DISTRIBUTED DENIAL OF SERVICE ATTACK DETECTION USING DENSITY BASED CLUSTERING AND INFORMATION FUSION TECHNIQUES

by

Muhammet Fatih Bayındır

B.S., Electrical and Electronics Engineering, Boğaziçi University, 2016

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

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University
2019

ACKNOWLEDGEMENTS

Firstly, I present my gratitude to my thesis advisor Professor Emin Anarım. He gave me the encouragement and guidance throughout my thesis. He led me to progress me in both career and academic level. I would like to thank each and every member of the Bogazici University Information and Communication Security Laboratory (BUICS) and BUSIM, my special thanks to Fırat Kimya, Çağatay Ateş, Can Altay, Samet Doksanoğlu, Derya Erhan and Süleyman Özdel for their support during my study. And to my friends in my office, Istanbul Settlement and Custody Bank Information Security and Risk Management team for exculpating me during these times. I am also grateful to Prof. Mutlu Koca and Prof. Tayfun Akgül for sparing time to join my thesis committee.

I am very grateful to my wife, Ceyda. Ceyda has been a tremendous source of motivation and support. She gave me the energy to work at the time I need. Furthermore, I thank my mom Hayrunnisa Bayındır and my dad Zeki Bayındır, and my sisters Serra Bayındır and Hacer Kılıç, for their support and endless love.

Especially, I present my gratitude for the financial support from TUBITAK BIDEB 2210-A Postgraduate Research Scholarship and for supporting young scholars for their research. In addition, this work is supported by the Scientific and Technological Research Council of Turkey (TUBITAK), under Cloud-Based Privileged Access Management System Project, Project No. 117R030.

Finally, I present my apologies to my wife, friends and family due to loading my time with preoccupation during my thesis. I can say having gone through from the dawn to the sleepless nights of hard work has definitely been paid off.

ABSTRACT

DISTRIBUTED DENIAL OF SERVICE ATTACK DETECTION USING DENSITY BASED CLUSTERING AND INFORMATION FUSION TECHNIQUES

The devices in networks are constantly under numerous attack threat in today's complex internet world. DDoS is the well-known type of attack since it is easy to launch and disrupt the target traffic. Attackers can implement various techniques to launch their attacks by masquerading their real identities behind false addresses. In order to ensure confidentiality, integrity and availability, the implementing an Intrusion Prevention Systems is of primary importance in order to establish a secure network infrastructure. In thesis, we propose DDoS attack prevention framework in which multiple metrics from packet headers are used and then fused to generate a collective judgment on whether an attack occurs.

Dempster-Shafer Theory is an information fusion approach for combining various evidences from various sources. An unsupervised BPA approach is employed to adapt assignment of beliefs to the most up to date attribute of the network traffic. The BPA approach is simple but highly effective. It has outstanding accuracy with low false alarms and very high attack identification rate.

It is hard to find publicly available datasets for DDoS schemes. We used CAIDA and Bogazici University datasets. Datasets with more comprehensive features would be better to test success of our method. A descriptive analysis of the produced results, for all the used datasets are given. The performance evaluation of the effectiveness of proposed scheme is measured by using different attack rates.

ÖZET

YOĞUNLUK TABANLI KÜMELEME VE BİLGİ FÜZYON TEKNİKLERİNİ KULLANARAK DAĞITIK HİZMET REDDİ SALDIRISI ÖNLEME

Ağlardaki cihazlar, günümüzün karmaşık internet dünyasında sürekli olarak sayısız saldırı tehdidi altındadır. DDoS iyi bilinen bir saldırı türüdür çünkü hedef trafiği etkilemek ve bozmak kolaydır. Saldırganlar, yanlış adreslerin arkasındaki gerçek kimliklerini maskeleyerek saldırılarını başlatmak için çeşitli teknikler uygulayabilirler. Gizliliği, bütünlüğü ve kullanılabilirliği temin etmek amacıyla İzinsiz Giriş Önleme Sistemlerinin uygulanması, ağlardaki güvenlik altyapılarının geliştirilmesinde temel öneme haizdir. Bu tez, paket başlıklarından gelen metriklerden yararlanan ve ardından bir saldırının gerçekleşip gerçekleşmeyeceği konusunda toplu bir karar vermek için onları birleştiren bir DDoS saldırı önleme sistemi önermektedir.

Dempster-Shafer Teorisi belirli kaynaklardan gelen çeşitli delillerden yararlanmak için bir bilgi birleştirme yaklaşımıdır. İnanç fonksiyonunu ağ trafiğinin en güncel özelliğine uyarlamak için denetimsiz BPA yaklaşımı kullanılmaktadır. Kullanılan BPA yaklaşımı basittir ancak oldukça etkilidir. Düşük yanlış alarmlar ve çok yüksek saldırı tespit oranı ile olağanüstü bir doğruluğa sahiptir.

DDoS programları için herkese açık veri setleri bulmak oldukça zordur. CAIDA ve Boğaziçi Üniversitesi veri setleri gibi kapsamlı özelliklere sahip veri kümeleri, yöntemin başarısını test etmek için daha iyi olacaktır. Bu tezde, değerlendirilen tüm veri setleri için oluşturulan sonuçların tam bir açıklaması verilmiştir. Önerilen programın etkinliğinin değerlendirilmesi, farklı atak oranları kullanılarak gerçekleştirilmektedir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$\mathcal{A}$	Algebra of a Subset
B	A Subset of $\Gamma(\omega)$
$\mathcal{H}$	Precise or Arbitrary Hint
Γ	Multivalued Mapping
Θ	Frame of Discernment
θ	Corresponding Uncertain Outcome of ω
Λ	An Additional Information to Closed World Assumption
Ω	Space
ω	Uncertain Outcome in Ω
bl	Belief Functions
P	Probability of an Event
pl	Plausibility Functions
q	Commonality Function
∞	Infinity
$\emptyset$	Empty Set

LIST OF ACRONYMS/ABBREVIATIONS

BOUN	Bogazici University
BPA	Basic Probability Assignment
CAIDA	Center for Applied Internet Data Analysis
DDoS	Distributed Denial of Service
DNS	Domain Name System
DoS	Denial of Service
DST	Dempster Shafer Theory
FN	False Negative
FNR	False Negative Rate
FP	False Positive
FPR	False Positive Rate
FTP	File Transfer Protocol
GA	Genetic Algorithm
HIDS	Host Based Intrusion Detection Systems
HR-DDoS	High Rate Distributed Denial of Service
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol
IDS	Intrusion Detection System
IP	Internet Protocol
IPS	Intrusion Prevention Systems
ISCS	Immediate System Call Sequence
ISP	Internet Service Provider
LAN	Local Area Network
LFC	Locality Frame Count
LR-DDoS	Low Rate Distributed Denial of Service
MAP	Maximum A Posteriori Probability
MSCD	Malicious System Call Execution Detection
NB	Naive Bayes

NIDS	Network Based Intrusion Detection Systems
OS	Operating System
OSI	Open Systems Interconnection
PCA	Principle Component Analysis
SIEM	Security Incident and Event Management
SMTP	Simple Mail Transfer Protocol
SOC	Security Operation Center
SSH	Secure Shell
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TPR	True Positive Rate
UDP	User Datagram Protocol
URL	Uniform Resource Locator
VM	Virtual Machine
WAF	Web Application Firewall

1. INTRODUCTION

With people and businesses relying on it for their communication and information needs, the Internet has become essential part for the today's world. Guaranteeing the availability of internet services is a challenging duty because of the growing volume of the traffic and the various communication standards that it has to comply with.

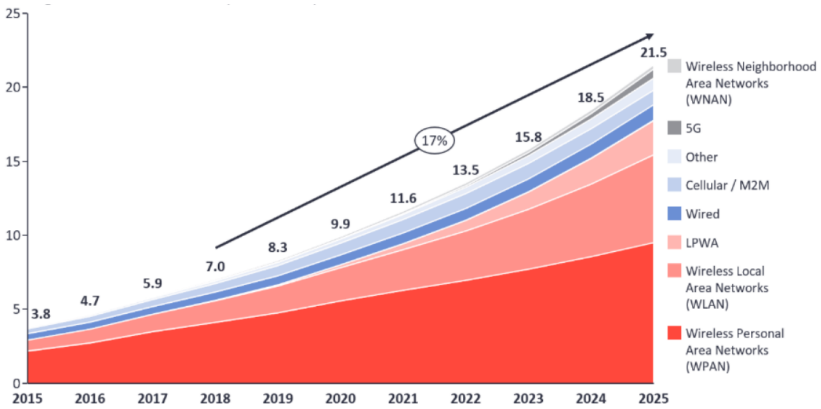

Figure 1.1. Global number of connected IoT devices [1]

The number of Internet-connected devices is expected to surpass 50 billion in 2020 compared to 17 billion now [4]. That is to say there is more computer-connected devices for each person in the future. The connection of these devices create many convenient benefits for the society. For instance, online banking facilitates faster purchases, thus leading to growing businesses and home appliances connected each of them thus creating an internet of things to smartly manage your home. Consequently, an increasing number of services and devices produce more data and traffic, and the modern society has to deal with its security. In order to maintain confidentiality, integrity and availability of next generation's large networks, security measurements must be taken accordingly. In lights of mentioned developments, we see tremendous growth of investment in cyber security technologies by corporations.

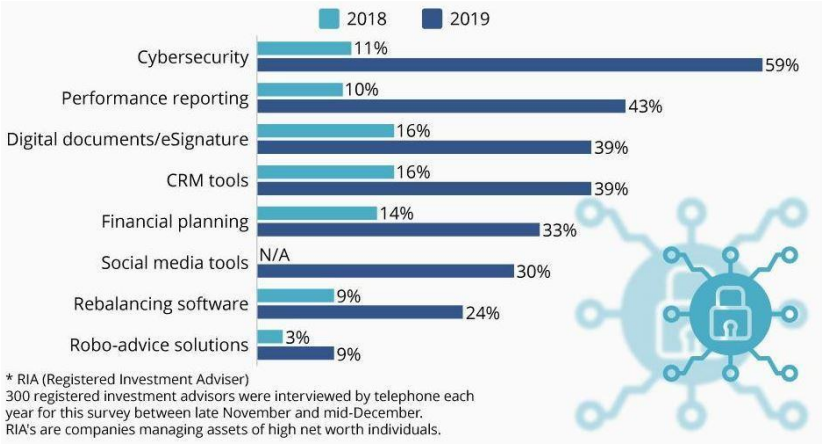

Figure 1.2. Percent of Registered Investment Agency considering investment in certain technologies [1]

A well-known way to interrupt the availability of a computer network is Distributed Denial of Service Attacks. A distributed denial-of-service (DDoS) attack is an ill will attempt to disrupt normal traffic by overwhelming the target itself or its background infrastructure with a large amount of illegitimate Internet traffic. An assaulter utilizes multiple compromised computer systems which named as zombies and manipulates them as sources of attack traffic. Computers and different network elements and IoT devices can be exploited in this process. Looking from a broader angle, a DDoS attack can be defined as a road hog clogging up the flow of the highway, thus preventing legitimate traffic to arrive at its destination.

In DDoS attack, an attacker needs to take control of a network of online remote machines to realize its intention. Remote computers and other appliances (e.g. IoT devices) are contaminated with malware, making them infected and turning them into a bot or zombie. By using this method, attacker can remotely direct the group of bots that form a botnet.

If a botnet is established, an attacker can instruct the remote machines by sending series of commands to all bots in it. As soon as the IP address of a victim is a target of a botnet, bots start sending packets to the victim, which potentially causes target to be overwhelmed. The resulting disruption in the normal traffic is called a DoS. Because

every bot is recognized as legal Internet machine, separating them as malicious traffic from normal traffic is hard task.

Different types of DDoS attack paths aim to exploit different components of a network layers. To comprehend how various DDoS attacks work, one should understand network connection layers and components. A network communication on Internet is the composition of different elements. It is similar to building an apartment from the ground to the very top, every step serves for a distinct purpose in the model. The OSI model, which provides a standard for different computer systems to be able to communicate with each other. Seven distinct layers add up to a conceptual framework designed for explaining network connectivity.

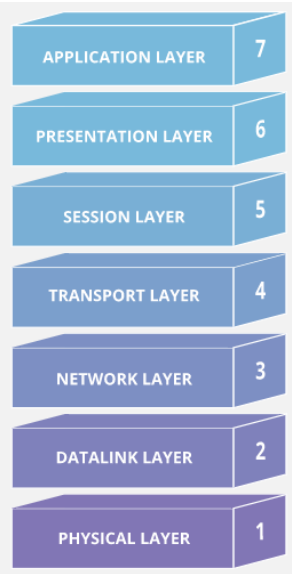

Figure 1.3. 7 Layer OSI model with its internal hierarchy [2]

(i) Physical Layer - Transmits bit stream over the physical raw medium
(ii) Datalink Layer - Describes the format of data on the network
(iii) Network Layer - Makes decision on which physical path the data take
(iv) Transport Layer - Transmits data utilizing transmission protocols e.g. TCP and UDP
(v) Session Layer - Establishes connections and is responsible for controlling ports and sessions

(vi) Presentation Layer - Ensures that data is in sound format and encryption occurs here

(vii) Application Layer - Human-computer interaction occurs, where applications makes connection to the network services

Almost every DDoS attack involves overflowing a target network with traffic. However, attacks can be considered in two main categories. An attacker can exploit a single or diverse attack vectors, or series of attack vectors by considering the potential countermeasures implemented by the target. Consequently, there are different types of attacks.

1.1. High Rate DDoS Attack

Traditional attacks are put in this category in which attacks known as brute force attacks. Thus, rate detection techniques can work with high accuracy in this type of attacks. Volumetric attacks presented in Section 1.3.2 can be considered as high rate DDoS(HR-DDoS) attack type.

1.2. Low Rate DDoS Attacks

Recently, we have a new type of attack named as low rate DDoS attack which is a kind of a DDoS offensive which exploits a low traffic stream which can target the application or device's resources. It is also known as slow-low DDoS attack. Slow and low assaults need very narrow bandwidth and can be hard to mitigate since distinguishing the attack from normal traffic is very hard. Because low and slow attacks do not require a lot of resources to deploy, LR-DDoS attack may effectively start campaigning only using a single machine. For launching a LR-DDoS attack, Slowloris and R.U.D.Y. are the popular tools.

LR-DDoS attackers assault thread-based web-servers with the purpose of piling up every thread with slow requests, which results in preventing legal users from attaining the service. It is realized in a way by transmitting data tardily, but fast enough to

avoid the server from timing out. To visualize LR-DDoS attack, think of an example in which we have a toll road with 3-lane. Each lane has a tollhouse before the entry. Each driver slow down at the tollhouse, pays their bill to enter the highway. After they pass, next cars are taken to the tollhouse and so on. At a moment, three drivers at each lane show up and act very slowly during payment such as handing over coins for the bill and operator consumes so much time counting the money thereby clogging up three lanes for minutes and blocking other cars from passing through. Assaulters use HTTP post requests, HTTP headers, and TCP traffic to carry out LR-DDoS. There are three common LR-DDoS attack examples. In the first example, we have Slowloris tool which makes connections to a web-server. After that it slowly sends part of HTTP headers which ends up the web-server to maintain the connection open waiting for the rest of the headers to complete freezing out the thread [2]. An alternative tool we have is R.U.D.Y. which produces HTTP-post requests for filling out form field options. Server asks how much data it should anticipate. Tool responds to server, however sends the data ponderously. The server maintains the connection open since it is expecting the rest of the data. Moreover, Sockstress attack is another type of LR-DDoS. This type abuses a characteristic of the TCP three-way handshake process, which leads to ambiguous connection.

Detecting the traffic rate as in the HR-DDoS attacks does not work here. One way to prevent LR-DDoS is to increase your network bandwith. If you have more connections simultaneously available, it is hard for a culprit to clog your network traffic. The difficulty with this solution is that an assaulter can easily increase the scale of their assault size to disrupt your network's capacity. Another resolution is load-balancing, which stop LR-DDoS attacks before reaching origin network. The protocol and application layer attacks which are presented in the next section can be considered as low rate DDoS attack.

1.3. Frequent Types of DDoS Attack

1.3.1. Application Layer Attacks

Sometimes referred to as a layer seven DDoS attack, by means of using these attacks an attacker aims to load exhaustion to the target resources. The layer in which generation of webpages is done and HTTP requests are replied on the server are targeted by the attacks. Execution of a HTTP request is easy on the client side whereas may be very pricy to reply at the server side. This is because the web servers frequently has to bring several files from database and run queries for reaching out to database to offer webpages. Because the traffic of Layer 7 is hard to flag as malign, this layer's assaults are not easy to mitigate.

1.3.1.1. HTTP Flood. Imagine so many different users simultanously keep pressing refresh button on a webpage for a long time. This leads to a great number of HTTP requests which is equivalent to flooding the target webpage which results in a DoS.

We have very complex HTTP flood attacks as well as simple ones. An example of a simple versions can reach a URL user agents, referrers and IP addresses within the identical range. On the other hand, complex implementations abuse assaulting IP addresses from wide variety of range. These attacks target random web links by employing random user agents and referrers.

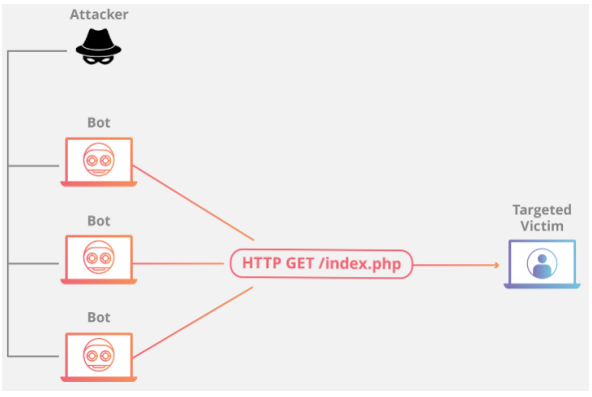

Figure 1.4. HTTP flood DDoS [2]

1.3.2. Volumetric Attacks

Volumetric attacks' goal is to generate massive traffic to prevent other traffic between target and internet. Huge loads of data are submitted to victim in the form of amplification and producing large amounts of traffic via other methods, for example a botnet making flood of requests.

1.3.2.1. DNS Amplification. Imagine a man calls a cafe and orders everything on the menu and asks waiter to call him back and tell him the entire order. The man gives waiter number of victim. A little action results in a loaded comeback.

IP address of the victim gets responses from DNS when an attacker sends a request to a public DNS server by spoofing its IP address as that of the victim. The attacker design its request in a way the punlic DNS replies victim with huge data. Finally, an amplified version of the beginning query arrives at victim side.

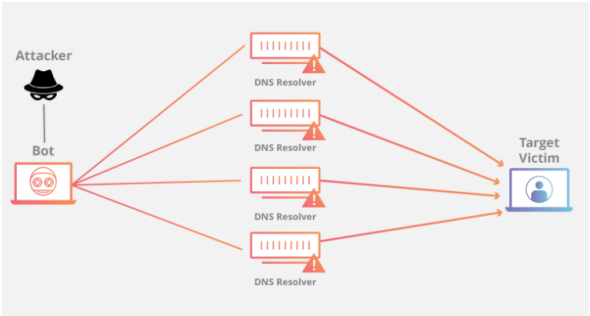

Figure 1.5. DNS Ampplification type DDoS Attack example [2]

1.3.2.2. UDP Flood. It is kind of a DoS attack where huge amount of UDP packets are sent to a victim with the goal of overflowing victim so that it cannot respond and process requests. UDP flood attacks can exhaust the the firewall of the victim that results in a DoS of legal traffic.

Primitively, the way that it works by exploiting a victim server's paces while responding to a UDP request made to its port. When a UDP packet sent to a specific port, steps below are taken in response at the server:

(i) First of all, the server looks for any running programs that are currently listening to requests made to particular port.

(ii) In case none of the programs receives any packet at that port, the server turns with a response to sender with a ping stating that destination is unreachable.

We can think of UDP flood as receptionist at a hotel who routes the calls coming to hotel. Initially, a caller demands her to make a connection to a particular room on the phone. Then, she requires to check all the rooms in order to ensure specified guest stays in the hotel and accepts calls. If she finds out the guest is not willing to take phone call, she picks up the phone and tell guest does not take any call. When all the other receptionists are called at the same time with similar requests, all the phone lines get overwhelmed simultaneously.

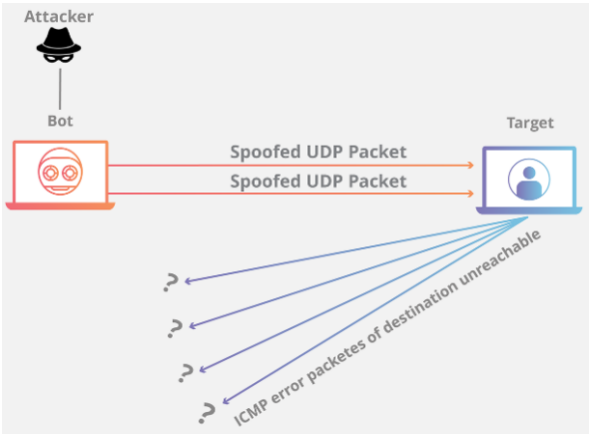

Figure 1.6. UDP flood DDoS [2]

1.3.3. Protocol Attacks(State-Exhaustion Attacks)

They result from a service interruption with the consumption of available state table capacity of application servers or protecting programs such as load balancers or WAF. Utilization of layer 3 and layer 4 weaknesses to render the victim unreachable are exploited.

1.3.3.1. TCP SYN Floods. An analogy with an operator working in backstore room who gets requests from the front side can be made for SYN flood attacks. After

receiving a request, he goes to get the packet and looking for an approval from front side to bring packet out. Then, the operator obtains large amount of packet requests without an approval till they cannot carry any more and the requests waiting for responses.

TCP connection works with the three-way handshake process which is exploited by SYN flood attacks. Generally, TCP connection presents three different steps to establish a connection [2].

(i) At the beginning, SYN packet is sent by client to victim server to initialize a communication.
(ii) Then, a response is returned with SYN/ACK packet from victim server in order to acknowledge the connection.
(iii) Consequently, the client returns ACK packet for acknowledging receipt from the victim server. After this process, communication is established to send and receive data.

Figure 1.7. Three-way handshake [2]

In order to launch a DDoS, an attacker make use of handshake process. As it is explained below:

(i) A large amount of SYN packets are sent to victim server, usually with fake IP addresses.
(ii) A response is given by victim server to each and every connection requests and leaves an open port waiting for the response.
(iii) Then, the victim waits another ACK packet to start the connection, that never comes, the attacker goes on sending extra SYN requests.

(iv) These new SYN requests stimulates victim to spare open port for some time until all the available ports get unavailable and can no longer process requests.

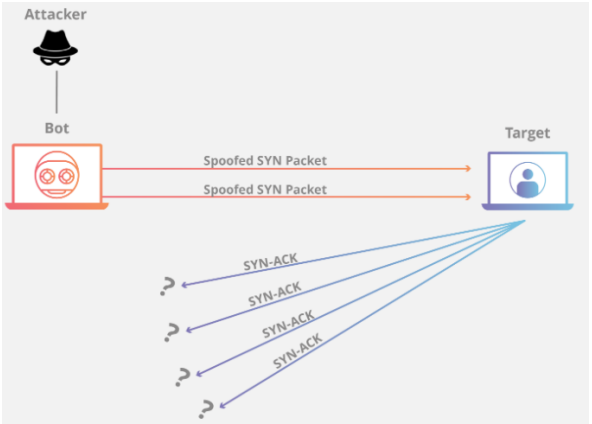

Figure 1.8. SYN flood DDoS [2]

In network terminology, if a machine has an open connection at the one side but the other machine does not, this connection is called half-open. Victim server has open connections which waits for all the requests to timeout before the ports can respond readily. As a result, this kind of attack is called "half-open attack". There are three different way that SYN flood can occur:

a) Spoofed Assault. An attacker may use fake IP address at every new SYN packet with the intention of block attempts from the victim side by avoiding being traced back. Although real IP addresses are hidden behind fake ones, they still can be discovered by ISP.

b) Distributed Assault. If a botnet is employed in an assault, tracing back to the original source is very hard. To increase the obfuscation, an assaulter all of its member of botnet's IP addresses spoofed which packets are sent from. If an assaulter make use of Mirai botnet, it should not hide IP addresses of its infected devices.

If an ill will actor wants a DDoS attack on a target, SYN flood attack requires less traffic than other kind of DoS attacks. SYN attacks solely has to be greater than the backlog capacity of victim side's operating system compared to volumetric attacks, that

aim to overwhelm the infrastructure of the victim. In the case of size of the backlog is known by bad actor, how much connection timeout is needed. With the key knowledge, the attacker is able to spoof the target with exactly the same parameters required to make the system unavailable, thereby generating the just minimum necessary amount of traffic to result in DDoS.

c) Direct Assault: If a victim is not spoofed with a fake IP address, this type of SYN flood attacks is called direct attack. In this kind, the bad actor does not mask their real identity at all. Since they use a sole source machine by their real IP address to launch the assault, they are likely to be discovered and mitigated. To make victim machine to leave a half-open state, the malicious user blocks their device to respond to the victim's SYN-ACK requests that is generally accomplished by creating a firewall rule which stop outgoing packets but only allowing SYN packets. This is also achieved by preventing SYN-ACK packets to come earlier than the arrival of the ill will users' device. Implementation of this method is very rare in practice since the it is easy to mitigate by just blocking all the malicious sources. Again, in the case of Mirai botnet, attacker does not bother spoof infected machine's IP address.

1.3.4. Mitigation of SYN Flood Attack

The vulnerability of SYN flood is well-known and there are various paths to mitigate. Three of them are explained below.

1.3.4.1. Increasing Backlog queue. A victim's OS has some kinds of half-open connections which allows for a remote machine to use. If a system admin increases the backlog capacity of its systems, he/she is able to handle more half-open connections. In order to have higher backlog queue size, they need a more memory space. Unless they have enough memory to increase the memory size, their system performance is negatively affected. However, this may still be preferable to DoS.

1.3.4.2. Recycling the Oldest Half-Open TCP connection. Discarding the oldest half-open connection on the backlog queue and writing incoming connection request to that capacity is another way to mitigate this kind of an attack. In order for this method to be successful, legitimate connection requests must be formed with fewer time than backlog is filled illegal SYN requests. This particular defense is destined to collapse if the attacker sends higher volume, or if the backlog queue is not large enough to make it applicable.

1.3.4.3. SYN cookies. A cookie is generated by the victim for this method. With the intention of preventing against to drop out the connections when the backlog queue is full, the defending machine replies to every connection request with a SYN-ACK packet. However, it then drops the SYN packet on the backlog and removes the request from memory and leaves the port open and ready to start a new connection. The server machine rebuilds the SYN backlog queue entry, if request comes from a legal entity and after a concluding ACK packet is sent from the legal client to the server. While Although this method reveals certain information on the TCP connection, it is still preferable to permitting DoS to happen to legitimate users.

1.4. Zero Day DDoS Attacks

The Zero-day as it name implies covers all the undiscovered or newly generated attacks in a way that exploits vulnerabilities since there is no patch to install yet. It is common in underground world trade of zero-day vulnerabilities by hackers since they cannot be mitigated easily.

1.5. History of Notorious DDoS Attacks

1.5.1. GitHub Attack - 2018

The greatest known DDoS attack ever known occurs in February 2018. A well-known online code repository service called GitHub was the victim of this attack which

is used by so many software developers. When the attack hit its peak bandwidth value, the incoming traffic rate was 1.3 Tbps(Terabyte/second), corresponding to 126.9 million packets/second. There was no botnet rather the assaulters deploy the amplification impact of a well-known database system called memcached. Memcached servers are flooded with spoofed requests, thereby they can magnify the impact of attack by 50,000 times. The world's greatest DDoS attack just lasted about 20 minutes because GitHub is able to detect and mitigate the attack quickly with its DDoS protection service.

1.5.2. The Dyn Attack - 2016

The second largest DDoS attack is targeted the Dyn which is a prime DNS provider in 2016's October. This one results in a devastation and generated service disruotion for many internet websites such as Reddit, GitHub, Netflix, AirBnB, Visa, PayPal, The New York Times and Amazon. This attack make use of the Mirai botnet. A Mirai botnet forms itself from infected IoT devices including smart TVs, cameras, printers, radios, baby monitors and so on. In order to produce the attack, these infected machines are all configured for sending illegal requests to a victim. Luckily, the attack is resolved by Dyn within one day. However, the motivation behind the attack is never revealed.

1.5.3. GitHub Attack - 2015

When it happens, it is known to be the biggest attack ever for the time. Again, the victim of the attack is GitHub. The motivation behind the attack is known to be politic and it lasts for a few days since the attackers change strategies against implemented DDoS protection methods. DDoS traffic is initiated from a source in China at the time. The traffic to create the attack is launched with the injection of a JavaScript code into the people's browsers visiting Baidu which is a popular search engine in China, which results that the infected browsers to send HTTP requests to the GitHub links. Websites that uses Baidu's services for analytics also inject the malicious code. After the attack is defeated, it is realized that the malicious code does not originate from Baidu, instead it is inserted by an intermediate service.

1.5.4. Spamhaus Attack - 2013

An attack on Spamhaus occurs in 2013 which is the greatest ever attack at the time. Spamhaus is a project helping combat spam emails or related activity. The project is responsible to filter out almost 80% of existing spam, that turns them into a target among ill will people who want to reach spam emails their desired recipients. Spamhaus receives 300 gigabit/second attack rate during the attack. Fortunately, attackers are not able to accomplish their aims, instead results in a major issues for LINX, the London internet exchange in UK. The main actor of the attack is recognized as hired teenager hacker.

1.5.5. Estonia Attack - 2007

Estonia is hit by a large DDoS attack in April 2007 nationwide. Major state services, banking industry and media outlets is targeted by the culprits. This is especially devastating for them because Estonia as a state almost practically completing digital transformation. Aftermath of the event causes the creation of laws for cyber warfare since it sparks conflict with other governments.

1.5.6. Mafiaboy Attack - 2000

Fifteen year old boy later known as Mafiaboy takes down certain websites such as Dell, CNN, Yahoo, eBay and E-Trade. This attack has major effects including making chaos in the stock market. Mafiaboy recognized to be Michael Calce is a high school boy at the time. He hijacks the internet networks of certain universities and using their servers to launch and coordinate the DDoS attack. After this incident, cybercrime laws were introduced by lawmakers which basically lead to today's cyber law framework.

1.6. Mitigation Approaches to DDoS Attacks

The main objective of the mitigation is to distinguish attack from normal traffic. For instance, at the time of new gadget release, the enterprise's website is hurled

with impatient customers. At this moment, cutting off the access is a catastrophic mistake. When the enterprise observe surge in its incoming traffic from unintended users, actions to eliminate the attack are necessary. The hardship is differentiating the eager customers from the illegitimate ones.

In the contemporary Internet, DDoS can be seen in various formats. For example, traffic can come legally from single source or sophisticated and adaptive trajectories may be formed to spoof destination. A multipath DDoS attack benefits from diverse attack vectors with the aim of flooding a target with alternate ways hence potentially by-passing preventing actions on any of the paths. An example assault targeting different protocol stacks simultaneously is combination of DNS amplification which assaults on layers three and four, and HTTP flood targeting layer seven.

1.6.1. Introspection Based Mitigation Methods

In order to counter against a multipath DDoS assault, one need to employ series of strategies. If an attacker is able to make his/her assault more complex, it is very hard to distinguish from normal traffic and mitigate that attack in general. Mitigation initiatives which include rate limiting or dropping network bandwidth indefinitely dismiss intended users out with the illegal ones. Also, attacker can adapt to circumvent preventing actions. To alleviate a complex network traffic at disruption, a multiple approach solution gives the outstanding results.

1.6.1.1. Blackhole Routing. A simple and primitive method to mitigate DDoS attack is to funnel traffic into that a route which visually goes nowhere. This is equivalent to cutting off all the network traffic. Unless blackhole routing is applied with sophisticated restriction rules, legal and malicious network traffic together are routed to a blackhole and they are not allowed in network traffic. If a network exposes to a DDoS attack, the ISP sends incoming traffic into a blackhole as a shielding action.

1.6.1.2. Rate Limiting. A network administrator limiting the packet per second a network can transmit over certain time window is also accepted as mitigation method of DDoS attacks. This method is successful in mitigating brute force login attempts and slowing down hackers from reaching private content. However, it is not sufficient to overcome a comprehensive DDoS attack by itself. Nonetheless, rate limiting can be effective vector in multipath DDoS mitigation strategy.

1.6.1.3. WAF(Web Application Firewall). WAF is a software which may be useful to defend against a layer seven or application layer DDoS attack. The WAF works as a reverse proxy if an administrator puts it between intranet newtork and Internet. Creating rules to filter out requests by DDoS tools, layer 7 assaults may be mitigated. Using successful WAF enables network admins to create effective rules to rapidly employ event-specific rules against an attack.

1.6.1.4. Load Balancing of Network Traffic. In this method, a small network is used to disperse the malicious traffic within a group of servers to absorb huge amount of traffic along the network. It is similar to channeling a river at the rapids into distributed smaller channels. By means of load balancing, effect of the DDoS attack traffic is reduced to the level at which it is controllable, preventing any disruptive capability.

1.6.2. Network Anomaly Based Mitigation Approaches

Systems that are employed at the network aim to defend information systems against unknown vulnerabilities. We have two kinds of systems that can realize this task, Intrusion Prevention Systems (IPS) and Intrusion Detection Systems (IDS). An intrusion can be described as externally-induced disruption originating from a successful assault [5].

Generally, the network packets that are transmitted on the network are monitored and controlled accroding to some predefined rules. In case of a malicious activity, IPS raise a flag. The major goal while detecting malicious activity is often to recognize

the real threats and maximize the accuracy by making the false positives as low as possible. IPS are able to prevent malicious attempts even before reaching the user. These systems observe the network flow, if they see a suspicious activity of an account, they terminate connections just by adding a rule to block IP address of the attacker and the user account. Usually, IPS employ means of different security controls such as router or firewall in order to protect the information systems against the attacks.

In modern world, the most of the enterprises are worried about the security of their information systems as they are under constant attack threat. In order to protect against this kind of hazards, internal security systems are implemented. These systems may be by-passed by unintended people and certain exposures may be faced. If an intruder tries to take system control by repeating series of actions, this is called intrusion attack. IDS's duty is to protect and defend against these kind of activities. In case of a malicious activity is detected, a countermeasure bust be taken immediately. This can be realized again in two paths on the network or on the host itself.

1.6.2.1. NIDS vs HIDS. In NIDS, network connection information specified according to some rule or formula are monitored the to detect a malicious activity. If an irregular event on the network is observed, the network admins are notified. On the other hand, in HIDS workstations are monitored on the host with specified rules to locate an attack. In the case of an attack, the systems admin is notified and demanded to check that specific host only. There are also additional IDS, for example, distribution mode in the analysis process (distributed, centralized) [6], the time of analysis (offline or online) [7] and the system analyzed the specific features (network data or log files data) [8]. These can be classified into signature-based, anomaly-based, and hybrid systems [9].

1.6.2.2. IDS Approaches. Signature-based system works by using a database in which all the past experience with attack model are added and new traffic with attack are added actively to system, thereby system learns how the attack works and protect the system. Attack patterns that previously experienced and incidents are tested with this trained model. When a match between an incoming attack and model, traffic

is classified to be an attack. Snort is a popular misuse based IDS [10]. Snort has various attributes. It analyze the traffic in real time, matches after searching the content by employing past identified attack pattern to reveal malicious behavior. This system is handy in order to capture the known attacks but unknown attacks cannot be identified. Therefore, this type of models is weak for classification and detection of zero-day attacks. New kind of an attack may be classified as normal since it is not in the attack history database [11].

Anomaly-based IDS. A training for the normal traffic in order to learn the properties of the system is involved in anomaly-based IDS models. A comparison is made according to this trained normal pattern and a decision is stated whether or not the new data is malicious. A major advantage of anomaly-based IDSs' is capability to catch previously unseen attack patterns whereas signatures only used in misuse detection. In normal conditions, new traffics which differ from normal model should not be in attack class sometimes. In some cases, trained model may give inaccurate information to classify attack traffic as normal. This leads to high operational cost and undesired false positive rates in practice [12].

Hybrid-based IDS. Hybrid IDS models are obtained after combining Anomaly-based IDS and Signature - based IDS. Complexity of computations for both anomaly detection and signature matching to analyze the incoming network traffic is the primary drawback of these combination systems.

1.6.2.3. Machine Learning Based Approaches. Using semi-supervised and unsupervised algorithm, one can make anomaly detection. For example, self organizing map [13], clustering algorithms [14] and SVM with one class [15]. Usage of machine learning algorithm facilitates catching zero-day attacks. [16] suggested an IDS technique which makes combination of hierarchical clustering with SVM and they had reduction in training time. Firstly, implementation of hierarchical clustering is realized via Dynamically Growing Self-Organizing Tree (DGSOT) algorithm [17] on training phase.

On the learning process, with every iteration brings more nodes to the tree. SVM is calculated with every iteration step in order to have reduction computational load. After SVM is trained, hyperplane is formed. A point which is close to these hyperplanes are denoted as support vectors. After this step, just formed support vectors pass to clustering with the aim of controlling the tree growing and thus just support vector nodes get bigger. Stopping criteria such as accuracy or tree size specifies iteration number which is to be repeated at every step i.e. stopping criteria. The method cannot present a accurate results for LR-DDoS like privilege escalation attacks

[18] presents Cloud based IDS to catch number of anomaly virtual machines on the cloud. sX-Engine subsystem (Tier 2) and uX-Engine subsystem (Tier 1) are known to be key constituents of the system. A Cloud Instance Monitor subsystem monitors the activity of the user on a specific directory of application and conveys it to Hlog-H for instant logging. Extraction of information from Hlog-H is done with an Audit Log Preprocessor (ALP) and pre-processession is performed to transform it so that uX-Engine which is a learning subsystem can understand it and then employs it for unsupervised classification in particular self organizing map [13] as a first tier system to identify the abnormality, normal behavior, or special permission. In order verify any abnormality, a connection is made to the Permission Recorder. The definition of second tier sX-Engine can be referred to an existing supervised classification method. The unsupervised learning method implemented by Cloud based IDS results in 89% true positive rate with 9% false positive rate. The performance of this technique is good in the detection of network traffic attacks like port scanning, DoS and IP spoofing. Yet, it is not successful at catching escalation of privilege attacks like worms, rootkit, virus and VM Escape attacks, since the network traces of this type of attacks look like normal traffic. Existing vulnerabilities in the OS or other programs can be exploited by attackers. If a malware script is triggered in the system, some system pattern displays in an abnormal way. Thus, close study of doubtful series of system calls, presence of peculiar traffics or processes can give certain insights to identify these attacks. Review of Program behavior analysis approaches is done below:

1.6.2.4. Behavior Analysis of Programs. Analyzing the network traffic is not enough to identify different types of attacks. In order to detect LR-DDoS attacks, program behavior analysis is helpful. This is classified into two groups Static Behavior Detection (SBD) and Dynamic Behavior Detection (SBD).

Dynamic behavior detection.. DBD methods gather the system calls of the certain programs in every probable execution scenarios and form a basic profile for advanced analysis. The methods suppose that malwares trigger certain system call patterns that makes a deviation from the execution of normal scenario. This dynamic behavior is retained even if the malware code is concealed. The DBD methods are named as frequency based approach, enumeration-based approach, machine learning approach and state-based approaches. Detailed explanations are given below:

Enumeration based methods. create a list with an order of system call sequences of programs to be monitored. In order to detect host based anomalies, one can adopt this method. [19] started the usage of system call analysis long times ago. At the beginning, a proposal which is based on look-ahead pair that contains a system calls database which is used with two values. Every element in this database denotes a system call, and an instant sub-sequence of system calls with n size of window.

In [20] an IDS based on analyzing system call analysis named as Immediate System Call Sequence (ISCS) is proposed to detect assaults in cloud. The method does not employ any learning based system rather it makes system calls database with the shaped format of key-value pair. A key denotes system call name which is unique while the amount signifies an instant system call sequence coming after in the duration of program execution. Observed programs are given in configuration file as determined by cloud admin. A match between the baseline ISCS snapshot with individual ISCS snapshot is monitored if it occurs cloud admin makes the detection. Any mismatch from baseline database corresponds to anomalous sequence. ISCS achieves 98% accuracy for intrusion detection.

[21] proposed an approach named as MSCD (Malicious System Call Execution Detection) based on program cum system-wide detection. The approach creates a program-wide MSCD database at each VM and system-wide MSCD database at cloud manager. Program-wide detection matches the current individual MSCD snapshot of each client VM with baseline MSCD snapshot at client VM. System-wide detection matches the current ISCS snapshots of client VMs with MSCD snapshots at cloud admin. The method presents reliable result although IDS daemon at client side VM is subverted, and at the system level intrusions can be detected. A test for MSCD for verification over University of New Mexico sent mail dataset accomplishes TPR of 80% with 3% FNR.

Frequency-based approaches. An uneven pattern can be seen due to an immediate normal user's misuse behavior like overflowing the buffer, invoking the unwanted function or an unexpected error. Only by mismatches, one may not be certain for the trace as anomalous. In many situation anomaly sequences can occur in a database and their frequency of occurrences may not be too high. Therefore, another pace to enhance the power of n-grams' discrimination, one can count pattern occurrences in the trace. Frequency distribution of abnormal and normal sequences of system calls are used to build frequency based methods. Some of the examples are:

Approach of [22] was enhanced by [23] and named as STIDE. A threshold for frequency was added to STIDE method, resulting in T-STIDE which stands for Time Delay Embedding with Frequency Threshold. The method makes advance control of occurrence frequency of the sequence in database if a match occurs. The entry is labeled as a rare one in the case occurrence frequency is below the predefined threshold. A check with LFC (Locality Frame Count) is done for each mismatch sequence and rare sequences. Count of LFC is computed for locality frame of 20 system calls, that gives number of mismatch in the last 20 sequences.

An anomaly detection approach by [24] employs frequency distribution of normal and anomalous sequences. Assumption of sequences which is rarely present in normal traces and frequent in intrusion traces are doubtful is made. Nevertheless, stationarity

is not observed in sequences within a trace.

A combination of machine learning methods with frequency-based approaches is made by [25]. Each sequence in the database is considered. Initially, the method transforms the input system call traces into numeric feature vector. This is named as 'Bag of system calls' that denotes each system call occurrences. Hence, the order information with system call is not known. Now, every feature is given by $X_i = \{f_1, f_2, ..., f_n\}$ in which f_i is the total amount of system call occurrences s_j and n is the total number of system calls in input sequence Z_i. With generated features which is observed to present accurate results for unknown entries than non-learning based systems, a classifier is trained.

In this thesis, we contribute to literature by using Dempster-Shafer Theory as classification method for detecting attack traffic. Further analysis are done using entropy distance between different header components as features for network traffic. In order to benchmark our study a comparison with SVM and NB are done after using DST. Firstly, we used Bogazici University UDP1, UDP2 and TCP dataset. Again, for a benchmark we used CAIDA 2007 and 2008 [26] datasets to compare our results with existing studies. MATLAB R2019a is used for simulations and analysis. Rest of the thesis is organized as follows Chapter 2 includes survey of existing DDoS attack mitigation related to our work. Chapter 3 is devoted to explaining Methodology used in thesis. Description of Dataset are in Chapter 4 . In Chapter 5, presents Features, Results and Analysis are presented and Chapter 6 gives concluding remarks.

2. SURVEY OF RELATED DDOS ATTACK MITIGATION METHODS

Nowadays, almost every month we hear a DDoS attack occurred to a large enterprise or we experience it on while using some mobile applications. There are several methods to defend and protect against a DDoS. There are pros and cons related to each method Furthermore, there are many research group who has their distinct dataset and a distinct approach to mitigate. Thereby, since it is confidential for their corporation there is no way to study on those dataset for other scientist. Therefore, benchmark studies stay limited a few public dataset.

In order to ensure routine system security low counter measures and numerical computations should be made to evade interruption discovery framework proposed by [27] in 2012. A recommendation by authors which is in contradiction with generating growing number of tenets the advancement streamlining methods like GNP (genetic network programming) can be employed. The GNP works with coordinated diagram. They focus on the issues related with security determined with sending data mining-based IDS.

A focus on interruption identification with the aid of grouping examination by [28] in 2011 is proposed. The key objective is to improve the detection rate and reducing the FPR. A changed dynamic K-means algorithm named MDKM to detect inconsistent entries is devised and reenactment tests for relating are given. First of all, the MDKM computation channels the commotion and detached concentrates on the information set. Furthermore, the high-thickness parameters and bunch parcel parameters, utilizing dynamic iterative process by computing the separations between all example information focuses, the k grouping focus is precisely obtained, then an anomaly detection model is presented. KDD CUP 1999 [29] information set is used to test the performance of the model. Their results indicate the framework has a better TPR and a lower FPR, it achieves high accuracy.

A GA (genetic algorithm) method with an improved introductory populace and recognition admin is presented by [30] in 2014 in order to efficiently identify differing types of system errors. GA was employed for streamlining the inquiry of attack instance in review records, on account of its great adjust investigation/misuse; as demonstrated by the authors it presents the subset of probabel attacks that are seen in the review document in a sensible preparing time. The test is done on KDD'99-NSL benchmark dataset is used to differentiate the misuse cases. Their IDS method with GA enhances TPR of the Network Intrusion Detection Model and lowers the false alarm rate.

A work by [31] in 2014, proposes a novel approach for PCA and SVM via streamlining the piece parameters employing programmed parameter choice strategy. Their technique reduces the testing and preparation time to detect anomaly thus improving the accuracy. Their method was tested on KDD dataset. The datasets are exactly separated into testing and preparing with consideration of the minor attack cases. For instance, R2L and U2R to be available in the testing set to distinguish the event of stealthy attack. Their results indicate that the method is successful in detecting interruptions.

It is noted by [32] in 2014 that for sound web empowered frameworks in the present world the network security is an essential element. As per the authors due to complex sequence of PCs the open doors for attacks and interruptions have grown. As a result, a requirement of large importance for locating the most reasonable routes conceivable to secure our frameworks. So the authors propose anomaly detection frame which assumes important part for security of PC. In order to solve problem of IDS, the most successful technique is machine learning. They monitored that the rising field of semi administered learning offers a guaranteed path for corresponding exploration. Thus, they offered a semi-administered method to lower FPR and to improve location of IDS rate.

Association with web might be both disadvantageous and worthwhile since it can offer to extent business further but at the same time hazard to end clients as it is suggested by [33] in 2014. An increase of data speed information stream further

advances corresponding to organizing along with certain variables there is plausible amount of attacks on PC.

IDS framework for remote sensing systems to examine territories is of great importance as it is proposed by [34] in 2011. A crucial part for a remote sensing system is intrusion discovery. We have two different types of interruption location system: signature based and oddity based as proposed by the authors. They identify a few assaults on WSN and principally focus only on the oddity based interruption location framework.

U2R attack class is asserted to be an open problem for academic researches as it is proposed in [35] in 2015. The main objective of their work is to detect the essential features to further enhance the identification rate and diminish the false alarm rate. The studied highlight subset choice approaches improve the general exactness, detection rate of U2R assault class moreover reduce the computational overhead. The experimental outcomes have showed an improvement in location rate of U2R assault class with highlight subset determination systems.

An intelligent IDS model is suggested by [36] in 2015. Considering the area of nerve and features of worldwide predominance of hereditary computation, the model modifies the neural system weights by using GA. Test outcomes show that the insightful way might improve the proficiency of the intrusion detection.

An emphasize is given to importance of anomaly based interruption identification processes in [37] in 2015. The essential outcomes of these works, the most recent methods and what is not abnormal from the future entries for the field. Furthermore, the technique of building client profiles affects detecting interruptions to be discovered. Consequently, the lights are shed on an offline method using MLP (Multi-Layer Perceptron) and SOM (Self Organizing Maps) that are successful approach for IDS.

In [38], counting difference of SYN-FIN packet numbers is proposed. Normally, every TCP connection requires to make a three-way handshake which begins with a

SYN packet, then ends with a RST or FIN packet. If there is no anomaly involved in a case, there must be equal amount of FIN and SYN packets after all. If there FIN packets are less than SYN packets which means an inequality, one can tell that attack is happening.

In addition to Sun *et al.*, in [39] it is suggested that observation of the difference in numbers of SYN/ACK packets and SYN packets may indicate an attack. Under normal condition, SYN/ACK packets are transmitted from a source and SYN packets arrive at the very same source at the time of 3-way handshake process. It is expected that there must be the similar or the same number of SYN/ACK and SYN packets after long time. Nonetheless, there is various SYN packets sent to target victim and it cannot send SYN/ACK packets back during an attack. Since DDoS attacks are often realized by requests with spoofed IP addresses, thorough study of the entropy of source IP addresses can appear to be reasonable feature to spot a DDoS attack [40]. Moreover, one victim is targeted by the bad actors during an attack. Hence, one can expect a decrease in entropy of destination IP addresses [41].

A group of packets with the same source IP, destination IP, source port, destination port are referred as flow. In the case of a SYN flood attack, there are number of packets with varying ports and IP addresses. However, in the case of flow, there are much less packets compared to normal traffic [42].

There is correlative flow if a group of packets have opposite source port - destination port and source IP - destination IP couples. For example, packet 1:destination IP is B source IP A source port is L destination port is K , packet 2: destination IP is A source IP B source port is K destination port is L. One can expect to see various incoming traffic wheres no outgoing traffic in case of a DDoS attack. Thereby, percentage of correlative flows is expected to decrease during an attack [43].

Connection size distribution is another feature which is used by Rahmani *et al.* to spot DDoS attacks. Connection size distribution is dependent on the service used by client, e.g. HTTP, FTP and DNS. Under normal conditions, the distribution is

assumed to be stable on short period of time. In addition, new IP addresses are put on the suspect table with this system. If a connection size of new IP address is not deemed to be safe, they are blocked and marked as malicious [44].

Furthermore, packet number and the distribution of source IP address on a period are extracted as features for detecting attacks. Hash functions are used by Lee *et al.* in order to map incoming traffic's source IP addresses and total number of packets are checked. Thereby, an increase in total number of packets and source IP of incoming traffic can be expected in case of a DDoS attack [45].

Investigation of the difference between entropy based features can be an accurate indicator of an attack. This is studied by Sharma *et al.* Uncertainty is expected to go up due to nature of DDoS attack when it happens. Thereby, using source IP, destination IP, source port, destination port, TCP flag set, protocol and length distributions in order to find their entropies and then computing entropies of these features to detect attack presence [46]. This methodology is adopted in this thesis.

Chen *et al.* [47] DST is employed to detect anomalies in email worms. They present the accuracy of DST classification by using two standard benchmark studies by combining multiple signals one can accomplish better results rather using a single signal. They use this method to implement to a real-world email dataset in which algorithm runs to detect email worm. DST is a promising approach to detect anomalies in which problems with two or more classes and multiple features (information sources), and. Hence, this approach is adopted in this thesis.

3. METHODOLOGY

3.1. Information Fusion Techniques for DDoS Attack Detection

Statistical modelling approaches have been employed to detect Distributed Denial of Service Attack to network systems. Statistical approaches include methods like Cluster analysis, Multivariate analysis, Principle Component Analysis, Bayesian analysis and Evidence Based analysis. Elements of these systems may contain incomplete or imprecise information which causes uncertainties in the defined metrics. By combining or fusing information from various metrics can help to eliminate these uncertainties. Reasoning is a common method to eliminate uncertainties.

Information fusion is a process to obtain information from several and heterogeneous data sources about different situations or incidents and then integrating them to reach final results with higher accuracy than that of single data source. In other words, information fusion is the approach of handling diverse sources of information which are not perfect in several paths and to obtain a clearer view of the situation at the end, hence reducing *uncertainty*.

The term uncertainty is used to cover different forms of imperfect knowledge; including incompleteness, ambiguity and vagueness. It can also rise from the measurement errors which is done by physical sensors or human observation (expert opinion). Furthermore, data preprocessing like converting data into other forms such as time windowing may disregard important information contained in the raw data.

It is obvious that uncertainty is intrinsic to nature of data. Nevertheless, uncertainty may arise in the fusion process as well. Problems related to uncertainty can be created in case the set of information coming from several sources about the same incident is inconsistent, or when certain information sources state varying degrees of belief to the same event. If a user cannot tell its choice because of lack of trust for the information sources, a decision based on the fusion of multiple beliefs is not certain.

Final choice requires to present the weight of multiple information inputs respectively, that generally results in uncertainty [48].

Before covering methods to deal with uncertainty, we briefly introduce the different classes of uncertainties.

3.2. Different Classes of Uncertainties

According to [49], [50], [51]; [52] different kinds of uncertainties must be handled in a way that is particular to its type. A grouping can be made among the following types of uncertainty [52]:

(i) Conflict
(ii) Imprecision
(iii) Incompleteness
(iv) Randomness

Firstly, uncertainty may appear because of partially conflicting information. For instance, two experts assert different opinion about a specific question. The situation in which the outcome presents with finite precision is referred as imprecision. Think of a situation that outside temperature is known to be in between 27.2 and 28.9 degrees Celsius. The incident where a future result is not certain whereas a probability distribution for the result is present is called randomness which is also referred as inherent variability. For example, outcome of throwing a die which is known to be fair. An event in which a distribution of probability or outcome is defined whereas the information available is not sufficient to identify this distribution of probability or outcome is referred as incompleteness. For instance, an evidence claiming the winner of a race is a female is just sufficient to predict the winner if there is only one female candidate is the winner. Otherwise, this evidence solely permits excluding male candidates.

With the purposes of reasoning, classification into two class can be made for uncertainty [53], [54]:

(i) Aleatory uncertainty
(ii) Epistemic uncertainty

Representing inherent variability is referred as statistical uncertainty or aleatory uncertainty. In other words, the observation of different results when repeating the same simulation over and over again.

In case a system has lack of knowledge, it is known as systemic uncertainty or epistemic uncertainty. The things that we principally know but cannot explain in practice are this type of uncertainty. A distinction can be made by the fact that aleatory uncertainty cannot be decreased or eliminated by inserting more information into the system on the other hand, epistemic uncertainty may not be alleviated [51] [53]. Think of the example of throwing a die where we know the underlying principle but, every time we observe a different result. However, throwing die more and more cannot supply any new data to decrease uncertainty on the result of a new throw. Thereby, aleatory type uncertainty is observed here. At the other side, when an unknown die thrown and a probabilistic distribution is constructed from the results, now more information help to eliminate uncertainty and an accurate model appears. This is an example of epistemic type. In an ideal world, one may want to alleviate all the epistemic uncertainty and deal with only the aleatory uncertainty. However, in real life, assumptions employed and practical constraints can limit the interpretation of uncertainty types [53].

If we have several uncertainty sources, then incompleteness, conflict and imprecision mean lack of knowledge. Thus, they can be assigned to epistemic uncertainty, whereas randomness can be grouped into aleatory uncertainty [52].

3.3. Methods of Reasoning under Uncertainty

In order to draw a line between the dots, among the different reasoning frameworks and the various uncertainty sources, a brief introduction to four famous frameworks to reason under uncertainty are given. Elaborate analysis of DST and Bayes reasoning methods are made and these are compared in this work.

3.3.1. Dempster-Shafer theory

DST framework [55], [56] [57]is an approach to work with incomplete knowledge. In order to work with this method, assignment of belief to group of elements in the defined set rather than assigning belief only to single element, which is done in Bayesian approach. There are several interpretations of the DST which are the lower and upper probabilities models are presented in different publications [58].

3.3.2. Bayesian probability theory

Theory of Probability [59] [60]is a simple and famous approach for reasoning with uncertainty. There are two interpretations of this method which are frequentist and subjective (Bayesian) approach [61]: We focus on the Bayesian approach whereas frequentists solely use present data, Bayesians use data to enhance their beginning belief, e.g. "data"+"initial belief" ="enhanced belief". Combining the data with initial belief can be beneficial in events in whick relatively less information and a priori knowledge are present [62].

3.3.3. Fuzzy logic

The fuzzy logic framework [63], [64]is used in order to study on perception-based information. Perception based information cannot be showed by using a single number, thus it is imprecise in nature. All the things are permitted to be graduated in fuzzy logic [65]. In fuzzy sense, proposed solution may be partially true. For instance, one may think of the hypothesis "The room is very hot". By using straightforward logic, this hypothesis is false or true. This can be true with a degree between 0 and 1 in fuzzy logic [66].

3.3.4. Possibility theory

Possibility theory is another way to work under incomplete information [50], [63]. Rather than a probability assignment to every single element of domain set as in the

Bayesian approach, two valuesis used by possibility theory which are necessity and possibility values which enables representing incomplete knowledge [67]. Given the event possibility is equivalent to zero if and only if its complement is of probability one and vice versa. An event's necessity being one is true if and only if the event is known to be true. In real world, representing binarly is frequently not wholely trusted and a possibility theory with graded notion is used [50]

An extensive comparison of DST and Bayesian approaches are given in this thesis due to their applicability to DDoS Attack Detection frameworks.

3.4. Bayesian Theory

Bayesian method can be defined as the mathematical approach used to compute the occurrence probability of an event, by using past experience. A resulting inference is acquired by the combination of some event probabilities. Conditional probability is a category that Bayesian operations are put into. With conditional probability concept additional information brings new probabilities which are selected from situations which occurs in the past [68]. In order to detect an intrusion, Bayesian would probably need a priori knowledge if an attack appears on the analyzed training set.

Bayesian approach gives event probability A being true under the condition that some evidences E are already given [69]. The needed evidences for computing the conditional probability are obtained using past events that happened in the past under likely experimental conditions with event. These events are in state that are mutually exclusive meaning which the system may present in just one of the states at a time [70]. Bayesian theory provides the conditional probability which is also recognized as posterior probability, is given in next Equation 3.1:

$$\frac{P(E|A)P(A)}{P(E)} = \frac{P(E|A)P(A)}{[P(A)P(E|A) + P(\bar{A})P(E|\bar{A})]} \tag{3.1}$$

A probability assignment cannot be made by Bayesian theory in the event above if an absence of information in regard to this formula. We may calculate the posterior

probability, just after evidence E is extracted. Three probabilistic terms are seen on the Equation 3.1. The term $P(A)$ presents the probability of a specific event being true in the case of lack of evidence. In general, it is denoted as the *a priori* probability. The probability of $P(A)$ is updated after each posterior repetition of the considered event.

Even though Bayesian theory is effective for some problems, there are certain challenges to be addressed upon its utilization. A primary challenge is the fact that the it needs the complete knowledge of both a priori and posterior probabilities of the problem. This is pointed out by most of the scientist working on probability. Defining two probabilities is often very hard and sometimes impossible [51]. Moreover, a problem can be formulated by multiple ways using Bayesian approach if we consider different or additional evidences for the same situations. The posterior probability probably differs for each evidence when these extra evidences are considered. In a similar way, the posterior probability can alter if the evidences are extracted from situations where various conditions occur for the same event. Furthermore, specific probability assignment to uncertainty is not allowed in Bayesian theory. However, it needs probabilities to be assigned to state occurrence at a specific time thus one cannot model the uncertainty [71], [72].

3.5. Dempster-Shafer Theory and Its Interpretations

Firstly, the probabilistic methods for DST are explained in the following subsection. After that, the non-probabilistic ones are given. It is realized with the sense of finite frames, since majority of the offered studies do the same. The infinite frames are included in separate subsection. In each section, the different methods are described with their historical order.

Later we give thje differing methods to compute uncertainty in DST framework. An then, the use of DST in our work is explained.

3.5.1. Probabilistic Approaches

3.5.1.1. Dempster's Multivalued Mappings. The exact wording cited from (with different symbols) Dempster [55]: "Consider a pair of spaces Ω and Θ a multivalued mapping Γ, which assigns a subset $\Gamma(\omega)$ to every $\omega \in \Gamma$. Suppose furthermore, that P is a probability measure on Ω, which assigns probabilities to the members of a class a of subsets of Ω. If P is acceptable for probability judgments about an uncertain outcome $\omega \in \Omega$, and if this uncertain outcome ω is known to correspond to an uncertain outcome $\theta \in \Gamma(\omega)$, what probability judgments may be made about the uncertain outcome $\theta \in \Gamma(\omega)$ ...". We may confirm that $\mathcal{A}$ the algebra of each subset of Ω Ω and Θ are finite sets in the next discussion. The scheme (Ω, P, Γ, Θ) described by Dempster might be presented in various interpretations, that is presented here and the other one in section 3.5.3.

Assume that a certain question, which has a unknown answer, should be worked on and the elements θ of Θ denote the probable answers to the problem. That is to say precisely one of the $\theta \in \Theta$ is the true answer, but it is not known which one. Yet, there is certain information or evidence at hand related to this problem. This evidence enables us to make different interpretations, with dependence on certain unknown cases and these interpretations are denoted by the entries ω from Ω. That is to say there is only one true interpretation ω of Ω, yet again it is not known which one. Not each interpretation is equally likely and a probability $p(\omega)$ denotes these varying likelihoods. The unknown answer θ is known to be in the set $\Gamma(\omega)$ if only $\omega \in \Omega$ is the true interpretation. This type of information is named a hint [57]. By using this approach, one can deduce a definite and clearer understanding to the further significant premises given by [55].

The hypothesis which the unknown outcome θ is in H might be considered if H is a subset of Θ, . To examine the hypothesis H with the information from the hint (Ω, P, Γ, Θ), one shall ask, that of the probable interpretations make H necessarily correct. All interpretations ω that has a non-empty set is in $\Gamma(\omega)$ that is included in H since $\theta \in \Gamma(\omega)$ if such an interpretation is true and hence necessarily $\theta \in H$. Defining

the $u(H)$ as:

$$u(H) = \{\omega \in \Omega : \emptyset \neq \Gamma(\omega) \subseteq H\} \tag{3.2}$$

Interpretations that could make H not necessarily correct, but just possible are given in:

$$v(H) = \{\omega \in \Omega : \Gamma(\omega) \cap H \neq \emptyset\} \tag{3.3}$$

because if $\omega \in v(H)$ is the true interpretation, then $\theta \in \Gamma(\omega)$ and it is at least probable which θ is in H, yet not certainly.

Observe that $v(\Theta)$ includes every interpretation with $\Gamma(\omega) \neq \emptyset$. Now, since Θ is thought to include the correct one, an interpretation ω that is not in $v(\Theta)$ is impossible. Hence, the true interpretation should be in $v(\Theta)$ and this additional information permits to go to the conditional probability $p(\omega|v(\Theta)) = p(\omega)/P(v(\Theta))$ for the probable interpretations in $v(\Theta)$.

Since the true interpretation is not known, it is impossible to verify if it is in $u(H)$ or $v(H)$, which is if H is only possible or true. Yet, it is at least probable to calculate the possibilities in which the true interpretation is in $v(H)$ or $u(H)$.

$$sp(H) = P(u(H)|v(\Theta)) = P(u(H)/P(v(\Theta)) \tag{3.4}$$

$$pl(H) = P(v(H)|v(\Theta)) = P(v(H)/P(v(\Theta)) \tag{3.5}$$

The *lower probability* of H named $sp(H)$ by Dempster and he asserted that $sp(H)$ might be treated as the minimum value of probability that may be transferred from Ω to results $\theta \in H$. In a similar fashion, he named $pl(H)$ the *upper probability* of H, that may be treated as the greatest probable amount of possibility, that might be transferred to results $\theta \in H$.

Taking into consideration each probable hypothesis $H \subseteq \Theta$, *pl* and *sp* occur to be functions from the power set of 2^{Θ} to [0, 1] and they are named plausibility and support (also belief) functions. From now on we call bl (belief) to support function in order to align with literature. The following theorem reflects the fundamental properties of these function.

Theorem 3.1. *If bl and pl are defined by (3) and (4) relative to a hint (Ω, P, Γ, Θ), then*

$$\begin{aligned} bl(\emptyset) = pl(\emptyset) = 0, \\ bl(\Theta) = pl(\Theta) = 1 \end{aligned} \tag{3.6}$$

$$bl(H) \leq pl(H) \text{for all} H \subseteq \Theta \tag{3.7}$$

$$bl(H) = 1 - pl(H^c), pl(H) = 1 - bl(H^c) \tag{3.8}$$

and

$$bl(H) \geq \sum \{(-1)^{|I|+1} bl(\cap_{i \in I} H_i) : \emptyset \neq I \subseteq \{1, 2, ..., n\}\} \tag{3.9}$$

for all $n \geq 1$ and sets H, H_i in Θ s.t. $H \supseteq H_i$

$$pl(H) \leq \sum \{(-1)^{|I|+1} pl(\cup_{i \in I} H_i) : \emptyset \neq I \subseteq \{1, 2, ..., n\}\} \tag{3.10}$$

for all $n \geq 1$ and sets H, H_i in Θ s.t. $H \subseteq H_i$

The inequality (8) states that bl is monotone of order ∞ and (9) says that pl is an alternating order ∞. Consequently, (8) and (5) states that bl is a Choquet capacity with ∞ order.

It is possible which one has two or more sources or hints of information $H_i = (\Omega_i, P_i, \Gamma_i, \Theta_i)$, $i = 1, 2, ..., m$ relative to the same question. Then, this data must be joined to achieve integration of each available information. With this information we have the combination as:

$$\mathcal{H}_1 \oplus \mathcal{H}_2, ..., \oplus\mathcal{H}_m = (\Omega_1 \times \Omega_2 \times ... \times \Omega_m, P_{1,2,...,m}, \Gamma, \Theta) \tag{3.11}$$

The combination of stochastically independent sources is named as *Dempster's rule of combination.*

The subsets $\Gamma(\omega)$ being all equal and identical to a subset B of Θ for all interpretations ω is possible. This simply states that the true answer is certainly in the subset B. Such a hint is named deterministic and is represented by B.

Theorem 3.2. *Let bl, $bl(. \mid B)$ and pl, $pl(. \mid B)$ represent the belief and plausibility functions of $\mathcal{H}$ and $\mathcal{H} \oplus B$ respectively. Then*

$$bl(H \mid B) = (bl(H \cup B_c) - bl(B_c))/(1 - bl(B_c)) \tag{3.12}$$

$$pl(H \mid B) = pl(H \cap B)/pl(B) \tag{3.13}$$

This is the definition of Dempster's rule of conditioning. Particularly, it is probable that $\Theta = B$. Now, the deterministic hint becomes vacuous since it carries no additional information concerning the question at hand. For a *vacuous* hint $bl(H) = 0$ and $pl(H) = 1$ for all $H \neq \Theta$. This is a perfect representation of **complete ignorance**. It is obvious we have $\mathcal{H} \oplus \Theta = \mathcal{H}$.

The hint is called *Bayesian* or *precise* by Shafer [56] if for each ω the sets $\Gamma(\omega)$ are singletons. It is now simply a random variable. In this situation, $bl = pl$ and bl became actually a measure of probability on Θ and with Dempster's rule of conditioning we have a reduction of general description of conditional probability. A generalization can be made in the following theorem:

Theorem 3.3. *(From Shafer [56]) Let $\mathcal{H}_1$ be an exact hint, $\mathcal{H}_2$ an arbitrary hint. Then $\mathcal{H} = \mathcal{H}_1 \oplus \mathcal{H}_2$ is also an exact hint. Represent by p; p_1 the probabilities induced by $\mathcal{H}$ and $\mathcal{H}_1$, and by pl_2 the plausibility function relative to $\mathcal{H}_2$. Then, for all $\theta \in \Theta$:*

$$\begin{aligned} p(\{\theta\}) =& k\ p_1(\{\theta\})pl(\{\theta\}) \\ k^{-1} =& \sum_{\theta \in \Theta} p_1(\{\theta\})pl(\{\theta\}) \end{aligned} \tag{3.14}$$

This asserts that single piec of information is required in the case of second hint $\mathcal{H}_2$. The singletons' plausibilities are sufficient, in fact it is even sufficient knowing just these plausibilities' relative values. This is a very interesting outcome, that allows linking **DST** with the general **Bayesian analysis**.

The next theorem explains how to acquire belief and plausibility functions for combination of hints in the usual case.

Theorem 3.4. *([57], [73]). Let $\mathcal{H}_i = (\Omega_i, P_i, \Gamma_i, \Theta_i)$, $i = 1, 2, ..., m$ be independent hints related to the same problem with belief and plausibility function bl_i; pl_i , and let bl and pl represent the belief and plausibility functions of $\mathcal{H} = \mathcal{H}_1 \oplus \mathcal{H}_2, ..., \oplus\mathcal{H}_m$. Then,*

for each subset H of Θ.

$$bl(H) = c\sum\{bl_1(H_c \cup (\cup_{i=2}^{m}\Gamma_i(\omega_i)^c))\prod_{i=2}^{m} P_i(\omega_i) : (\omega_2, ..., \omega_m) \in \Omega_2 \times ... \times \Omega_m\} - (c-1) \tag{3.15}$$

$$pl(H) = c\sum\{pl_1(H \cap (\cap_{i=2}^{m}\Gamma_i(\omega_i)))\prod_{i=2}^{m} P_i(\omega_i) : (\omega_2, ..., \omega_m) \in \Omega_2 \times ... \times \Omega_m\} \tag{3.16}$$

where

$$c^{-1} = \sum\{pl_1(\cap_{i=2}^{m}\Gamma_i(\omega_i)))\prod_{i=2}^{m} P_i(\omega_i) : (\omega_2, ..., \omega_m) \in \Omega_2 \times ... \times \Omega_m\} \tag{3.17}$$

Dempster defined another set function that has no distinct interesting interpretation, yet that has a simple transforming with Dempster's rule of combination which is:

$$q(H) = \sum\{p(\omega) : \Gamma(\omega) \supseteq H\} \tag{3.18}$$

is named as **commonality function**. This function can simply be multiplied when hints are combined:

Theorem 3.5. *([55], [56]) Let* $\mathcal{H}_i = (\Omega_i, P_i, \Gamma_i, \Theta_i)$, $i = 1, 2, ..., m$ *be independent hints relative to the same question with commonality functions* q_i, *and let* $\mathcal{H} = \mathcal{H}_1 \oplus \mathcal{H}_2, ..., \oplus\mathcal{H}_m$ *with commonality function* q. *Then for every subset of* H *of* Θ

$$q(H) = q_1(H)...q_m(H) \tag{3.19}$$

This property enables especially theoretical researches of Dempster's rule. It is shown next theorems which from the commonality function both the belief and the plausibility functions can be acquired and vice versa.

3.5.1.2. Random Sets. The mapping $\Gamma : \Omega \to 2^{\Theta}$ may be explained as a random set. The probabilities:

$$m(H) = \sum\{p(\omega) : \Gamma(\omega) = H\}, \text{ for all } H \in 2^{\Theta} \tag{3.20}$$

describe a probability measure on 2^{Θ}. The probability distribution of m denotes a random set in and might be used rather than $(\Omega, P, \Gamma, \Theta)$ to denote hints related to a question at hand. Observe that:

$$\begin{aligned} bl(H) &= \sum\{m(B) : \emptyset \neq B \subseteq H\}/(1 - m(\emptyset)) \\ pl(H) &= \sum\{m(B) : B \cap H \neq \emptyset\}/(1 - m(\emptyset)) \end{aligned} \tag{3.21}$$

Theorem 3.6. *([56]) Let $\mathcal{H}_i = (\Omega_i, P_i, \Gamma_i, \Theta_i)$, $i = 1, 2, ..., m$ be independent hints related to the same question with commonality functions q_i, and and let m represent the random set defined by the hint $\mathcal{H} = \mathcal{H}_1 \oplus \mathcal{H}_2, ..., \oplus\mathcal{H}_m$. Then*

$$1 - m(\emptyset) = \sum\{(-1)^{|B|+1} \prod_{i=1}^{m} q_i(B) : \emptyset \neq B \subseteq H\} \tag{3.22}$$

Moreover, Dempster's rule of combination may also be defined with terms of m. In fact, let $m_i, i = 1, 2, ..., r$, denote r random sets, expressing r independent sources of information related to the same question. Then Dempster's rule of combination results in a random set defined by m that is described:

$$m(H) = \sum\{m_1(B_1)...m_r(B_r) : B_1 \cap B_2 \cap ... \cap B_r = H\} \tag{3.23}$$

This is corresponding to an intersection of the m random sets. The random set point of view is discussed by Nguyen, [74], [75] in particular also in the framework of infinite frames Θ.

3.5.1.3. Propositional Logic with Uncertain Arguments. In Assumption Based Truth Maintenance (ABTM) method in AI [76] framework, knowledge and information are defined by a set of clauses $\sum = \xi_1, \xi_2, ..., \xi_r$ on certain set of propositions N. A clause is a disjunction $x_1 \vee x_2 \vee ...x_t \vee$ of literals and a literal is a proposition p or its negation $\neg p$. There is a subset of propositions $A = \{a_1, a_2, ..., a_s\}$ of N named as assumptions, that are uncertain, but for that probabilities of truth $q_i = p(a_i)$ are given. This kind of assumptions serve for instance to define uncertain implications or rules such as $(p \vee a_i \rightarrow q) = (\neg p \vee \neg a_i \vee q)$. The implication $p \rightarrow q$ is valid if the assumption a_i holds, otherwise this implication does not hold.

This framework makes it possible the powerful approaches of propositional logic and also of reliability theory of binary systems to evidence theory. This leads also to various computational methods to DST.

3.5.1.4. Evidence Versus Partially Known Probabilities. The evidence and degree of trustness is different. The difference may be examplified by a method suggested by Fagin and Halpern [77]. These authors point out that when defining probabilities of certain events in Θ, it might be impossible to determine probabilities on the entire algebra 2^{Θ}. The available data renders it just possible to identify probabilities P on a subalgebra $\mathcal{A}$ of 2^{Θ} . Nonetheless, the outer and inner probabilities P_* and P^* of P denote upper and lower limits for any extension $P^{'}$ of P on the whole algebra 2^{Θ} : $P_*(H) \leq P^{'}(H) \leq P^*(H)$ for any subset H of Θ. This is a typical method of defining partially known probability. The fact that the outer measure P^* and the inner measure P_* have precisely the same properties as plausibility and belief functions [77] assert that they satisfy the assertions in theorem 1, 8 and 9. This becomes clearer, just if conditioning on an event $B \subseteq \Theta$ is taken into consideration. In the context here, if it becomes known that B holds, then all probability measures $P_{\prime}$ in the family $\mathcal{F}$ of all probability measures on 2^{Θ} defined by $P_*(H) \leq P_{\prime}(H) \leq P^*(H)$ should be conditioned on B in the general way of probability theory, for instance the new bounds are:

$$P_*(H \mid B) = inf\{P^{'}(H \mid B) : P^{'} \in \mathcal{F}\} = inf\{P_{'}(H \cap B)/P^{'}(B) : P^{'} \in \mathcal{F}\}$$
$$P^*(H \mid B) = sup\{P^{'}(H \mid B) : P^{'} \in \mathcal{F}\} = sup\{P_{'}(H \cap B)/P^{'}(B) : P^{'} \in \mathcal{F}\}$$

Fagin and Halpern (1991) show that:

$$P_*(H \mid B) = P_*(H \cap B)/(P_*(H \cap B) + P_*(H^c \cap B))$$
$$P^*(H \mid B) = P^*(H \cap B)/(P^*(H \cap B) + P^*(H^c \cap B)) \tag{3.24}$$

which is different from Dempster's rule of conditioning (see theorem 3.2). And this is not a secret, because P_* and P^* are not degrees of belief and plausibility for H respectively, but upper and lower limits of the only partially known probability of H, which is quite different matter.

3.5.2. Non-probabilistic Approaches

The proposed methods explained before are all created as implementation of probability theory. However, the DST can also be useful on its own by means of an axiomatic theory of math without referring theory of probability. In this subsection, we review two such theories. Yet, as [56] states, "In order to use the theory, yet, we need something more. At the very least, we need canonical examples with which to compare and calibrate actual evidence. Ideally, these canonical examples should motivate the axioms and rules ...". On the other hand, there may be canonical examples which is not based on theory of probability, majority of them is. At the end, theory is again connected to theory of probability.

3.5.2.1. Shafer's Theory of Belief Functions. In order to build theory of evidence axiomatically, one can postulate set functions that has the basic conditions of belief functions. A method to build theory of evidence axiomatically is postulating set func-

tions satisfying the basic properties of belief functions as in Theorem 3.1. Assume a frame of discernment Θ for a exact problem, postulate numerical measure existence of the degree of belief $Bl(H)$ induced by the available information on each hypothesis H imposes now the next conditions as axioms upon the set function Bl (compare with theorem 3.1):

$$\begin{aligned} &bl(\emptyset) = 0 \\ &bl(\Theta) = 1 \\ &bl(H) \geq \sum\{(-1)^{|I|+1} bl(\cap_{i \in I} H_i) : \emptyset \neq I \subseteq \{1, 2, ..., n\}\} \end{aligned} \tag{3.25}$$

for all $n \geq 1$ and sets H, H_i in Θ s.t. $H \supseteq H_i$. From these axioms, we have the following theorem:

Theorem 3.7. *(Shafer 1976) If bl is a belief function, then*

$$m(H) = \sum\{(-1)^{|H-B|} bl(B) : B \subseteq H\} \tag{3.26}$$

is a set function $m : 2^{\Theta} \rightarrow [0, 1]$ *that satisfies:*

$$\begin{aligned} &m(\emptyset) = 0 \\ &m(H) \geq 0 \text{ for all } H \subseteq \Theta \\ &\sum\{m(H) : H \subseteq \Theta\} = 1 \end{aligned} \tag{3.27}$$

and such that

$$bl(H) = \sum\{m(B) : B \subseteq H\}, \text{ for all } H \subseteq \Theta \tag{3.28}$$

On the other hand, *bl* defined by (3.27) is a belief function if a function$m : 2^{\Theta} \rightarrow [0, 1]$ satisfying (3.26). This function m reminds of course the random set model of evidence, with the exception which $m(\emptyset)$ is imposed to vanish. The set function m, that satisfies three conditions of (3.26) is named as **BPA (basic probability assignment)**. It is interpreted as a distribution of a probability mass over the nonempty subsets of Θ, such that $m(B)$ is understood as "the measure of belief that is committed exactly to H" [56]. $bl(H)$ is then the total belief committed to H, which is the sum of all beliefs exactly committed to its subsets (see (3.27)). Sets B for which $m(B) > 0$ are called **focal sets**.

The belief committed to H_c denotes the doubt into H, $Dou(H) = bl(H^c)$ and the lower the doubt in H, the more plausible is H; $pl(H) = 1 - Dou(H) = 1 - bl(H^c)$. The functions *bl*, m and *pl* are hence all linked together and the information of one of them allows the rebuilding of the other two. Thus, one can begin with any of these functions, either m with properties (3.26) or *bl* with axioms (3.26) or also *pl* with axioms similarly to *bl*, except that (3.8) is replaced by (3.9). The function described by $bl(H) = 1 - pl(H^c)$ is then a belief function [56]. There is more the commonality function:

$$q(H) = \sum \{m(B) : B \supseteq H\} \tag{3.29}$$

that is linked to the other functions by the next theorem:

Theorem 3.8. *([56] Shafer). Suppose that bl is a belief function over Θ; pl is its plausibility function and q is its commonality function. Then for all nonempty subsets*

H of Θ*:*

$$\begin{aligned} bl(H) &= \sum\{(-1)^{|B|}q(B) : B \subseteq H^c\}, \\ pl(H) &= \sum\{(-1)^{|B|+1}q(H) : \emptyset \neq B \subseteq H\}, \\ q(H) &= \sum\{(-1)^{|B|}bl(B^c) : B \subseteq H\}, \\ &= \sum\{(-1)^{|B|+1}pl(B) : B \subseteq H\} \quad . \end{aligned} \tag{3.30}$$

Dempster's rule of combination is in this framework best defined using the BPA. The definition can be modeled after (22), but taking into account that in the BPA $m(\emptyset) = 0$. Thus, if $m_i, i = 1, 2, .., r$ are r BPAs for r independent belief functions, then the BPA of the combined belief function is given by:

$$\begin{aligned} m(H) = c\sum\{m_1(B_1)...m_r(B_r) : B_1 \cap ... \cap B_r = H\} \text{ for all } H \neq \emptyset \\ c^{-1} = 1 - \sum\{m_1(B_1)...m_r(B_r) : B_1 \cap ... \cap B_r = \emptyset\} \end{aligned} \tag{3.31}$$

3.5.2.2. Smets' Transferable Belief Model. Another axiomatic method to belief functions is defined by Smets (see for instance [58]). He begins with an allotment $m(H)$ of a total belief mass 1 to the frame of discernment Θ subsets H denoting the probable solution to the problem to be researched. This is very much the same as BPA, except which Smets accepts that probably certain positive belief mass gets closer to $\emptyset$. This denotes the probability which the correct solution is not included in Θ, however is outside Θ, that is named the open world assumption (in contradiction to the closed world assumption supposed so far). $bl(H)$, $pl(H)$ and $q(H)$ are defined as above, except that in (27) the sum extends only over $m(B)$, $B \neq \emptyset$. The distinction between closed and open world assumptions is not really essential. An eentry λ might be put into Θ, that implies "something else" and with $\Theta \cup \{\lambda\}$ we have a closed world. This might even be more prudent, since the empty set is subset of any frame, and if relative to one frame

Θ_1 certain belief is allocated to $\emptyset$, then this belief is automatically also a belief that something is outside Θ_2, that has no meaning.

The difference in the approach of the transferable belief model comes in the treatment of conditioning and combination. The model first defines **conditioning**: If it becomes known the truth is in a subset B of Θ, then the allocation m must be changed into a new allocation $m^{'}$ in the following way:

(i) If $H \subseteq B$, then the evidence that the truth is in B does not modify the belief associated with $H : m^{'}(H) = m(H)$
(ii) If both $H \cap B \neq \emptyset$, and $H \cap B^c \neq \emptyset$, then the belief allocated to H must be transferred to $H \cap B$, because we learn that the truth is not in $H \cap B^c$
(iii) If $H \subseteq B^c$, then the belief allocated to H must be transferred to λ because our initial belief in H now accounts for "something else" relative to B.

This leads to the following definition of the new allocation of belief:

$$m^{'}(H) = \sum \{m(H \cup C) : C \subseteq B^c\} \text{ for all } H \subseteq B \tag{3.32}$$

This is corresponding to Dempster's rule of conditioning. The rule of combination of two belief function bl_1 and bl_2 into a new belief function $bl_{12} = bl_1 \oplus bl_2$ is now defined axiomatically using the rule of conditioning. Here are the first four axioms of [58]:

(i) *associativity* : $(bl_1 \oplus bl_2) \oplus bl_3 = bl_1 \oplus (bl_2 \oplus bl_3)$
(ii) *symmetry* : $bl_1 \oplus bl_2 = bl_2 \oplus bl_1$
(iii) *conditioning*: If bl_2 is such that $m_2(B) = 1$
(iv) *compositionality*: bl_{12} is a function of bl_1 and bl_2 only, then bl_{12} is defined from m_{12} which is itself obtained from (31) by replacing $m^{'}$ by m_{12} and m by m_1.

[58] introduces four more axioms, that are more like technical, and gives proof then that these axioms mean the existence and unity of bl_{12} and also the product

rule for commonality functions (3.19), hence Dempster's rule of combination (without normalization).

3.5.3. Different Methods to Compute Uncertainty

There are number of approaches for handling uncertainty [šek]. In 1948, Shannon asserted: "Information is used to eliminate random uncertainty" and suggested the concept of "information entropy" (using the concept of entropy in thermodynamics) to solve the problem of information measurement [25]. The concept of entropy is the derivation from physics, it is a measure of disorder and uncertainty. A system which has more uncertainty has larger entropy, that also includes more information.

3.5.3.1. Shannon Method for Uncertainty Calculation. The Shannon entropy H is derived as

$$H = -\sum_{i=1}^{N} p_i log_b p_i \tag{3.33}$$

where N is the number of basic states in a system, and p_i is the probability of state i appears satisfying $\sum_{i=1}^{N} p_i = 1$

Shannon entropy has an importance to handle a basic probability problem, and there are some limitations of Shannon entropy . The concept of entropy in the framework of DST is an open issue. Many researchers have extended many measured functions based on it, such as:

3.5.3.2. Dubois and Prade's Method of Uncertainty Calculation. Dubois and Prade weighted Hartley entropy of BPA was shown [50]:

$$H_{dp}(m) = -\sum_{A \subseteq 2^{\Theta}} m(A) log(\mid A \mid) \tag{3.34}$$

3.5.3.3. Höhle's Method of Uncertainty Calculation. One of the former confusion measures for DST was because of Höhle [öhle]:

$$H_0(A) = -\sum_{A \subseteq 2^{\Theta}} m(A) log(bl(A)) \tag{3.35}$$

3.5.3.4. Yager' s Method of Uncertainty Calculation. Dissonance measure of BPA was defined by Yager, as follows [78]:

$$H_y(m) = -\sum_{A \subseteq 2^{\Theta}} m(A) log(pl(A)) \tag{3.36}$$

3.5.3.5. Klir and Ramer's Method of Uncertainty Calculation. Another discord measure of BPA was defined by Klir and Ramer, as follows [64]:

$$H_{kr} = -\sum_{A \subseteq \Theta} m(A) log \sum_{B \subseteq \Theta} m(B) \frac{\mid A \cap B \mid}{B} \tag{3.37}$$

3.5.3.6. Klir and Parviz's Method of Uncertainty Calculation. Klir and Parviz defined entropy [79]:

$$H_{kp} = -\sum_{A \subseteq \Theta} m(A) log \sum_{B \subseteq \Theta} m(B) \frac{\mid A \cap B \mid}{\mid A \mid} \tag{3.38}$$

3.5.3.7. George and Pal's Method of Uncertainty Calculation. George and Pal suggested a definition of conflict measure [80]:

$$H_{gp} = \sum_{A \subseteq \Theta} m(A) \sum_{B \subseteq \Theta} m(B) \mid 1 - \frac{A \cap B}{A \cup B} \mid \tag{3.39}$$

3.5.3.8. Deng's Method of Uncertainty Calculation. It can easily be showed that these methods are based on the Shannon entropy. There are also some studies which give a detailed introduction to these functions [šek], and these entropies have their own basic properties, such as consistency with DST semantics, probability consistency, non-negativity, and later Deng suggested the concept of Deng Entropy [81], that is a new function to measure uncertainty. The Deng entropy is described as follows [81]:

$$H_d = -\sum_{A \subseteq \Theta} m(A) log \frac{m(A)}{2^{|A|} - 1} \tag{3.40}$$

where $\mid A \mid$ is the cardinality of A. As the above, Deng Entropy is very similar to Shannon Entropy, but Deng Entropy uses $2^{|A|} - 1$ to deal with the BPA of multifocal elements, which is more advantageous than Shannon Entropy. In addition, additivity and boundary are expanded.

It is easy to see in today's world that this theory might be seen in several, yet basically equal formats. Some of them are built on probability theory, some are axiomatic theories, *apriori* without a referring to probability theory. In light of this point, one may categorize the methods widely into non-probabilistic approaches and probabilistic ones. The latter ones are trying to integrate evidence theory into the framework of classical probability theory, on the other hand the former deliberately go further classical probability. Indeed, it might be more accurate to name them non-standard probability theories than non-probabilistic ones.

3.5.4. Use of DST in Our Work

DST [55] [56] is a major information fusion method that intrinsically manages uncertainty reasoning and using knowledge present. DST is proven to be an effective and strong information fusion technique in different fraud detection related fields and network security system frameworks [53], [82], [83], [78], [84].

Dempster-Shafer is a useful mathematical method in combination the information with evidences from various and different sources to deduce a fused final belief for the probability of a situation. DST is initiated after froming the frame of discernment:

$$\Theta = \{\theta_1, \theta_2, ..., \theta_n\} \tag{3.41}$$

, where θ_n includes the all probable mutually exclusive results of a specific domain of problem. Now imagine an incident with a frame of discernment indicating two probable outcomes.

$$\Theta = \{A, N\} \tag{3.42}$$

the hypotheses are then defined as the power set for this problem.

$$2^{\Theta} = \{A, N, \{A|N\}, \emptyset\} \tag{3.43}$$

$\{A|N\}$ (either A or N) corresponding to uncertainty and $\emptyset$ represents empty set. In this thesis, we would like to determine if the considered network traffic is normal or malicious. Therefore, the frame of discernment consists of $N = Normal$ and $A = Attack$.

A belief value assignment to the each hypothesis is done within the range [0, 1]. This process is known to be Basic Probability Assignment. By using PMF m, that denotes the evidence extracted directly from the hypothesis and the notation is:

$$m : 2^{\Theta} \rightarrow [0, 1] \tag{3.44}$$

is called Belief Function if

$$m(\emptyset) = 0 \tag{3.45}$$

$$m(H) \geq 0 \quad where \quad \forall H \subseteq \Theta \tag{3.46}$$

$$\sum_{H \subseteq 2^{\Theta}} m(H) = 1 \tag{3.47}$$

are satisfied. Note that Equation (3) corresponds to null space which must be an empty set.

Belief function for Y is defined to be:

$$bel(Y) = \sum_{X|X \subseteq Y} m(X) \tag{3.48}$$

Plausibility function for Y is defined to be:

$$pl(Y) = \sum_{X|X\cap Y\neq\emptyset} m(X) \tag{3.49}$$

Note that $Belief \leq Plausibility$ is always the case. And in this study, we use only belief function as it is done by [71] since it has accurate results for our computations.

Table 3.1. An example of source probabilities to be combined with DST

	pmf (Source 1)	pmf (Source 2)
Neither	0	0
Normal	0.85	0.85
Attack	0.05	0.1
Either(Uncertainty)	0.1	0.05

In this case, there is a little ignorance since uncertainty for two sources are relatively low(0.1 and 0.05). Also, there is little conflict because two sources have almost the same opinions about the incident.

3.5.5. Dempster's Rule of Combination

Now, in order to combine different belief we use the Dempster's rule of combination. It basically makes calculation of the orthogonal sum , thereby fusion of information is realized by obtaining a single belief. The formulation is given in

$$\begin{aligned} m(H) &= \frac{\sum_{X\cap Y=H} m_1(X)m_2(Y)}{1-\sum_{X\cap Y=\emptyset} m_1(X)m_2(Y)} \quad \forall H\neq\emptyset \\ &= \frac{\sum_{X\cap Y=H} m_1(X)m_2(Y)}{1-K} \quad \forall H\neq\emptyset \end{aligned} \tag{3.50}$$

where

$$K = \sum_{X \cap Y = \emptyset} m_1(X) m_2(Y) \quad \forall H \neq \emptyset \tag{3.51}$$

in which $m_1(H)$ and $m_2(H)$ denote the beliefs of sources 1 and 2 in the hypothesis H.

Note that K is a measure of the amount of conflict between the two mass sets. In a similar way, $X \cap Y = H$ denotes each evidence combination that yield H and $X \cap Y = \emptyset$ indicates the hypothesis H's mutually exclusive subsets. Thus, intersection of X and Y is an empty set.

With Dempster's rule one can only combine evidences from two sources at a time. Dempster's rule can be used in an iterative manner to combine more evidences. This is realized by placing new evidence and output of the initial combination process into another fusion process. One can easily observe that associative property is satisfied by Dempster's rule. Hence, the final combined belief values is not affected by the order where the belief values are fused. We can move on with more sources.

If we ignore the attachment of probability to the entire frame of discernment, the two methods(Dempster's rule of combination and Bayes' MAP rule) produce the same results when there is no ignorance [85].

Nonetheles, there are certain issues associated with DST that need to be presented. Specifically, the creation of the BPA values is of fundamental importance. Even though DST is sensitive to the BPA, one does not have to use a particular methodology to derive the belief values. Because of this fact, a few of the previously offered methods may robustly and dynamically produce the BPA values in order to be deployed at the time of information fusion process of IPS. Thus, this problem is continuing challenge to be addressed by the opponents of DST. In addition, there are other disadvantages related to DST. The computational complexity raises with exponential magnitude if possible number of event outcomes increases. For example, suppose the system has n

possible outcomes, it has up to $2^n - 1$ hypothesis for analysis. With regards to DDoS Attack Detection framework, the system has two probable outcomes; a frame can be malicious or normal. Hence, the algorithm's computational complexity with DST is expected to be low which can be disregarded. Another major drawback is the conflicting beliefs management, that is known to researchers as problematic situation. An example from [51] indicates the conflicting belief phenomenon very well.

3.6. Advantages of Dempster-Shafer Theory

The use of probability boxes and DST structures in statistical analyzes presents various important benefits over a past probabilistic approaches [51]. They provide comprehensive and convenient methods to alleviate some of the most practical serious problems faced by scientists, such as:

(i) Non-negligible measurement uncertainty,
(ii) Poorly known or even unknown dependencies,
(iii) Imprecisely specified distributions,
(iv) Small sample size,
(v) Model uncertainty
(vi) Non-stationary of distributions
(vii) Inconsistency in the quality of input data,
(viii) Non-detects or other censoring in measurements, .

While representing theory of uncertainty, a significant problem is encountered by [48] when using these tools regarding the kind of knowledge available in input of information fusion implementation. Indeed, the denotation of information absence is egregiously used in the probability theory. In fact, conditional and prior probabilities have to be determined and put into probabilistic model. This obligation generally result in use of a symmetry argument (minimax error) assigning prior probabilities to random variables consider having 0.5 probabilities for binary decision in which no data is given about which is more likely). Nevertheless, any information included in the

missing conditionals and priors is not used in the DST if it can be acquired directly — and then probably it is available for computation via Bayes equations [48].

A primary advantage of DST over theory of probability is therefore permitting us to determine a grade of ignorance in a frame rather being forced to find prior probabilities. By using this capability, one can explicitly assign an ignorance degree. Consequently, DST becomes very appealing due to advantages it presents to researchers.

The other advantage is that DST does not require a priori probability or a priori knowledge on the probable domain states as Bayesian. In some unknown and vague scenarios, this property is very useful [86]. More importantly, in duties of IPS, DST is suitable to detect previously unseen attacks since it doesn't need a priori knowledge. The reasons for using DST in this thesis is also observed in [71] on the results where the author makes a comparison of various information fusion approaches and reaches a conclusion stating DST is more promising than Bayesian.

3.7. The Proposed Solution

Nowadays, in network anomaly detection domain, an increase in use of information theory based detection metrics is observed. The main benefit of using information theory-based metrics can be stated as (a) time and space complexity is small as only header information is used for calculation, (b) they can easily characterize the different kinds of network traffic using few packet header features , (c) high sensitivity , (d) low false positive rate, and (e)high scalability [41]. In order to analyze a sample of time interval T with n total samples/time window, it takes $O(T_n)$ time for the information theory-based metrics. It means even if we perform multi-variate analysis which means analyzing different packet features at the same time, it does not have an impact on overall time complexity of the information theory-based detection systems.

Network anomalies have some attributes that identify them e.g. unique source IP addresses and single Destination IP addresses . Entropy is a useful measure to detect these uncertainties. The block diagram in Figure 1 shows the flow of the used work.

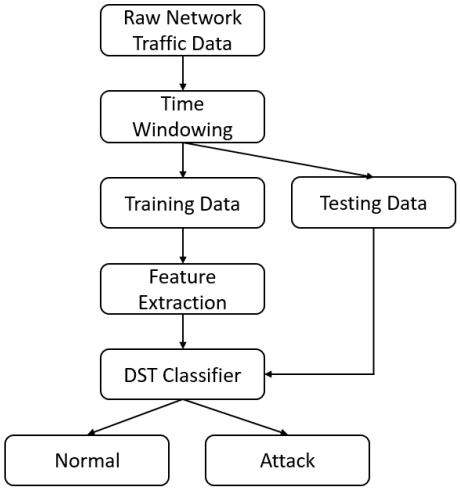

Figure 3.1. Block Diagram of Proposed Solution

Feature extraction is realized after converting data to 50 ms time windows and then determining the distributions of frame header which contains Time, Frame Length, Source IP Address, Destination IP Address, Source Port, Destination Port and Protocol. An example of frames given in the figure below.

	Time	frame_	src_ip	dst_ip	sourcel	destina	transpc
2	0	68	92.45.54.1	193.140.20	8908	6500	UDP
3	0.000218	900	193.140.19	31.13.84.8	49218	443	TLSv1.2
4	0.000233	171	31.13.84.8	192.168.66	443	40991	TLSv1.2
5	0.000235	1500	192.168.68	54.225.245	54602	443	TLSv1.2
6	0.000466	126	192.168.79	64.15.113.	55251	443	TCP
7	0.000859	126	192.168.79	64.15.113.	55251	443	TCP
8	0.001107	2906	74.125.133	193.140.19	443	50936	TLSv1.2
9	0.001402	70	193.140.19	74.125.133	50936	443	TCP
10	0.001407	1396	193.140.19	216.58.208	55606	443	UDP
11	0.001482	1396	193.140.19	216.58.208	55606	443	UDP
12	0.001488	1396	193.140.19	216.58.208	55606	443	UDP
13	0.00149	1396	193.140.19	216.58.208	55606	443	UDP
14	0.001492	1396	193.140.19	216.58.208	55606	443	UDP
15	0.001503	78	92.45.54.1	193.140.20	7816	6430	UDP

Figure 3.2. Sample frames from UDP-2 flood data

After having probability distribution of each header element, the corresponding entropy values are calculated. For example, first 50 ms time window of UDP2 flood BOUN dataset contains 597 frames and 88 different Source IP Addresses. Each Source

IP Address' probability is obtained and entropy is computed according to Shannon's formula:

$$H(x) = -\sum_{x} P(x)\log_2(P(x)) \tag{3.52}$$

where

$$P(x) = \frac{\#\quad of\quad times\quad x\quad appears\quad in\quad Window}{Window\quad Size} \tag{3.53}$$

At the beginning, in order to make an assignment of mass values to domain set, BPAs are determined. After that, obtained mass values are used to compute hypothesis beliefs. Then, the combination of the beliefs are done according to Dempster's rule of combination in order to see the overview belief for the domain sets, as indicated in Equation 3.11.

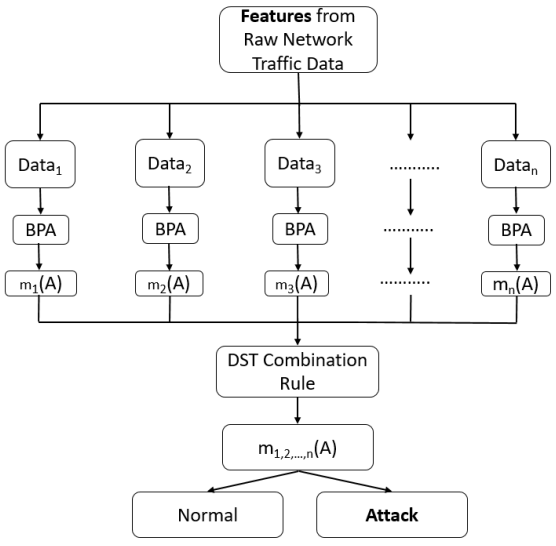

Figure 3.3. Classification Approach with Evidence Theory

3.7.1. Basic Probability Assignment

DST is previously used in the anomaly detection studies to order to enhance the accuracy rate. In [63], [49], [50], the researchers prove DST as an efficient and powerful approach to be implemented in IPSs. Nonetheless, a significant challenge to be studied stays open in DST. That is th employment of a self-adaptive and unsupervised BPA process.

The BPA assignment is of fundamental importance to the success of DST [67]. A crucial problem to implement DST in IPS is to automatically specify the BPA values which must be built from the current attribute of the network traffic [65]. Probability assignments can be made to domain set for anomaly detection by data mining methods as well as empirical approaches [71]. Nonetheless, none of the indicated methods look for a way to adapt an unsupervised process of BPA, and some of the methods may be used fine tuning period or without a previous training. IN this section, we address the necessity of an automatic BPA approach by employing a simple formula from [47] to obtain appropriate belief values and able to change its capabilities to the current attribute of the network, not requring an action from security administrator. implementation in real time is possible since the proposed methodology has low computational complexity.

3.7.1.1. Method to Assign Belief in Attack:. As we can observe from the Figure 3 and 4, it is easy to see the jumps in the attack region in both features. Therefore, a simple but efficient and self-adaptive BPA is used. Assigning 10% uncertainty we have:

$$m(A) = \frac{1}{1 + e^{(threshold - dist)}} \tag{3.54}$$

$$m(A \quad or \quad N) = 0.1 \tag{3.55}$$

$$m(N) = 0.9 - m(A) \tag{3.56}$$

Observe that the equations 3.15,3.16 and 3.17 adds up to 1. This ensures Dempster's rule of combination. Belief to normal is assigned 0.01 probability to avoid conflicting belief phenomenon as it was shown in [82].

In this thesis, the threshold is determined by making histogram of the entropy distances. We know that the $\frac{3.49}{84.7} \cong 4\%$ of the dataset is with attack traffic. Thus, we look at the lower and upper 4% of the data with wider dispersion. When we work on the $feature_1$, we calculate the threshold as 1.85 and for $feature_2$ 0.80.

The BPA values are assigned to each data item, for two feature based on that feature value (entropy distance). Dempster's Rule Combination are applied on mass values and overall mass values of the hypothesis normal and of the hypothesis attack are computed. If the mass value of the 'normal' hypothesis is less than the mass value of the 'attack' hypothesis, then it is classified as attack; otherwise as normal.

3.7.2. Support Vector Machines

The input of the detection part is the distance values which are constructed in the previous section. Each of them can be used for detecting a different kind of network anomalies. Applying a static threshold or using some statistical models could find anomalies, but it is hard to adjust thresholds for a different type of network models. Also, it may result in higher false alarms. In order to increase the detection rate,

decrease the false alarm rate and also make the system more adaptable for a different type of anomalies and network models, SVM is used in the detection part in order to benchmark with other classification approaches [87]. Support Vector Machine is a supervised classification algorithm. It uses class labels in the training phase. It separates data into its classes using hyperplane. It is the maximum separable line between the different classes of data.

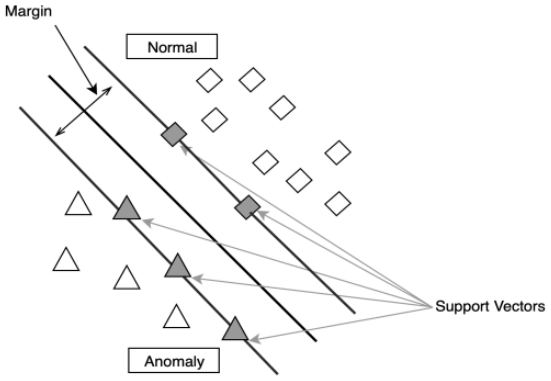

Figure 3.4. The Hyper-Plane of SVM

The hyperplane of the SVM can be divided into two classes given in Figure 3.4.

4. DESCRIPTION OF DATASETS

There are various study in literature to protect and defend against DDoS attacks the information systems. In the existing dataset, legitimate and attack traffics are basically mixed, and then researches are done to find differences and similarities. They are beneficial to observe the different sides of normal and attack traffic. Nonetheless, they do not offer a realistic frame.

Stealthy DDoS attack are generated with attacks packet with low rate which are formed in a sophisticated way in order to by-pass IPSs. This kind of attacks is hard to spot since it sends less number of malicious packets, as compared to legitimate traffic [88]. Since we would like to study NIDS datasets, not victim or host based datasets, we need a detection scheme for attack by monitoring the whole network. It would be easy to detect the attack if just one machine is monitored, because there is a sudden spike in traffic. If the entire network is observed, there will be only small amount of change which will look like a random moves. Liu points out that detection of an attack with packet number that has no difference than normal period is a challenging task [89].

Rahmani *et al.* asserts that it is really hard finding a real DDoS dataset. He uses a dataset with content that just attack traffic is present and it is synthetic which is injected into normal traffic. It is noted that studying with dataset including solely attack traffic, is not useful for compact detection of DDoS problem [44].

A major challenge in DDoS research is to acquire a dataset which is voiced by most of the DDoS attack researchers. Instead, they improvise synthetically fabricated datasets or their network's traffc and they do not make the dataset public because of the privacy concerns. Consequently, we, as DDoS researchers, are not able to make comparison of our results with new works.

4.1. BOUN Dataset

In BOUN dataset, the attack is kept on a victim or host server within the campus. The dataset is not taken from the victim instead from the one major switch of the campus, thereby including very diverse kinds of network traffic along with attack traffic. By collecting traffic on a main switch, the dataset is made to be suitable for NIDS researches. UDP flood and TCP-SYN flood attack scenarios are included in this dataset. The flood attack topology of the can be seen in Figure 4.1. The data trace is collected in between backbone switch and LAN switch, which has over 4000 host connection and having an average of 19000 packet/second rate. The attack is launched from EE department in KB building in North Campus. IP spoofing is done and the destination port was 80. All the attack scenarios last for 484 seconds. We observe 80 seconds waiting period, and then 20 seconds attack period in each of them. Different packet ratios are used in each attack period.

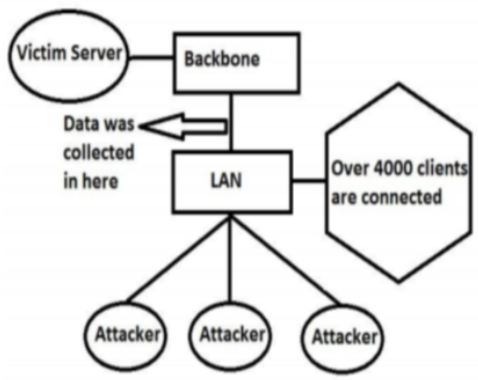

Figure 4.1. Topology of Bogazici University DDoS attack dataset [3]

There are three different attack scenarios in this dataset. In each of them, 80 second waiting period, then 20 seconds attack period is applied. Each of them lasted 8 minutes. The first attack is the UDP Flood attack with the name of UDP1 attack. Packet rates are 1000,1500,2000 and 2500 packets/second. The other attack is the TCP SYN Flood attack. Packet rates of attacks are 1000,1500,2000 and 2500 packets/second. In addition, the final one is a UDP flood attack with name of UDP2 flood. Higher packet rates compared to previous one are used which are 2000,3000,4000 and

5000 packets/second.

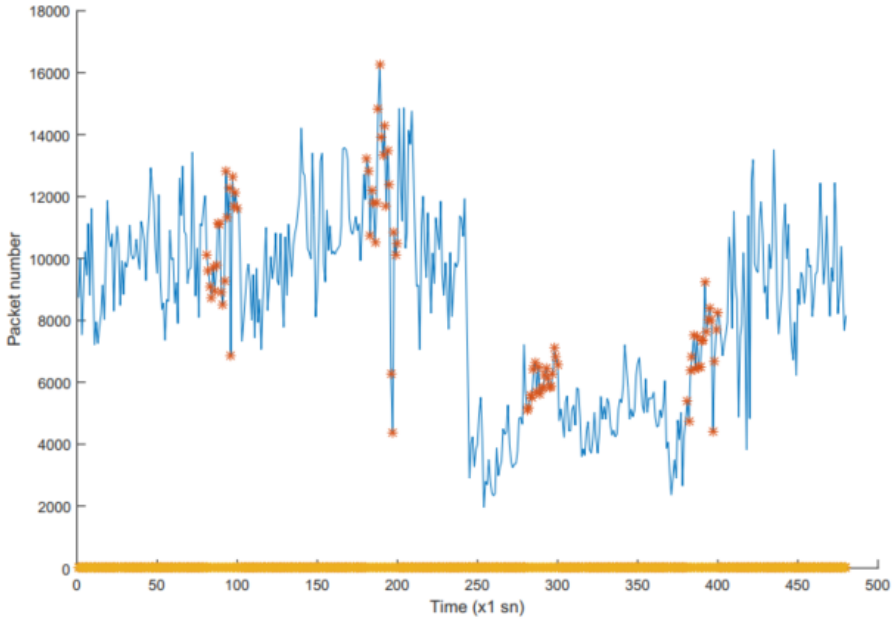

Figure 4.2. Bogazici University UDP1 flood packets/second (stars mean attacked traffic)

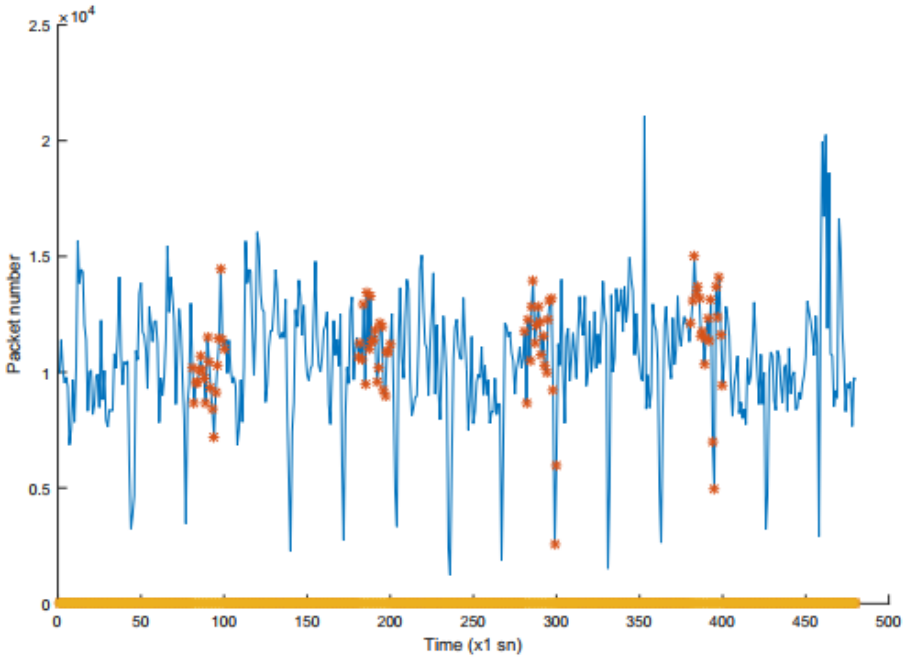

Figure 4.3. Bogazici University TCP SYN flood packets/second (stars mean attacked traffic)

4.2. CAIDA Dataset

In CAIDA, several different types of data at topologically and geographically different locations are collected , and made this data reachable to the scientific community to the extent probable by protecting the privacy of people and enterprises which donate their network access or data. Its attack and normal dataset are separate which makes it suitable to work.

There are lots of different datasets in literature for data analysts to test their proposed methods on network security platform. CAIDA is a platform on network security for collection and sharing for Internet traffic data [29]. CAIDA 2007 is attack traffic whereas CAIDA 2008 is normal traffic. CAIDA traffic is TCP based and almost 80% of the traffic on Internet uses TCP transport layer 4 protocol [30].

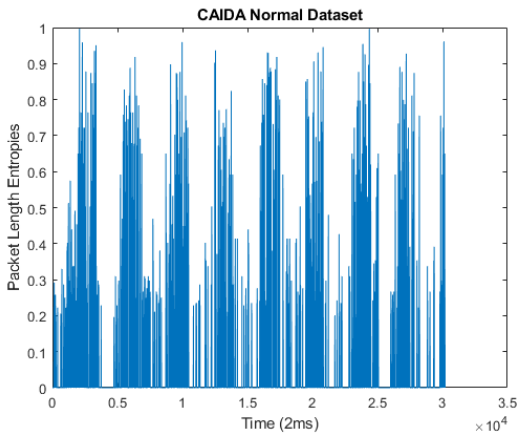

Figure 4.4. CAIDA Normal Dataset Frame Length Entropies

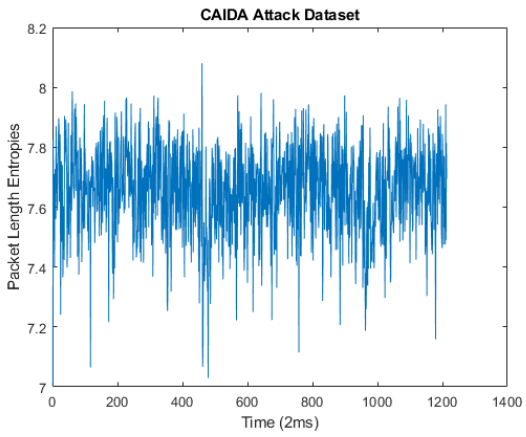

Figure 4.5. CAIDA Attack Dataset Frame Length Entropies

While working with CAIDA we used 2ms time windows since the data is very dense compared to BOUN datasets. Because there is so much data in CAIDA, we only selected 62.83 second first 60ms being normal traffic and the remaning seconds being attack traffic.

5. FEATURES, RESULTS AND ANALYSIS

5.1. Feature Extraction

Features are extracted with the method described in Section 3.7. As a result, we have a total of ten features to work with since there is five header element to calculate distance among each two of them.

5.1.1. Entropy Distance between Source IP and Destination IP Addresses - $feature_1$

In order to extract how data is distributed along different values, entropy is a useful metric to work. When DDoS attack takes places in a network traffic, the entropy distance between Source IP adresses and Destination IP addresses increases because they come from distributed sources to single destination. With the assumption entropy distance should increase during a DDoS attack [20]. Figure x.x in the Results section is an example of entropy distance between Source IP and Destination IP addresses.

5.1.2. Entropy Distance between Source Port and Destination Port Numbers - $feature_2$

Normally, the network traffic contains frequently used port numbers such as 80 for HTTP, 443 for HTTPS, 25 for SMTP and 22 for SSH. However, there are so many unfamiliar source port numbers during the DDoS attack. On the other hand, destination port numbers do not differ significantly during the attack. Therefore, measuring the distance between source port and destination port numbers gives promising results.

5.1.3. Entropy Distance between Source IP Address and Destination Port Number - $feature_3$

Again with the same motivations, we know that we have varying packets with Source IP addresses and more stable destination port numbers during attack. Thus, entropy distance between source IP addresses and Destination Port numbers can be good measure for DDoS attack indication.

5.1.4. Entropy Distance between Source Port Number and Destination IP Address - $feature_4$

We have various source port numbers during an attack in some cases they even come in sorted numbers from 1 to 65535. Hence, it would be a good idea to look for a major distance in DDoS attack.

5.1.5. Entropy Distance between Source IP Address and Source Port Number - $feature_5$

At the time of DDoS attack, Source IP address and Source Port number distance can be an indicator of an attack.

5.1.6. Entropy Distance between Destination IP Address and Destination Port Number - $feature_6$

Again during the attack, this value can present useful data.

5.1.7. Entropy Distance between Source IP Address and Packet Length - $feature_7$

Packet length is a useful information in packet header since it indicates how much data stored in packet. The size of a packet can be 64 bytes minimum. If it is shorter than the 64 bytes, they are padded to make them 64 bytes. Generally, attack packets

are observed to be 64 bytes length which is easy to construct. Thus, it would be a good idea to closely check on frame length. Entropy of frame length is expected to be low in attacked traffic. The entropy distance between source IP and frame length is selected as feature.

5.1.8. Entropy Distance between Source Port and Packet Length - $feature_8$

Source Port differs at the time of attack and packet length becomes 64 bytes length. Thus, we can choose the distance between them as a feature.

5.1.9. Entropy Distance between Destination IP Address and Packet Length - $feature_9$

Intuitively, their entropy values correlatively fluctuates along the time. We extract the distance between them as feature.

5.1.10. Entropy Distance between Destination Port and Packet Length - $feature_{10}$

Destination port are generally 80 which means HTTP traffic at the attack traffic and packet length is equal to 64 bytes which shortest possible length. The distance between them is chosen as feature.

5.2. Results

5.2.1. Simulations with BOUN TCP Flood Dataset

In TCP-SYN flood attack, three way handshake process is exploited. It always begins with an empty SYN packet sent to the server by client which makes this first step. In this starting packet, a request is made to the server to allocate resource to client so that a connection can start. Then in the second step, the server sends back a SYN/ACK packet after receiving this packet client replies with ACK packet in the

third step. The trick is that an attacker opens up a connection request with SYN packet up until the server gets busy with requests and can no longer answer with SYN/ACK packet. Now, legitimate clients cannot establish a connection with the server. So, server becomes unavailable.

This dataset contains relatively high rate DDoS attack. Again, 80 second normal network traffic, then 20 seconds of attack period present. This goes on a cycle with total of 8 minutes duration. Packet rates are 1000,1500,2000 and 2500 packets/second in each period. These rates lower compared to UDP-2 flood dataset which are 2000,3000,4000 and 5000 packets/second.

5.2.1.1. Training Dataset. The first 1688 windows are used as 70 windows with attack and 1618 windows with normal traffic in the training dataset. To determine the accuracy of results, we compare the results on the following criteria:

$$classification \quad accuracy = \frac{\#\quad of\quad correctly\quad classified\quad items}{Total\quad \#\quad of\quad items} \tag{5.1}$$

As it was previously shown, entropy distance between these different header elements is a good indicator of anomalies. Thus, we simply computed the difference between header entropy values. From the first 84.36 second of the TCP flood data, the entropy distance between Source IP and Destination IP Addresses' is given in Figure 5.1.

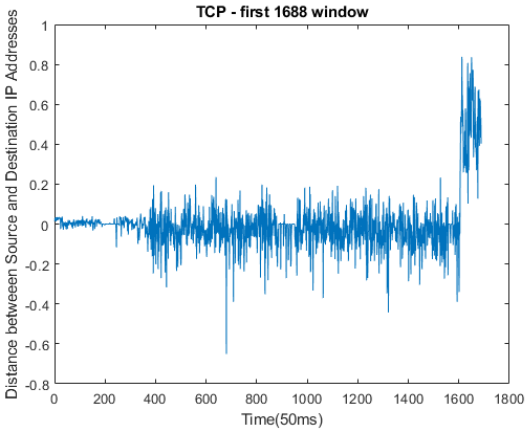

Figure 5.1. Distance between Source IP and Destination IP Addresses($feature_1$)

Since TCP-SYN flood attacks make up 90% of all connection type, we decide to use it as leading example in our study. The first 1688 windows are used as training vectors, as it showed in the Figure 5.1. These 1688 windows are also used for SVM and NB classifier. After having threshold parameters, we go to the whole data which has 9653 window and test our result with the trained classifier.

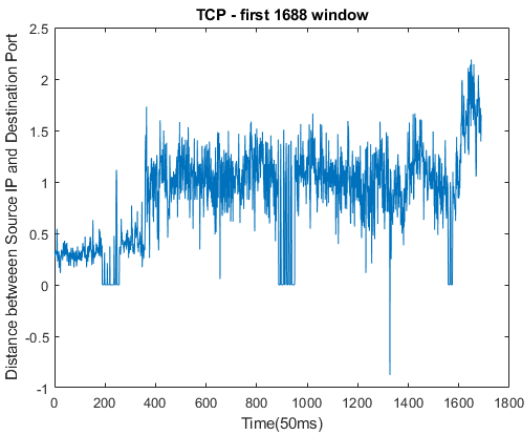

Figure 5.2. Distance between Source and Destination Ports($feature_3$)

This is another feature that is efficient to use in detecting DDoS attacks. The process for defining threshold for the DST classifier, we follow the next steps.

After having entropy distribution of all of the training data attack and normal inclueded. We extract the histogram of the distribution. The top 5% and the lowest

5% are compared. We use the one with more dispersion. By using this method, we make our learning unsupervised. In this case, the top 5% has more dispersion. The lowest value of the 5% is selected as threshold and BPA values are computed according to this value with Equation 3.15.

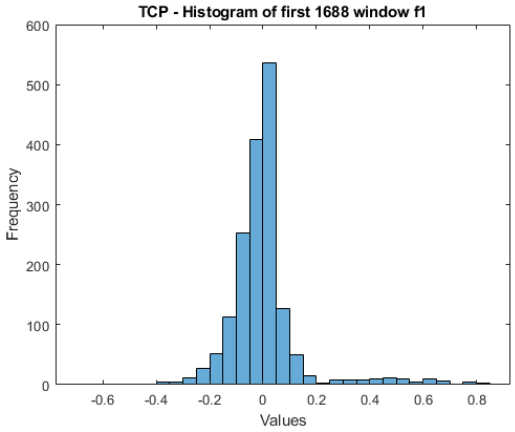

Figure 5.3. The histogram of all(attack and normal) distance values between Source IP and Destination IP Addresses(to find threshold for $feature_1$)

As you can see above, we have more dispersion on the highest 5% of the training data. So the value of 0,2 will be our threshold value for the feature 1 in DST in TCP SYN flood data. The computations of the BPAs will be done according to this threshold with Equation 3.54.

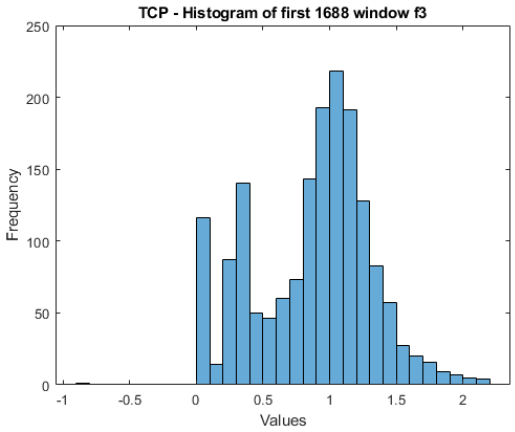

Figure 5.4. The histogram of distance values between Source and Destination Ports(to find threshold for $feature_3$)

For each feature the same training is done and by using this training set we build the classifiers for DST, SVM and NB. Results for three classifier and four different dataset are given at the end of this section.

5.2.1.2. Testing Dataset. In TCP/SYN flood attack, attackers start TCP connection with SYN request yet they do not complete the three way handshake process and server waits for the connection to complete. So the detection of these attacks is of more important than UDP connection. And a close watch of Source IP difference gives important results as it is given below.

We used whole dataset to test results. We have total of time 9653 windows in TCP dataset. For each feature we have the figures below.

DDoS assaults may impact various network features. These values fluctuates on different kinds of DDoS attacks. Hence, analysis of the differences in header parameters will be done

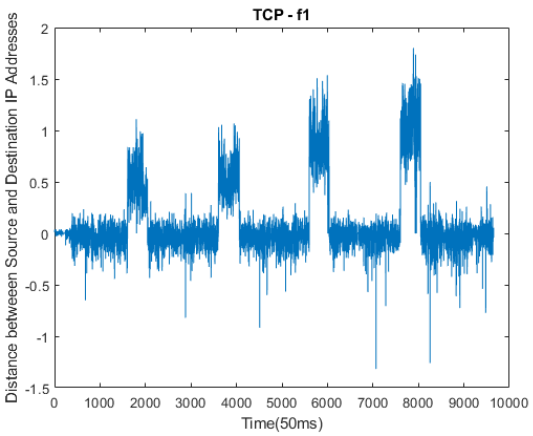

Figure 5.5. Distance between Source IP and Destination IP Addresses ($feature_1$)

When a TCP/SYN flood attack occur, attackers generally use randomly selected group of spoof IP adresses. At TCP/SYN flood attacks a packets destination IP is generally the target victim's IP. So, these fake source IP adresses send their spoofed packets to victim. We see lots of this kind of packet during SYN flood repeatedly. As a result, entropy of destination IP of network traffic data is expected diminish during an

attack. So the difference between source IP and destination IP addresses is expected to spike at the time of attacks which is exactly the case on Figure 5.5.

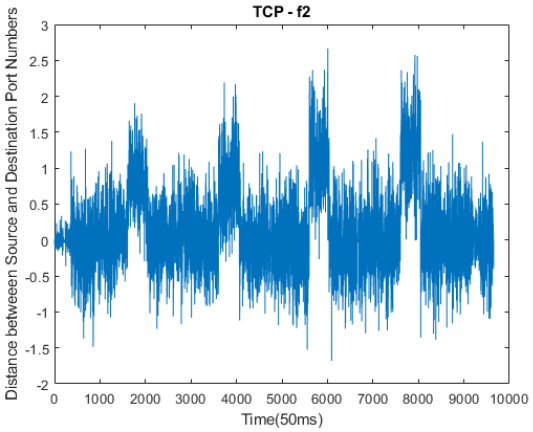

Figure 5.6. Distance between Source Port and Destination Port Numbers ($feature_2$)

In general case we have less information from source ports. However, in our BOUN dataset we have varying source port actually their sorted from 1 to 65536 which is basically the all ports possible. In fact, an expert system could easily demolish this attack scenario without requiring anomaly detection methods because rare ports are used as source ports. On the other hand, we have destination port statistics. Attackers may sometime are smart enough to change their victim IP's destination port so they are not noticed by expert system. However, in this case we do not see that.

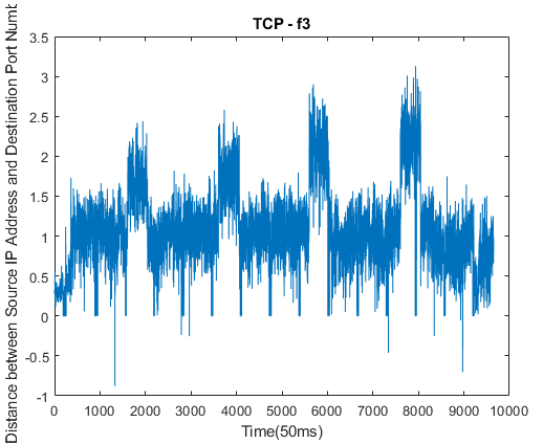

Figure 5.7. Distance between Source IP Addresses and Destination Port Numbers ($feature_3$)

Since source IP adresses varies during an attack, and destiantion port variation decreases, inspecting entropy distance between these two header element is beneficial. This is found to be most second accurate feature for DDoS attack detection.

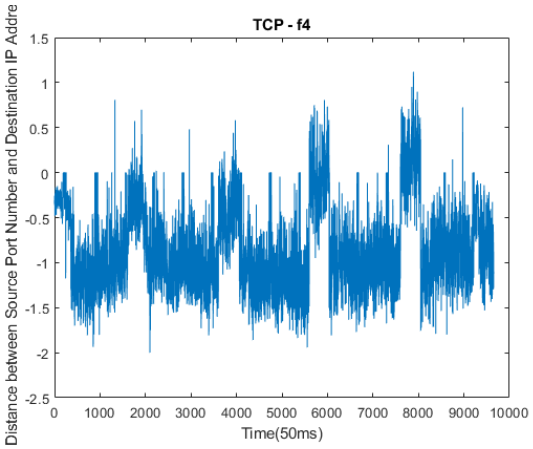

Figure 5.8. Distance between Source Port Numbers and Destination IP Addresses ($feature_4$)

Variation of source Port numbers increase during an attack, however, we have lots of packet along with attack packet and their destination port numbers are generally the most known port numbers such as 80, 443, 22, 25 and so on. So the statistic of other packets in network trace data disturbs this.

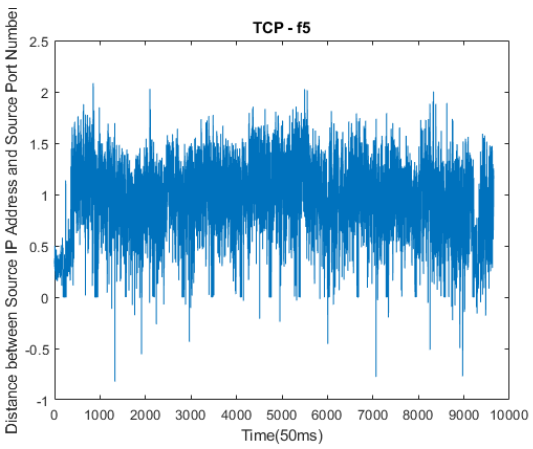

Figure 5.9. Distance between Source IP Addresses and Source Port Numbers ($feature_5$)

We do not see a spike at the times of attack because source IP addresses and source port adresses differentiate increasingly during an attack,.

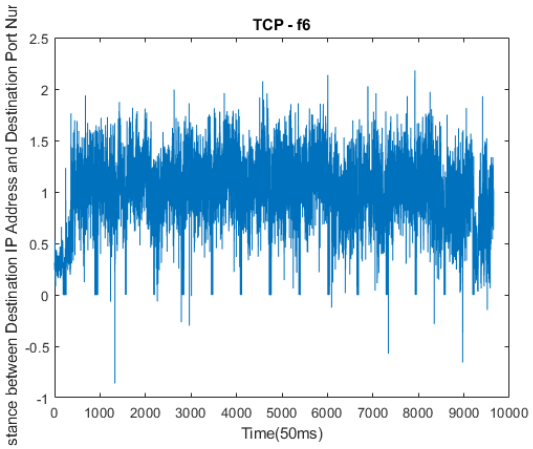

Figure 5.10. Distance between Destination IP Addresses and Destination Port Numbers ($feature_6$)

Since destination IP addresses and destination port adresses vary less during an attack, we do not see a spike at the times of attack.

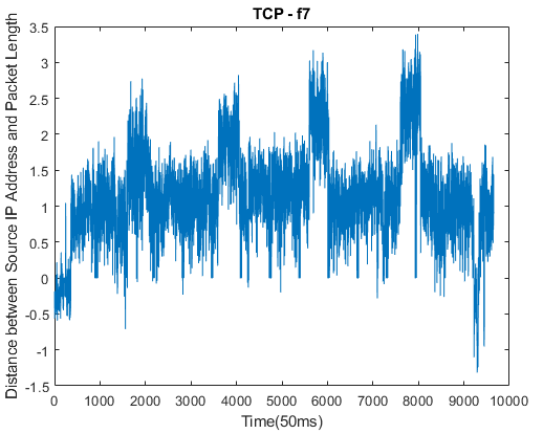

Figure 5.11. Distance between Source IP Addresses and Packet Length ($feature_7$)

From our previous knowledge we know that when a TCP/SYN flood occurs we ought to see so many packets with the same frame length. So, analysis of frame lengths is considered to be beneficial. We expect packets with 64 length packets because attackers do not vary their packet length. We cannot observe this clearly here because there are also other legitimate packets in network that carries 64 byte length packets

which is the smallest possible frame length for network packet by definition. Thus we cannot see the distinction in feature 8, feature 9 and feature 10.

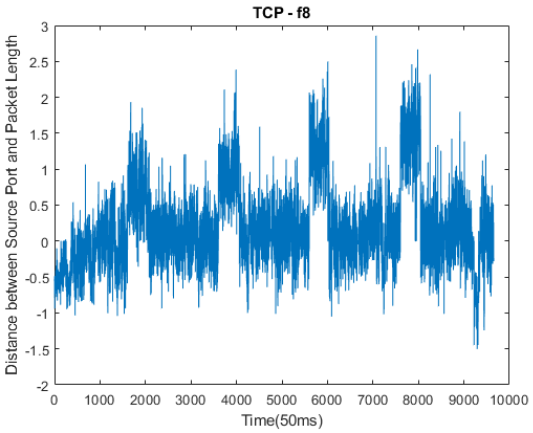

Figure 5.12. Distance between Source Port Numbers and Packet Length ($feature_8$)

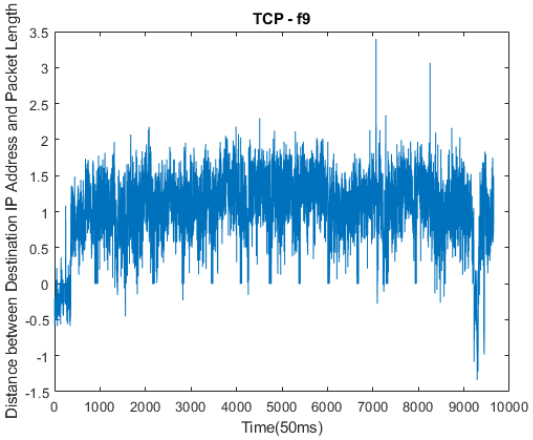

Figure 5.13. Distance between Destination IP Addresses and Packet Length ($feature_9$)

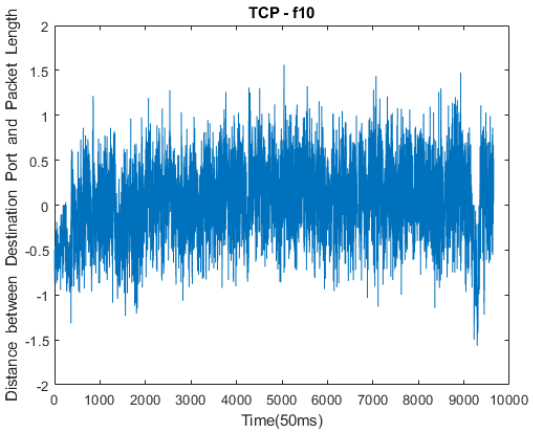

Figure 5.14. Distance between Destination Port and Packet Length ($feature_{10}$)

5.2.1.3. TCP Results with DST Classification. Feature1 and feature3 are good incdicator of an attack since attack vectors are in the same areas, we can combine feature1 and feature3. One can observe that attack and normal traffic are separated with very high accuracy as can be seen in Figure 5.15. As it is observed, results are enhanced when the two are combined. As you can see on the next figure, attacks are segregated when plotting features on two dimension. Blue dots represent the attack cases, red ones denote normal cases. Only a few blue dot is in the red cluster as they are true negatives.

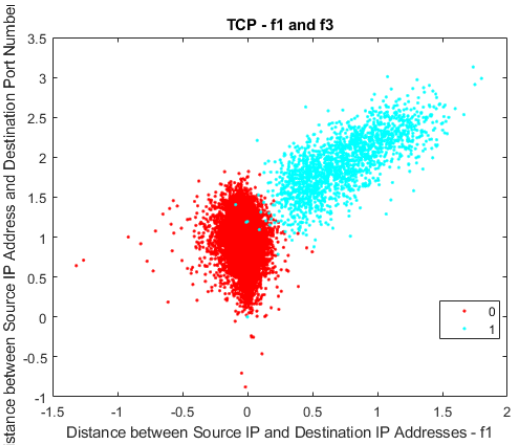

Figure 5.15. TCP - ($feature_1$) vs ($feature_3$)

When only $feature_1$ is used, we have 95.55% TPR and 0.44% FPR which result in 98.82% accuracy rate. False positives are 577^{th}, 594^{th}, 870^{th}, 1246^{th}, 1299^{th}, 1306^{th}, 1330^{th}, 1393^{rd}, 1399^{th} and 1418^{th} from the training set. True negatives are 1637^{th}, 1657^{th}, 1658^{th}, 1669^{th}, 1670^{th}, 1671^{th}, 1675^{th}, 1676^{th} and 1677^{th} from the training set. The statistics of the result are given in the following table:

Table 5.1. Confusion Matrix for TCP classification with DST with only $Feature_1$

DST		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1676**	78	1754
	Negative	**35**	7808	7899
	Total	1711	7886	9653

Table 5.2. Confusion Matrix for the classification with DST with only $Feature_3$

DST		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1707**	47	1754
	Negative	**154**	7745	7899
	Total	1861	7792	9653

When only $feature_3$ is used, we have 97.32% TPR and 1.95% FPR which results in 97.91% accuracy. 685^{th}, 1016^{th}, 1063^{th}, 1071^{th}, 1082^{th}, 1143^{th}, 1149^{th} and 1215^{th} windows are false positives from the training set. 1619^{th}, 1622^{nd}, 1623^{rd}, 1624^{th}, 1631^{st}, 1638^{th}, 1651^{st}, 1658^{th}, 1659^{th}, 1662^{nd}, 1670^{th}, 1671^{st}, 1672^{nd}, 1676^{th}, 1677^{th}, 1678^{th}, 1679^{th} and 1687^{nd} windows are the true negatives from the training set. The statistics of the result are given in the table:

Table 5.3. Confusion Matrix for TCP classification with DST with the combination of $Feature_1$ and $Feature_3$

DST		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1679**	75	1754
	Negative	**12**	7887	7899
	Total	1691	7962	9653

When combination of $feature_1$ and $feature_3$ is used, we have 95.72% TPR and 0.15% FPR which result in 99.10% accuracy rate. 1661^{st}, 1670^{th}, 1671^{st}, 1677^{th} and 1678^{th} windows are true negatives that are not correctly classified. This is because of the attack traffic rate is relatively low at these windows. 594^{th}, 659^{th}, 684^{th}, 964^{th}, 1126^{th}, 1214^{th}, 1246^{th}, 1306^{th}, 1330^{th}, 1398^{th} and 1418^{th} windows are the false positives from the training set. The statistics of the enhanced result are given in the following table:

As a result, True Positive Rate increases when combination is done. Also, we also have increasing False Positive rate. However, we have better accuracy rate overall.

The hypothesis that combination of features using DST improves accuracy is proven to be correct for the BOUN TCP dataset. Furthermore, another assumption is also proven to be correct, i.e. a few badly selected features do not negatively affect the results, given that most chosen features are suitable. These two characteristics make DST very effective for solving real world IDS problems.

5.2.1.4. TCP Results with Naive Bayes Classification. By using the 1688 windows in Figures 5.1 and 5.2, we train NB classifier for TCP data. After having classifiers, we apply them to entire 9653 windows in dataset. We have better results as it is explained below. This process is realized using built-in functions in MATLAB2019a software.

Since supervised learning is involved in Bayesian approach, we have higher results 99.26% TPR and 0.24% FPR which result in 99.67% accuracy. 1657^{th}, 1670^{st}, 1675^{th} and 1677^{th} are true negatives which is not classified as attack with Naive Bayes Classifier. 684^{th}, 1015^{th}, 1070^{th} and 1214^{th} windows are the false positives.

Table 5.4. Confusion Matrix for TCP classification with NB using $Feature_1$ and $Feature_3$

NB		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1741**	13	1754
	Negative	**19**	7880	7899
	Total	1760	7893	9653

5.2.1.5. TCP Results with SVM Classification. By using the 1688 windows in Figures 5.1 and 5.2, we train SVM classifier for TCP data. After having classifiers, we apply them to entire 9653 windows in dataset. We have outstanding results as it is explained below. This process is realized using built-in functions in MATLAB2019a software.

For binary classification SVM has high accuracy. The results confirms the high classification rate. However, we have less TPR than NB which is surprising.

Table 5.5. Confusion Matrix for TCP classification with SVM using $Feature_1$ and $Feature_3$

SVM		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1735**	19	1754
	Negative	**2**	7897	7899
	Total	1737	7916	9653

We have 98.92% TPR and 0.02% FPR which result in 99.73% accuracy with SVM.

5.2.2. Simulations with BOUN UDP1 Flood Dataset

We plot the feature1 vs feature3 since they show spike during the attack cases. Again blue dots represent the attacks cases whereas red ones denote normal ones. Red and blue area overlaps because UDP1 attack is low rate and entropy is density oriented in our formula definition thus it cannot differentiate explicitly. Yet, again it is highly successful detecting attacks.

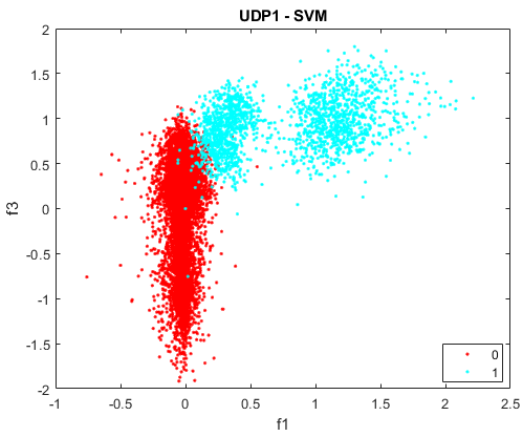

Figure 5.16. $(feature_1)$ vs $(feature_3)$ for BOUN UDP1 dataset

BOUN UDP1 dataset is low rate. Especially, first attack region with 1000 packets/second is hard to detect.

5.2.2.1. UDP1 Results with DST. : Apart from other dataset, we implemented another technique in this case. We also analyzed the effect of uncertainty on accuracy of classification. DST threshold values are computed using the same method as explained in section 5.2.1.1 with histogram figures 5.3 and 5.4. Thresholds are found to be 0.52 and 0.43. This process is realized using and coding some functions in MATLAB2019a software.

Table 5.6. Confusion Matrix for the classification with 1% uncertainty DST using $Feature_1$ and $Feature_3$

DST		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1665**	81	1746
	Negative	**726**	7142	7868
	Total	2391	7223	9614

During the work on UDP1 data, we realized that there is more uncertainty in its features since it contains LRDDoS type attack. So tuning the uncertainty value computed by Equation 3.55 may give improved results as it is proposed by [71]. Thus, we decided to compute for different uncertainty values 1% and 10% for this case.

We have 86.66% TPR and 0.98% FPR which result in 96.77% accuracy with DST.

Table 5.7. Confusion Matrix for the classification 10% with DST using $Feature_1$ and $Feature_3$

DST		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1513**	233	1746
	Negative	**77**	7791	7868
	Total	1590	8024	9614

We work with 1% uncertainty in TCP-SYN flood and UDP2 cases. However, using 1% uncertainty in UDP1 data has less accurate results compared to DST performance in TCP-SYN and UDP2. Observe that increasing the uncertainty value to 10% has better result in table 5.7. So the proposal from [71] is proven to be correct here again.

Even though the accuracy is lower than TCP and UDP2 datasets, it still performs remarkably considering other benchmark studies. So the DST is proven to be successful at detecting low rate DDoS attacks as well. Although DST is an unsupervised learning approach, its performance is very close to supervised methods NB and SVM even for low rate dataset.

5.2.2.2. UDP1 Results with NB. :

Using NB as classifier gives more accurate result than DST again. However, it cannot achieve higher or the same result as it is done for TCP-SYN or UDP2 data since less dense attack is hard for NB classification too.

Table 5.8. Confusion Matrix for UDP1 classification with NB using $Feature_1$ and $Feature_3$

NB		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1626**	120	1746
	Negative	**12**	7856	7868
	Total	1638	7976	9614

For this case of NB, we have 93.13% TPR and 0.15% FPR which result in 98.62% accuracy. It has very low FPR and high TPR which is accepted as successful for DDoS attack detection mission.

Although it has high accurate result, it is not suitable for real life applications. This is because not all the time prior or posterior conditions can be guessed. So the building classifier may not always be possible with NB.

5.2.2.3. UDP1 Results with SVM. : SVM has the best result among other methods again. Since with its training set examples, every instance is marked as pertaining to attack or normal categories, an SVM training algorithm constructs a model that puts new instances to a attack or normal, turning it to a non-probabilistic binary classification. This process is realized using built-in functions in MATLAB2019a software.

Table 5.9. Confusion Matrix for UDP1 classification with SVM using $Feature_1$ and $Feature_3$

SVM		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1642**	104	1756
	Negative	**7**	7861	7868
	Total	1649	7965	9614

We have 93.51% TPR and 0.09% FPR which result in 98.85% accuracy with NB.

Although it has high accurate result, it is not suitable for real life applications. This is because not all the time prior or posterior conditions can be guessed. So the building classifier may not always be possible with SVM.

5.2.3. Simulations with UDP2 Flood Dataset

We plot the feature1 vs feature3 since they show spike during the attack cases. Again blue dots represent the attacks cases whereas red ones denote normal ones. There is a more distinction between red and blue area because UDP2 attack is more dense compared to UDP1.

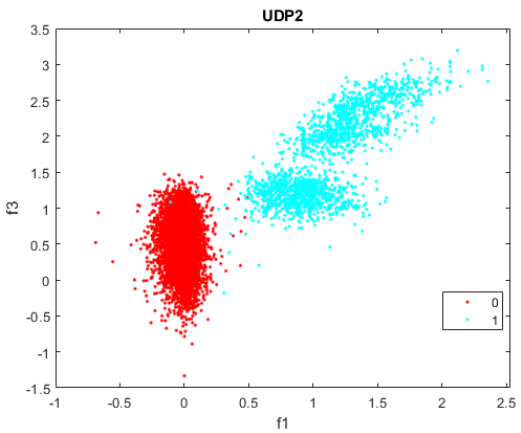

Figure 5.17. $(feature_1)$ vs $(feature_3)$ for BOUN UDP2 dataset

5.2.3.1. UDP2 Results with DST. : Because the UDP2 dataset contains more dense region of attack compared to UDP1, the detection is easier. Again, histograms for the first 1688 windows are formed as it is done in Figures 5.3 and 5.4 for the feature 1 and feature3. The threshold values are now computed to be 0.70 and 0.78 respectively for feature1 and feature3. And then, we built the classifiers. After that, we use these classifiers the to test on whole data which consists of 9625 windows. In order to avoid overload of figures we do not give their histograms here. This process is realized using and coding some functions in MATLAB2019a software.

Table 5.10. Confusion Matrix for UDP2 classification with 3% uncertainty DST using $Feature_1$ and $Feature_3$

DST		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1791**	5	1796
	Negative	**14**	7815	7829
	Total	1805	7820	9625

We have 99.72% TPR and 0.18% FPR which result in 99.80% accuracy with DST. This accuracy rate can be considered to be very successful for an unsupervised method. So, it should be studied further with other benchmark studies to see its success.

5.2.3.2. UDP2 Results with NB. : Again, NB enhances detection accuracy as it seen in Table 5.11. Since the dense attack is involved here, a better accuracy rate is observed. If the conditional probabilities were to be computed easily, NB would be the better approach for this kind of problems. However, finding the conditionals are not always possible especially in cases of realtime detection [47]. Thus, DST approach which does not require prior or posterior probabilities is more applicable to our problem. This process is realized using built in functions in MATLAB2019a software.

Table 5.11. Confusion Matrix for UDP2 classification with NB using $Feature_1$ and $Feature_3$

NB		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1793**	3	1796
	Negative	**12**	7817	7829
	Total	1805	7820	9625

We have 99.83% TPR and 0.15% FPR which result in 99.84% accuracy with NB.Even though NB has very high result, we would not be able to build its classifiers

in a real time case. Therefore, DST is more acceptable approach.

5.2.3.3. UDP2 Results with SVM. : SVM has the better accuracy than DST again. Since with its training set examples, every instance is marked as pertaining to attack or normal categories, an SVM training algorithm constructs a model that puts new instances to a attack or normal, turning it to a non-probabilistic binary classification. This process is realized using built-in functions in MATLAB2019a software.

Table 5.12. Confusion Matrix for UDP2 classification with SVM using $Feature_1$ and $Feature_3$

SVM		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1792**	4	1796
	Negative	**3**	7826	7829
	Total	1795	7830	9625

We have 99.78% TPR and 0.04% FPR which result in 99.93% accuracy with SVM. This is because SVM is very successful at binary classification and UDP2 dataset is has very dense attack region so it is convenient for detection of DDoS attacks.

5.2.4. Simulations with CAIDA Dataset

We have clear distinction between attack and normal traffic regions since the CAIDA dataset is so much dense during attack times, Again blue dots denote attacks cases on the other hand red ones represent normal traffic windows. Because there is very dense attack case here, we can completely extract attack situations.

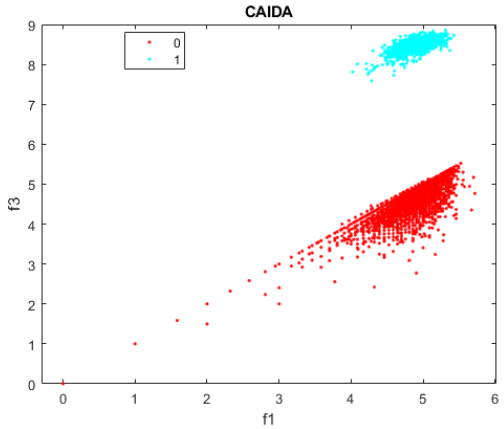

Figure 5.18. ($feature_1$) vs ($feature_3$) for CAIDA Dataset

5.2.4.1. CAIDA Results with DST. : The detection is easier. Again, histograms for the first 1688 windows are formed as it is done in Figures 5.3 and 5.4 for the feature 1 and feature3. Since the CAIDA dataset includes more dense region of attack compared to UDP1, UDP2 and TCP-SYN, it has perfect accuracy of detection. The threshold values are now computed to be 0.5 and 0.56 respectively for feature1 and feature3. And then, we built the classifiers. After that, we use these classifiers the to test on whole data which consists of 30204 windows. However, we used 2ms windowing not 50 ms. This is because of the fact that CAIDA is in nature is much more crowded than BOUN dataset. So, this windowing is found to be appropriate. In order to avoid overload of figures we do not give their histograms here. This process is realized using and coding some functions in MATLAB2019a software.

Table 5.13. Confusion Matrix for CAIDA classification with DST using $Feature_1$ and $Feature_3$

DST		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1213**	0	1213
	Negative	**0**	30204	30204
	Total	1213	30204	31417

5.2.4.2. CAIDA Results with NB. Now, one can clearly observe NB cannot make any improvement on detection accuracy since the perfect accuracy is achieved by DST above. Because the dense attack traffic is involved here, a clear distinction is seen. For this kind of problem, we can wholeheartedly use DST thanks to its perfect accuracy. This is because finding the conditionals are not always possible especially in cases of realtime detection [47]. Therefore, DST approach which does not require prior or posterior probabilities is more applicable to our problem. This process is realized using built in functions in MATLAB2019a software.

Table 5.14. Confusion Matrix for CAIDA classification with NB using $Feature_1$ and $Feature_3$

NB		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1213**	0	1213
	Negative	**0**	30204	30204
	Total	1213	30204	31417

5.2.4.3. CAIDA Results with SVM. SVM is expected to have better performance than DST again. However, perfect accuracy is already achieved by DST. We cannot analyze the effect of the learning process since a clear distinction is present between attack and normal traffic entropies. This process is realized using built-in functions in MATLAB2019a software.

Table 5.15. Confusion Matrix for CAIDA classification with SVM using $Feature_1$ and $Feature_3$

SVM		Predicted Values		
Actual Values		Positive	Negative	Total
	Positive	**1213**	0	1213
	Negative	**0**	30204	30204
	Total	1213	30204	31417

5.2.5. Comparison of DST, SVM and NB Classifiers

In order to have broad perspective on the performance of DST, NB and SVM, we would like to give them their performance on each dataset in a single table. Observe that in each case NB and SVM has higher accuracy than DST. Nonetheless, their accuracy is coming from their supervised model whereas DST employs unsupervised approach.

Table 5.16. Performance Evaluation of the Proposed Scheme(DST)

DST	True Positive Rate	False Positive Rate	**Accuracy**
BOUN UDP1	86.66%	0.98%	**96.77%**
BOUN UDP2	99.72%	0.18%	**99.80%**
BOUN TCP	95.72%	0.15%	**99.10%**
CAIDA'07&'08	100%	0%	**100%**

When a comparison of the performance with other unsupervised methods such as k-means approaches by [12], DST is very accurate. It is just slightly lower than supervised model that are showed in this study. Adoption of the DST method in network based intrusion detection systems would be beneficial since it can give successful result on real time dataset. An implementation on SIEM (Security Incident and Event Management) tools could easily prove its high performance. Low FPR rate is also a big plus because the SOC professionals usually deal with false positive most of their time.

Table 5.17. Performance Evaluation of the NB

NB	True Positive Rate	False Positive Rate	**Accuracy**
BOUN UDP1	93.13%	0.15%	**98.62%**
BOUN UDP2	99.83%	0.15%	**99.84%**
BOUN TCP	99.26%	0.24%	**99.61%**
CAIDA'07&'08	100%	0%	**100%**

We have used Naive Bayes classifiers in order to show that how close results have DST to this superior methods by also offering knowledge processing on real time. As you can see, NB has the best result for BOUN UDP2 which is dense attacks are involved. So we can say that NB is better than SVM on the cases with high density.

Table 5.18. Performance Evaluation of the SVM

SVM	True Positive Rate	False Positive Rate	**Accuracy**
BOUN UDP1	93.51%	0.09%	**98.85%**
BOUN UDP2	99.78%	0.04%	**99.93%**
BOUN TCP	98.92%	0.02%	**99.73%**
CAIDA'07&'08	100%	0%	**100%**

SVM has the best accuracy for low density datasets BOUN TCP and BOUN UDP1. The success of SVM on binary classification is again proven to be correct here.

Even though DST has slightly lower accuracy than NB and SVM, it is very successful for classification. NB and SVM are involved in supervised learning whereas DST uses unsupervised learning. This makes it very useful for detection of real time activities.

5.3. Analysis

Since we make 50 ms time windows, we have 1688 windows for 84.36 second. Attacks start at the 80.87 second, which corresponds to 1618^{th} window. In other words, first 80.87 second is normal network traffic data and last 3.49 second is network traffic data with attacks included. The whole UDP-1 flood dataset lasts 484 second. As it was previously explained first 80 second traffic is normal and the next 20 second contains attack and 100-180 second normal, then 180-200 second is with attack and so on.

As one can observe from Figure 5.1, the entropy distance drops suddenly when the attacks occur. Then, fluctuations go on at the lower level. We can distinguish attack frames by using this abrupt change. Methods like Thresholding and Naive Bayes Classification can simply find the attacks. However, we solve the problem with Evidence Theory(DST).

Simulation results on the UDP1, UDP2, TCP SYN Flood and CAIDA'07 and CAIDA'08 datasets are shown in the thesis. To apply algorithms, the duration of the 50ms time window is selected according to the data rate. Since CAIDA datasets are very dense, 2 ms time window is selected for them. After choosing the time window, probability distributions of packet header elements are computed. Then, we extract the histogram of the distribution. The top 5% and the lowest 5% are compared. We use the one with more dispersion. By using this method, we make our learning unsupervised. In this case, the top 5% has more dispersion. The lowest value of the 5% is selected as threshold and BPA values are computed according to this value with Equation 3.15. For each feature the same training is done and by using this training set we build the classifiers for DST, SVM and NB. Results for three classifier and four different dataset are given at the end of this section.

Main characteristics of DDoS attack are abrupt traffic change, flow dissymmetries such as distributed source IP addresses and concentrated destination IP addresses. In addition, it is observed that the distribution of port numbers and packet lengths are affected by this attack. Higher distance metric values result from the concentration difference of the probability distributions. Conversely, when the distance metric is low, concentrations of the probability distributions are similar. When there is no attack, distance values are more stable with each other.

TCP Flood data with slower packet rates are given in details. It is observed that seven of ten features are beneficial in order to detect DDoS attacks. In various kind of attacks, this number is likely to change. The best indicator of the DDoS attack is the distance between the source and destination IP addresses given in Figure 5.5. A similar relation is observed between the source IP addresses and destination ports given

in Figure 5.7. IP addresses and ports at the same side tend to move together during the DDoS attack. Nonetheless, the limited number of source ports compared to source IP addresses affects results. In Figure 5.9 and Figure 5.10, separation of distance values is not clear like in Figure 5.5 and 5.7 because of the limitedness of source ports. In 5.13, it is observed that both features are concentrated. The distance between them does not change reasonably but in Figure 5.14, due to the limitedness of source ports, during a DDoS attack, the distance between these features decreases slightly. In the nature of a network, there is randomness between the length of packets. However, in the attack case dispersed source IP addresses send a similar length of packets to the destination. This case is observed in Figure 5.11 and 5.12. Destination side shows similar behavior with the length of packets. Therefore, a certain change in the distance values are not observed in Figure 5.13 and 5.14.

Simulation results are given in previous section. UDP-1 Flood corresponds to the slower rate of DDoS attack data. UDP-2 Flood corresponds to the faster rate of DDoS attack data. 5% of the data is used as a training set. It means that every 1 of 20 sample of the results are added to the training set. Also, for CAIDA dataset, 1% the data is used as a training set.

Even though Bayesian theory is very powerful in some situations, there exist certain problems related with its utilization. A major problem that asserted by the most of the researchers the Bayes should have completeness with knowledge of both the conditional and prior probabilities. These are very hard and sometimes is not possible to specify in real life challenges [54]. Moreover, one may have multiple paths to formulate a specific incident via Bayesian, different or additional evidences are considered about the same situation. If extra or different evidences are considered, the posterior probability is more likely to alter under different conditions. In a similar way, the posterior probability is likely to differ when the evidences are obtained from situations that occurs under diverse experimental conditions. Additionally, Bayesian theory cannot permit the assignment of a specific probability for uncertainty. It states that the probabilities should be assigned to the state occurrence at a time [59], [61].

Although DST has slightly lower accuracy than NB and SVM, it is very successful for classification. NB and SVM are involved in supervised learning whereas DST uses unsupervised learning. This makes it very useful for detection of real time activities. With all the classifiers, we are able to perfectly detect every attack in CAIDA. Because CAIDA attack dataset is ver dense compared to normal one, entropy of the time windows changes immediately, thereby, we can detect them in an accurate manner.

6. CONCLUSION

If a multipaths method is adopted by an IPS, it is observed it has better outcomes compared to single-layer approach. Because DST offers number of advantages over other information fusion techniques in detection systems, it is selected as the most appropriate. A theoretical and mathematical explanation of the DST framework is given and in order to present how this method works certain practical situations are given as example. In the past, DST is employed in IPS researches in order to improve the detection accuracy rate. Nonetheless, an unsupervised process of BPA, based on the measurements of metrics system under consideration, which can detect occurrences of attacks in real time is adopted in this thesis.

In this work, the distance between two probability distributions with different sizes is used as an indicator of an attack. The probability distributions of the packet headers are calculated. In our simulations it is observed that these calculated distance values are a great sign of the DDoS attacks. These values are very effective to detect DDoS attacks. Thus, giving a small portion of the dataset into a training set (5% in Bogazici University dataset, 1% in CAIDA datasets is enough to take high accuracy and true positive rates.

For future work, different kind of attacks with different datasets will be simulated using this algorithm. Also, to improve the detection method from supervised learning to unsupervised or semi-supervised learning, new methods will be studied. This system will be designed to operate in Security Information and Event Management (SIEM) systems.

REFERENCES

1. Feldman, S., *2019 Is a Big Year for Cyber Security Investments*, 2019, `https://www.statista.com/chart/17935/cybersecurity-tech-investments/`, accessed in July 2019.

2. Cloudflare, 2019, `https://www.cloudflare.com`, accessed in July 2019.

3. Erhan, D., *BOUN DDOS Attack Dataset*, 2015.

4. Cisco, *At a Glance Internet of Things*, 2018, `https://www.cisco.com/c/dam/en/us/products/collateral/se/internet-of-things/at-a-glance-c45-731471.pdf`, accessed in July 2019.

5. Powell, D., R. Stroud, S. Creese, M. Dacier, Y. Deswarte and J.-C. Laprie, *Malicious and Accidental Fault Tolerance for Internet Applications - Conceptual Model and Architecture*, 2001.

6. Debar, H., M. Dacier and A. Wespi, "A Revised Taxonomy for Intrusion-Detection Systems", *Annales Des Telecommunications*, Vol. 55, pp. 361–378, 2000.

7. Debar, H., M. Dacier and A. Wespi, "Towards a Taxonomy of Intrusion-Detection Systems", *Computer Networks*, Vol. 31, pp. 805–822, 1999.

8. Axelsson, S., *Intrusion Detection Systems : A Survey and Taxonomy*, 2000.

9. Aleroud, A. and G. Karabatis, "Toward Zero-Day Attack Identification using Linear Data Transformation Techniques", pp. 159–168, 2013.

10. *Snort*, 2013, `https://www.snort.org`, accessed in July 2019.

11. Modi, C., D. R. Patel, B. Borisaniya, H. Patel, A. Patel and M. Rajarajan, "A Survey of Intrusion Detection Techniques in Cloud", *J. Network and Computer*

Applications, Vol. 36, pp. 42–57, 2013.

12. Liao, H.-J., C.-H. R. Lin, Y.-C. Lin and K.-Y. Tung, "Intrusion Detection System: A Comprehensive Review", *J. Network and Computer Applications*, Vol. 36, pp. 16–24, 2013.

13. Kohonen, T., "The Self-Organizing Map", Vol. 78 (9), pp. 1464 – 1480, 1990.

14. Casas, P., J. Mazel and P. Owezarski, "Unsupervised Network Intrusion Detection Systems: Detecting the Unknown without Knowledge", *Computer Communications*, Vol. 35(7), pp. 772–783, 2012.

15. Yang, J., T. Deng and R. Sui, "An Adaptive Weighted One-class SVM for Robust Outlier Detection", *Proceedings of the 2015 Chinese Intelligent Systems Conference*, pp. 475–484, 2016.

16. Khan, L., M. Awad and B. Thuraisingham, "A New Intrusion Detection System using Support Vector Machines and Hierarchical Clustering", *International Journal on Very Large Data Bases*, Vol. 16 (4), pp. 507–521, 2007.

17. López-Rubio, E., R. Luque-Baena, E. Palomo and E. Domínguez, "Dynamic Tree Topology Learning by Self-Organization", *Neural Computing and Applications*, pp. 1–14, 2016.

18. Srinivasan, M., K. Sarukesi, A. Keshava and P. Revathy, "ecloudids Tier-1 UX-Engine Subsystem Design and Implementation using Self-Organizing Map (SOM) for Secure Cloud Computing Environment", *Recent Trends in Computer Networks and Distributed Systems Security*, pp. 432–443, 2012.

19. Forrest, S., S. Hofmeyr, A. Somayaji and T. Longstaff, "A Sense of Self for Unix Processes", *IEEE Symposium on Security and Privacy*, pp. 120–128, 1996.

20. Gupta, S. and P. Kumar, "An Immediate System Call Sequence Based Approach

for Detecting Malicious Program Executions in Cloud Environment", *Wireless Personal Communications*, Vol. 81 (1), pp. 405–425, 2015.

21. Gupta, S. and P. Kumar, "System Cum Program-Wide Lightweight Malicious Program Execution Detection Scheme for Cloud", *Information Security Journal: A Global Perspective*, Vol. 23 (3), pp. 86–99, 2014.

22. Warrender, C., S. Forrest and B. Pearlmutter, "Detecting Intrusions using System Calls: Alternative Data Models", *Proceedings of the IEEE Symposium on Security and Privacy*, p. 133–145, 1999.

23. Hofmeyr, S., S. Forrest and A. Somayaji, "Intrusion Detection Using Sequences of System Calls", *Journal of Computer Security*, Vol. 6 (3), p. 151–180, 1998.

24. Helman, P. and J. Bhangoo, "A Statistically Based System for Prioritizing Information Exploration Under Uncertainty", *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol. 27 (4), p. 449–466, 1997.

25. Kang, D., D. Fuller and V. Honavar, "Learning Classifiers for Misuse and Anomaly Detection using a Bag of System Calls Representation", *IEEE Information Assurance Workshop*, p. 118–125, 2005.

26. CAIDA, *CAIDA Attack and Normal Dataset: Attack 2007 and Normal 2008*, 2008, `https://www.caida.org/data`, accessed in July 2019.

27. Prasenna, P., R. Kumar, A. Ramana and A. Devanbu, "Network Programming and Mining Classifier for Intrusion Detection using Probability Classification", *International Conference on Pattern Recognition, Informatics and Medical Engineering*, p. 204–209, 2012.

28. Han, L., "Using a Dynamic k-means Algorithm to Detect Anomaly Activities", *IEEE International Conference on Computational Intelligence and Security*, p. 1049–1052, 2011.

29. *KDD Dataset*, 1998, `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`, accessed in August 2019.

30. Benaicha, S., L. Saoudi, S. Guermeche and O. Lounis, "Intrusion Detection System using Genetic Algorithm", *IEEE Science and Information Conference*, pp. 564–568, 2014.

31. Thaseen, I. and C. Kumar, "Intrusion Detection Model using Fusion of PCA and Optimized SVM", *International Conference on Contemporary Computing and Informatics*, pp. 879–884, 2014.

32. Wagh, S. and S. Kolhe, "Effective Intrusion Detection System using Semi-supervised Learning", *International Conference on Data Mining and Intelligent Computing*, pp. 1–5, 2014.

33. Sayar, A., S. Pawar and V. Mane, "A Review of Intrusion Detection System in Computer Network", *International Journal of Computer Science and Mobile Computing*, Vol. 3 (2), pp. 700–703, 2014.

34. Islam, M. and S. Rahman, "Anomaly Intrusion Detection System in Wireless Sensor Networks: Security Threats and Existing Approaches", *International Journal of Advanced Science and Technology*, Vol. 36 (1), pp. 1–8, 2011.

35. Bahl, S. and S. Sharma, "Improving Classification Accuracy of Intrusion Detection System using Feature Subset Selection", *International Conference on Advanced Computing Communication Technologies*, pp. 431–436, 2015.

36. Yan, C., "Intelligent Intrusion Detection based on Soft Computing", *International Conference on Measuring Technology and Mechatronics Automation*, pp. 577–578, 2015.

37. Haidar, G. and C. Boustany, "High Perception Intrusion Detection System using Neural Networks", *International Conference on Complex, Intelligent, and Software*

Intensive Systems, pp. 497–501, 2015.

38. Sun, C., J. Fan and B. Liu, "A Robust Scheme to Detect SYN Flooding Attacks", pp. 397 – 401, 2007.

39. Nashat, D., X. Jiang and S. Horiguchi, "Router Based Detection for Low-Rate Agents of DDoS Attack", pp. 177–182, 2008.

40. Singh, J., M. Sachdeva and K. Saluja, "Detection of DDoS Attacks Using Source IP Based Entropy", *International Journal of Computer Science Engineering and Information Technology Research*, Vol. 3, pp. 201–210, 2013.

41. Monowar, H. B., K. B. Dhruba and K. K. Jugal, "Information Metrics for Low-Rate DDoS Attack Detection: A Comparative Evaluation", *Seventh International Conference on Contemporary Computing*, pp. 80–84, 2014.

42. Tu, X., H. Dake and L. Yu, "DDoS Attack Detection Based on RLT Features", *2007 International Conference on Computational Intelligence and Security (CIS 2007)*, pp. 697–701, 2007.

43. Dongqi, W., Y. Zhu and J. Jia, "A Multi-core Based DDoS Detection Method", *International Conference on Computer Science and Information Technology*, Vol. 4, pp. 115–118, 2010.

44. Hamza, R., S. Nabil and K. Farouk, "DDoS Flooding Attack Detection Scheme Based on F-divergence", *Computer Communications*, Vol. 35, pp. 1380–1391, 2012.

45. Sang, M. L., S. K. Dong, H. L. Je and S. P. Jong, "Detection of DDoS Attacks Using Optimized Traffic Matrix", *Computers Mathematics with Applications*, Vol. 63, pp. 501–510, 2012.

46. Sharma, S., S. K. Sahu and S. K. Jena, "On Selection of Attributes for Entropy Based Detection of DDoS", pp. 1096–1100, 2015.

47. Chen, Q. and U. Aickelin, "Dempster-Shafer for Anomaly Detection", *ArXiv*, Vol. abs/0803.1568, 2006.

48. Bellenger, A. and S. Gatepaille, "Uncertainty in Ontologies: Dempster-Shafer Theory for Data Fusion Applications", *The Computing Research Repository*, 2011.

49. Dubois, D. and H. Prade, "Possibility Theory, Probability Theory and Multiple-valued Logics: A Clarification", *Annual Mathematical Artificial Intelligence*, Vol. 32(1-4), pp. 35–66, 2001.

50. Dubois, D., H. Prade and P. Smets, "Representing Partial Ignorance", *IEEE Transaction Systems, Man Cybernetics, Part A: Systems and Human*, Vol. 26(3), pp. 361–377, 1996.

51. Ferson, S. and L. Ginzburg, "Different Methods Are Needed to Propagate Ignorance and Variability", *Reliable Engineering Systems Safety*, Vol. 54(2), pp. 133–144, 1996.

52. Verbert, K., R. Babuška and B. De Schutter, "Bayesian and Dempster–Shafer Reasoning for Knowledge-based Fault Diagnosis – A comparative study", *Engineering Applications of Artificial Intelligence*, 2017.

53. Kiureghian, A. and O. Ditlevsen, "Aleatory or Epistemic? Does It Matter?", *Struct. Saf.*, Vol. 31(2), pp. 105–112, 2009.

54. Billinton, R. and D. Huang, "Aleatory and Epistemic Uncertainty Considerations in Power System Reliability Evaluation", *Proceedings of the 10th International Conference on Probabilistic Methods Applied to Power Systems*, pp. 1–8, 2008.

55. Dempster, A., "Upper and Lower Probabilities Induced by a Multivalued Mapping", *Annals of Mathematical Statistics*, pp. 325–339, 1967.

56. Shafer, G., "A Mathematical Theory of Evidence", Vol. 1, 1976.

57. Kohlas, J. and P. Money, "A Mathematical Theory of Hints: An Approach to the Dempster-Shafer Theory of Evidence", *Springer-Verlag*, 1995.

58. Smets, P., *What Is Dempster-Shafer's model? In: Yager, R., Kacprzyk, J., Fedrizzi, M. (Eds.), Advances in the Dempster-Shafer Theory of Evidence*, John Wiley Sons, 1994.

59. Laplace, P. and D. Huang, *A Philosophical Essay on Probabilities*, Dover Publications Inc., 1814.

60. Savage, L. and D. Huang, *The Foundations of Statistics*, John Wiley Sons, 1954.

61. Howie, D., *Interpreting Probability: Controversies and Developments in the Early Twentieth Century*, Cambridge University Press, 2002.

62. Goldstein, M., "Subjective Bayesian Analysis: Principles and Practice", *Bayesian Anal.*, Vol. 1 (3), pp. 403–420, 2006.

63. Zadeh, L., "Fuzzy Sets as a Basis for a Theory of Possibility", *Fuzzy Sets Systems*, Vol. 100(Supplement 1), 1999.

64. Klir, G. and B. Yuan, *Fuzzy Sets and Fuzzy Logic 4*, Prentice Hall, 1995.

65. Zadeh, L., "Fuzzy Logic and Approximate Reasoning", *Synthese*, Vol. 30(3-4), 1975.

66. Alexandros, G., Fragkiadakis, A. S. Vasilios and T. Apostolos, "Effective and Robust Detection of Jamming Attacks", *2010 Future Network Mobile Summit*, pp. 1–8, 2010.

67. Dubois, D., "Possibility Theory and Statistical Reasoning", Vol. 51(1), 2006.

68. Triola, M., *Bayes' Theorem*, Addison Wesley, elementary statistics edn., 2010.

69. Shih, C. and G. Kochanski, *Bayes' Theorem*, 2006, `http://kochanski.org/gpk/teaching/0401Oxford/Bayes.pdf`, accessed in July 2019.

70. Siaterlis, C. and B. Maglaris, "Towards Multisensor Data Fusion for DoS Detection", *Proceedings of the ACM Symposium on Applied Computing*, 2004.

71. F.J.Aparicio-Navarro, "Using Metrics From Multiple Layers to Detect Attacks in Wireless Networks", *PhD Thesis*, 2014.

72. Boston, J., "A Signal Detection System based on Dempster-Shafer Theory and Comparison to Fuzzy Detection", *IEEE Transactions on Systems, Man, and Cybernetics Part C: Applications and Reviews*, Vol. 30 (1), 2000.

73. Vakili, A., "Approximation of Hints", *Inst. for Automation and Op. Res.*, 1993.

74. Goodman, I. and H. Nguyen, *Uncertainty Models for Knowledge-Based Systems*, 1985.

75. Hestir, K., H. Nguyen and G. Rogers, "A Random Set Formalism for Evidential Reasoning", *Goodman, I.R. et al. (eds.)*, pp. 309–344, 1991.

76. Kleer, J. D., "An Assumption-based TMS", *Artificial Intelligence*, Vol. 28(2), pp. 127–162, 1986.

77. Halpern, J. and R. Fagin, "Two Views of Belief: Belief as Generalized Probability and Belief as Evidence", *Artificial Intelligence*, Vol. 54, pp. 275–317, 1992.

78. Yager, R. R., "On the Dempster-Shafer Framework and New Combination Rules", *Inf. Sci.*, Vol. 41 (2), pp. 93–137, 1987.

79. Pal, N., J. Bezdek and R. Hemasinha, "Uncertainty Measures for Evidential Reasoning: A Review", *Int. J. Approx. Reason.*, Vol. 7, p. 165–183, 1992.

80. George, T. and N. Pal, "Quantification of Conflict in Dempster–Shafer Framework: A New Approach", *Int. J. Gen. Syst.*, Vol. 24, p. 407–423, 1996.

81. Deng, Y., "Deng Entropy", *Chaos Solutions Fractals*, Vol. 91, p. 549–553, 2016.

82. Panigrahi, S., A. Kundu, S. Sural and A. Majumdar, "Credit Card Fraud Detection: A Fusion Approach using Dempster–Shafer Theory and Bayesian Learning", *Information Fusion*, Vol. 10, 2009.

83. Chen, T. M. and V. Venkataramanan, "Dempster-Shafer Theory for Intrusion Detection in Ad Hoc Networks", *IEEE Internet Computing*, Vol. 9(6), 2005.

84. Sentz, K. and S. Ferson, *Combination of Evidence in Dempster-Shafer Theory*, 2002.

85. Buede, D. M. and P. Girardi, "A Target Identification Comparison of Bayesian and Dempster-Shafer Multisensor Fusion", *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, Vol. 27 (5), pp. 569–577, 1997.

86. Chatzigiannakis, V., G. Androulidakis, K. Pelechrinis, S. Papavassiliou and V. Maglaris, "Data Fusion Algorithms for Network Anomaly Detection: Classification and Evaluation", *Proceedings of the 3rd International Conference on Networking and Services*, 2007.

87. Mountrakis, G., J. Im and C. Ogole, "Support Vector Machines in Remote Sensing: A Review", *Journal of Photogrammetry and Remote Sensing*, Vol. 66, pp. 247–259, 2011.

88. Makke, A., O. Salem, M. Assaad, H. Moungla and A. Mehaoua, "Flooding Attacks Detection in Backbone Traffic using Power Divergence", pp. 15–20, 2012.

89. Liu, H. and M. S. Kim, "Real-Time Detection of Stealthy DDoS Attacks using Time-Series Decomposition", *IEEE International Conference on Communications*,

pp. 1–6, 2010.