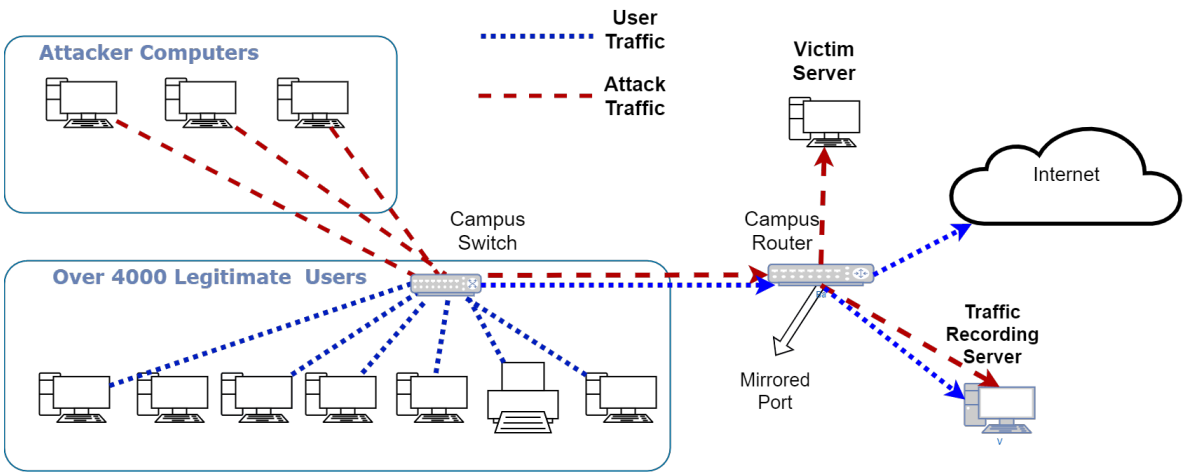

Figure 3.7. Network Topology of BOUN DDoS Dataset traffic generation and recording.

The design concept of Network-based intrusion detection systems is detecting attacks from the network's end, on the router, or on the backbone switch. This dataset is produced for the evaluation of network-based intrusion detection methods. In the network topology shown in Figure 1, the traffic is taken from campus routers port by the mirroring method. The mirroring operation on routers interfaces provides our traffic recording server the exact copies of incoming and outgoing packets flowing through the

mirrored interface. Traffic is recorded and converted to .csv file format using Wireshark software [68].

The dataset includes two different attack scenarios. In both situations, randomly generated spoofed destination IP addresses are used in a flooding manner. For TCP flood attacks, TCP port 80 is used as the destination port. All of the datasets lasted 8 minutes. In each of them, 80 seconds waiting period, then 20 seconds attack period is practiced. Different packet rates are used to let researchers evaluate their detection methods concerning different packet rates.

The TCP SYN Flood and UDP flood datasets include attack rates of 1000, 1500, 2000, and 2500 packets/second, respectively. The topology of the network for obtaining an attack dataset is given in Figure 1. Both legitimate and DDoS attack traffics mirrored the recording server. Attack packets can be distinguished from attack-free packets using the destination IP address of packets. The victim's IP address is 10.50.199.86.

Figure 1 shows the network topology used in the generation of the dataset. We carried out the TCP SYN flood and UDP flood attacks towards a server connected to the campus backbone. Over 4000 active internet user traffic was flowing over the campus router simultaneously to the attack traffic. We used the hping3 software installed on three computers for attacks. Attack packets contain spoofed source IP addresses. Since the source IP addresses of the attack packets are generated randomly and uniquely, it appears as attacks come from many different sources when viewed from the router's port. In other words, the attack packets in the dataset come from multiple sources.

Datasets are given as two tables in the comma-separated value (.csv) file format. The names of the files are BOUN_TCP_Anon.csv corresponding to TCP SYN flood attacks and BOUN_UDP_Anon.csv corresponding to the UDP flood attack dataset. The tables in the files of the dataset have the following columns:

- Time: Time values start from zero and have a resolution of 0.000001 seconds.

Time values are expressed in seconds.

- Frame Number: Frame number is simply the incremental count of packets in the dataset.
- Frame_length: Frame length is the length of that packet in bytes.
- Source_ip: Source IP address of the packet.
- Destination_IP: Destination Ip address of the packet.
- Source_Port: Source TCP port of the packet. If it is not a TCP packet, this field is empty.
- Destination_Port: Destination TCP port of the packet. If it is not a TCP packet, this field is empty
- SYN: This value is "Set" if the packet is a TCP packet and its SYN flag is equal to one, it is equal to "Not Set" if the packet is a TCP packet and its SYN flag is equal to zero. If the packet is not a TCP packet, this field is empty.
- ACK: This value is "Set" if the packet is a TCP packet and its ACK flag is equal to one, it is equal to "Not Set" if the packet is a TCP packet and its ACK flag is equal to zero. If the packet is not a TCP packet, this field is empty.
- RST: This value is "Set" if the packet is a TCP packet and its RST flag is equal to one, it is equal to "Not Set" if the packet is a TCP packet and its RST flag is equal to zero. If the packet is not a TCP packet, this field is empty.
- TTL: Time to live value of the packets.
- TCP_Protocol: This value can be TCP or UDP if the packet belongs to a transport layer IP protocol. Else this value can have different values.

Table 3.2 and Table 3.3 gives some statistics and information about attacks in datasets. Each attack dataset contains 4 attack instances. The columns of tables are explained as follows:

- Attack Period: There are four attack periods for TCP SYN and UDP flood datasets.
- Start Time: Each dataset timer starts from zero. The start time column corresponds to the start time of the attack in seconds.

- End Time: The end time of the attacks in seconds.
- Start Frame: The frame number of the first packet of the attack.
- End Frame: The frame number of the last packet of the attack.
- Attack Packets: The number of attack packets in the attack instance.
- Legitimate packets: The number of attack-free packets in the attack instance.
- Density: The ratio of the number of attack packets to the number of attack-free packets. This ratio is calculated in the time window where the attack packet exists.

Table 3.2. Information about attack instances in BOUN TCP SYN Flood attack dataset.

Attack	**Start (sec)**	**End (sec)**	**Start Frame**	**End Frame**	**Attack Pkt.**	**Normal Pkt.**	**Base Rate**
1	80.22269	102.20233	2335362	2335362	19035	370746	0.05134
2	180.17426	203.08441	4240070	4240070	27121	428168	0.06334
3	279.97402	301.79111	5959329	5959329	35936	352296	0.10201
4	380.10981	402.35755	7885602	7885602	43465	401553	0.10824

Table 3.3. Information about attack instances in BOUN UDP Flood attack dataset.

Attack	**Start (sec)**	**End (sec)**	**Start Frame**	**End Frame**	**Attack Pkt.**	**Normal Pkt.**	**Base Rate**
1	80.87054	102.68198	1354950	1354950	37216	268882	0.13841
2	180.94241	203.55186	2931244	2931244	55029	337036	0.16327
3	280.59444	303.16265	4702829	4702829	75023	393450	0.19068
4	381.01394	403.65057	6513625	6513625	93378	404330	0.23095

We used the same network topology shown in Figure 3.7 to create the UDP and TCP SYN flood datasets. The setup differs only in the generated attack packets for UDP and TCP SYN flood attack datasets. We used hping3 software to generate attack packets with randomly generated spoofed source IP addresses. Network-based intrusion detection systems aim to detect intrusions by monitoring traffic to and from all devices.

They perform detection by analyzing all traffic passing through the gateway of the user networks. They are generally connected to the gateway of the network or the backbone router. We produced the BOUN DDoS dataset to evaluate network-based intrusion detection approaches. We recorded the network traffic from the mirrored router port. Port mirroring on the backbone router sends a copy of all network packets seen on the mirrored router port to another interface for monitoring purposes. Wireshark software running on a server running with a windows processing system was used to record the traffic. Traffic is initially saved in .pcap file format and then converted into the .csv file format to make it available to use in research software applications. Payloads of packets are deleted, and A-class virtual IP addresses replace source IP addresses using text editing software to preserve the confidentiality of end-users. TCP flood attack dataset includes 7.3 million packets, where 260646 packets are attack packets with a base rate of %3,54.

3.1.3. The CAIDA Datasets

The DDoS dataset provided by CAIDA (Center for Applied Internet Data Analysis) includes only TCP SYN flood attacks. Also, CAIDA provides attack-free traffic data for modeling internet traces. Both CAIDA 2007 Anonymized DDoS traces dataset and CAIDA 2008 Anonymized Internet Traces 2008 dataset are combined using the packet replay method to use the dataset in this work. CAIDA 2008 traffic is used as normal background traffic, while a portion of the CAIDA 2007 attack dataset is used as DDoS traffic.

The CAIDA 2007 [44]. The DDoS dataset contains approximately one hour of anonymized traffic traces from a DDoS attack on August 4, 2007. This includes a TCP SYN flood attack, which aims to deny target service by consuming computer resources. The attack-free traffic is removed from CAIDA 2007 dataset resulting in only two-way attack traffic left in the dataset.

The CAIDA 2008 dataset contains anonymized passive traffic traces from CAIDA's

Equinix-Chicago, and Equinix-Sanjose monitors on OC192 Internet backbone links [43]. However, the dataset is used to recreate the background traffic of DDoS attacks. In order to complete our experiments with available computing resources, only a part of the dataset that contains 26 million anonymized packets without payload is used.

3.2. Traffic Features

This section contains information about feature generation and the behavior of features in different traffic classes. We also briefly describe the preprocessing and normalization steps of the feature generation process.

3.2.1. Preprocessing

Features must be first extracted from tcpdump files to carry out calculations. There are various tools available for extracting packet features from tcpdump files. In real-time, intrusion detection features of packets must be recorded on the fly. This process needs a high amount of computing power. For this purpose, deep packet inspection-focused ASICs (Application-Specific Integrated Circuits) are produced. Since we don't have ASIC designed explicitly for our work, we have to work offline anomaly detection. The preprocessing step needs a high amount of computing power and memory.

Sixteen candidate features were used in this work. These features represent the distribution of properties of network packets for a given time window. These types of features can be called time window-based features. The smallest time window that can be defined .pcap traffic capture format is 0.1 millisecond, which introduces enough resolution for determining different time window ranges for our methodology.

3.2.2. Feature Generation

Initially, features are extracted for each time window w. All packet going through the network are collected, and properties of these packets are selected to achieve a scalar

number as an element of the feature that represents the chosen property of the packets. As an example, we can choose the number of packets per time window feature. All packets going through the network are counted in the specific time window, and the number of packets is achieved in that particular time window.

Lets define n^{th} feature vector as $f_n = f_{n1}, f_{n2} \ldots, f_{nk}$ where k is defined as the fraction between total time of network data and length of the time window and n is the features number that can be seen from 3.4.k is defined as:

$$k = \frac{total\,duration\,of\,data}{length\,of\,time\,window}. \tag{3.2}$$

There are two types of feature vectors generated for this work.

- First, we obtain one-dimensional traffic attribute vectors calculated by counting properties of network packets in every time window, and its denoted by f.
- For every time window w_i and $i = \{1, 2, 3, \ldots, k\}$ a characteristic feature vector y is created from one dimensional traffic attribute vectors.

For each time window w_i at i^{th} time interval a scalar value is calculated by counting properties of network packets that flow throughout the network. For each time interval we obtain a characteristic feature vector with length m set that can be defined as:

$$y_i = \{f_{1i}, f_{2i}, \ldots, f_{ni}, \ldots, f_{mi}\}, \tag{3.3}$$

where m is the number of all traffic attribute vector values and y_i is used as a characteristic feature vector that defines state of network at i^{th} time window w_i.

3.2.3. Normalization

The normalization of different feature vectors is not a straightforward job because some features, such as packets per time window, have large scales and high variation. On the contrary, the number of UDP packets per time window has a smaller scale and variation. We have the possibility of losing data in low-varying attributes if we normalize the characteristic feature vector. Because of this, every feature vector is normalized within itself.

All training data is collected from network and feature vectors $f_1, f_2 \ldots, f_n, \ldots, f_m$ are constructed from collected network packets. All feature vectors have different scales and deviations; as an example number of TCP packets can vary between zero and a thousand, while the number of ICMP packets can be a pretty lower highest value. We have to normalize each feature vector within itself to compensate for the difference of variances of feature vectors.

The standard feature scaling method is used for feature normalization. Feature scaling is a method used to standardize the range of independent variables or features of data. It is also known as data normalization and is generally performed during the data preprocessing step. Feature normalization can be shown as:

$$\hat{f}_n = \frac{f - min(f_n)}{max(f_n) - min(f_n)}. \tag{3.4}$$

We have to form characteristic feature vectors y from normalized features $\hat{f}$ for each time window defined in 3.3. Since we have a k time window in our training dataset, we obtain k characteristic feature vectors.

Table 3.4. Traffic Features and Descriptions.

Attribute Name	Description
The number of SYN packets [69]	The Number of packets that have SYN flag bit set to 1 per time window.
The number of RST packets	The number of packets that have the RST flag bit set to 1 per time window.
The number of ACK packets	The number of packets that have ACK flag bit set to 1 per time window.
The number of packets [25, 70]	The number of packets that have ACK flag bit set to 1 per time window.
Average packet size [39, 71, 72]	The average packet size of packets in time window.
Data Transferred	The total payload in bytes going through the network per time window.
Number of TCP, UDP, and ICMP packets	The number of TCP, UDP, and ICMP packets per time window and can be used to distinguish TCP, UDP, and ICMP attacks from each other.
The number of hosts [73]	The number of unique source and destination IP addresses per time window.
The number of flows [74]	The number of unique communicating pairs (flows) per time window.
Packet per flow	The average number of packets per unique flow per time window.
Data per-flow	The amount of data in bytes flowing in each unique flow per time window.
TCP, UDP, ICMP packet per flow	Packet count per-flow regarding their transfer layer protocol.

3.2.4. Features

Many features can be generated from packet headers and payloads. As an example, KDD 99 intrusion detection dataset includes 41 different features. We obtain 16 distinct features in this work as shown in Table 3.4. According to the destination IP addresses of packets, we can label each packet as an attack packet or a normal packet. We can generalize the generated features in this work into two categories, namely flow-based features and packet-based features.

3.2.4.1. Packet Based Features. Packet-based features include the number of all packets, SYN, ACK, RST, UDP, TCP, ICMP packets. The average packet size and total data transferred in the time window features are also calculated in a packet-based feature generation manner.

Number of SYN packets feature corresponds to the number of packets that have the SYN flag bit set to 1. Any attack that alters TCP three-way handshake protocol is expected to affect this feature. One of the most popular attacks of this category is the TCP SYN flood attack. Legitimate traffic includes random SYN packets at random times, depending on the number of nodes in the network. In the case of the SYN flood attack, the attack traffic will include SYN packets that follow a pattern such as flat, ramp, or increasing. The pattern may not be seen by human inspection when attack traffic is lower than whole traffic. A systematic method is needed to detect these attacks.

Number of RST packets feature includes the number of packets that have the RST flag bit set to 1. This feature can be affected in the case of a reflected SYN flood attack. Attackers send packets that have spoofed victim IP addresses to legitimate nodes. Nodes will answer these packets sending ACK or RST packets through the victim node. Reflected SYN flood attacks can be detected using SYN, RST, or ACK packets. If the dump is taken from the edge of the network, includes victim node, ACK, and RST packets will generate an alarm.

Number of ACK packets feature includes the number of packets that have the ACK flag bit set to 1. The effect of attacks on this feature is explained in the previous paragraph.

Number of all packets feature corresponds to the number of all packets in a time window. All types of flood attacks affect the number of packets on the whole traffic. However, this feature strictly depends on the ratio of legitimate attack bandwidth and attack bandwidth. If this ratio is too small, like %5, this feature becomes insufficient for detection.

Most anomaly detection systems are based on only this feature. However, most DDoS attacks targeting a single target can be successful without making a slight change in this feature. In a large network that includes high bandwidth and contains an increased number of hosts, such as traffic of campus network, this feature gives little information of low scale successful DDoS attacks. As a real-life example, DDoS attacks generated for this work in the Boğaziçi University campus successfully denied target host communication while making an unrecognizable change in this feature.

Average packet size feature is useful for detection of DDoS attacks that generate a large number of small packets. In case of attack, generally, the average size of packets decreases. In normal circumstances, this feature has random values depending on the usage of hosts in the network.

In general DDoS attacks include many packets coming from different sources. Most attacks use spoofed source IP addresses to stay anonymous. There are two cases of ongoing flood attacks. If normal traffic includes a large number of hosts, the number of unique hosts feature will be decreased. In general DDoS attacks increase the number of hosts per second. As an example, a botnet with 100 hosts attacks a victim web server with spoofed IP addresses. Each host sends 1000 packets per second, and they send these packets with spoofed source addresses. If the source addresses are randomly created, the number of unique hosts seen from the victim network will significantly

increase. However, if all hosts use one IP per host, the number of individual hosts in the victim network can be decreased.

Total data transferred feature includes total data going through the network in a second. This feature is a slow-changing feature in large networks and can be affected easily in small networks.

Number of TCP, UDP, and ICMP packets features represent the number of packet that has corresponding transport protocol in a specified time window. These features can be used to distinguish TCP, UDP, and ICMP attacks from each other.

3.2.4.2. Flow Based Features. In packet switching networks, traffic flow is defined as a sequence of packets from a source host to a destination host. It can be defined as traffic flow as "an artificial logical equivalent to a call or connection. In our work, traffic flow is defined as a group of packets sharing flowing through the network in a specific time window sharing the same source IP address, destination IP address, source port, and destination port. On the basis of this definition, flow-based features are obtained for TCP and UDP traffic.

A network flow is determined by four fields in the packet header. These are:

- Source IP address
- Destination IP address
- Source Port
- Destination Port

Every packet flowing through the network that has the same values in these fields is considered as in the same flow. Flow information is collected from all packets, and every unique flow is counted as one. As an example, there can be 1000 packets in the dataset while these can packet belong to only 50 flows. For this feature, these 50 unique flows are used.

Any kind of DDoS attack can have an effect on this feature. Unlike the case of the DoS attack, which includes only one source and one target, DDoS attacks include many clients attacking the same target, resulting in an increase in this feature.

Packet per flow feature contains the average number of packets in each unique flow in the time window.

Data per flow feature contains the amount of data in bytes flowing in each unique flow per time window.

TCP, UDP, ICMP packet per flow features include packet count per-flow regarding their transfer layer protocol.

3.2.5. Feature Selection

The environment where the IDS is located significantly affects the features that can be used by IDS. It is clear that different sources of data produce various kinds of features. For example, if the network traffic capturer is a flow-based capturer, it provides completely different information than a packet capturer does. In addition to network traffic, there are other sources of data from which a subset of features is selected or extracted using feature analysis methods.

In order to reduce the training time and the risk of overfitting in modeling, we use feature selection. Feature selection is a process of selecting the most relevant features using the given dataset. In this study, two different feature selection approaches are used to ensure consistency.

3.2.5.1. Chi-Squared Feature Selection. Chi-square is a statistical test that measures divergence between the expected distribution and observed variables. Chi-square-based feature selection is used under the assumption that the features are independent of the occurrence of the classes. The chi-square value is calculated between each feature

and the class value, and the desired number of features is selected with the highest chi-squared value. The number of feature occurrences in different classes is used to calculate the Chi-Squared value, which is calculated with the following formula [75,76]:

$$\chi^2 = \sum_{i=1}^{c} \sum_{i=1}^{m} \frac{(A_{ij} - E_{ij})^2}{E_{ij}}, \tag{3.5}$$

where E_{ij} is the expected frequency of A_{ij},m is the attributes in the feature vector, and c is the number of classes which is equal to 2 in our case. A_{ij} is the observed frequency, which is the number of patterns in the i^{th} interval, and j^{th} class.

3.2.5.2. Information Gain Feature Selection. The information gain of a feature is related to an attack group is the information quality of the feature that shows if a connection belongs to the attack group or not. In other words, Information Gain measures how much information a feature gives us about the classes. If we rearrange the labels 1 and 0, such that one indicates attack packets exist in a specified time window while 0 shows a lack of attack packets in the specified time window. Expected information, which is the highest level of information gain of a feature, can be calculated by [77,78]:

$$I(s_1, s_2) = -\sum \frac{s_1}{s_2} log_2(\frac{s_1}{s}), \tag{3.6}$$

where s_1 is the number of samples in the group, and s_2 is the number of samples out of the group. To calculate the information gain of a feature, the entropy of the feature on the label class, which is 0 or 1 in this case, is subtracted from the expected information of the label class given above. If $\mathbf{f}$ feature vector that has value set of $f_1, f_2, ...f_n$, entropy of the feature is expressed as:

$$E(F) = \sum_{i=1}^{n} \frac{s_{1i} + s_{2i}}{S} I(S_{1i}, S_{2i}). \tag{3.7}$$

That is, s_{1i} is the number of the samples whose feature value is in the group of the label. The same value subsets of an ideal discriminating feature match altogether into the group subset or out of the group subset of the label. Thus, a sample can be concluded

to be in the label group or not by means of feature value. The information gained from the F feature is,

$$Gain(F) = I(s_1, s_2) - E(F). \tag{3.8}$$

Table 3.5. Information Gain and Chi-Squared Feature Selection for Different Feature Vectors in BOUN TCP Flood Dataset.

Chi-Squared Value	**Information Gain**	**Feature Name**
4456.10	0.71	Number of Unique Sources
4427.81	0.70	Number of SYN Packets
4393.43	0.70	Data per Flow
4274.73	0.68	Average Packet Size
4242.40	0.68	Number of TCP Packets per Flow
4220.15	0.67	Packets per Flow
3972.34	0.64	Number of UDP Packets per Flow
3929.75	0.63	Number of Hosts
3478.06	0.55	Number of Flow
1345.60	0.24	Number of Packets
1305.10	0.23	Number of TCP Packets
1185.12	0.22	Number of ACK Packets
706.42	0.13	Number of UDP Packets
161.03	0.03	Number of Unique Hosts
48.84	0.01	Number of RST Packets
36.24	0.01	Number of ICMP Packets

The resulting Information Gain for sixteen features is shown in Table 3.5. As the Information Gain decreases, the ability of a feature vector to distinguish between the attack and attack-free classes decreases. For convenience, the first six features that have the highest information Gain are chosen for probability distribution fitting.

4. DETECTION USING AUTO-REGRESSIVE MODELING

We have seen in Chapter 3that correlation coefficients between features change when they are under attack. We can make use of this result to detect DDoS attacks using multiple feature vectors. One way to use multiple feature vectors for anomaly detection is proposed in [20]. They offer an abrupt change detection using AR modeling that uses SNMP MIB variables and combines them with an operator matrix to form a traffic health function. We use traffic features and correlation coefficient matrix instead of SNMP MIB variables and operator matrix.

We use AR-based change detection for DDoS detection in this chapter. This method considers network features as time series. Although many methods use a time series-based approach for anomaly detection, we focus mainly on abrupt change detection based on AR modeling.

4.1. Abrupt change detection

An abrupt change is defined as any change in the parameters of a time series and can be modeled using an autoregressive (AR) process [20].

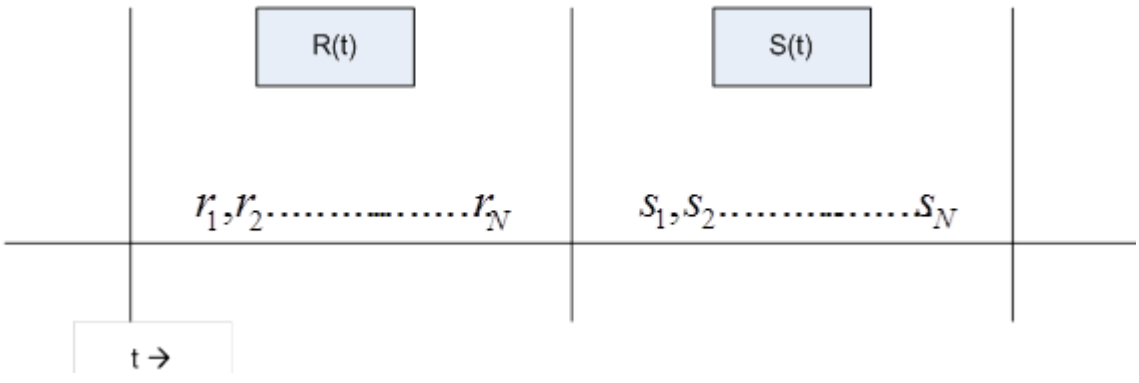

Figure 4.1. Piece-wise stationary segments.

We compare the variance of the residuals obtained from two adjacent piecewise stationary time windows for abrupt change detection. These are referred to as the learning and test windows, as shown in Figure 4.1. We initially used AR modeling on t-both windows and obtained residuals from this process.

Thus the data were divided into ten time lags piecewise stationary windows. The data are linearly modeled using a first-order AR process within a time window of size $N(N = 10)$. Piecewise stationary segments $R(t)$ and $S(t)$ shown in figure 4.1 $R(t)$ is learning window and $S(t)$ is test window. Non-overlapping windows were used in order to obtain less correlated residuals. An order p autoregressive process $AR(p)$ can be defined as:

$$X_t = \alpha_1 X_{t-1} + ... + \alpha_p X_{t-p} + \varepsilon_t, \tag{4.1}$$

where ε_t is an iid sequence of $N(0, \sigma^2)$ random variables, and the polynomial $\alpha(z) = 1 - \alpha_1 z - ... - \alpha_p z^p$. We can define $R(t)$ is the $AR(p)$ process specified in 4.1 with mean μ ,

$$R(t) = \{r_1(t), r_2(t), r_3(t), ...r_N\}. \tag{4.2}$$

Here we can state any $r_i(t)$ as $\tilde{r}_i(t)$ where $\tilde{r}_i(t) = r_i(t) - \mu$ and μ is the mean of the segment R(t). Now $\tilde{r}_i(t)$ can be estimated as an AR order p process (p=1) with a residual error ε_i,

$$\varepsilon_i(t) = \sum_{k=0}^{p} \alpha_k \tilde{r}(t-k), \tag{4.3}$$

where $\alpha_R = \{\alpha_1, \alpha_2, ...\alpha_p\}$ are the AR parameters, and $\varepsilon_i(t)$ is assumed to be white noise. The joint probability density function of $\varepsilon_1(t), \varepsilon_2(t)...\varepsilon_i(t)$ is given by,

$$f(\varepsilon_1, ...\varepsilon_i) = (2\pi\sigma^2)^{-\frac{N}{2}} exp\{-\frac{1}{2\sigma^2}\sum_{t=1}^{N}\varepsilon^2\}. \tag{4.4}$$

Change detection is performed using a hypothesis test based on the generalized likelihood ratio (GLR) [79]. The abnormality indicator value varies between 0 and 1. Substituting for $\varepsilon_i(t)$ from (4.4) and making use of the $x_{1-p}, x_{2-p}, ...x_0$ the likelihood function L of the $\varepsilon_i(t)$ is

$$L = (2\pi\sigma^2)^{-\frac{N}{2}} exp\{-\frac{1}{2\sigma^2}\sum_{t=1}^{N}(\sum_{i=0}^{p}\alpha_i\chi_{t-i})^2\}, \tag{4.5}$$

which can rewritten as in logarithmic likelihood format,

$$L = (2\pi\sigma^2)-N/2exp\{(-1/2\sigma^2)N\{a^TCa\}, \tag{4.6}$$

where a is the column vector given by $\{a^T\} = [1, \alpha_1, ...\alpha_p]$, and $C = [c_ij]$ is the $(p+1)(p+1)$ covariance matrix given by:

$$c_{ij} = \frac{1}{N}\sum_{t=1}^{N}\chi_{t-i}\chi_{t-j} \quad i, j = 0, 1, ...p. \tag{4.7}$$

To obtain the maximum-likelihood estimates of σ^2 and α we must maximize L with respect to σ_2 and α . This leads to the estimates σ_2 and α where $\tilde{\sigma}$ (covariance estimate)

$$\tilde{\sigma}^2 = a^TCa. \tag{4.8}$$

When we turn back to our notation, error is $N(0, \sigma_r^2)$ distribution. The joint likelihood of the residual time series was obtained as:

$$p(\varepsilon_{p+1}, ...\varepsilon_{NR}/\alpha_1, ...\alpha_p) = \left(\frac{1}{\sqrt{2\pi\sigma_R^2}}\right)^{N_R} exp\left(\frac{-\tilde{N}_R\hat{\sigma_R^2}}{2\sigma_R^2}\right), \tag{4.9}$$

where σ_R^2 is the variance of residual in segment $R(t)$ and $\tilde{N}_R = N_R - p$ and $\hat{\sigma_R}$ is the

covariance estimate of σ_R . Joint likelihood L of the residuals $R(t)$ and $S(t)$ is

$$l = \left(\frac{1}{\sqrt{2\pi\sigma_R^2}}\right)^{\tilde{N}_R} \left(\frac{1}{\sqrt{2\pi\sigma_S^2}}\right)^{\tilde{N}_S} exp\left(\frac{-\tilde{N}_R\hat{\sigma_R^2}}{2\sigma_R^2}\right) exp\left(\frac{-\tilde{N}_S\hat{\sigma_R^2}}{2\sigma_S^2}\right), \tag{4.10}$$

where $\tilde{N}_s = N_s - p$, σ_s^2 is variance of the residual in the segment $S(t)$. Two hypotheses are H_0 implying that no change. H_1 implying a change. Under the hypothesis H_0 we have; $\alpha_R = \alpha_S$ and $\sigma_R^2 = \sigma_S^2 = \sigma_p^2$ where σ_p^2 is the pooled variance is

$$l_p = \left(\frac{1}{\sqrt{2\pi\sigma_p^2}}\right)^{\tilde{N}_R+\tilde{N}_S} exp\left(\frac{-(\tilde{N}_R+\tilde{N}_S)\hat{\sigma}_P^2}{2\sigma_P^2}\right). \tag{4.11}$$

Under the hypothesis H_1 we have; $\alpha_R \neq \alpha_S$ and $\lambda_R^2 \neq \lambda_S^2$ and under hypothesis H_0 and $\alpha_R = \alpha_S$, $\lambda_R^2 = \lambda_S^2$ Furthermore on using the maximum likelihood estimates for the variance terms, we get the log likelihood ratio is

$$-\ln l = \tilde{N}_R(\ln\hat{\sigma_p} - \ln\hat{\sigma_R}) + \tilde{N}_S(\ln\hat{\sigma_p} - \ln\hat{\sigma_R}). \tag{4.12}$$

The log likelihood ratio $\ln l$ is compared with an optimally chosen threshold h where the threshold was exceeded were considered to be change points. That is

$$-\ln l > h ==> H_0 \quad \text{change} \quad -\ln l \leq h ==> H_1 \quad \text{no change.} \tag{4.13}$$

Let us define the hypothesis again, H_0 is implying no change, and H_1 is implying change. The expression for l is a sufficient statistic and is used to perform a binary hypothesis test. Under the hypothesis H_0, implying that no change is observed between the two Windows H_0 implying no change; likelihood lo under hypothesis H_1, implying that a change is observed between the two windows we have, $l_1 = 1$. In order to obtain a value for the likelihood ratio η that is bounded between $[0\,1]$, we define η as

$$\eta = \frac{l_0}{l_1 + l_0}. \tag{4.14}$$

Furthermore, on using the maximum likelihood estimates for the variance terms in equations (4.10) and (4.11) we get likelihood as

$$\eta = \frac{\sigma_R^{-\tilde{N}_R}\sigma_T^{-\tilde{N}_T}}{\sigma_R^{-\tilde{N}_R}\sigma_T^{-\tilde{N}_T} + \sigma_R^{-(\tilde{N}_R+\tilde{N}_T)}}, \tag{4.15}$$

where σ_R and σ_T are the variance of the residual in the learning window and the test window, respectively, $\tilde{N}_R = N_R - p$ and $\tilde{N}_T = N_T - p$, where p is the order of AR process, and N_R is the length of the learning window. There is a new definition of Likelihood ratio in [80] Qingtao et al. have called Log-likelihood Ratio (LLR) η_L as follows:

$$\eta_L = \frac{l_1}{l_0}, \tag{4.16}$$

After simplification, we get

$$\eta_L = \frac{\hat{\sigma}_p^{-(\tilde{N}_R+\tilde{N}_T)}}{\hat{\sigma}_T^{-\tilde{N}_T}\hat{\sigma}_R^{-\tilde{N}_R}}. \tag{4.17}$$

We form an abnormality vector by collecting the abnormality indicators obtained from the individual features in different time windows. The abnormality vector is used as a measure of the changes in normal network behavior. First, a (1 x n) input vector φ is constructed with components of likelihood ratio; η and $\varphi(t)$ is the Abnormality Vector which defined as

$$\varphi = [\eta_1, ...\eta_n]. \tag{4.18}$$

The method mentioned above is suitable for a one-dimensional feature vector. However, we want to include multiple feature vectors in DDoS detection. In order to overcome this drawback of the model, Thotthan et al. have suggested a combination matrix [22] which is also suggested in [20]. The individual abnormality vectors must be combined for alarm generation. Like Thottan's approach, we use a linear operator combining

the feature vectors to obtain a traffic health function. These kinds of operators are also used in quantum mechanics [81]. The linear operator is designed based on the correlation between the chosen feature vectors. In particular, the quadratic function is used to generate a continuous scalar indicator of network health. It is defined as

$$\xi_t = \varphi \mathbf{A} \varphi^T. \tag{4.19}$$

This network health indicator is interpreted as a measure of abnormality in the network obtained from individual traffic feature vectors.The operator matrix $\mathbf{A}$ is designed to obtain a scalar value for the measure of the transformation. For example $\varphi(t)$ can be chosen like $\varphi_{xyz} = \alpha[\eta_x, \eta_y, \eta_z]$. x, y and z represent the three feature vectors. So, operator matrix $\boldsymbol{A}$ is represented as

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}. \tag{4.20}$$

Elements of matrix $\mathbf{A}$ are composed of spatial correlation between features. Such as the coupling between x, y, and z. There was a different operator $\mathbf{A}$ matrix definition in [21] which is based on an abnormality vector. Anomaly detection is performed by applying a threshold to ξ_t at equation (4.19). Calculated health function for attack traffic can be seen in the proceeding sections.

We perform DETER testbed simulations, including ICMP, UDP, and TCP SYN flood attacks. Before applying AR(1) method, we select appropriate features using information gain from a set of candidate features.

4.1.1. ICMP Flood Attack

We generate 12 ICMP attacks attacks in total in four experiments using DETER testbed. We obtain sixteen feature vectors shown in Figure 4.2.

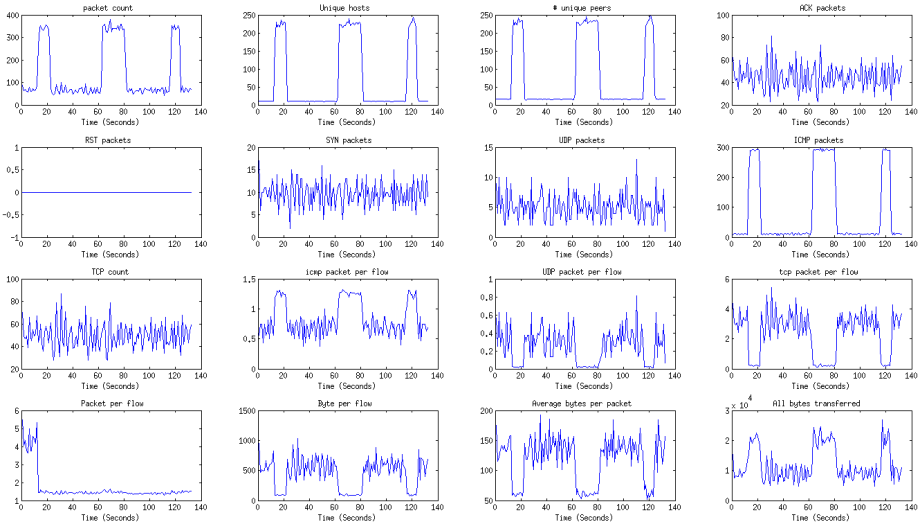

Figure 4.2. Traffic features obtained from ICMP flood attacks.

Figure 4.3 shows the traffic health function calculated by applying AR (1) method, and 4 of these attributes are selected with the information gain method. The red lines in Figure 4.3 show the actual time windows under attack. It is seen that the abrupt change detection method successfully detects the starting and ending points of the ICMP flood attack. Following features selected with information gain calculation:

- Number of packets.
- Number of unique hosts.
- Number of unique peers.
- Number of ICMP packets.

Alarm generation is performed by applying a threshold to the health function. The performance metrics are shown in Table 4.1

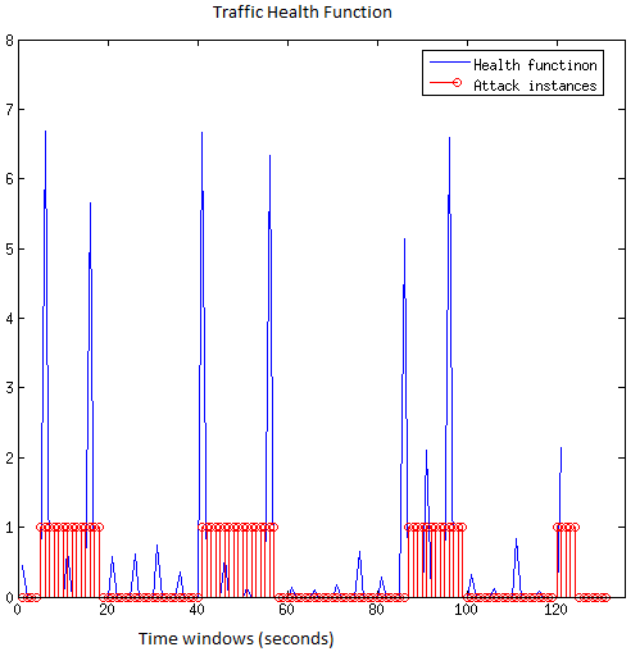

Figure 4.3. Traffic health function for one of the ICMP flood attack experiments in DETER testbed using information gain selected features and attack instances.

4.1.2. TCP SYN Flood Attacks

Unlike ICMP flood attack, we perform 1 TCP SYN flood experiment with DETER testbed, including ten attacks. Figure 4.4 show the traffic features obtained from the experiment as we can see from the figure that it is not straightforward to see attacks from feature vectors.

Following features are selected from sixteen candidate feature functions using the IG feature selection algorithm:

- Number of Bytes per-flow
- Average length of packets
- Number of TCP packets per-flow
- Number of Packets per-flow

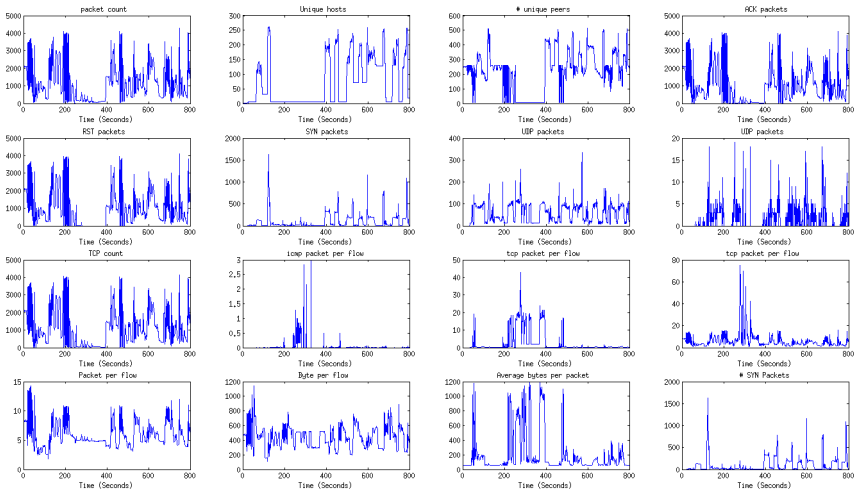

Figure 4.4. Traffic features obtained from TCP SYN flood attacks.

After selecting appropriate features, we perform abrupt change detection. As we can see in Figure 4.5, the health function has peaked at the beginning and end times of the attacks. The evaluation of TCP SYN flood detection is given in Table 4.1.

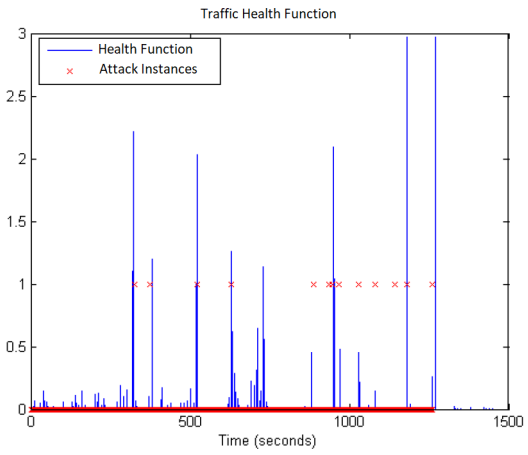

Figure 4.5. Traffic health function for TCP flood attacks using information gain selected features and attack instances.

4.1.3. UDP Flood Attacks

We have created three experiments for UDP flood attacks in DETER testbed. We perform a total of nine attacks and successfully detect 16 starting and ending time instances of attacks out of 18.

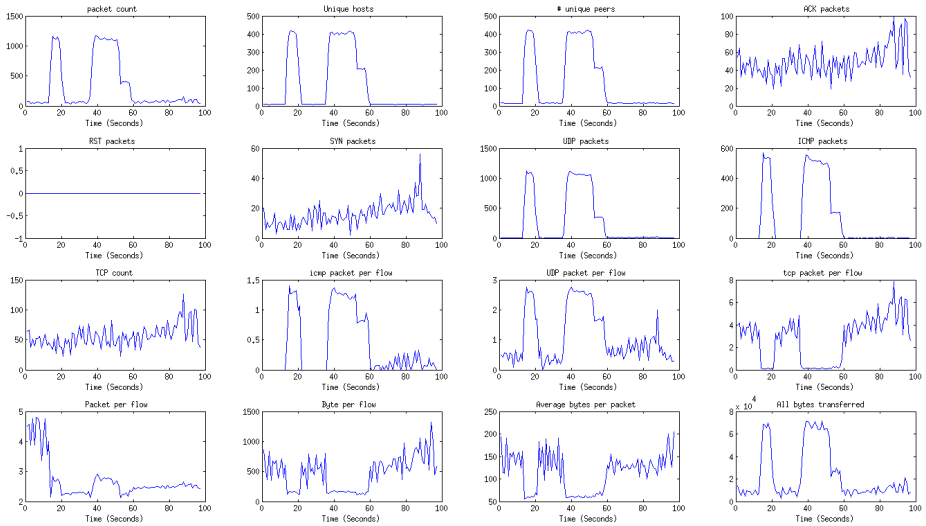

Figure 4.6. Feature vectors that can used for manual selection of features.

Selected features with information gain method are as follows:

- Number of Bytes per flow.
- Average length of packets.
- Number of Unique Hosts.
- Number of unique peers.

Figure 4.6 shows one of the experiments for UDP flood attacks we performed in DETER testbed. There are two attacks in that experiment, and as we can see from Figure 4.7, there are more false alarms compared to TCP and ICMP attacks. There can be several reasons for this result. One possible solution to this problem is increasing candidate feature space and performing the feature selection phase.

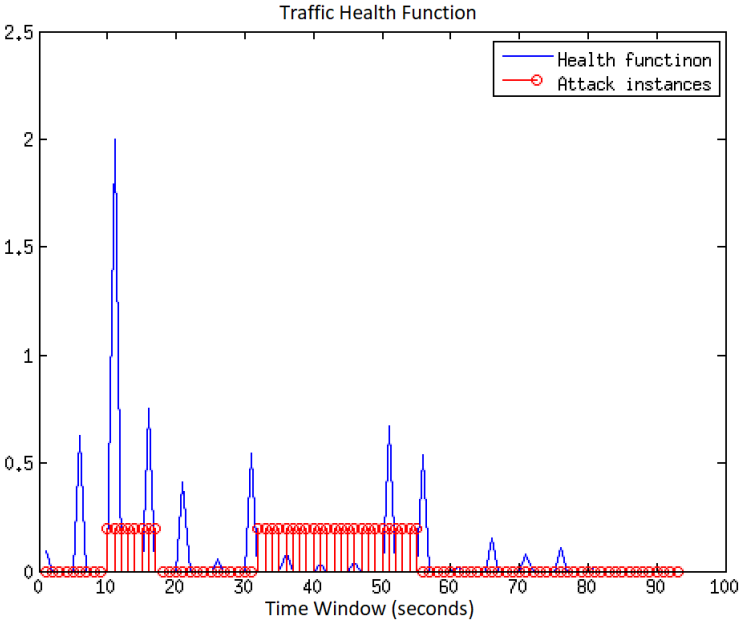

Figure 4.7. Traffic health function for UDP flood attacks.

4.2. Results and Conclusion

This chapter proposes a novel DDoS attack detection method based on the abrupt change detection method. This approach is evaluated with the attack experiments created with the simulations in the DETER testbed environment. The evaluation of the abrupt change detection using three different flood attack types is shown in Table 4.1. We can see from the results that this method efficiently detects change points in the features caused by DDoS attacks.

Table 4.1. Performance metrics obtained for ICMP UDP and TCP SYN flood attacks.

Attack Type	Number Of Attacks	True Positive	False Positive	False Negative
ICMP	12	24	1	0
TCP	10	20	3	0
UDP	9	16	5	2

As we can see from the results, although the proposed method helps generate

alarms at the start and end moments of attacks, it does not provide information about how long the attacks last. Because of this, we need to find a methodology that can use multiple traffic features for DDoS detection.

5. DDoS DETECTION with SIGNAL REPRESENTATION METHODS

In this chapter, we discuss Wavelet and MP-based DDoS detection approaches. These are MPMP, Adaptive Matching Pursuit Based Detection (AMP), and Wavelet-based intrusion detection methods. Initially, we evaluate and compare these approaches using the CAIDA datasets. CAIDA datasets are used to combine two datasets containing only DDoS attacks and only attack-free traffic. CAIDA datasets are frequently used to evaluate DDoS detection methods, including high DDoS attack density. These methods were evaluated and compared using the BOUN DDoS dataset, which has a lower DDoS attack intensity. We combine these methods with a decision-making mechanism using an Artificial Neural Network (ANN) to form a hybrid intrusion detection system. We evaluate the hybrid framework with datasets containing TCP SYN flood and UDP flood attacks.

We perform both an anomaly and misuse type of detection on DDoS attacks. The methods are combined with the decision module to detect multiple DDoS attack classes to form a hybrid detection mechanism. The decision engine is also combined with the Wavelet and the Matching Pursuit Mean Projection (MPMP) methods to compare similar signal representation methods. Performances of these three approaches are compared with each other using the CAIDA [43, 44], and Boğaziçi University DDoS attack dataset (BOUN DDoS) [37]. MP is a greedy algorithm that represents any signal as a linear expansion of atoms chosen from a redundant dictionary [82].

5.1. Sparse Signal Representation

A signal is said to be a sparse signal is it can be compactly expressed as a linear combination of basis vectors. Using an over-complete dictionary $\boldsymbol{D} \in \mathbb{R}^{nxK}$ that contains K basic signal atoms for columns, $\{\mathbf{d}_i\}_{i=1}^{K}$. Signal $\boldsymbol{f} \in \mathbb{R}^n$ can be represented as a linear combination of these atoms.

The representation of $\boldsymbol{f}$ can be either exact or approximate $\boldsymbol{f} \approx \mathbf{Dx}$. The vector $\mathbf{x} \in \mathbb{R}^K$ contains representation coefficients of signal $\boldsymbol{f}$.

Sparse representation of a signal over a dictionary is achieved by optimizing an objective function. In other words, we have to minimize the signal reconstruction error and optimize sparsity. Typical norms used for measuring the deviation are the l^p norms for $p = 1, 2, and, \infty$.

If $n < K$ and $\mathbf{D}$ is a full rank matrix, an infinite number of solutions can be obtained for sparse representation problem. Objective functions for sparse representation are as

$$\min_x \|\mathbf{x}\|_0 \quad subject \quad to \quad \mathbf{y}{=}\mathbf{Dx}, \tag{5.1}$$

or,

$$\min_x \|\mathbf{x}\|_0 \quad subject \quad to \quad \|\mathbf{y}\text{-}\mathbf{Dx}\|_2 \leq \epsilon, \tag{5.2}$$

where $\|.\|_0$ is the l^0 norm, and ϵ is a fixed value.

In sparse coding stage the coefficients x are calculated, given signal y and the dictionary $\mathbf{D}$ with solving Equations 5.1 or 5.2. The matching pursuit algorithm finds an approximate solution for these equations. Sparse representations of signals are used in many applications such as JPEG2000 coding, compression, computer communication.

The MP algorithm aims to achieve a sparse representation of a given signal $f \in IR^n$ using an overcomplete dictionary $\mathbf{D} \in IR^{nxK}$ where K is the number of atoms $g_{j}{}_{j=1}^{K}$ in the dictionary and $g \ll K$.

Let us define the representation of $f = Dx$ or $f \approx Dx$ subject to $\| f - Dx \|_p \leq \epsilon$

for some small number ϵ.The sparsest representation is the solution to either

$$(P_0) \quad min_x \parallel x \parallel_0 \quad subjectto \quad f = \mathbf{D}x, \tag{5.3}$$

or,

$$(P_0) \quad min_x \parallel x \parallel_0 \quad subjectto \quad \parallel f - \mathbf{D}x \parallel \leq \epsilon, \tag{5.4}$$

where $\parallel . \parallel_0$ l^0 is the counting of the non-zero entries of a vector.

5.1.1. Matching Pursuit

Matching Pursuit is a well-known technique for sparse signal representation. Mallat and Zhang first introduced the MP algorithm in 1993 [83]. Sparse signal representations are frequently used for solving problems related to signal processing and communication areas.

MP is a greedy algorithm that decomposes any signal into a linear expansion of waveforms selected from a redundant dictionary of functions. MP finds linear approximations of signals by iteratively projecting them over a redundant set of signals called a dictionary. Waveforms in the dictionary are chosen to express the signal best way.

Since MP is a greedy algorithm, it may give a suboptimal approximation [84]. However, it is useful for approximations when it is hard to develop optimal orthogonal approximations, as in high dimensional signals or images. The matching pursuits algorithm has been successfully applied to a variety of problems like signal representation [85], image and video compression [86] and target identification [87].

The approximations derived from a matching pursuit proposed in [88] can be refined by orthogonalizing the directions of projection. The resulting orthogonal Pursuit converges with a finite number of iterations of infinite-dimensional spaces, which is not

the case for a nonorthogonal pursuit. However, computational cost per iteration is increased due to the Gram-Schmidt procedure.

Tree-based search orthogonal matching pursuit algorithm (TB-OMP) is introduced in [89] to improve the approximation performance of the matching pursuit algorithm. Although the TB-OMP algorithm improves the approximation performance, its computation time requirement increases exponentially, making it impractical for specific applications. In [90] the flexible tree-search based orthogonal matching Pursuit (FTB-OMP) was proposed. The algorithm provides design parameters that give the flexibility to establish a trade-off between approximation performance and experimental time complexity.

An over-complete dictionary to use in the MP algorithm can be either chosen as a predefined set of functions or by adapting dictionary content to a given signal set. Selecting a predefined set of functions is simpler and leads to faster dictionary generation algorithms. In general, wavelets, curvelets, contourlets, short-time Fourier transforms can be used to generate these kinds of dictionaries. The effectiveness of using such dictionaries depends on the sparse representation application. As an example, some applications use signals that is a linear combination of well-predefined functions. In this thesis, we use both types of dictionaries.

5.1.2. Basic Matching Pursuit Algorithm

Let H be a Hilbert Space. Matching Pursuit decomposes a signal $\boldsymbol{f} \in H$ over a redundant dictionary $D = \alpha_1, \alpha_2, ..., \alpha_{n_d} \subseteq H$. Each $\alpha_i \in H$ is called atom in dictionary where $\| \alpha_i \| = 1$ and n_d is the number of atoms in dictionary. Given a vector f , the MP generates an approximation of y as a linear combination of p atoms from D as:

$$\hat{\boldsymbol{f}} \approx x_0\boldsymbol{\alpha_0} + x_1\boldsymbol{\alpha}_1 + \ldots + x_{p-1}\boldsymbol{\alpha}_{p-1}. \tag{5.5}$$

The difference between $\boldsymbol{f}$ and its approximation $\hat{f}$ is called the residue R_p and it is calculated as

$$\mathbf{R}_p = \boldsymbol{f} - (x_0\boldsymbol{\alpha}_0 + x_1\boldsymbol{\alpha}_1 + \ldots + x_{p-1}\boldsymbol{\alpha}_{p-1}). \tag{5.6}$$

The algorithm starts with finding $\boldsymbol{\alpha}_0$ that gives a maximum projection of $\boldsymbol{f}$. It can be shown as

$$\boldsymbol{\alpha}_0 = argmax_j \langle \boldsymbol{f}\boldsymbol{\alpha}_j \rangle. \tag{5.7}$$

The residue is updated by subtracting $\boldsymbol{\alpha}_0$ times its magnitude of projection x_0 from $\boldsymbol{f}$ and it is calculated as

$$\boldsymbol{R}_1^{(x)} = \boldsymbol{f} - \boldsymbol{\alpha}_0 x_0, \tag{5.8}$$

where $\mathbf{x}_0 = \langle \boldsymbol{f}, x_0 \rangle$ is called coefficients of x_0.This Process continues iteratively by projecting $\mathbf{R}_i$ on dictionary atoms and updating $\mathbf{R}_{i+1}$. After p iterations $\boldsymbol{f}$ can be written as

$$\boldsymbol{f} = \sum_{i=0}^{p-1} x_i \boldsymbol{\alpha}_i - \boldsymbol{R}_p. \tag{5.9}$$

Iterations can continue until some stopping criteria have been met. The energy of represented signal $\mathbf{f}$ is conserved in sparse representation process. Since all atoms $\mathbf{d}$ have unit energy conservation of energy is can be written as

$$\|pmbf\|^2 = \sum_{n=0}^{p} \|x_n\|^2 + \|\mathbf{R}_{p-1}\|^2. \tag{5.10}$$

5.1.2.1. K-SVD Dictionary Generation Algorithm. With ever-growing computational capabilities, the computational cost may become secondary in importance to the improved performance achievable by methods that adapt dictionaries for special classes

of signals. We can consider a different route for using the dictionary by designing it D based on learning from data. For this purpose, we use the K-SVD dictionary generation algorithm proposed in [91]. The goal of the K-SVD algorithm is to find the dictionary D that yields sparse representations for the training signals.

Dictionary used for MP approximation is an overcomplete set of vectors defined in a Hilbert space. Overcompleteness of a set means that it has more members than the dimensionality of its members. The advantage of the overcompleteness of a dictionary is its robustness in case of noisy or degraded signals. Also, it introduces a greater variety of shapes in the dictionary, thus leading to sparser representations of a variety of input signals.

We believe that training dictionaries with sample data can lead to the adoption of detection algorithms to traffic itself. It is also necessary to model traffic in order to perform anomaly detection. For this purpose, the dictionary of the MP algorithm is used as a model for attack detection.

The modeling used in this study is simply a dictionary creation process for the MP algorithm (Figure 5.1) using the training feature set. Different dictionaries are created for attack and legitimate traffic using the K-SVD algorithm, which is an iterative optimization algorithm [91, 92]. Aharon et al. proposed the K-SVD algorithm as a generalized k-means clustering algorithm. Initially, characteristic feature vectors are generated from training network traffic dumps. These characteristic feature vectors are used to construct dictionaries of attack and legitimate training datasets.

A code book used to represent a wide family of vectors $\mathbf{F} = f_{i i=1}^{N} \quad (K \ll N)$ by nearest neighbor assignment. The objective function of the K-SVD algorithm is

$$\min_{\mathbf{D},\mathbf{X}} \| \mathbf{F\text{-}DX} \|_F^2 \quad subjectto \quad \forall i, \| x_i \|_0 \leq T_0. \tag{5.11}$$

In the sparse coding stage, where it is assumed that $\mathbf{D}$ is fixed, and consider the

above optimization problem as a search for sparse representations with coefficients summarized in the matrix $\mathbf{X}$ using

$$\| \mathbf{F\text{-}DX} \|_F^2 = \sum_{i=1}^{N} \| y_i - \mathbf{D}x_i \|_2^2. \tag{5.12}$$

By using equation 5.12 in equation 5.11 we can rewrite objective as:

$$\min_{\mathbf{D},\mathbf{X}} \| \mathbf{F\text{-}DX} \|_2^2 \quad subjectto \quad , \| x_i \|_0 \leq T_0, \forall i = 1, 2, \ldots, N. \tag{5.13}$$

In order to update the dictionary together with the non-zero coefficients. Assume that both $\mathbf{X}$ and $\mathbf{D}$ are fixed, and we want to update only one column in the dictionary $\mathbf{d}_k$ and coefficients corresponding to it using

$$\begin{aligned} \| \mathbf{F\text{-}DX} \|_F^2 &= \| \mathbf{F} - \sum_{j=1}^{K} d_j x_T^j \|_F^2 \\ &= \| \sum_{j \neq 1} d_j x_T^j - d_k x_k^T \|_F^2 \\ &= \| \mathbf{E}_k - d_k x_T^k \|_F^2 . \end{aligned} \tag{5.14}$$

The matrix $\mathbf{E}_k$ stands for the error for all N the examples when the k^{th} atom is removed. $K-1$ term is assumed to be fixed with this equation, and the k^{th} term is updated.

5.2. Adaptive Matching Pursuit Based DDoS Detection

In this section, we give detailed information about the AMP approach. The AMP approach contains three main parts: Feature Generation Module, Dictionary Generation Module, and Alarm Generation Module. Figure 5.1 shows the block diagram of the AMP approach.

The feature generation module extracts attributes of network packets from network-based traffic data. The module partitions the dataset into equally spaced time windows and calculates 16 attributes for every time window. The characteristic feature

vectors are created from the normalized attribute vectors. Characteristic feature vectors obtained from the training dataset are classified into multiple classes depending on whether they belong to normal or attack traffic.

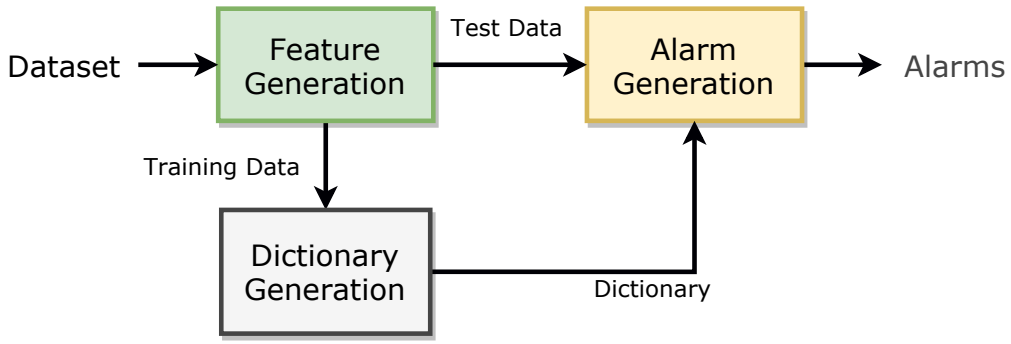

Figure 5.1. Block diagram of the AMP DDoS Detection approach.

The dictionary generation module generates dictionaries from training data. Generating dictionaries from network traffic provides the adaptation of the AMP method to training data. Separate dictionaries are generated from different traffic classes.

The alarm generation module calculates the norms of residuals for every time window and generates abnormality values.

5.2.1. Features and Feature Generation

Packet flowing throughout the network contains various properties, including source-destination IP addresses, TCP flags, and source/destination ports, traffic flow information [93]. This diversity results in high-dimensional attribute space. Attribute diversity examples include traffic flow information [94, 95], router SNMP MIB variables [96], TCP header information [50], entropy-based features [97]. One of the challenges about attributes is finding the best set of features that represents different types of DDoS attacks from a wide variety of attributes. Besides, multiple traffic attributes are subject to change simultaneously under DDoS attack [98–100].

In the feature generation phase, we extract numerical traffic attributes from the raw network packets dataset. Initially, we divide network traffic into equally spaced time windows. Then we count some specific properties of network packets in the time window and form attribute vectors $\boldsymbol{f}$ defined in Table 3.4.

The network traffic is handled in two different ways in this work as traffic attributes and characteristic feature vectors. One-dimensional attribute vectors are affected by DDoS attacks in various ways. The effect of a DDoS attack on the attributes varies according to the intensity/type of the attack, the size of the victim network, and the variety of attacking IP addresses. Sixteen different flow-based and packet-based traffic attributes are obtained from network traffic. The attributes used in this study are chosen based on their potential to reveal the properties of DDoS attacks. The explanations of these attributes and some of the academic studies using these attributes are shown in Table 3.4.

Packet-based attributes are obtained by computing the characteristics of the packets in the network traffic. Flow information is not taken into account while generating these attributes. The packet-based attributes are the number of SYN, RST, ACK, TCP, UDP, and ICMP packets. These are counted using packet header information.

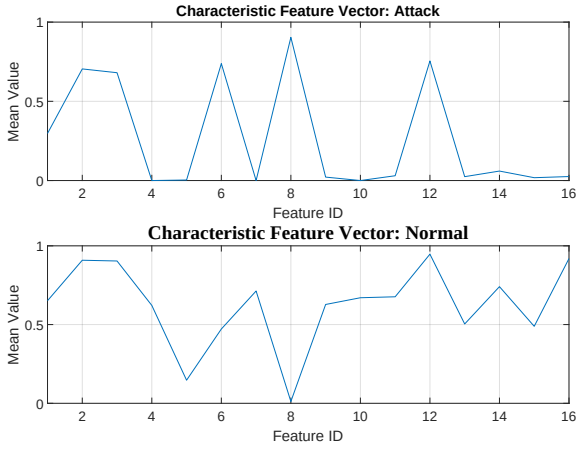

Figure 5.2. Mean of the characteristic feature vectors for Attack and Normal samples in CAIDA'07 and CAIDA'08 dataset.

Traffic flow is characterized as a sequence of packets that share common properties, such as the same source/destination IP addresses and source/destination ports. This work creates network flows by considering source/destination IP address pairs and source/destination TCP ports pairs of network packets. The flow-based attribute vectors used in this work are the number of flow, packet per flow, data per flow, TCP, UDP, ICMP packets per flow. Here while calculating average packet size, data is counted as the length of the payload of the packets.

To capture the effect of DDoS attacks on different traffic attributes simultaneously, we propose the characteristic feature in this work. The main idea behind building this feature is to model normal and attack traffic attributes for every time window. The characteristic feature vectors obtained from the attack data differ from those obtained from attack-free data, as seen in Figure 5.2.

For every time window, a characteristic feature vector is generated. Every attribute vector is normalized within itself and combined to form characteristic feature vectors as follows:

$$\boldsymbol{f}_i = \{\hat{f}_{1i}, \hat{f}_{2i}, \ldots, \hat{f}_{ni}, \ldots, \hat{f}_{16i}\}, \qquad i = \{1, 2, \ldots, k\}, \tag{5.15}$$

where, $\hat{f}_{ji}$ is the i^{th} element of j^{th} normalized attribute vector $\hat{\boldsymbol{f}}$. The attribute index j ranges between 1 and 16 because there are 16 attribute vectors.

5.2.2. Dictionary Generation

Different types of dictionaries are generated from the training dataset in this work. From attack samples of feature vectors in training data, a misuse dictionary is generated. Similarly, from the attack-free samples of feature vectors in training data, an anomaly dictionary is created. Figure 5.3 shows the block diagram of the dictionary generation process.

To obtain dictionaries, an iterative optimization algorithm K-SVD is used. K-SVD is a generalized k-means clustering algorithm proposed in [91, 92]. In the dictionary generation process, a dictionary that consists of K atoms is produced from the training set of features. Dictionary size is determined experimentally in this work. Let us construct a matrix $\boldsymbol{Y}$ from characteristic feature vectors obtained from the training dataset. The objective function of the K-SVD algorithm is as follows:

$$\min_{\boldsymbol{D},\boldsymbol{x}} \| \boldsymbol{Y} - \boldsymbol{D}\boldsymbol{x} \|_F^2 \quad subject\ to \quad \forall i, \| \boldsymbol{x}_i \|_0 \leq \epsilon \tag{5.16}$$

Where $\| \, . \, \|_F^2$ is Frobenius norm and $\| \, . \, \|_0$ is the L_0 norm of a vector. According to the equation (5.16), the K-SVD algorithm aims to produce the dictionary that gives the smallest residual value in the Frobenius norm sense using the given data set. The training dataset can contain different traffic classes like attack-free traffic classes and various types of attacks. For every traffic class, a separate dictionary is created. As a result, Frobenius norms of the residuals obtained using these dictionaries of a specific traffic class have smaller values for vectors that are in the same category. Similarly, the vectors of different traffic classes result in higher norms.

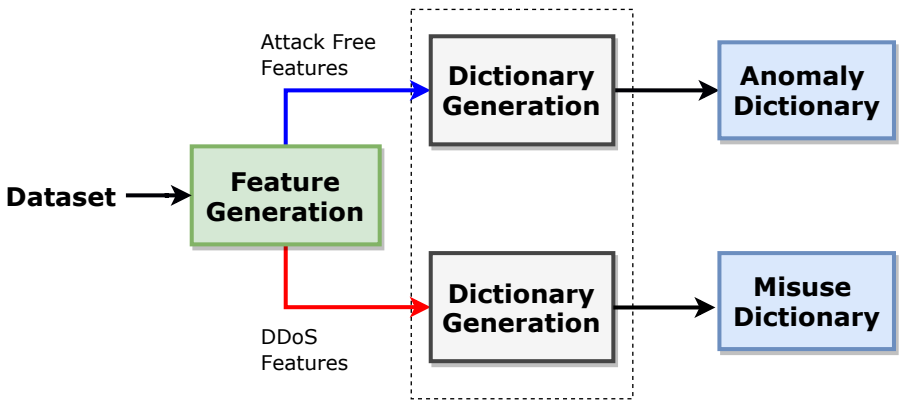

Figure 5.3. Block diagram of dictionary generation module.

5.2.3. Alarm Generation

For every time window in the test dataset, a characteristic feature vector is obtained using the feature generation module. These feature vectors are decomposed by

the MP algorithm using the dictionaries obtained from the dictionary generation module. The abnormality indicator value is calculated from the resulting residual vectors as follows:

$$\psi_i = \| \boldsymbol{r}_i \|^2, \tag{5.17}$$

where ψ_i is the abnormality indicator value obtained for i^{th} time window, and $\| \, . \, \|^2$ is the L_2 norm of a vector. Alarms are created by applying a threshold to the abnormality indicator vector $\boldsymbol{\psi}$. The pseudo-code for alarm generation is shown in Figure 5.4.

Input: Dictionary generated from training dataset $\mathbf{D}$, characteristic feature vectors $\boldsymbol{f} = \{\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_k\}$, maximum number of iterations M, threshold τ, maximum number of time windows k.

Output: Alarm

Initialization; $i \leftarrow 1$

Repeat: Find

$R_M = y_i - \sum_{i=0}^{M-1} c_i \boldsymbol{\alpha}_i$

Calculate: $\psi_i = \|R_M\|^2$

using $\boldsymbol{D}$ and MP algorithm.

Alarm generation

if $\psi_i \leq \tau$ $alarm_i = 0$ **else** $alarm_i = 1$

Until: $i = k$

Figure 5.4. AMP Alarm Generation pseudo-code.

The abnormality indicator vectors are evaluated differently according to the dictionary type. If the anomaly dictionary is utilized, the abnormality value is expected to increase under attack. Similarly, the abnormality indicator value is expected to decrease under attack when the misuse dictionary is utilized. Since the K-SVD algorithm generates dictionaries that give minimum residual norm with a maximum number of non-zero elements, this behavior is expected as a result of the objective function of the K-SVD algorithm. The same approach is accurate for misuse dictionaries and

legitimate traffic.

The histograms of abnormality indicator vectors obtained for CAIDA datasets using misuse and anomaly dictionaries are shown in Figure 5.5. It can be seen from Figure 5.5 that the distribution of abnormality vectors obtained from the misuse dictionary has higher values for attack-free data when compared to attack data.

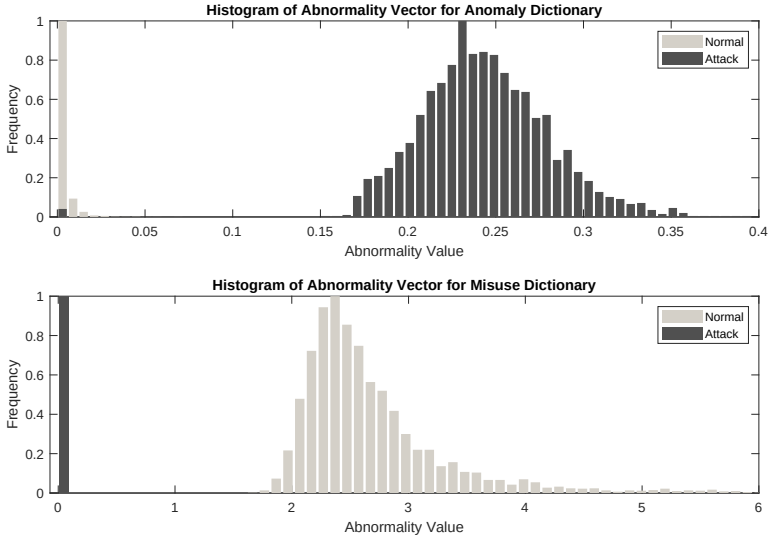

Figure 5.5. Normalized histograms of abnormality vectors obtained using anomaly and misuse dictionaries for the CAIDA dataset in the AMP method.

5.3. Detection Using Matching Pursuit Mean Projection

Structured predefined dictionaries are commonly used in matching pursuit methods. The anomaly detection method proposed in [16] practices a dictionary that consists of atoms of Gabor base functions. Gabor base functions provide optimal joint time-frequency localization.

Structured predefined dictionaries are commonly used in matching pursuit methods. The anomaly detection method proposed in [16] practices a dictionary that consists of atoms of Gabor base functions. Gabor base functions provide optimal joint

time-frequency localization. A real Gabor function can be expressed as:

$$g_\gamma(t) = K(\gamma)\exp\{-\pi(\frac{t-u}{s}))\sin(2\pi\frac{w}{N}(t-u)+\phi), \tag{5.18}$$

where N is the size of signal for which dictionary, $K(\gamma)$ is normalizing constant to achieve atom unit energy such that $\|g_\gamma\| = 1$. And $\gamma = \{u, w, s, \phi\}$ denotes parameters of the dictionary functions corresponding to time, frequency, scale, and time shift. Dictionary used to calculate MPMP consist of one-dimensional Gabor base functions. The dictionary, $\boldsymbol{D}$ is built using ten different scales and 50 different frequencies to create an over-complete set of base functions.

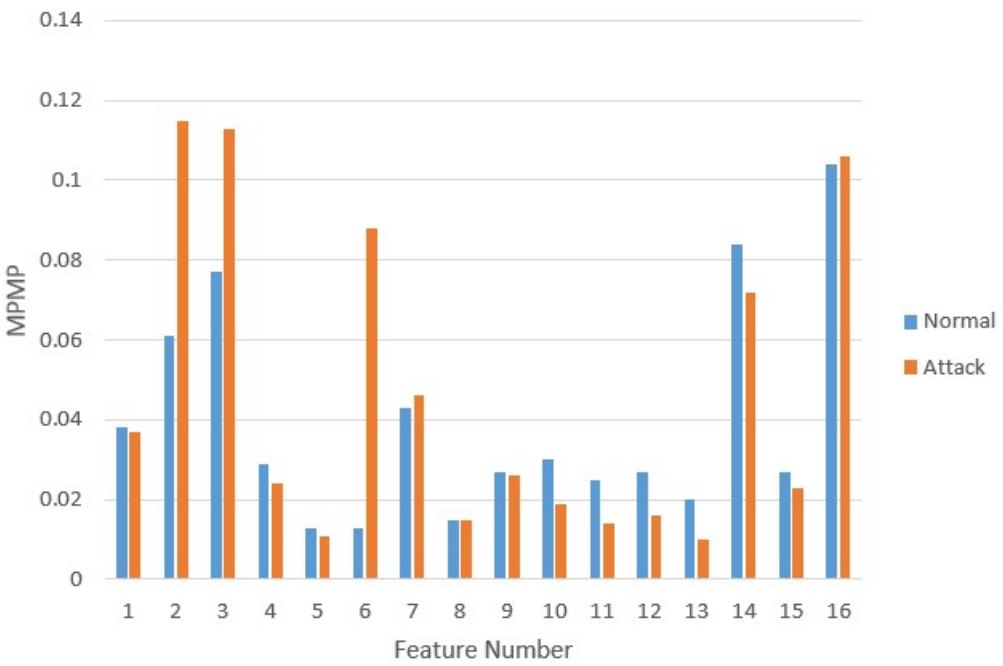

Figure 5.6. Average MPMP values obtained for attack and normal traffic using 16 feature vectors.

One-dimensional traffic attributes are partitioned into signal windows of 10. This signal is decomposed with the use of the MP algorithm and structured Gabor dictionary. After MP decomposition, we achieve projection coefficients of c_k, which are used for creating normal traffic profiles. MP algorithm give 3 outputs corresponding atoms

$\boldsymbol{\alpha}$, residues $\boldsymbol{r}$ and weights c. Matching pursuit mean projection is defined as:

$$MPMP = \frac{1}{M} \sum_{i=0}^{M-1} c_i. \tag{5.19}$$

The main idea behind this approach is to utilize the relation between the energy of the signal and MPMP. If the energy of the specified attribute increases, the MPMP value also increases.

Table 5.1. Obtained mean MPMP for TCP Flood Attack.

Atrributes	**Normal**	**Attack**
Number of Unique Hosts	0.038	0.037
TCP packet per flow	0.061	0.115
UDP packet per flow	0.077	0.113
Packet per flow	0.029	0.024
Data per flow	0.013	0.011
Average packet length	0.013	0.088
Total data transferred	0.043	0.046
Number of unique flows	0.015	0.015
Number of UDP packets	0.027	0.026
Number of Packets	0.030	0.019
Number of ACK packets	0.025	0.014
Number of TCP packets	0.027	0.016
Unique destinations	0.020	0.010
Number of SYN packets	0.084	0.072
Number of RST Packets	0.027	0.023
Number of ICMP packets	0.104	0.106

The difference between MPMP of the time window in the test dataset and the average MPMP obtained from attack-free samples in the training dataset is used as an anomaly indicator value. When this value exceeds a certain threshold, an alarm

is generated. We can see the average MPMP values obtained for attack and normal traffic using 16 feature vectors. As we can see that attack and normal MPMP values are different in features with high information gain.

Traffic attributes in Table 5.1 are used to create a 1D signal with length 10. This signal is decomposed with the use of MP transformation. After MP decomposition, we achieve projection coefficients c_k, used to create normal traffic profiles.

The difference between the MPMP value obtained from the current time window and the average attack-free MPMP value is used for detection.

5.4. Detection Using Wavelet Transform

Wavelet decomposition represents a signal using a series of orthogonal wavelets. To detect DDoS attacks using Wavelet, we decompose the input signal into sub-bands and calculate the differences of energies of sub-bands. The concept of Wavelet transform is defined in [101] as follows:

$$W_d f(m,n) = \sum f(x).\Psi_{m,n}(x), \tag{5.20}$$

where $\Psi_{m,n}$, means a family of discrete Wavelet functions. Detection method in this section is proposed in [15]. Deaubechies type Wavelet is used to decompose network traffic features as proposed in [15]. Detection is performed by using energy of three DWT sub-bands $E_W(i)$ using approximation coefficients $\boldsymbol{ca}_1, \boldsymbol{ca}_2, \boldsymbol{ca}_3$. The energy of i^{th} sub-band, $E_W(i)$ using K coefficient is calculated as:

$$E_W(i) = \sum_{n=1}^{K} \boldsymbol{ca}_i^2(n). \tag{5.21}$$

The difference En between $E_W(i)$ of three different sub-bands are used as an abnormality indicator value for the Wavelet-based detection approach. Alarms are generated by thresholding abnormality indicator value En.Wavelet transformation with three de-

composition levels is presented in figure 5.7.

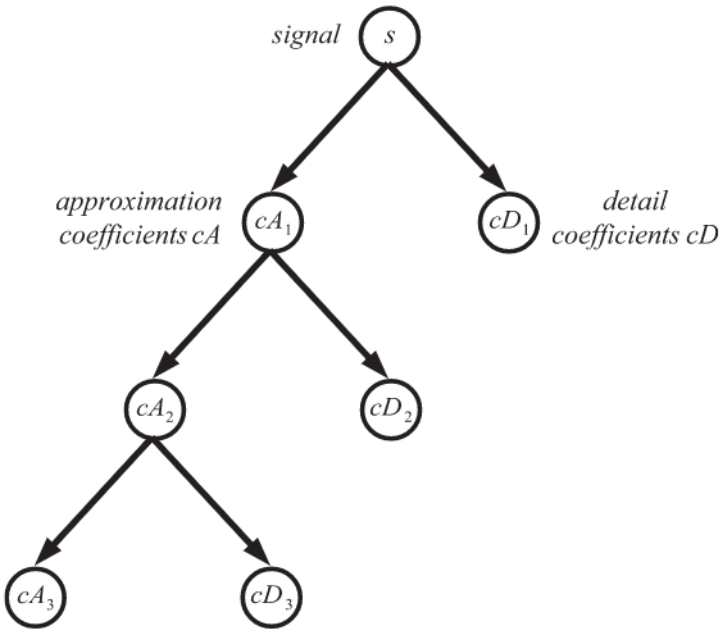

Figure 5.7. Wavelet transformation with three decomposition levels.

5.5. Hybrid Detection Framework with Signal Representation Methods

Using the AMP method, it is possible to generate dictionaries to perform anomaly and misuse detection. We obtain abnormality indicator values using the MP algorithm for each dictionary. Combining the abnormality vectors obtained from different dictionaries with a decision module, we obtain a hybrid detection framework. Additionally, the hybrid detection framework can detect together to achieve multi-class detection. This section proposes a hybrid detection framework that can concurrently identify multiple traffic classes. The proposed hybrid detection framework combines the AMP method with a decision module. The proposed framework combines anomaly and misuse methods to obtain a hybrid intrusion detection.

5.5.1. Decision Module

In this work, the ANN is employed as the decision mechanism. ANN is the combination of a large number of interconnected processing elements (nodes) that demonstrate the ability to learn and classify data using the information in the training patterns of data. ANN is a supervised classification algorithm and requires training. The ANN topology used in the decision module is shown in Figure 5.8.

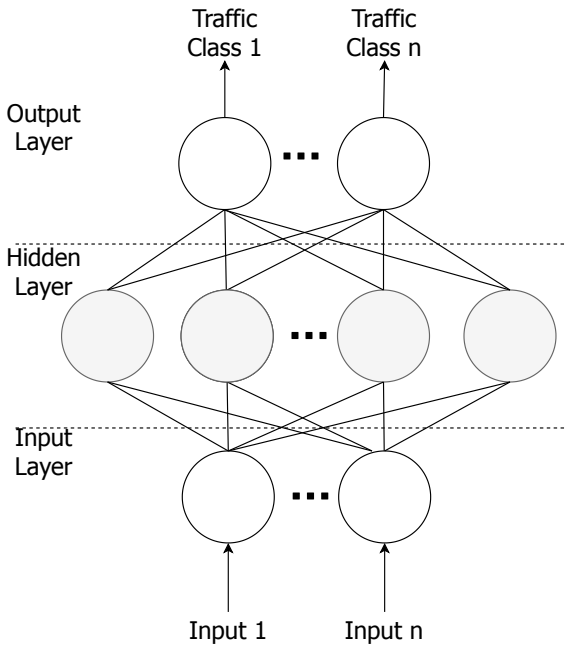

Figure 5.8. Artificial Neural Network structure used as a decision module. The number of input layer changes depending on the detection method. The number of output layer changes depending on the number of traffic classes to be detected.

Training the ANN includes adjusting the values of the weights and biases of the network to optimize network performance. We use feed-forward ANN and mean squared error as the performance function calculated as:

$$e_{mse} = \frac{1}{N} \sum_{i=1}^{N} (t_i - a_i)^2. \tag{5.22}$$

The transfer function used in neural network is Hyperbolic Tangent Sigmoid Transfer

Function and calculated as:

$$t(x) = \frac{2}{1 + exp(-2^x)} - 1, \tag{5.23}$$

where a is the output of the neural network, N is the sample size, and the t are the target outputs.

The ANN used in this work has 20 nodes in the hidden layer and three neurons for the output layer used in three traffic classes and two neurons for the output layer used in two traffic classes detection.

The ANN topology used in the decision module is approximately the same with different approaches. The number of inputs of the ANN differs according to the utilized detection method. MPMP and Wavelet approach generated 16 inputs to the decision module, corresponding MPMP, and En of 16 feature vectors.

The AMP method generates one abnormality indicator value for each dictionary. So for the dataset that includes two traffic classes, the AMP method generates two abnormality indicator vectors. Similarly, the AMP method generates three abnormality indicator vectors. As a result, two inputs in 2 traffic class cases and three inputs for three traffic class cases are used for ANN.

5.5.2. Hybrid Detection Framework Training and Alarm Generation

The overall hybrid detection system with the AMP, MPMP and Wavelet methods are shown in Figure 5.9. The decision module generates alarms using abnormality values generated by the AMP, MPMP, and Wavelet methods.

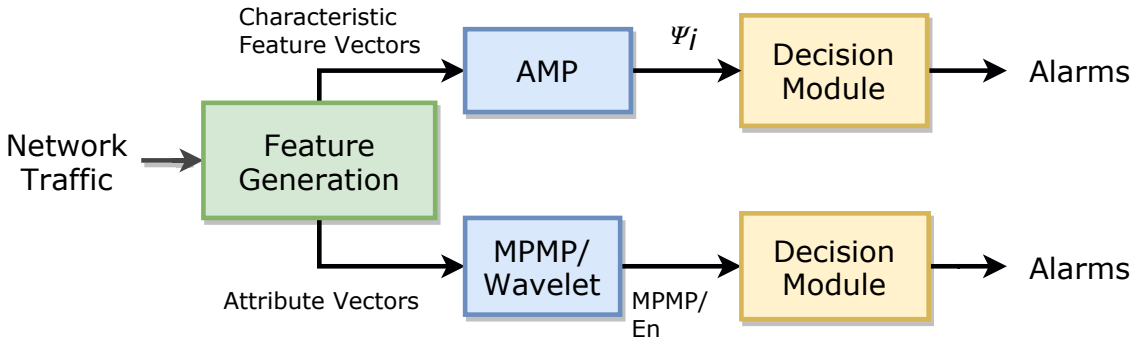

Figure 5.9. Block diagram of AMP-based Hybrid DDoS detection framework, MPMP, and Wavelet methods with decision module.

The hybrid detection framework using the AMP method requires to be trained using a training dataset. The training has two phases corresponding to dictionary generation and training of the ANN in the decision module, as seen in Figure 5.10. Initially, a separate dictionary is generated for each network class. The decision module is trained using the abnormality indicator vectors obtained from the training data.

For every time window, an abnormality indicator value is calculated using dictionaries corresponding to each network class in the alarm generation phase. The decision module generates alarms using the abnormality indicator values using the trained ANN.

5.5.3. Hybrid Detection Method using Wavelet and MPMP

In this section, we describe the usage of MPMP and Wavelet methods with the decision module. As we mentioned before, we use ANN in the decision module. In the case of MPMP and Wavelet approaches, similar to the AMP method, the framework calculates abnormality values defined in Equations (5.19) and (5.21) and generate alarms using them with the decision module.

The MPMP value calculated for each attribute vector is used to train the ANN network. Also, they are fed into ANN to generate alarms in the test dataset.

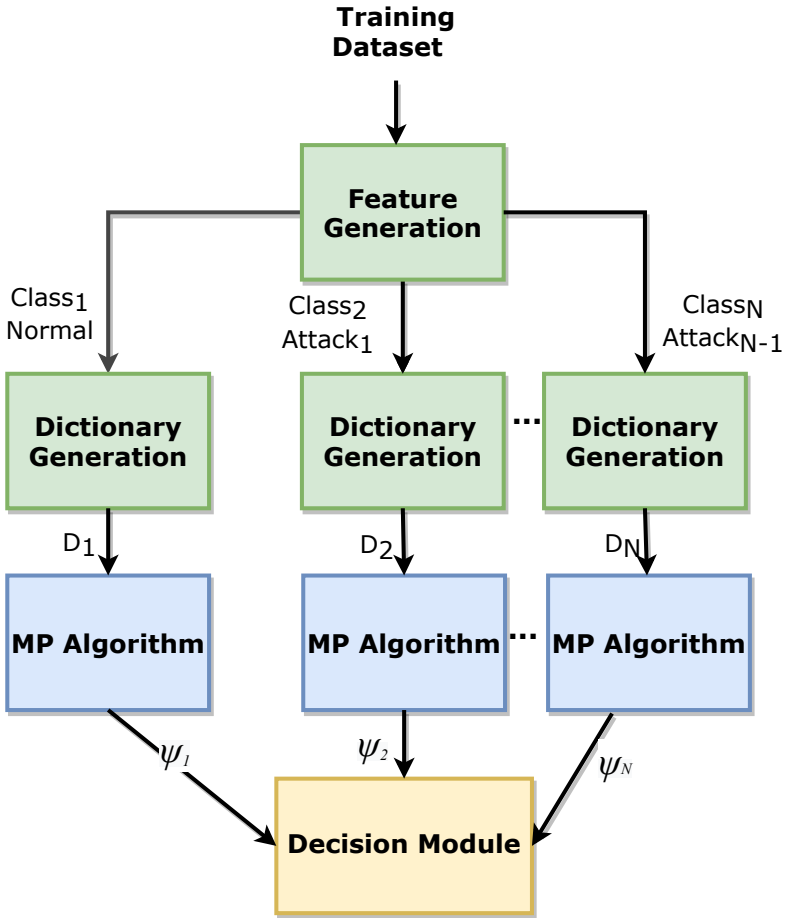

Figure 5.10. Training Hybrid DDoS detection framework based on the AMP method for multiple traffic classes.

Similarly, for Wavelet, the energy difference En between the different layers of attribute vectors, shown in the equation (5.21) are used to train the ANN network.

We do not build models from the training dataset in MPMP and Wavelet approaches. However, the ANN in the decision module learns about the normal and attack behaviors of these approaches. Therefore the resulting framework will be called a hybrid detection framework using Wavelet and MPMP.

5.6. Evaluation

In this section, three methods mentioned in this study were evaluated using two data sets. MPMP and Wavelet methods handle one-dimensional attribute vectors each time. As a result, the anomaly indicator value is calculated for 16 different attribute vectors. In evaluation, only the result obtained for the attribute that gives the highest CID value is included in the result tables.

5.6.1. Algorithm Complexity Analysis

The complexity analysis of the proposed algorithms is divided into two phases as training and detection. We compare three different approaches in terms of computational complexity.

The AMP training phase includes dictionary generation and training of ANN. The complexity of K-SVD algorithm is $O(NM^2K)$ [102] for dictionary $D \in \mathbb{R}^{(MxN)}$, using N number of data vectors. There is no training phase in the Wavelet approach. For the MPMP method, we need to calculate the MPMP of attack-free samples of the training dataset.

We can calculate the complexity of ANN using the number of operations between neurons. We have an ANN with one hidden layer. The time complexity for training a neural network with three layers with respectively i, j, and k nodes, with n data samples. The computational complexity of the ANN is $O(n(ij + jk.))$. The values n, j, k are the same for the MPMP, Wavelet, and AMP methods.

As a result, complexities can be compared using the difference between input neurons. The number of inputs i is equal to the number of dictionaries of the AMP method. This number equals 3 for three traffic classes and 2 for two traffic classes. In the case of the Wavelet and MPMP methods, the number of inputs depends on the number of traffic attributes shown in Table 3.4. As a result, $i = 16$ for two and three

traffic classes. The AMP method produces lesser input for ANN. Hence the complexity is reduced compared to the MPMP and Wavelet methods.

5.6.2. Evaluation for Two Traffic Classes

The results obtained using CAIDA datasets are shown in Table 5.2. The CAIDA dataset includes two separate datasets containing normal and attacks traffic. As a result, we obtain nearly perfect detection scores for these datasets. Because of a lower false-positive rate, AMP DDoS detection has a higher CID value. Detection using a misuse dictionary provides perfect detection, and detection using an anomaly dictionary provides %99.6698 TPR with zero FPR. The Wavelet and MPMP based detection use one-dimensional traffic attribute vectors. As a result of this, we obtain 16 different results using these methods. Only the best detection performance achieved upon 16 results is included in the Tables 5.2, 5.3, and 5.4.

Table 5.2. Detection performance of AMP, MPMP and Wavelet based approaches using CAIDA dataset.

Method	**CID**	**TPR (%)**	**FPR (%)**	**AUC**	**Acc (%)**
Wavelet	0.98	99.75	0.04	0.99	99.89
MPMP	0.98	99.75	0.04	0.99	99.98
Misuse AMP	1	100	0	1	100
Anomaly AMP	0.98	99.66	0	0.99	99.89

It is not a surprise to achieve perfect detection for CAIDA datasets since its a combination of two different datasets, including an attack-free dataset and a DDoS dataset. BOUN DDoS datasets are used to achieve a better comparison of the methods of this work.

Table 5.3. Detection performance of AMP, MPMP and Wavelet based approaches using BOUN TCP SYN flood dataset.

Method	**CID**	**TPR (%)**	**FPR (%)**	**AUC**	**Acc (%)**
Wavelet	0.84	93.26	0.28	0.98	98.39
MPMP	0.96	98.91	0.01	0.99	99.78
Misuse AMP	0.95	99.47	0.46	1	99.49
Anomaly AMP	0.96	99.09	0.17	1	99.69

Table 5.4. Comparison of AMP, MPMP and Wavelet based DDoS detection for BOUN UDP flood dataset

Method	**CID**	**TPR (%)**	**FPR (%)**	**AUC**	**Acc (%)**
Wavelet	0.896	95.532	0.052	0.982	99.088
MPMP	0.947	97.872	0.001	0.995	99.585
Misuse AMP	0.924	99.110	0.667	0.998	99.152
Anomaly AMP	0.958	99.778	0.407	0.999	99.628

The comparative results for the TCP and UDP dataset are shown in Table 5.3 and Table 5.4. The MPMP and Wavelet methods results are obtained using the attribute vector, which gives the highest CID value among 16 attribute vectors. For the simplicity of this thesis, we did not provide the results for all feature vectors for comparison. The following inferences should be considered when evaluating these tables:

- In the evaluation of the MPMP approach, the number of unique hosts provides the highest CID (0.96) value for the TCP SYN flood attack. The second highest CID is 0.92, and it is obtained using the TCP SYN packets attribute. For other attribute vectors, an average CID of 0.17 is obtained. This average value indicates that other attribute vectors do not provide proper detection using the MPMP method.
- The closest CID is the unique flows attribute with a CID value of 0.85.

- Similarly, the number of unique hosts provides the highest CID (0.95) value for the BOUN UDP flood dataset using the MPMP method. For other feature vectors, we get an average CID of 0.25, which indicates that other feature vectors do not provide good detection by the MPMP method.
- The conditions mentioned in the above two phrases also apply to the Wavelet method. It can be concluded that Wavelet and MPMP methods are not efficient unless the right attribute is selected.
- The AMP method achieved a higher TPR than the other two methods, even by modeling only attack-free traffic in the data set, without the need for attribute selection.

5.6.3. Evaluation of Hybrid AMP Framework

The proposed framework is evaluated separately using two and three traffic classes. Similar to previous sections, the dataset is divided into training and test sets, which include %30 and %70 of the network traffic. The two traffic classes include attack and attack-free traffic, while three traffic classes include two attacks. In two class evaluations, we discuss the detection of TCP and UDP attacks separately. Because there is no publicly available DDoS dataset that contains more than one type of flood attack, we use the BOUN dataset is for three traffic class cases.

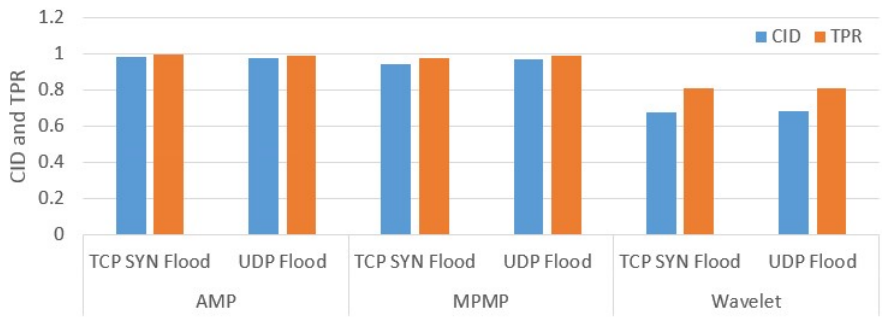

Figure 5.11. Comparison of CID and TPR values for AMP, MPMP and Wavelet based hybrid DDoS detection framework using two traffic classes in BOUN UDP and TCP SYN flood datasets.

Table 5.5. Comparison of AMP, MPMP and Wavelet based hybrid DDoS detection framework for two traffic classes using BOUN UDP and TCP SYN flood datasets.

Method	**Dataset**	**CID**	**TPR(%)**	**FPR(%)**	**AUC**	**Acc**
AMP	TCP SYN Flood	**0.983**	99.658	0.055	1.000	0.999
	UDP Flood	**0.980**	99.556	0.051	0.999	0.999
MPMP	TCP SYN Flood	0.946	97.826	0.000	0.990	0.990
	UDP Flood	0.971	98.936	0.000	0.995	0.990
Wavelet	TCP SYN Flood	0.680	80.926	0.000	0.907	0.953
	UDP Flood	0.688	81.455	0.011	0.912	0.957

The hybrid detection framework is applied separately for BOUN UDP flood and BOUN TCP SYN flood data sets for two traffic class cases. The evaluation metrics can be seen from Table 5.5 and the graphical comparison of CID and TPR values can be seen in Figure 5.11. The TCP and UDP flood datasets are combined to obtain traffic that contains more multiple attack classes.

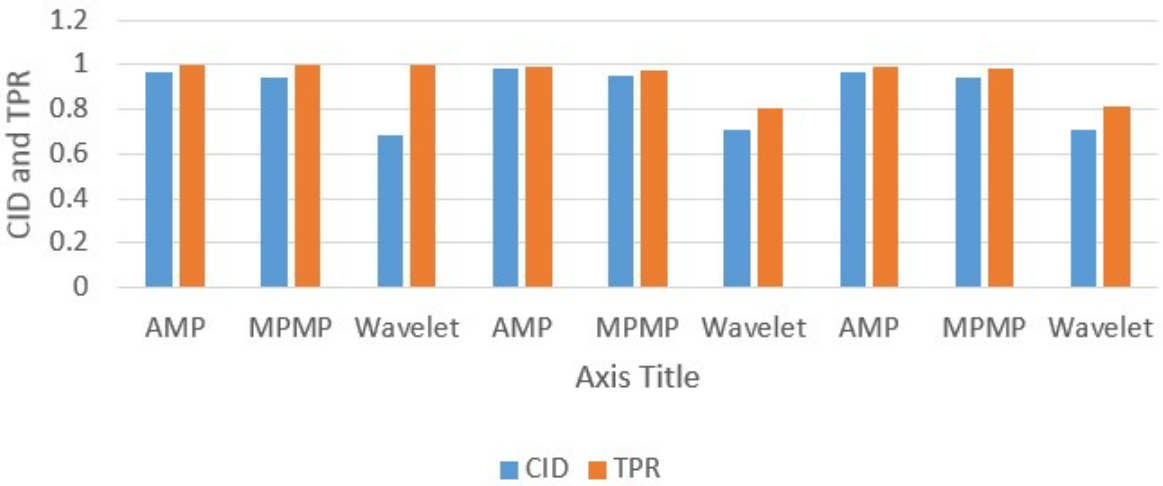

Figure 5.12. Comparison of CID and TPR values for AMP, MPMP and Wavelet based hybrid DDoS detection framework using two three classes in BOUN UDP and TCP SYN flood datasets.

The AMP, MPMP, and Wavelet methods are used with Neural Network decision mechanism; the AMP-based framework provides better results with higher CID and

TPR rates. In Wavelet and MPMP methods, alarms are generated using abnormality vectors produced from all attribute vectors.

As mentioned in previous sections, we cannot achieve discriminative abnormality vectors from all attribute vectors using Wavelet and MPMP approaches. That is the main reason we obtained lower evaluation metrics from the Wavelet approach.

The TCP and UDP flood datasets are combined to obtain traffic that contains more multiple attack classes. When traffic data has more than two attack classes, the AMP-based hybrid framework performs better than MPMP and Wavelet-based methods, as seen in Table 5.6. We give the graphical comparison of CID, and TPR values Figure 5.12 AMP-based framework achieves higher than 0.97 CID value for all attack and attack-free classes. Although the MPMP method has high-performance metrics, it still gives lower CID and TPR for UDP flood and attack-free classes. Also, MPMP gives a higher FPR for TCP SYN flood attacks. The AMP-based method works better even when traffic types are unknown, and only normal traffic is modeled.

Table 5.6. A Comparison of Hybrid Detection Framework based on AMP, MPMP and Wavelet using three traffic classes dataset.

Method	**Class**	**CID**	**TPR (%)**	**FPR (%)**	**AUC**	**Acc (%)**
AMP	TCP Flood	0.97	99.93	0.62	1.00	99.83
	UDP Flood	0.98	99.32	0.01	1.00	99.92
	Normal	0.97	99.22	0.06	1.00	99.87
Wavelet	TCP Flood	0.68	99.99	18.99	0.90	95.52
	UDP Flood	0.71	80.74	0.00	0.91	97.75
	Normal	0.71	81.27	0.00	0.90	97.77
MPMP	TCP Flood	0.94	99.98	2.69	0.99	99.46
	UDP Flood	0.95	97.83	0.00	1.00	99.79
	Normal	0.94	97.87	0.12	1.00	99.68

5.7. Summary and Conclusion

In this chapter, we propose the AMP method for DDoS detection that uses the MP algorithm. We also introduce the characteristic feature vector generated from a combination of multiple one-dimensional traffic attributes. Furthermore, this study provides adaptation to the traffic data to the MP algorithm by creating dictionaries from the training dataset.

Because there is no recent study that uses the MP algorithm to detect DDoS attacks, the proposed methodology is compared with the MPMP and Wavelet methods. We practice these methods using CAIDA and BOUN datasets. The experimental results show that the AMP method performs better with higher CID values comparing with the Wavelet and the MPMP approaches.

Additionally, a hybrid intrusion detection framework is proposed in this study that combines the abnormality indicator values obtained from different dictionaries. The abnormality indicator values are combined with an intelligent decision mechanism that uses ANN. MPMP and Wavelet methods are designed for only anomaly detection. We also include these methods in our Hybrid framework by combining them with the decision module utilizing the abnormality indicator vectors obtained for each traffic attribute vector.

Evaluation results show that the hybrid detection framework using the AMP approach performs better than MPMP and Wavelet-based methods for all traffic classes, including attack-free traffic class.

6. DDoS DETECTION USING STATISTICAL METHODS

This chapter focuses on the statistical properties of traffic features under DDoS attacks and proposes a DDoS detection methodology using statistical modeling. We use probability distribution function fitting and binary hypothesis testing on network traffic features in the first section. We show that various probability distribution functions can be used to represent the probabilistic behavior of network features. In the second section, we propose a novel DDoS detection method using statistical modeling. This work is partly presented in [38]. The proposed statistical DDoS detection methodology uses clustering and distance metrics.

6.1. Statistical Properties of DDoS Attacks

The main contribution of this section is to show researchers of the DDoS detection area that the frequently used DDoS features can be represented with various probability distribution functions and well-known probabilistic models. The hypothesis testing performed using the log-likelihood ratio shows that Gaussian, GEV, and logistic distributions show similar performance. Besides fitting ten different probability distributions to DDoS features, three frequently used probability distribution functions in anomaly detection are discussed in this chapter. These distributions are Weibull, Gaussian, and Logistic distributions. Weibull probability distribution function is handled as is a part of generalized extreme value (GEV) distributions. This family of distributions are generally used for representation and modelling of traffic features, such as TCP trace inter-arrival time and mean to time failure [103–105], DDoS detection [104], and novelty detection [106, 107]. The Gaussian probability distribution is also frequently used in network traffic representation and anomaly detection [108, 109]. In the area of network traffic modeling and attack detection, logistic regression models are also used [110–112]

6.1.1. Probability Distribution Fitting

In order to find the best distribution fit for selected features of attack and normal traffic, three criteria are used. These are the log-likelihood, Bayesian information criterion, Akaike information criterion.

6.1.2. Likelihood Function

In statistical inference, a likelihood function is a function of the parameters of a statistical model, given specifically observed data [113, 114]. The likelihood function is used to find the one probability density function among all possible densities that the model of interest prescribes, is most likely to produce the given sample data. The likelihood function is defined using the data vector $\mathbf{x}$ and the parameter vector θ in $p(\mathbf{x}|\theta)$ as follows:

$$L(\theta|x) = p(x|\theta). \tag{6.1}$$

Thus $L(\theta|y)$ represents the likelihood of the parameter θ given the observed data vector $\mathbf{x}$ and, as such, is a function of θ.

6.1.3. Akaike Information Criterion

Akaike Information Criterion (AIC) is an unbiased estimator of the expected relative Kullback-Leibler distance between the model g and approximate model f [115]. It is calculated as follows:

$$AIC = 2K - 2ln(L), \tag{6.2}$$

where K is the number of estimated parameters, and L is the likelihood function.

6.1.4. Bayesian Information Criterion

The Bayesian information criterion (BIC) is a criterion for model selection among a finite set of probability distribution models. It is based on the likelihood function, and it is closely related to the Akaike information criterion (AIC) [116]. Bayesian Information Criterion (BIC) is defined as:

$$BIC = -2L + Klogn, \tag{6.3}$$

where K is the number of estimated parameters, and n is the sample size. Given equal prior for all competing models, choosing the model with the smallest BIC is equivalent to selecting the maximum posterior probability.

The features are divided into two classes to choose the best probability distribution that represents selected features of the DDoS dataset. These are attack and attack-free classes. The probability distributions are chosen according to the probability fit criterion listed in the previous section. Gamma, Generalized Extreme Value (GEV), Generalized Pareto, Inverse Gaussian, Logistic, Nakagami, Gaussian, Rayleigh, Rician, and t location-scale (TLS) probability distributions are fitted to the selected feature vectors using maximum likelihood estimation for estimating parameters of the distributions.

Two probability distributions to represent attack and normal are chosen for each feature vector. We choose the best three distribution fits for each feature vector attack and normal states. This examination provides us that besides popular distributions such as Gaussian and Weibull, statistical properties DDoS features can be represented with various types of probability models.

Table 6.1 show best fit of probability distribution functions of feature vectors. As we can see from the table, feature vectors are most fitted to the Gaussian, GEV, and Logistic distributions. The main reason for this is heavy-tailed distributions better

represent a low number of values with higher values in the feature vector. In addition to the statistical representation of DDoS features, different types of probability distribution functions can be used instead of one type of pdf for all feature vectors.

Table 6.1. Probability Distributions Fit for Selected Features Normal and Attack Classes of BOUN DDoS Dataset.

Feature Name	***Dataset***	**Fitted Distribution Pairs**		
Unique Sources	Attack	GEV	Weibull	TLS
	Normal	TLS	Logistic	GEV
SYN Packets	Attack	GEV	Weibull	Rician
	Normal	GEV	Weibull	Nakagami
Data per Flow	Attack	Logistic	Logistic	TLS
	Normal	TLS	GEV	Logistic
Average Data	Attack	Rician	Gaussian	Nakagami
	Normal	Logistic	TLS	Gaussian
TCP Flow	Attack	Logistic	GEV	Nakagami
	Normal	GEV	TLS	Logistic
Packets per Flow	Attack	Beta	Gamma	GEV
	Normal	GEV	TLS	Logistic

Instead of displaying sixteen feature vectors in this thesis, in order to give an idea about the estimated probability distributions for feature vectors, we plot the feature vector that has a maximum, medium, and minimum Information Gain in Figure 6.1. The feature that represents the number of Unique sources has the maximum Information Gain and Chi-Squared values and is shown in Fig 6.1(a). It is seen that the attack and normal classes are separable in the feature vector. When Information Gain decreases, the separation ability using the feature vector decreases, as can be seen from Figure 6.1(b) and Figure 6.1(c).

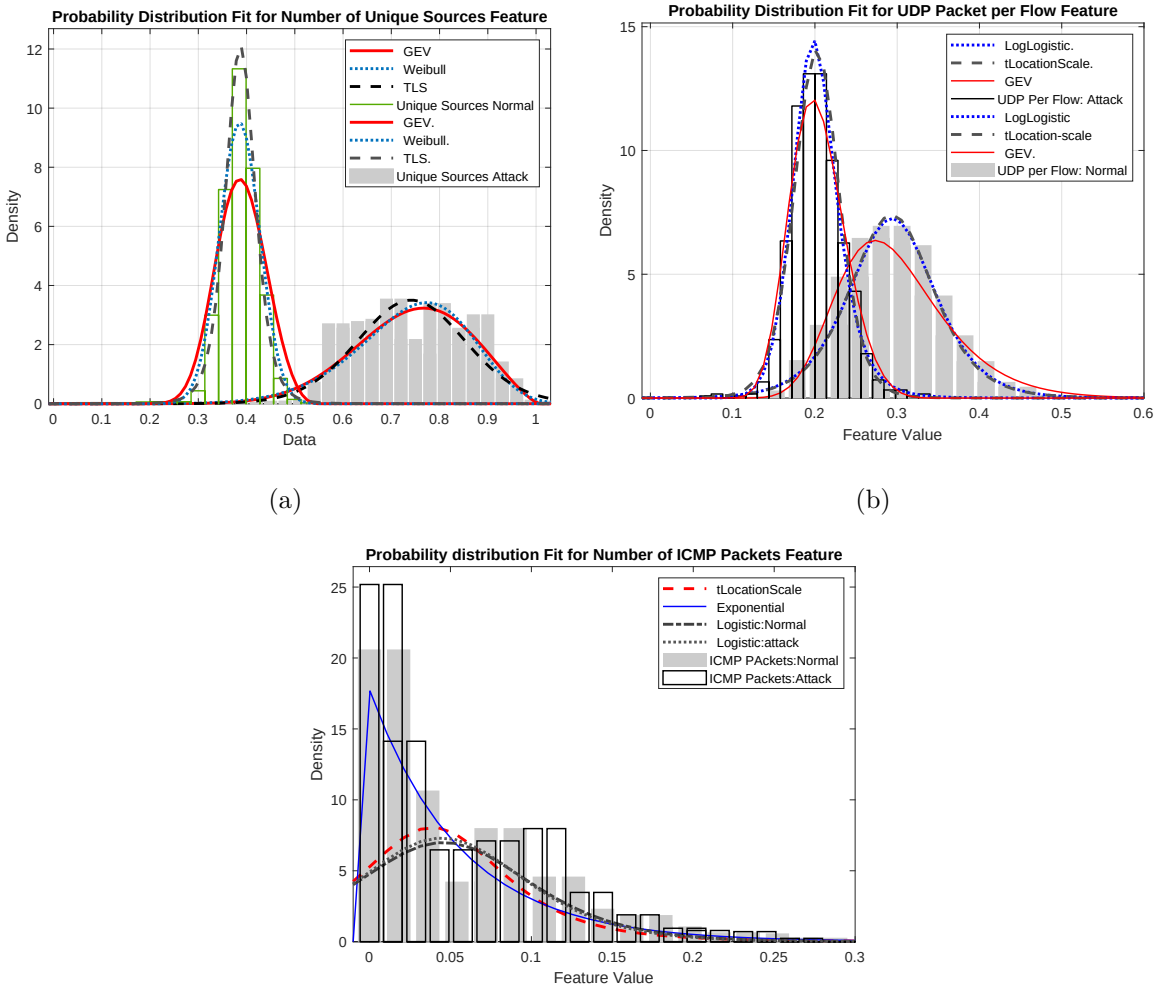

Figure 6.1. Examples of probability distribution fits for attack and normal classes of (a) number of unique sources, (b) number of UDP packets per flow and (c) number of ICMP packets feature vectors in BOUN DDoS dataset.

6.1.5. Probability Distribution Functions

In this section, the most frequently used probability distribution functions in network traffic representation and DDoS detection are discussed.

6.1.5.1. Gaussian (Normal) Distribution. Gaussian, namely normal distribution, is a commonly used probability distribution because of its ability to represent the statistical behavior of many phenomena in natural and social sciences. In addition, it is also

frequently used in network traffic representation, and anomaly detection [108, 109]. The Gaussian distribution function is written as follows:

$$N(\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} exp(\frac{x-\mu}{2\sigma^2}), \tag{6.4}$$

where μ is the mean value, and σ is the standard deviation of the distribution.

6.1.5.2. Generalized Extreme Value Distribution. Extreme-value distributions are usually used in modeling extreme events, from meteorological events to financial fluctuations. In addition, this family of distributions are generally used for representation and modelling of traffic features, such as TCP trace inter-arrival time and mean to time failure [103,104], DDoS detection [104], and novelty detection [106,107]. Extreme Value distribution is categorized into three main types, which are Gumbel (Type I), Frechet (Type II), and Weibull (Type III) distributions [117]. The Gumbel distribution function is written as follows:

$$f(x) = \frac{1}{\sigma} exp(-z - exp(-z)), \quad Gumbel, \tag{6.5}$$

where $z = \frac{x-\mu}{\sigma}$,μ is the location parameter, and σ is the distribution scale satisfying ($\sigma > 0$). Frechlet distribution can be written as follows:

$$f(x) = \frac{\alpha}{\beta}(\frac{\beta}{\alpha})^{\alpha+1} exp(-\frac{\beta}{x}^{\alpha})), \quad Frechet, \tag{6.6}$$

where, α is the shape parameter satisfying ($\alpha > 0$), and β is the scale parameter satisfying ($\beta > 0$). This distribution is with zero where ($x > 0$). Weibull distribution is also a special case of extreme value distribution family defined as:

$$f(x) = \frac{\alpha}{\beta}(\frac{\beta}{\alpha})^{\alpha-1} exp(-\frac{\beta}{x}^{\alpha})), \quad Weibull. \tag{6.7}$$

Weibull distribution is defined for ($x > 0$), and also $\alpha > 0$ and $\beta > 0$. In this work, both Weibull, Frechet, and Gumbel distributions are categorized as the GEV distribution.

6.1.5.3. Logistic Distribution. The logistic distribution is used for various growth models, and it is used for logistic regression models [118]. The logistic regression models are also used in network traffic modeling and attack detection [110–112]. Logistic probability distribution function is defines as:

$$f(x) = \frac{e^z}{\sigma(1+e^z)^2}, \tag{6.8}$$

where $z = \frac{x-\mu}{\sigma}$ and σ,μ are the location and scale parameters.

6.1.6. Binary Hypothesis Testing

In the previous section, we have found the best probability distribution fit for each selected feature. We have also seen that probability distributions differ depending on the state of the network, namely attack and normal. To detect the presence of the DDoS attack, we define the binary the hypothesis-testing problem is as below:

- H_0: There is no DDoS attack and has the probability distribution f_0,
- H_1: There is DDoS attack and has the probability distribution f_1,

where the probability distribution functions f_0 and f_1 are fitted to the features for normal and attack samples, the estimated parameter sets are θ_0 and θ_1, respectively.

Building likelihood ratio tests for all the probability distribution sets shown in Table 6.1 are not straightforward. For this purpose, the probability distributions are estimated from the training data of the feature vectors.

Let us consider feature vector $\mathbf{x}$ of length N consisting of N random variables. All random variables are assumed as independent and identically distributed (i.i.d), the joint pdfs of the two hypotheses are defined as:

$$f_0(\mathbf{x}) = \prod_{i=1}^{N} f_0(x_i), \tag{6.9}$$

and,

$$f_1(\mathbf{x}) = \prod_{i=1}^{N} f_1(x_i). \tag{6.10}$$

The likelihood ratio test of the N random variable in terms of $f_1(\mathbf{x})$ and $f_0(\mathbf{x})$ is defined as:

$$\Lambda(x) = \frac{f_1(\mathbf{x})}{f_0(\mathbf{x})} \gtrless \tau, \tag{6.11}$$

where $\Lambda(\mathbf{x})$ is the likelihood ratio and τ is the decision threshold which helps in making decision either the hypothesis H_0 or H_1. On considering the natural logarithm of both the sides of 6.11, we obtain:

$$log(\Lambda(\mathbf{x})) = log(\frac{f_1(\mathbf{x})}{f_0(\mathbf{x})}) \gtrless \eta. \tag{6.12}$$

Above equation is can be written as:

$$\hat{\Lambda}(\mathbf{x}) = \sum_{i=1}^{N} log(f_1(x_i))) - \sum_{i=1}^{N} log(f_0(x_i)) \gtrless \eta, \tag{6.13}$$

where $\eta = log(\tau)$ and $\hat{\Lambda}(\mathbf{x})$ is the log likelihood ratio. Equation 6.12 represents the log likelihood ratio test of different distributions estimated from test data.

6.1.7. Experimental Results and Discussion

In this section, the detection performed using the likelihood ratio test is evaluated. Binary hypothesis testing is performed on selected feature vectors using Gaussian, GEV, and logistic probability distribution functions for Boun DDoS and CAIDA datasets. Accuracy obtained for these datasets are listed in Tables 6.2, 6.3.

Table 6.2. Accuracy obtained from likelihood ratio test for GEV, Gaussian and Logistic distribuitons for BOUN DDoS dataset.

Feature Name	**GEV**	**Gaussian**	**Logistic**
Unique Sources	0.9973	0.9973	0.9973
SYN Packets	0.8043	0.8043	0.8043
Data per Flow	0.9447	0.9472	0.9472
Average Data	0.8295	0.8716	0.8043
TCP Flow	0.8734	0.8747	0.8665
Packets per Flow	0.8194	0.8651	0.8043

Table 6.3. Accuracy obtained from likelihood ratio test for GEV, Gaussian and Logistic distribuitons for CAIDA datasets.

Feature Name	**GEV**	**Gaussian**	**Logistic**
Unique Sources	0.6706	0.6706	0.6706
SYN Packets	0.6706	0.6706	0.6706
Data per Flow	0.6706	0.6706	0.6706
Average Data	0.6706	0.6706	0.6706
TCP Flow	0.6783	0.6706	0.6760
Packets per Flow	0.6706	0.6706	0.6706

Tables 6.2 and 6.3 show that using likelihood ratio using Gaussian, GEV and Logistic distributions exhibit similar performance in case of accuracy metric. Although features created from CAIDA datasets have high information about attack, we achieve lower detection performance when compared to BOUN DDoS dataset.

Table 6.4. Accuracy obtained from likelihood ratio test for Selected 3 distributions for CAIDA datasets.

Feature Name	**Selected Probability Distribution**		
Unique Sources	0.9973	0.9958	0.9973
SYN Packets	0.8043	0.8206	0.8043
Data per Flow	0.9469	0.9472	0.9472
Average Data	0.8043	0.8043	0.8043
TCP Flow	0.8747	0.8741	0.8665
Packets per Flow	0.8043	0.8194	0.8043

In Table 6.4, the results of binary hypothesis testing using are shown. We used the probability distributions that give the highest values in probability distribution fitting. These are Tlocation-scale, Rician, Nakagami, Beta, and Gamma distributions. As we can see from this result, the features created for DDoS attacks can be represented by distributions other than the GEV, Gaussian, and logistic distributions. However, this is not a general diagnosis. It just opens a door for expanding the set of probability distributions that can be used in network traffic features.

6.2. DDoS Detection Using Statistical Modelling

In this section, a statistical modeling-based DDoS detection method is proposed. To build statistical models of the network traffic, we first generate empirical probability distributions using protocol and flag information in packet headers. We generate an empirical distribution for every time window in the training dataset. We obtain a model that represents network traffic by clustering these distributions. We use cluster centers as models and generate alarms by thresholding the minimum divergences between empirical distributions obtained from the test dataset and these models.

We divide the network traffic into equal-length time windows and compute an empirical distribution within each time window. Initially, we obtain a large number of empirical distributions for each time window in the training dataset. We use clustering

to generate statistical models from these distributions with K-Means and DBSCAN clustering algorithms. Cluster centers are utilized as statistical models, and minimum divergence from these centers are called anomaly indicator values. We then apply a threshold to anomaly indicator values to create alarms. The system block diagram can be seen in Figure 6.2.

In this work, we generate two types of models, namely anomaly model, and misuse model. Anomaly model is generated from attack-free samples of network traffic, while misuse model is generated from attack samples. The thresholding operation differs according to the type of model used in the detection phase. We will explain alarm generation phase in Section 6.2.2.

BOUN DDoS TCP SYN flood attacks dataset [37] is used to evaluate the proposed methodology. The dataset includes legitimate and attacks traffic simultaneously, recorded from Boğaziçi University's core network. The TCP SYN Flood in the dataset includes attack rates of 1000, 1500, 2000, and 2500 packets/second, respectively. The dataset contains spoofed IP addresses as source addresses to achieve many distributed sources for a network-based intrusion detection system.

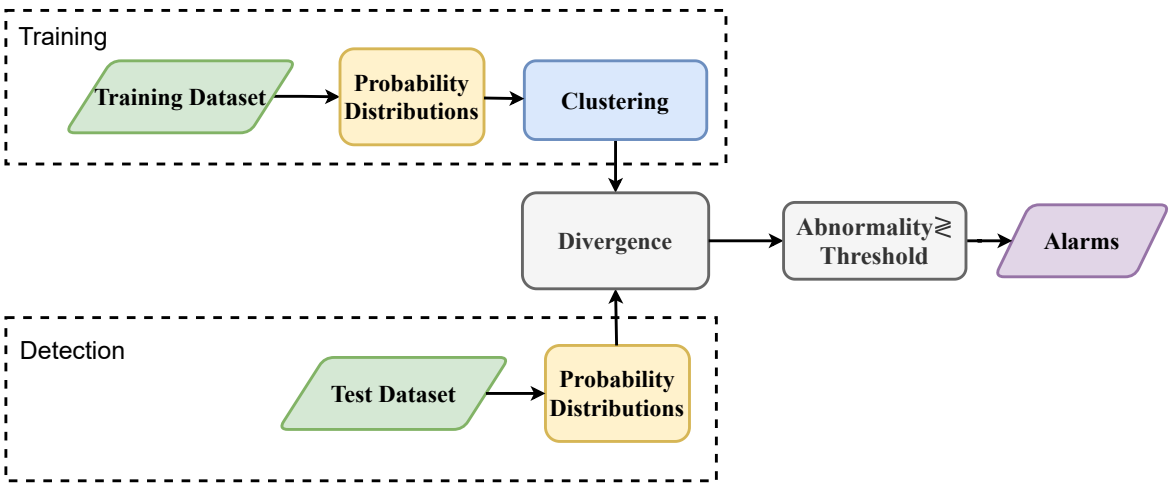

Figure 6.2. Overall block diagram of DDoS detection using statistical modeling methodology.

The dataset is separated into two subsets corresponding to training and test, including 30%, and 70% of the traffic, respectively. The training dataset is used to generate statistical models of the network traffic, while the test dataset is used to evaluate the detection performance of the proposed approach. Alarms are generated depending on the network state used in the modeling phase. If the model is generated from attack-free samples of the training datasets (i.e., anomaly model), the abnormality value tends to increase in the case of attack. If the model is generated from attack samples of the training dataset (i.e., misuse model), the abnormality value tends to decrease in case of attack. Thresholds are applied considering this behavior.

6.2.1. Empirical Model Generation

For building statistical models in the training dataset, empirical distributions are constructed for every time window. Let $\omega = w_1, w_2 \ldots, w_m$ the set of property classes according to TCP protocols and TCP flags, including Transport Control Protocol (TCP), Internet Control Message Protocol (ICMP), User Datagram Protocol (UDP). Given a sequence of network packets $S = x_1, x_2, \ldots, x_n$ in a specific time window t, empirical distribution $P_t(w)$ over ω for the time window t is calculated as follows:

$$P_t(w) = \frac{1}{n} \sum_{i=1}^{n} 1(x_i \in w), \tag{6.14}$$

where $1(x \in w)$ is the indicator function that takes value 1 when the condition $x \in w$ holds and $\frac{1}{n}$ is the normalization constant. After obtaining an empirical distribution for every time window, the distributions are clustered to obtain a statistical model using K-means and DBSCAN clustering algorithms. We will briefly discuss these clustering methods.

K-means algorithm finds locally optimal solutions using the sum of Euclidean Distances between each element and its cluster centers. Suppose the set of empirical

distributions is defined as:

$$\Phi = P_1(w), P_2(w), \ldots, P_m(w), \tag{6.15}$$

where K is the number of non-empty clusters such as $S = C_1, C_2, \ldots, C_K$, K-means algorithm maps space ω to S using the square error criterion, such as:

$$e = \sum_{i=1}^{K} \sum_{j=1}^{K_i} ||P_{ij} - \hat{P}_i||^2, \tag{6.16}$$

where K is the number of clusters, K_i is the number of objects of the cluster i, P_{ij} the j^{th} object of the i^{th} cluster and P_i is the centroid of the i^{th} cluster which the is defined as:

$$F(C_1, C_2, \ldots, C_K) = \sum_{i=1}^{K} \sum_{j=1}^{K_i} ||P_{ij} - \hat{P}_i||. \tag{6.17}$$

Cluster centers are $\hat{P}_i$ are used as statistical network traffic models. Normalization is applied to the cluster centers.

DBSCAN is a non-parametric algorithm that groups elements closer to each other than a given distance [119, 120]. DBSCAN uses two parameters; minimum points M with the distance epsilon ϵ around core objects. The DBSCAN clustering finds the elements classified as core objects with the least M elements within distance epsilon. Initially, all elements in the dataset are considered unassigned to any cluster. DBSCAN then chooses an arbitrary unassigned object from the data set and tries to find at least M elements in the dataset in the epsilon neighborhood of that point. If DBSCAN finds that the object is not a core object, then the object is considered to be noise, and DBSCAN moves onto the next unassigned object. Once every element in the dataset is assigned, the algorithm stops. The elements that are further away from any core objects are marked as outliers. Unlike the K-Means algorithm, DBSCAN does not assign every element in the dataset into a cluster. The core elements of the DSCAN algorithm are chosen to be models for the detection phase.

6.2.2. Alarm Generation

We use four distance measures to calculate divergence between models and empirical distributions obtained from the test dataset. These are K-L divergence, J-S divergence, Greedy algorithm, and MAnhattan distance. We take the minimum distance between empirical distribution generated from the test dataset and cluster centers/core elements of the clusters as anomaly indicator values.

The most well-known and used measure between probability distribution functions is Kullback-Leibler divergence (K-L divergence). Solomon Kullback and Richard Leibler introduced the K-L divergence in [121]. They introduced a metric that is then directed by the divergence between two distributions. K-L divergence, also called relative entropy, measures how one probability distribution function is different from a second. K-L divergence, shown as $D(P||Q)$, measures the inefficiency of assuming that the distribution is Q when the true distribution is P. For discrete probability distribution functions P and Q, the KL-divergence from Q to P is defined as:

$$D_{KL}(P \parallel Q) = -\sum_{i} P(i) log \frac{Q(i)}{P(i)} \tag{6.18}$$

We also used the Jensen and Shannon divergence, which is an asymmetrical version of the K-L divergence. Jensen and Shannon's divergence is defined as follows in [122]

$$D_{JS}(P, Q) = \sum_{i} P(i) log \frac{P(i)}{\frac{1}{2}P(i) + \frac{1}{2}Q(i)}. \tag{6.19}$$

In [123], an alternative metric to K-L divergence is proposed. In their paper, they define a distance between two probability distribution functions $d(\boldsymbol{\phi}, \boldsymbol{\psi})$ on distinct sets , by choosing the joint distribution on $\boldsymbol{\theta}$. They choose $\boldsymbol{\theta}$, which has marginal distributions and also has minimum entropy, maximizing the mutual information between the random variables having the two distributions. They used a greedy algorithm to solve the problem. The details of this algorithm are not discussed in this thesis.

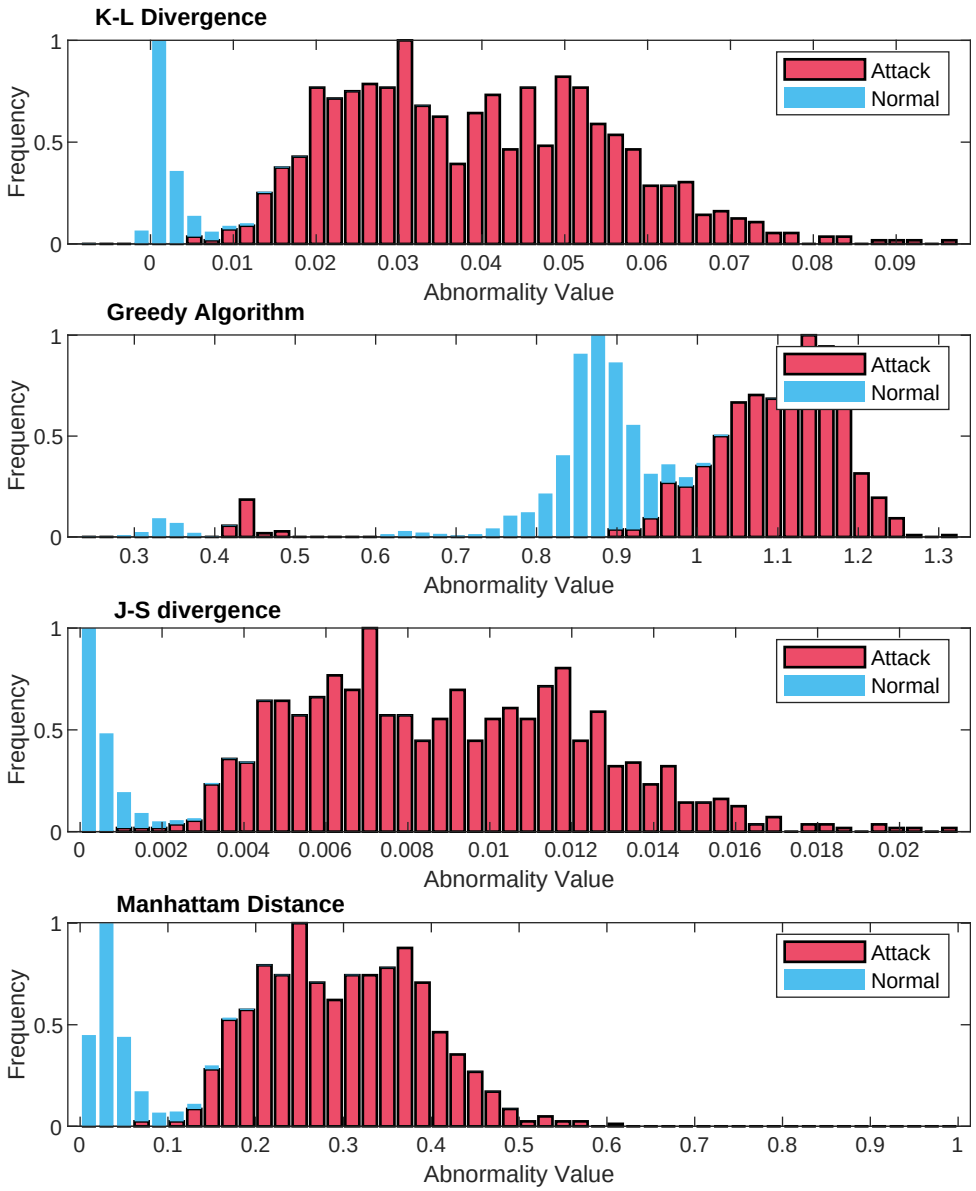

Figure 6.3. Normalised histograms of four divergence measures calculated for attack and attack-free samples. These divergences are calculated using anomaly model generated by K-Means Clustering.

The Manhattan distance is also called the L_1 distance, is used to calculate the distance between two points or multidimensional vectors in space. The Manhattan distance between two length n vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ is calculated using

$$D_M = \sum_{i=1}^{n} |x_i - y_i|, \tag{6.20}$$

where x_i and y_i are the i^{th} element of vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. The threshold value and alarm generation process vary according to the type of model used. If an anomaly model is used, alarms are generated for values lower than the threshold value.

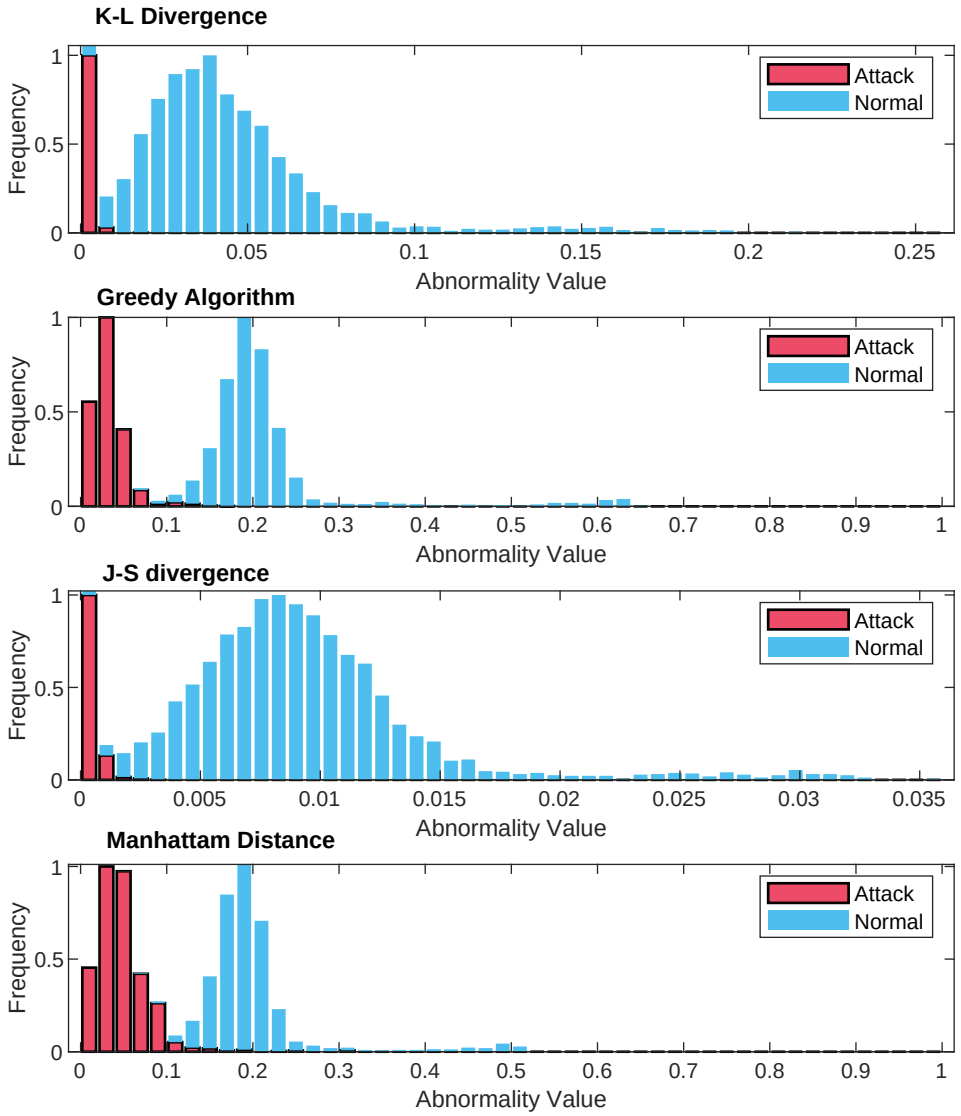

Figure 6.4. Normalised histograms of four divergence measures calculated for attack and attack-free samples. These divergences are calculated using misuse model generated by K-Means Clustering.

For the Misuse model, an alarm is generated for values higher than the threshold value. This is because the proposed approach always considers the minimum divergence between the empirical distributions and the model.

If the model is generated from attack data, empirical distributions close to the model indicate an attack in the time window in which they were calculated. This can be seen in Figure 6.3.

Similarly, the anomaly values obtained using the misuse model will be distant from the empirical distributions created for the attack time windows. This relationship is shown in Figure 6.4.

6.3. Experimental Results and Discussion

In this section, we evaluate the performance of the proposed statistical DDoS detection approach. We mention the performance evaluation metrics briefly. We will also present the performance evaluation metrics we obtain for both misuse and anomaly detection methodology.

Table 6.5. Evaluation of misuse-based DDoS attack detection.

Cluster	**Distance**	**CID**	**TPR(%)**	**FPR(%)**	**AUC**	**Acc**
DBSCAN	K-L Divergence	0.941	98.142	0.114	0.997	0.985
	Greedy Alg.	0.825	95.330	1.115	0.966	0.982
	Jensen-Shannon	0.944	98.248	0.114	0.999	0.986
	Manhattan	0.812	97.903	2.506	0.991	0.978
K-Means	K-L Divergence	0.939	98.036	0.114	0.997	0.984
	Greedy Alg.	0.788	92.027	0.796	0.971	0.961
	Jensen-Shannon	0.946	98.355	0.114	0.999	0.986
	Manhattan	0.933	99.204	0.570	0.999	0.993

The assessment of the proposed approach with five different performance metrics are shown in Tables 6.6 and 6.5. We obtain higher TPR with higher FPR values than the ones shown in the results. But we need to choose a single metric to compare different approaches. For this purpose, we choose the operation point from ROC analysis to give the maximum CID value.

Table 6.6. Evaluation of anomaly-based DDoS attack detection.

Cluster	**Distance**	**CID**	**TPR(%)**	**FPR(%)**	**AUC**	**Acc**
DBSCAN	K-L Divergence	0.892	98.519	1.008	0.998	0.989
	Greedy Alg.	0.807	94.191	1.088	0.974	0.982
	Jensen-Shannon	0.891	98.975	1.221	0.998	0.988
	Manhattan	0.900	94.191	0.716	0.996	0.991
K-Means	K-L Divergence	0.943	98.646	0.227	0.997	0.988
	Greedy Alg.	0.79	92.369	0.743	0.973	0.963
	Jensen-Shannon	0.946	97.771	0	0.999	0.982
	Manhattan	0.941	99.316	0.478	0.999	0.995

In this work, we employ anomaly detection and misuse detection. For this purpose, we generate anomaly and misuse models. Table 6.5 includes the results obtained using misuse model, while Table 6.6 shows the results obtained using anomaly model.

Comparing detection metrics listed in Tables 6.6 and 6.5, it can be observed that misuse based modeling using DBSCAN as clustering algorithm gives poorer detection when compared other results. Another conclusion that should be emphasized is that the greedy algorithm gives poorer results than K-L and Jensen Shannon divergence. This result is not surprising since the greedy algorithm sets an upper limit to the distance between the two statistical distributions. Additionally, as expected, Jensen and Shannon and K-L divergence provide similar results.

6.4. Summary and Conclusion

This chapter aims to give the researchers of the DDoS detection area insight into the probabilistic behavior of the traffic features of DDoS attacks. After finding best-fitted distributions, it has been seen that DDoS features are mostly best fitted to GEV, Gaussian, and Logistic distributions. These distributions are frequently used in network traffic modeling and anomaly detection.

All the calculations are performed using a BOUN TCP SYN flood. As future work, this method can be performed using different datasets and different types of DDoS attacks.

In addition, this chapter also proposes a methodology of attack detection that uses statistical modeling. To employ statistical modeling, we divide network packets into equal-length time windows and create empirical distributions for every time window. Empirical distributions are clustered using K-means and DBSCAN algorithms to obtain a model for detection. Cluster centers obtained from K-means and core elements obtained from DBSCAN algorithms are used as models for misuse and anomaly-type attack detection. The minimum divergence between statistical models and empirical distributions obtained from network traffic is used to generate alarms. We use K-L divergence, Greedy Algorithm, Jensen-Shannon divergence, and Manhattan distance to calculate divergence between models and empirical distributions.

Experimental results show that the proposed DDoS detection using a statistical modeling approach achieves high detection rates and low false-positive rates.

7. CONCLUSION and FUTURE WORK

DDoS attacks have long been a severe threat to the computer services of many large and small businesses, including government agencies, especially financial organizations. Although there have been many studies in the literature on detecting DDoS attacks, there is still a need for practical and intelligent methods in this regard. This thesis presents practices on understanding DDoS attack mechanisms, discovering attributes used for attack detection, and novel DDoS attack detection approaches.

To find the most suitable method for DDoS detection, first of all, in this thesis, features specific to these attacks should be discovered and produced. Packet and flow-based features proposed in the literature were examined, generated, and tested through the experiments carried out on the DETER testbed. We have gained the necessary experience to create a DDoS data set with the simulations created on the DETER testbed. In addition, we examined the online datasets and the approaches we recommend, including the frequently used CAIDA datasets. We wanted to evaluate the proposed approaches with a more realistic dataset. We generated the BOUN DDoS dataset for this purpose. The difference of the BOUN DDoS dataset from the DETER testbed is that it contains the actual network traffic. If this dataset is different from the CAIDA dataset, it contains both attack and normal traffic simultaneously.

In this thesis, 16 different attributes have been created for the detection of DDoS attacks. These attributes can be divided into package-based and flow-based. In addition, in this thesis, we examined the statistical behaviors of these features and the scores when the feature selection is applied.

To detect DDoS attacks, we first implement an AR-based change detection approach. We handle the feature vectors as time series and use error variances obtained from AR(1) estimation of two adjacent windows in features. We calculate an abnormality value from those error variances for each feature vector. We combine these

abnormality indicators using the correlation coefficient matrix between these features to obtain a traffic heath value for alarm generation. We evaluate the approach with the dataset obtained from DETER testbed for three types of flood attacks. Results show that although we can successfully determine the starting and end of the attacks, we need to determine ongoing attacks and generate alarms. With this methodology, we can detect all ICMP flood attacks, 88%of the UDP flood attacks, and all TCP flood attacks.

We also present novel methodologies and implemented methods from the literature using sparse signal representation methods for detecting DDoS attacks. We proposed an adaptive MP-based DDoS detection method in anomaly and misuse detection manner. We also implemented Wavelet and MPMP based DDoS detection approaches using two different datasets. Sparse representation methods express signals with a linear combination of basis vectors. We use matching pursuit and wavelet approaches in this thesis. We first combine normalized traffic attributes to introduce a characteristic feature vector representing multiple attributes in a particular time window. Unlike other sparse representation-based anomaly detection approaches, we can simultaneously use multiple attributes in the AMP-based DDoS detection approach.

We use CAIDA and BOUN DDoS datasets to test and compare the AMP, Wavelet, and MPMP based methods. We obtain similar performance from the proposed AMP method with the MPMP and Wavelet methods for the CAIDA dataset. However, we achieved much better detection performance with the AMP approach for TCP and UDP flood attacks. We reached a TPR of more than 99% with an FPR lower than 0.05% for the anomaly, and misuse detection AMP approaches for TCP SYN and UDP flood attacks.

After completing the evaluation of the AMP approach, we decided to combine anomaly and misuse detection methods to obtain a hybrid detection framework based on MP. The output of the misuse and anomaly-based AMP method is connected using ANN to form a hybrid detection framework. We also connected these methods with

an ANN to compare the AMP hybrid method with MPMP and Wavelet methods. We evaluate the hybrid detection framework for both one attack type and two attack types. Comparing the hybrid AMP with MPMP and Wavelet methods, we obtain much higher TPR with a lower FPR for two class and three class cases. The performance of the proposed approach has 99% TPR with FPR lower than 0.06% for two-class and three-class detection.

We also examine the statistical properties of traffic features under DDoS attacks. For this purpose, we try to fit probability distribution functions to traffic features. DDoS features are essentially best fitted to the GEV, Gaussian, and Logistic probability distribution functions. These distributions are also frequently used in network traffic modeling and anomaly detection in the literature. We use features obtained from BOUN DDoS dataset.

In addition, this thesis includes a methodology of attack detection that uses statistical modeling. We have created the empirical distributions from test and training datasets. Empirical distributions are clustered using K-means and DBSCAN algorithms to obtain a smaller statistical model space. Cluster centers obtained from K-means and DBSCAN algorithms are used as models for the testing phase. In the alarm generation phase, we generate empirical distributions from the test dataset and compared them with the generated model using KL, Jensen-Shannon divergence, Greedy Algorithm, and Manhattan distances. We perform both anomaly and misuse detection approaches. The results show that detection using a statistical modeling approach achieves high detection rates and low false-positive rates.

I plan to examine the effects of DDoS attacks in IoT and SDN environments in my future studies. In addition, as future work, my plans include finding the most effective detection method for these attacks in these environments.

REFERENCES

1. David, J. and C. Thomas, "Efficient DDoS Flood Attack Detection Using Dynamic Thresholding on Flow-Based Network Traffic", *Computers & Security*, Vol. 82, pp. 284–295, 2019.

2. *Worldwide Infrastructure Security Report*, 2020, `https://www.netscout.com/report/`, accessed in May 2021.

3. *State of the Internet, Security: 2020 — A Year in Review*, 2020, `https://www.akamai.com/content/dam/site/en/documents/state-of-the-internet/soti-security-a-year-in-review-report-2020.pdf`, accessed in May 2021.

4. Warburton, D. and E. Ojeda, *DDoS Attack Trends for 2020*, 2021, `https://www.f5.com/labs/articles/threat-intelligence/ddos-attack-trends-for-2020`, accessed in May 2021.

5. Deshmukh, R. V. and K. K. Devadkar, "Understanding DDoS Attack & Its Effect in Cloud Environment", *Procedia Computer Science*, Vol. 49, pp. 202–210, 2015.

6. Depren, O., M. Topallar, E. Anarim and M. K. Ciliz, "An Intelligent Intrusion Detection System (IDS) for Anomaly and Misuse Detection in Computer Networks", *Expert Systems with Applications*, Vol. 29, No. 4, pp. 713–722, 2005.

7. Zarpelao, B. B., R. S. Miani, C. T. Kawakani and S. C. de Alvarenga, "A Survey of Intrusion Detection in Internet of Things", *Journal of Network and Computer Applications*, Vol. 84, pp. 25–37, 2017.

8. Chertov, R., S. Fahmy, P. Kumar, D. Bettis, A. Khreishah and N. B. Shroff, *Topology generation, instrumentation, and experimental control tools for emulation testbeds*, 2006.

9. Chan, P. K. and M. V. Mahoney, "Modeling Multiple Time Series for Anomaly Detection", *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pp. 8–pp, IEEE, 2005.

10. Salvador, S., P. Chan and J. Brodie, "Learning States and Rules for Time Series Anomaly Detection.", *FLAIRS Conference*, pp. 306–311, 2004.

11. Vlachos, M., M. Hadjieleftheriou, D. Gunopulos and E. Keogh, "Indexing Multi-Dimensional Time-Series with Support for Multiple Distance Measures", *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 216–225, ACM, 2003.

12. Saganowski, Ł., M. Choraś, R. Renk and W. Hołubowicz, "Signal-based Approach to Anomaly Detection in IDS Systems", *International Journal of Intelligent Engineering and Systems*, Vol. 1, No. 4, pp. 18–24, 2008.

13. Saganowski, Ł., M. Choraś, R. Renk and W. Hołubowicz, "A Novel Signal-Based Approach to Anomaly Detection in IDS Systems", *International Conference on Adaptive and Natural Computing Algorithms*, pp. 527–536, Springer, 2009.

14. Andrysiak, T. and Ł. Saganowski, "Anomaly Detection System Based on Sparse Signal Representation", *Image Processing & Communications*, Vol. 16, No. 3-4, pp. 37–44, 2011.

15. Choraś, M., Ł. Saganowski, R. Renk and W. Hołubowicz, "Statistical and Signal-Based Network Traffic Recognition for Anomaly Detection", *Expert Systems*, Vol. 29, No. 3, pp. 232–245, 2012.

16. Renk, R., L. Saganowski, W. Holubowicz and M. Choras, "Intrusion Detection System Based on Matching Pursuit", *2008 First International Conference on Intelligent Networks and Intelligent Systems*, pp. 213–216, 2008.

17. Eriksson, B., P. Barford, R. Bowden, N. Duffield, J. Sommers and M. Roughan,

"BasisDetect: A Model Based Network Event Detection Framework", *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, pp. 451–464, 2010.

18. Erhan, D., E. Anarım and G. K. Kurt, "DDoS Attack Detection Using Matching pursuit algorithm", *Signal Processing and Communication Application Conference (SIU), 2016 24th*, pp. 1081–1084, 2016.

19. Xia, H., B. Fang, M. Roughan, K. Cho and P. Tune, "A Basis Evolution Framework for Network Traffic Anomaly Detection", *Computer Networks*, Vol. 135, pp. 15–31, 2018.

20. Thottan, M. and C. J. C. Ji, "Anomaly Detection in IP Networks", *IEEE Transactions on Signal Processing*, Vol. 51, No. 8, pp. 2191–2204, 2003.

21. Thottan, M. and C. J. C. Ji, "Adaptive Thresholding for Proactive Network Problem Detection", *Proceedings of the IEEE Third International Workshop on Systems Management*, pp. 108–116, 1998.

22. Thottan, M. and C. Ji, "Fault Prediction at the Network Layer Using Intelligent Agents", *Integrated Network Management VI Distributed Management for the Networked Millennium Proceedings of the Sixth IFIPIEEE International Symposium on Integrated Network Management Cat No99EX302*, pp. 745–759, 1999.

23. Andrysiak, T., Ł. Saganowski and M. Choraś, "DDoS Attacks Detection by Means of Greedy Algorithms", *Image Processing and Communications Challenges 4*, pp. 303–310, Springer, 2013.

24. Bhuyan, M. H., D. Bhattacharyya and J. K. Kalita, "An Empirical Evaluation of Information Metrics for Low Rate and High Rate DDoS Attack Detection", *Pattern Recognition Letters*, Vol. 51, pp. 1–7, 2015.

25. Mousavi, S. M. and M. St-Hilaire, "Early Detection of DDoS Attacks Against

SDN Controllers", *2015 International Conference on Computing, Networking and Communications (ICNC)*, pp. 77–81, 2015.

26. Mehmood, A., M. Mukherjee, S. H. Ahmed, H. Song and K. M. Malik, "NBC-MAIDS: Naïve Bayesian Classification Technique in Multi-Agent System-Enriched IDS for Securing IoT Against DDoS Attacks", *The Journal of Supercomputing*, pp. 1–15, 2018.

27. Nooribakhsh, M. and M. Mollamotalebi, "A Review on Statistical Approaches for Anomaly Detection in DDoS Attacks", *Information Security Journal: A Global Perspective*, Vol. 29, No. 3, pp. 118–133, 2020.

28. Khalaf, B. A., S. A. Mostafa, A. Mustapha, M. A. Mohammed and W. M. Abdullah, "Comprehensive Review of Artificial Intelligence and Statistical Approaches in Distributed Denial of Service Attack and Defense Methods", *IEEE Access*, Vol. 7, pp. 51691–51713, 2019.

29. Mishra, S., R. Sagban, A. Yakoob and N. Gandhi, "Swarm Intelligence in Anomaly Detection Systems: an Overview", *International Journal of Computers and Applications*, pp. 1–10, 2018.

30. Dehkordi, A. B., M. Soltanaghaei and F. Z. Boroujeni, "The DDoS Attacks Detection Through Machine Learning and Statistical Methods in SDN", *The Journal of Supercomputing*, pp. 1–33, 2020.

31. Devi, B. K. and T. Subbulakshmi, "Cloud-Based DDoS Attack Detection and Defence System Using Statistical Approach", *International Journal of Information and Computer Security*, Vol. 11, No. 4-5, pp. 447–475, 2019.

32. Hoque, N., D. K. Bhattacharyya and J. K. Kalita, "FFSc: A Novel Measure for Low-Rate and High-Rate DDoS Attack Detection Using Multivariate Data Analysis", *Security and Communication Networks*, Vol. 9, No. 13, pp. 2032–2041,

2016.

33. Machaka, P. and A. Bagula, "Statistical Properties and Modelling of DDoS Attacks", *Context-Aware Systems and Applications, and Nature of Computation and Communication*, pp. 44–54, Springer, 2020.

34. Çakmakçı, S. D., T. Kemmerich, T. Ahmed and N. Baykal, "Online DDoS Attack Detection Using Mahalanobis Distance and Kernel-based Learning Algorithm", *Journal of Network and Computer Applications*, Vol. 168, p. 102756, 2020.

35. Ahmed, M. E., S. Ullah and H. Kim, "Statistical Application Fingerprinting for DDoS Attack Mitigation", *IEEE Transactions on Information Forensics and Security*, Vol. 14, No. 6, pp. 1471–1484, 2018.

36. Ateş, Ç., S. Özdel and E. Anarım, "Network Anomaly Detection Using Header Information with Greedy Algorithm", *2019 27th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, 2019.

37. Erhan, D. and E. Anarım, "Boğaziçi University Distributed Denial of Service Dataset", *Data in Brief*, Vol. 32, p. 106187, 2020.

38. Erhan, D. and E. Anarım, "Istatistiksel Yöntemler ile DDoS Saldırı Tespiti DDoS Detection Using Statistical Methods", *2020 28th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, 2020.

39. Erhan, D., E. Anarım, G. K. Kurt and R. Koşar, "Effect of DDoS Attacks on Traffic Features", *2013 21st Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, 2013.

40. Erhan, D. and E. Anarım, "DDoS Detection Using Statistical Modelling", *International Conference on Digital Image and Signal Processing (DISP 19)*, 2019.

41. Erhan, D. and E. Anarım, "Statistical Properties of DDoS Attacks", *2019 6th*

International Conference on Control, Decision and Information Technologies (CoDIT), pp. 1238–1242, IEEE, 2019.

42. Erhan, D. and E. Anarim, "Hybrid DDoS Detection Framework Using Matching Pursuit Algorithm", *IEEE Access*, Vol. 8, pp. 118912–118923, 2020.

43. *The CAIDA UCSD Anonymized Internet Traces 2008*, 2008, `http://www.caida.org/data/passive/passive_2008_dataset.xml`, accessed in 2010.

44. *The CAIDA UCSD "DDoS Attack 2007" Dataset*, 2007, `http://www.caida.org/data/passive/ddos_20070804_dataset.xml`, accessed in 2010.

45. Benzel, T., R. Braden, D. Kim, A. D. Joseph, B. C. Neuman, R. Ostrenga, S. Schwab and K. Sklower, "Design, Deployment, and Use of the DETER Testbed.", *DETER*, 2007.

46. *Denial-of-Service Developments*, Tech. rep., CERT Coordination Center, 2000.

47. Tariq, U., M. Hong and K.-s. Lhee, "A Comprehensive Categorization of DDoS Attack and DDoS Defense Techniques", *International Conference on Advanced Data Mining and Applications*, pp. 1025–1036, Springer, 2006.

48. Mirkovic, J. and P. Reiher, "A Taxonomy of DDoS Attack and DDoS Defense Mechanisms", *SIGCOMM Comput. Commun. Rev.*, Vol. 34, pp. 39–53, April 2004.

49. Specht, S., "Distributed Denial of Service: Taxonomies of Attacks, Tools, and Countermeasures.", *Citeseer. Proceedings of the 17th International Conference*, 2004.

50. Douligeris, C. and A. Mitrokotsa, "DDoS Attacks and Defense Mechanisms: Clas-

sification and State-of-the-Art", *Computer Networks*, Vol. 44, No. 5, pp. 643–666, 2004.

51. Champagne, D. and R. Lee, "Scope of DDoS Countermeasures: Taxonomy of Proposed Solutions and Design Goals for Real-World Deployment", *on Systems and Information Security SSI*, 2006.

52. Mahajan, R., S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson and S. Shenker, "Controlling high bandwidth aggregates in the network", *ACM Computer Communication Review*, Vol. 32, pp. 62–73, 2002.

53. Asosheh, A. and N. Ramezani, "A Comprehensive Taxonomy of DDoS Attacks and Defense Mechanism Applying in a Smart Classification", *WSEAS Transactions on Computers*, Vol. 7, No. 4, pp. 281–290, 2008.

54. Gu, G., P. Fogla, D. Dagon, W. Lee and B. Skorić, "Measuring Intrusion Detection Capability: an Information-Theoretic Approach", *Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security*, pp. 90–101, ACM, 2006.

55. Cover, T. M. and J. A. Thomas, *Elements of information theory*, John Wiley & Sons, 2012.

56. Ye, T., D. Veitch, G. Iannaccone and S. Bhattacharya, *First International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities*, pp. 262–271, IEEE, 2005.

57. Barford, P., J. Kline, D. Plonka and A. Ron, "A signal analysis of network traffic anomalies", *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pp. 71–82, 2002.

58. Lee, Y. and B. Mukherjee, "Traffic Engineering in Next-Generation Optical Networks", *Communications Surveys Tutorials, IEEE*, Vol. 6, No. 3, pp. 16 –33,

quarter 2004.

59. Cao, J., W. Cleveland, Y. Gao, K. Jeffay, F. Smith and M. Weigle, "Stochastic models for generating synthetic HTTP source traffic", *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, Vol. 3, pp. 1546 –1557 vol.3, March 2004.

60. Mirkovic, J., *Benchmarks for Evaluation of Distributed Denial of Service (DDOS)*, Tech. rep., Delaware University Newmark, 2008.

61. Sommers, J., H. Kim and P. Barford, *Harpoon: a Flow-Level Traffic Generator for Router and Network Tests*, p. 392, ACM, 2004.

62. Ateş, Ç., S. Özdel and E. Anarım, "Graph–Based Anomaly Detection Using Fuzzy Clustering", *International Conference on Intelligent and Fuzzy Systems*, pp. 338–345, Springer, 2019.

63. Ateş, Ç., S. Özdel and E. Anarım, "Clustering Based DDoS Attack Detection Using the Relationship Between Packet Headers", *2019 Innovations in Intelligent Systems and Applications Conference (ASYU)*, pp. 1–6, 2019.

64. Ateş, Ç., S. Özdel and E. Anarım, "A New Network Anomaly Detection Method Based on Header Information Using Greedy Algorithm", *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*, pp. 657–662, 2019.

65. Ateş, Ç., S. Özdel, M. Yıldırım and E. Anarım, "DDoS Attack Detection Using Greedy Algorithm and Frequency Modulation", *2019 27th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, 2019.

66. Fontugne, R., P. Borgnat, P. Abry and K. Fukuda, "Mawilab: combining diverse anomaly detectors for automated anomaly labeling and performance benchmarking", *Proceedings of the 6th International COnference*, pp. 1–12, 2010.

67. "DDoS Evaluation Dataset (CICDDoS2019)", `https://www.unb.ca/cic/datasets/ddos-2019.html`.

68. "Wireshark Network Protocol Analyzer.", `https://www.wireshark.org`.

69. Adi, E., Z. Baig and P. Hingston, "Stealthy Denial of Service (DoS) Attack Modelling and Detection for HTTP/2 Services", *Journal of Network and Computer Applications*, Vol. 91, pp. 1–13, 2017.

70. Santanna, J. J., R. van Rijswijk-Deij, R. Hofstede, A. Sperotto, M. Wierbosch, L. Z. Granville and A. Pras, "Booters an Analysis of DDoS as a Service Attacks", *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pp. 243–251, 2015.

71. Oo, T. T. and T. Phyu, "A Statistical Approach to Classify and Identify DDoS Attacks Using UCLA Dataset", *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, Vol. 2, No. 5, pp. 1766–1770, 2013.

72. Wang, B., Z. Li, D. Li, F. Liu and H. Chen, "Modeling Connections Behavior for Web-Based Bots Detection", *2010 2nd International Conference on E-business and Information System Security*, pp. 1–4, 2010.

73. Nezhad, S. M. T., M. Nazari and E. A. Gharavol, "A Novel DoS and DDoS Attacks Detection Algorithm Using ARIMA Time Series Model and Chaotic System in Computer Networks", *IEEE Communications Letters*, Vol. 20, No. 4, pp. 700–703, 2016.

74. Lim, S., J. Ha, H. Kim, Y. Kim and S. Yang, "A SDN-oriented DDoS Blocking Scheme for Botnet-Based Attacks", *2014 Sixth International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 63–68, 2014.

75. Novaković, J., "Toward Optimal Feature Selection Using Ranking Methods and

Classification Algorithms", *Yugoslav Journal of Operations Research*, Vol. 21, No. 1, 2016.

76. Liu, H. and R. Setiono, "Chi2: Feature Selection and Discretization of Numeric Attributes", *Proceedings of 7th IEEE International Conference on Tools With Artificial Intelligence*, pp. 388–391, 1995.

77. Kayacik, H. G., A. N. Zincir-Heywood and M. I. Heywood, "Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets", *Proceedings of the Third Annual Conference on Privacy, Security and Trust*, Vol. 94, pp. 1723–1722, Citeseer, 2005.

78. Yang, Y. and J. O. Pedersen, "A Comparative Study on Feature Selection in Text Categorization", *Icml*, Vol. 97, pp. 412–420, 1997.

79. Van Trees, H. L., *Detection, Estimation, and Modulation Theory*, John Wiley & Sons, 2004.

80. Wang, G. W. G., Z. W. Z. Wang and X. L. X. Luo, "Research of Anomaly Detection Based on Time Series", *2009 WRI World Congress on Software Engineering*, Vol. 1, pp. 444–448, 2009.

81. Cohent-Tannoudji, G., B. Diu and F. Laloë, *Quantum Mechanics*, 2000.

82. Cohen, L., "Time-Frequency Distributions a Review", *Proceedings of the IEEE*, Vol. 77, No. 7, pp. 941–981, 1989.

83. Mallat, S. and Z. Zhang, *Matching Pursuit With Time-Frequency Dictionaries*, Tech. rep., Courant Institute of Mathematical Sciences New York United States, 1993.

84. Mazhar, R., P. D. Gader and J. N. Wilson, "Matching-Pursuits Dissimilarity Measure for Shape-Based Comparison and Classification of High-Dimensional Data",

IEEE Transactions on Fuzzy Systems, Vol. 17, No. 5, pp. 1175–1188, 2009.

85. M. R. McClure, L. C., "Matching Pursuits with a Wave-Based Dictionary", *IEEE Trans. Signal Processing.*, Vol. 45, 1997.

86. R. Neff, A. Z., "Very Low Bit-Rate Video Coding Based on Matching Pursuits", *IEEE Trans. Circuits and Systems for Video Technology.*, Vol. 7, 1997.

87. P. K. Bharadwaj, L. C., P. R. Runkle, "Target Identification with Wave-Based Matched Pursuits and Hidden Markov Models", *IEEE Trans. Antennas and Propogration.*, Vol. 47, 1999.

88. Pati, Y. C., R. Rezaiifar and P. S. Krishnaprasad, "Orthogonal Matching Pursuit: Recursive Function Approximation With Applications to Wavelet Decomposition", *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, pp. 40–44, 1993.

89. Cotter, S. and B. Rao, "Application of Tree-Based Searches to Matching Pursuit", *EEE International Conference on Acoustics, Speech, and Signal Processing*, Vol. 6, 2001.

90. Karabulut, G., "Integrating Flexible Tree Searches to Orthogonal Matching Pursuit Algorithm", *IEEE Proceedings - Vision, Image, and Signal Processing*, 2006.

91. Aharon, M., M. Elad and A. Bruckstein, "K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation", *IEEE Transactions on Signal Processing*, Vol. 54, No. 11, pp. 4311–4322, 2006.

92. Rubinstein, R., T. Peleg and M. Elad, "Analysis K-SVD: A Dictionary-Learning Algorithm for the Analysis Sparse Model", *IEEE Transactions on Signal Processing*, Vol. 61, No. 3, pp. 661–677, 2012.

93. Singh, P. K., S. K. Jha, S. K. Nandi and S. Nandi, "ML-Based Approach to Detect

DDoS Attack in V2I Communication Under SDN Architecture", *TENCON 2018-2018 IEEE Region 10 Conference*, pp. 0144–0149, 2018.

94. Lu, W. and A. A. Ghorbani, "Network Anomaly Detection Based on Wavelet Analysis", *EURASIP Journal on Advances in Signal Processing*, Vol. 2009, p. 4, 2009.

95. Tang, T. A., L. Mhamdi, D. McLernon, S. A. R. Zaidi and M. Ghogho, "Deep Learning Approach for Network Intrusion Detection in Software Defined Networking", *2016 International Conference on Wireless Networks and Mobile Communications (WINCOM)*, pp. 258–263, 2016.

96. Boyar, O., M. Özen and B. Metin, "Detection of Denial-of-Service Attacks with SNMP/RMON", *2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES)*, pp. 000437–000440, 2018.

97. Wang, R., Z. Jia and L. Ju, "An Entropy-Based Distributed DDoS Detection Mechanism in Software-Defined Networking", *2015 IEEE Trustcom/BigDataSE/ISPA*, Vol. 1, pp. 310–317, 2015.

98. Jankowski, D. and M. Amanowicz, "Intrusion Detection in Software Defined Networks With Self-Organized Maps", *Journal of Telecommunications and Information Technology*, 2015.

99. Bawany, N. Z., J. A. Shamsi and K. Salah, "DDoS Attack Detection and Mitigation Using SDN: Methods, Practices, and Solutions", *Arabian Journal for Science and Engineering*, Vol. 42, No. 2, pp. 425–441, 2017.

100. Osanaiye, O., K.-K. R. Choo and M. Dlodlo, "Distributed Denial of Service (DDoS) Resilience in Cloud: Review and Conceptual Cloud DDoS Mitigation Framework", *Journal of Network and Computer Applications*, Vol. 67, pp. 147–165, 2016.

101. Grossmann, A. and J. Morlet, "Decomposition of Functions Into Wavelets of Constant Shape, and Related Transforms", *Mathematics+ Physics: Lectures on Recent Results (Volume 1)*, pp. 135–165, World Scientific, 1985.

102. Eksioglu, E. M. and O. Bayir, "K-svd Meets Transform Learning: Transform K-SVD", *IEEE Signal Processing Letters*, Vol. 21, No. 3, pp. 347–351, 2014.

103. Çiflikli, C., A. Gezer, A. T. Özşahin and Ö. Özkasap, "BitTorrent Packet Traffic Features over IPv6 and IPv4", *Simulation Modelling Practice and Theory*, Vol. 18, No. 9, pp. 1214–1224, 2010.

104. Osanaiye, O., K.-K. R. Choo and M. Dlodlo, "Change-Point Cloud DDoS Detection Using Packet Inter-Arrival Time", *Computer Science and Electronic Engineering (CEEC), 2016 8th*, pp. 204–209, 2016.

105. Bollmann, C., M. Tummala, J. McEachen, J. Scrofani and M. Kragh, "Techniques to Improve Stable Distribution Modeling of Network Traffic", *Proceedings of the 51st Hawaii International Conference on System Sciences*, 2018.

106. Roberts, S. J., "Novelty Detection Using Extreme Value Statistics", *IEE Proceedings-Vision, Image and Signal Processing*, Vol. 146, No. 3, pp. 124–129, 1999.

107. Hugueny, S., D. A. Cliftony and L. Tarassenko, "Novelty Detection with Multivariate Extreme Value Theory, part II: An Analytical Approach to Unimodal Estimation", *2009 IEEE International Workshop on Machine Learning for Signal Processing*, pp. 1–6, IEEE, 2009.

108. Fadlil, A., I. Riadi and S. Aji, "A Novel DDoS Attack Detection Based on Gaussian Naive Bayes", *Bulletin of Electrical Engineering and Informatics*, Vol. 6, No. 2, pp. 140–148, 2017.

109. Sun, D., K. Yang, Z. Shi and Y. Wang, "A Distinction Method of Flooding DDoS

and Flash Crowds Based on User Traffic Behavior", *Trustcom/BigDataSE/ICESS, 2017 IEEE*, pp. 65–72, 2017.

110. Arora, K., K. Kumar and M. Sachdeva, "Characterizing DDoS Attack Distributions from Emulation Based Experiments on DETER Testbed", *International Conference on Advanced Computing, Networking and Security*, pp. 541–550, Springer, 2011.

111. Yadav, S. and S. Selvakumar, "Detection of Application Layer DDoS Attack by Modeling User Behavior Using Logistic Regression", *2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)(Trends and Future Directions)*, pp. 1–6, 2015.

112. Mok, M. S., S. Y. Sohn and Y. H. Ju, "Random Effects Logistic Regression Model for Anomaly Detection", *Expert Systems with Applications*, Vol. 37, No. 10, pp. 7162–7166, 2010.

113. Edwards, A., "Likelihood Cambridge", *UK Cambridge University*, 1972.

114. Posada, D. and T. R. Buckley, "Model Selection and Model Averaging in Phylogenetics: Advantages of Akaike Information Criterion and Bayesian Approaches Over Likelihood Ratio Tests", *Systematic biology*, Vol. 53, No. 5, pp. 793–808, 2004.

115. Akaike, H., "Information Theory and an Extension of the Maximum Likelihood Principle", *Selected Papers of Hirotugu Akaike*, pp. 199–213, Springer, 1998.

116. Huang, L., Y. Xiao, K. Liu, H. C. So and J.-K. Zhang, "Bayesian Information Criterion for Source Enumeration in Large-Scale Adaptive Antenna Array", *IEEE Transactions on Vehicular Technology*, Vol. 65, No. 5, pp. 3018–3032, 2016.

117. Resnick, S. I., *Extreme Values, Regular Variation and Point Processes*, Springer, 2013.

118. Bennett, S., "Log-logistic Regression Models for Survival Data", *Applied Statistics*, pp. 165–171, 1983.

119. Ester, M., H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.", *Kdd*, pp. 226–231, 1996.

120. Erman, J., M. Arlitt and A. Mahanti, "Traffic Classification Using Clustering Algorithms", *Proceedings of the 2006 SIGCOMM Workshop on Mining Network Data*, pp. 281–286, ACM, 2006.

121. Kullback, S. and R. A. Leibler, "On Information and Sufficiency", *The Annals of Mathematical Statistics*, Vol. 22, No. 1, pp. 79–86, 1951.

122. Lin, J., "Divergence Measures Based on the Shannon Entropy", *IEEE Transactions on Information theory*, Vol. 37, No. 1, pp. 145–151, 1991.

123. Vidyasagar, M., "A Metric Between Probability Distributions on Finite Sets of Different Cardinalities and Applications to Order Reduction", *IEEE Transactions on Automatic Control*, Vol. 57, No. 10, pp. 2464–2477, 2012.