

FILTERING BASED DEFENSE MECHANISMS AGAINST DDOS ATTACKS FOR
CORE NETWORKS

by

Kübra Kalkan

B.S., Computer Science and Engineering, Sabanci University, 2009

M.S., Computer Science and Engineering, Sabanci University, 2011

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

Graduate Program in
Boğaziçi University

2016

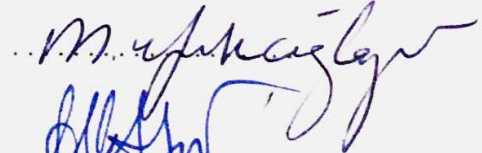
FILTERING BASED DEFENSE MECHANISMS AGAINST DDOS ATTACKS FOR
CORE NETWORKS

APPROVED BY:

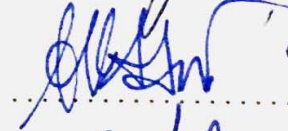
Prof. Fatih Alagöz
(Thesis Supervisor)



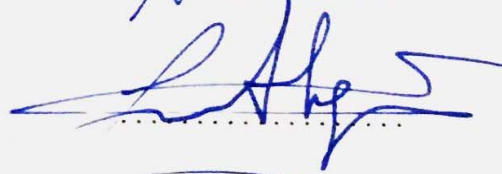
Prof. M. Ufuk Çağlayan



Prof. Albert Levi



Prof. Sema Oktuğ



Prof. Tuna Tuğcu



DATE OF APPROVAL: 7.10.2016

ACKNOWLEDGEMENTS

Firstly, I would like to thank my advisor Prof. Dr. Fatih Alagöz for his guidance and encouragement throughout these years. He is not only a motivating advisor but also removes tedious hierarchy between the Professor and the student. This makes you feel free and removes the pressure on you, which makes it easier to research.

I would like to thank my Master thesis advisor Prof. Dr. Albert Levi for his continuous guidance in my academic life. He and Prof. Dr. Ufuk Çağlayan also contributed to my PhD thesis during the progress presentations. I would also like to thank my final jury members Prof. Dr. Ufuk Çağlayan, Prof. Dr. Albert Levi, Prof. Dr. Sema Oktuğ and Prof. Dr. Tuna Tuğcu for their time and comments about my thesis.

I would especially want to thank Dr. Gürkan Gür, for his support during my thesis. He was like an academic elder brother for me, who improves my scientific skills and academic thinking. He reviewed my all papers and improved their quality significantly. I would also thank to Levent Altay for being a supportive colleague during our projects.

During this thesis, I was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) 2211 National PhD Scholarship. Thanks to this support that I can concentrate on my research without financial concerns.

I cannot find words to express my appreciation for my parents. I feel very lucky to have such a family. Since they are also academicians they understand, advice and support me deeply. My father is the first advisor for me who helps me to draw my academic path, whereas my mother is the first prototype to be a successful academician and a mother. I am also extremely grateful to my sister Rana. She was the source of joy in my life while she was a little girl, but now she became the best friend and life saver. She became a wonderful aunt who is literally more than half of a mother.

I also need to thank to my husband's family especially his sisters, for becoming a real sister for me. They incredibly supported me with their help on taking care of my son whenever I needed, even in distant lands. Last but definitely not the least, my deepest gratitude goes to my two special guys: my husband Hakan and my son Orhan. Thanks for my husbands unconditional love, patience, and continual support during my academic life. He did not only supported me but also boosted me. I am such a lucky girl that my husband encourages me to go abroad for my career even we had a little son. He always supported me in my decisions without any hesitations. Also, my son was my foremost luck who helped me by being not a naughty kid. He was the source of inspiration and happiness when I stucked during my research. He also helped me with playing his toys without giving any discomfort, while I am working on this thesis. He gave wonderful patience even he is three years old. This thesis is dedicated to my awesome large family.

ABSTRACT

FILTERING BASED DEFENSE MECHANISMS AGAINST DDOS ATTACKS FOR CORE NETWORKS

In this thesis, we present filtering based defense mechanisms against Distributed Denial of Service (DDoS) attacks for core networks. Initially, several filtering techniques are analyzed and their advantages and disadvantages are presented. A comparative classification of these methods is provided for security analysts. Classification results suggest that there are a few filtering methods that are both proactive and collaborative. Proactivity provides prevention of attacks before it spreads whereas collaboration enables getting knowledge about different points of the network and deciding filters together. Thus, we proposed a proactive and collaborative model called ScoreForCore. It is a statistical packet based defense mechanism that selects the most appropriate attributes for current attack traffic. Our results suggest that the success of the system's behavior on legal and attack packets increased considerably. This strategy is also convenient for current emerging technology for core networks, called Software Defined Networking (SDN). It has several problems related to security that are largely induced by the centralized control paradigm. In that regard, DDoS attacks are specifically valid for SDN environment. Several defense mechanisms in SDN environment are analyzed and comparative classification is provided for rendering the current state of the art in the literature. Then, our defense strategy is applied on SDN environment with capable switches. This mechanism's (SDNScore) results suggest that it gives perfect results for several known attacks and 84% success for an unknown attack. Since there is a trade-off between SDN paradigm and capable switches in SDNScore, we improved it and proposed another model called Joint Entropy based Scoring for SDN (JESS) that carries all burden to the controller and does not need capable switches. The results suggest that it is an elegant defense method for SDN environment.

ÖZET

ÇEKİRDEK AĞLARDA DDOS SALDIRILARINA KARŞI FİLTRELEME TABANLI SAVUNMA MEKANİZMALARI

Bu tezde çekirdek ağlarda Dağılmış Hizmet Engelleme Saldırısı (DDoS)'na karşı filtreleme bazlı savunma mekanizmaları geliştirilmiştir. İlk olarak, çeşitli filtreleme teknikleri analiz edilmiş ve avantajları ve dezavantajları sunulmuştur. Güvenlik analistleri için bu metotların karşılaştırılmalı sınıflandırılması yapılmıştır. Sınıflandırma sonuçları, hem proaktif hem de işbirlikçi modellerin çok az sayıda olduğunu ortaya koymuştur. Bir metodun proaktif olması saldırı yayılmadan engellemesini sağlarken, işbirlikçi olması ağın farklı noktaları hakkında bilgi edinip filtrelemlere birlikte karar vermesini sağlamaktadır. Biz de bu sebeple proaktif ve işbirlikçi bir model olan Score-ForCore'u öne sürdük. Bu metot, anlık saldırı trafiği için en uygun öznelilikleri seçen, istatistiksel, paket bazlı bir savunma mekanizmasıdır. Sonuçlarımıza göre bu mekanizmanın legal ve saldırı paketleri üzerindeki başarısı oldukça yüksek çıkmıştır. Öne sürdüğümüz strateji, çekirdek ağlar için yeni gelişmekte olan Yazılım Tabanlı Ağ (YTA) teknolojisi için de uygulanabilir. Bu sebeple, bu çalışmada ilk olarak YTA ortamında literatürdeki en yeni gelişmeleri sunmak için çeşitli savunma mekanizmaları analiz edilmiş ve karşılaştırmalı sınıflandırılması yapılmıştır. Daha sonra da, bizim savunma stratejimiz, YTA ortamında kabiliyetli anahtarlayıcılarla uygulanmıştır. Bu mekanizma (SDNScore), bilinen saldırılar için çok iyi sonuçlar vermiştir. Hatta bilinmeyen saldırılar için bile 84% başarı göstermiştir. SDNScore modelinde kabiliyetli anahtarlayıcılar ve YTA paradigması arasında bir seçim yapma durumu olduğu için, biz bu modeli geliştirip, tüm yükü yönetici makineye taşıyan ve kabiliyetli anahtarlayıcılara ihtiyaç duymayan Joint Entropy Based Scoring on SDN (YTA için Ortaklaşa Entropi Tabanlı Skorlama) isimli başka bir mekanizma daha öne sürdük. Sonuçlara göre bu mekanizma YTA ortamı için güçlü bir savunma ortamı sağlamıştır.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	iii
ABSTRACT	v
ÖZET	vi
LIST OF FIGURES	x
LIST OF TABLES	xiii
LIST OF SYMBOLS	xv
LIST OF ACRONYMS/ABBREVIATIONS	xvii
1. INTRODUCTION	2
2. RELATED WORK	6
2.1. Filtering Based DDoS Defense Mechanisms	6
2.1.1. Ingress/Egress Filtering	6
2.1.2. Route Based Distributed Packet Filtering (RDPF)	7
2.1.3. Source Address Validity Enforcement (SAVE)	9
2.1.4. Hop Count Filtering (HCF)	10
2.1.5. PacketScore	10
2.1.6. Secure Overlay Services (SOS)	11
2.1.7. Pushback	12
2.1.8. Active Internet Traffic Filtering (AITF)	12
2.1.9. StopIt	14
2.1.10. Probabilistic Filter Scheduling (PFS)	15
2.1.11. Adaptive Probabilistic Filter Scheduling (APFS)	16
2.2. Solutions Against DDoS Attacks in SDN Environment	18
3. ANALYSIS OF DDOS DEFENSE FILTERING MECHANISMS	24
3.1. Research areas about DDoS	24
3.1.1. DDoS Detection Mechanisms	25
3.1.2. DDoS Prevention Mechanisms	29
3.2. Analysis of Filtering Techniques in Traditional Networks	29
3.2.1. Classification of Filtering Techniques	30
3.2.2. Attribute-based Comparison of Filtering Mechanisms	33

3.3.	Analysis of Filtering Techniques in SDN Environment	34
3.3.1.	DDoS attack scenarios on SDN	37
3.3.2.	Classifications of Solutions Against DDoS Attacks in SDN . . .	39
3.4.	Chapter Summary	42
4.	A STATISTICAL FILTERING DEFENSE MECHANISM: ScoreForCore . .	44
4.1.	ScoreForCore Mechanism	45
4.1.1.	Profiling	45
4.1.2.	Comparison	46
4.1.3.	Collaboration	47
4.1.4.	Score calculation	50
4.1.5.	Threshold calculation	50
4.1.6.	Selective discarding	50
4.2.	Dataset and Simulation Details	52
4.2.1.	Dataset	52
4.2.2.	Simulation Environment	53
4.2.3.	Network Generation	53
4.2.4.	Attack Generation	54
4.3.	Performance Evaluation	55
4.3.1.	Performance Metrics	55
4.3.2.	Attribute Distribution Analysis in Network Topology	57
4.3.3.	Results	58
4.3.4.	Storage Analysis	60
4.4.	Discussion	62
4.5.	Chapter Summary	63
5.	DEFENSE MECHANISM IN SDN ENVIRONMENT: SDNScore	65
5.1.	SDNScore Mechanism	67
5.1.1.	SDNScore Modules for Switches	67
5.1.2.	SDNScore Module for Controller: PairProfiler	70
5.2.	Simulations and Performance Evaluation	71
5.2.1.	Network Topology and Dataset	71
5.2.2.	Attack Types	72

5.2.3. Performance Metrics	73
5.2.4. Experimental Results	74
5.3. Chapter Summary	77
6. JOINT ENTROPY BASED SCORING MECHANISM AGAINST DDOS AT-	
TACKS IN SDN ENVIRONMENT:JESS	79
6.1. JESS Mechanism	81
6.1.1. Nominal Stage	81
6.1.2. Infant Stage	84
6.1.3. Lifelong Stage	85
6.2. Simulations and Performance Evaluation	89
6.2.1. Dataset	89
6.2.2. Simulation Environment	89
6.2.3. Network Generation	89
6.2.4. Packet Analysis in Mininet	90
6.2.5. Attack Types	90
6.2.6. Performance Metrics	92
6.2.7. Experimental Results	92
6.2.8. Time Analysis	98
6.2.9. Storage Analysis	101
6.2.10. Communication Analysis	101
6.3. Discussions	102
6.4. Chapter Summary	103
7. CONCLUSION	104
REFERENCES	106

LIST OF FIGURES

Figure 2.1.	Ingress/Egress Filtering [1]. Ingress and egress filters are installed at edge routers. In this example configuration, egress filter does not allow any IP out of 220.25.112.0/24 range. Similarly, since incoming IP address space is known, ingress filters do not allow any “alien” IP.	7
Figure 2.2.	Route Based Distributed Packet Filtering (RDPF) [2]. Attacker is at AS6 which spoofs AS1 and Victim is AS3. RDPF installed at AS5 will block packets from AS1 to AS3 since that transmission is not possible in this topology.	8
Figure 2.3.	Source Address Validity Enforcement (SAVE) [3]. Router E generates forwarding table from SAVE messages. This table contains expected IP ranges from links 1, 2, 3 and 4.	9
Figure 2.4.	Pushback [4]. Router 10 is the congested router. It requests Router 7 and 8 to rate-limit. Then, Router 7 sends request to Router 2 whereas Router 8 sends request to Router 4 and 5. These pushback messages propagate upstream until the source of traffic.	13
Figure 2.5.	Active Internet Traffic Filtering (AITF) [5]. Filter is sent from victim (V) to its gateway (VGW). Then it is propagated to the attacker’s gateway (AGW) which requests attacker (A) to stop sending traffic.	13

Figure 2.6.	StopIt [6]. Victim V sends filter to its access router (Rv) which communicates with its StopIt server. Then StopIt server of AS3 communicates with StopIt server of AS1 in order to reach attacker. StopIt server of AS1 communicates with access router (Ra) which blocks traffic after filter installation.	15
Figure 2.7.	Probabilistic filter scheduling schemes.	17
Figure 3.1.	Classification of DDoS attack studies.	25
Figure 3.2.	A sample SDN topology with DDoS attackers	39
Figure 3.3.	Classification of solutions against DDoS Attacks in SDN	40
Figure 4.1.	Collaboration in ScoreForCore	48
Figure 4.2.	Pseudo Code for Collaboration of ScoreForCore	49
Figure 4.3.	Operations in ScoreForCore	51
Figure 4.4.	Network Topology utilized in ScoreForCore simulations	53
Figure 4.5.	Metrics utilized in ScoreForCore evaluation	55
Figure 5.1.	SDNScore architecture: switch and controller modules.	67
Figure 5.2.	Network topology used in SDNScore experiments.	72
Figure 6.1.	Protocol of JESS Mechanism	82
Figure 6.2.	Network topology used in the experiments.	90

Figure 6.3.	FPR performance of JESS during DDoS attacks	94
Figure 6.4.	Accuracy performance of JESS during TCP SYN Flood, NTP attack, DNS Amplification and Generic attacks	95
Figure 6.5.	Network topology used in the mixed attack experiments of JESS .	96
Figure 6.6.	Accuracy and FPR performance of JESS during Mixed Attack . .	96
Figure 6.7.	FPR during TCP SYN Flood, NTP attack and DNS Amplification with Sliding Windows	98
Figure 6.8.	Performance of JESS during different attack intensities	99

LIST OF TABLES

Table 3.1.	Classification of DDoS Detection Mechanisms [7]	26
Table 3.2.	Classification of Defense Mechanisms Against DDoS Attacks	31
Table 3.3.	Key Attributes and Comparison of Defense Mechanisms	35
Table 3.4.	Classification of Solutions According to Defense Functionality and Switch Intelligence	41
Table 4.1.	Pair Nominal Profile (<i>OPNP</i>) Example	45
Table 4.2.	Single Nominal (<i>SingleNP_{protocol}</i>) and Single Current <i>SingleCP_{protocol}</i> Profiles of Protocol Attribute	46
Table 4.3.	Single Nominal and Single Current Profiles of Packet Size Attribute	46
Table 4.4.	Number of Routers reaching the <i>SuspiciousPair</i>	58
Table 4.5.	Precision, Recall and F-Measure Results of ScoreForCore	59
Table 4.6.	True Negative Rate , Negative Predictive Value and F-Measure Complement Results of ScoreForCore	59
Table 4.7.	Accuracy and Attack Prevention Efficiency Results of ScoreForCore	60
Table 4.8.	The Number of Different Values for Each Attribute	62
Table 5.1.	Pair Nominal Profile Example	68

Table 5.2.	Precision, Recall, Accuracy and F-Measure Results of SDNScore .	75
Table 5.3.	Precision, Recall, Accuracy and F-Measure Results of SDNScore for Unknown Attacks	77
Table 6.1.	An Example of a Pair Nominal Profile for TTL Value & Destination Port Attributes	83
Table 6.2.	An Example of a Rule Table (RT_C) for the <i>SuspiciousPair</i> TTL Value & Destination Port Attributes	87
Table 6.3.	RT_P for the <i>SuspiciousPair</i> TTL Value & Destination Port At- tributes	88
Table 6.4.	Attributes in Headers	91
Table 6.5.	Time Complexity Analysis for steps in JESS	100

LIST OF SYMBOLS

A	Attribute A
$A = a_p$	Attribute A with a value in packet p
AP	The total number of attack packets
B	Attribute B
$B = b_p$	Attribute B with b value in packet p
CP	Profile in an attack period
dev_{an}	The deviation for attribute value $A = a_n$
Dev_A	Maximum deviation for attribute A
$DifRT$	Differentiated Rule Table
dis_i	The discard hop of attack packet i
$DJ_{A,B}$	The difference between $JECP_{A,B}$ and $JENP_{A,B}$
HOPY	Y number of hops
$JECP_{A,B}$	Joint Entropy of a current profile for attribute pair A and B
$JENP_{A,B}$	Joint Entropy of a nominal profile for attribute pair A and B
MEMX	X number of nominal profiles
N	First hop neighbor list
N'	Second hop neighbor list
N''	Third hop neighbor list
NA	The Number of attributes
NE	The Number of entries in a profile
NP	A Profile in an attack free period
NPP	The Number of packets in a profile
$npcp_{an}$	The number of packets with $A = a_n$ in current profile
$npnp_{an}$	The number of packets with $A = a_n$ in nominal profile
$OPNP$	Own pair nominal profile created by considering <i>OwnPair</i>
$OPNPList$	A list consists of <i>OPNPs</i>
<i>OwnPair</i>	Random attribute pair owned by the router
<i>OwnPairList</i>	A list consists of <i>OwnPairs</i>

p_i	The path length of attack packet i in terms of hop
$ScoreList$	A list that contains scores of packets
$ScorePair$	The pair that is used for generating current pair profiles
$PC(A = a, B = b)$	The probability of a packet to have the property of $A = a$ and $B = b$ in a current profile
PCP	Current pair profile used for score operations
$PCP_{(A=a_p, B=b_p, \dots)}$	The number of packets in a current profile that have the property of $A = a_p$ and $B = b_p$
$PN(A = a, B = b)$	The probability of a packet to have the property of $A = a$ and $B = b$ in a nominal profile
PNP	Nominal pair profile used for score operations
$PNP_{(A=a_p, B=b_p)}$	The number of packets in a nominal profile that have the property of $A = a_p$ and $B = b_p$
RT_C	Current Rule Table
RT_P	Previous Rule Table
S	Score value of an entry
$SingleNP$	A single attribute profile in an attack free period
$SingleCP$	A single attribute current profile in an attack period
S_p	Score value of packet p
$SPNP$	Nominal profile created by considering <i>SuspiciousPair</i>
ST	Score table
$SuspiciousPair$	The attribute pair with the most probable signs for the current attack
Th_C	Current Threshold
Th_P	Previous Threshold
Th	Threshold score value for packet discarding
$TPCP$	The total number of packets in a current profile
$TPNP$	The total number of packets in a nominal profile
ϕ	Acceptable traffic
ψ	Total current incoming traffic

LIST OF ACRONYMS/ABBREVIATIONS

ACC	Accuracy
AITF	Active Internet Traffic Filtering
APE	Attack Prevention Efficiency
APFS	Adaptive Probabilistic Filter Scheduling
ARP	Address Resolution Protocol
AS	Autonomous Systems
BGP	Border Gateway Protocol
CPS	Cyber Physical Systems
DNS	Domain Name System
DoS	Denial of Service Attacks
DDoS	Distributed Denial of Service Attacks
FM	F-Measure
FMC	F-Measure Complement
FN	False Negative
FP	False Positive
FPR	False Positive Rate
FR	Filter Router
GAD-2A	Generic Attack with Determined two Attributes
GAD-3A	Generic Attack with Determined three Attributes
GAD-4A	Generic Attack with Determined four Attributes
HCF	Hop Count Filtering
HTTP	Hypertext Transfer Protocol
ICT	Information Communication Technologies
IoT	Internet of Things
IP	Internet Protocol
IT	Information Technology
JESS	Joint Entropy based Scoring on SDN
KB	Kilobytes

MB	MegaBytes
NAT	Network Address Translation
NFV	Network Function Virtualization
NPV	Negative Predictive Value
NTP	Network Time Protocol
PFS	Probabilistic Filter Scheduling
PN	Precision
PS	PacketScore
RDPF	Route Based Packet Filtering
RL	Recall
SAVE	Source Address Validity Enforcement
SDN	Software Defined Networking
SFC	ScoreforCore
SOS	Secure Overlay Services
SQL	Structured Query Language
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TN	True Negative
TNR	True Negative Rate
TP	True Positive
TTL	Time to Live
UDP	User Datagram Protocol
XML	Extensible Markup Language

1. INTRODUCTION

Denial of Service (DoS) attacks have become a serious problem with the ubiquitous proliferation of IP based systems. Although it was already known in the 1980s, it became a predominant issue for network and information security in the last two decades. In 1984, DoS attacks were defined as the case of “intruder preventing legitimate users to access shared resources by using almost all available services” [8]. In other words, the simple strategy behind a DoS attack is to deny the use of system services/resources to legitimate users and degrade system availability. The fundamental mechanism for DoS attack execution is to send a flood of superfluous network traffic to the target so that it cannot respond to genuine requests for services or information. If multiple sources are used by attacker(s), that type is called as *Distributed Denial of Service (DDoS)* attack which is much more catastrophic than DoS [9]. Currently, botnet tools available on the Internet provide attackers with massive DDoS resources and a high level of anonymity against countermeasures. Other DoS attacks include the physical destruction of computer hardware and the use of electromagnetic interference designed to destroy unshielded electronics via current or voltage surges [10].

As a general phenomenon, the scale and size of DDoS attacks have been increasing recently. According to Prolexic 2014-Q1 Global DDoS Global Attack Report [11], compared to the same quarter one year ago, the total attacks increased 18 percent as seen by Prolexic. Although average attack duration went down 24 percent, from 22.88 to 17.38 hours, average peak attack bandwidth increased 114 percent from 4.53 Gbps to 9.70 Gbps and peak packets-per-second rate went up by 87 percent from 10.60 Mpps to 19.80 Mpps. The increasing bandwidths for Internet end-points and hyperconnectivity have also eased the practicality of these attack types. Moreover, the proliferation of network systems has transformed previously uncharted territories such as Software Defined Networking (SDN) into a part of the problem domain.

Software Defined Networking (SDN) is a recent emerging technology which defines a new design and management approach for networking [12]. The main property of this paradigm is the separation of control and data planes. In traditional networks, routers apply high level routing algorithms and decide where data packets should be forwarded. In SDN, decision and forwarding functionalities are separated. Decision process is provided by SDN controller whereas data forwarding is handled by switches. Since decision algorithms do not run on network devices, simpler network devices can be utilized rather than complicated routers [13]. Moreover, in traditional networks, each router has its own security, link failure and forwarding mechanisms. If any of these mechanisms needs to be updated, each network device should be handled individually. However, one can manage all these issues at a central point in SDN architecture.

One of the vulnerabilities of SDN architecture regarding DDoS attacks is the limited passive capabilities of switches. Since they send all packets with unknown IPs to the controller, their medium becomes attractive for DDoS attacks. Moreover, they do not possess sufficient resources for very large volumes of traffic. On the controller side, because of central management property, DDoS attack can cause catastrophic results if the controller is saturated with attack traffic. In the literature, there are some works proposed for DDoS defense in SDN. However, it is still an immature area since there is no dominant solution and all models have some drawbacks.

Broadly speaking, DDoS attacks are serious problems for both traditional and recent networks. Thus, in this thesis we proposed a defense strategy that is applied on both traditional network and SDN environment.

In this thesis, in Chapter 2 we review the related work of defense mechanisms against DDoS attacks in traditional network and SDN environment.

Chapter 3 presents a comprehensive survey on filtering based defense mechanisms against Distributed Denial of Service (DDoS) attacks in traditional networks and SDN environment. Firstly, several filtering techniques in traditional networks are analyzed and their advantages and disadvantages are presented. In order to help network se-

curity analysts choose the most appropriate mechanism according to their security requirements, a comparative classification of these methods is provided. The relevant research efforts are identified and discussed for rendering the current state of the art in the literature. This classification will also help researchers to address weaknesses of these filtering methods and thus mitigate DDoS attacks using more effective defense mechanisms. Then, filtering techniques in SDN environment are analyzed. Software Defined Networking (SDN) is a pivotal technology emerging as a promising paradigm for alleviating challenges of current as well as Future Internet. It relies on the fundamental idea of decoupling control and data planes in the network. This property provides several advantages such as flexibility, simplification and lower costs. However, it also brings along several drawbacks that are largely induced by the centralized control paradigm. Security is one of the most significant challenges related to centralization. In that regard, Distributed Denial of Service (DDoS) attacks are specifically valid for SDN environment. This chapter presents a comprehensive survey on solutions against DDoS attacks in SDN environment. Moreover, several mechanisms are analyzed and a comparative classification is provided for rendering the current state of the art in the literature. This analysis will help researchers to address weaknesses of these solutions and thus mitigate such attacks using more effective defense mechanisms.

In Chapter 4, we present a proactive and collaborative filtering based defense mechanism against Distributed Denial of Service (DDoS) attacks. Proactivity provides prevention of attacks before it spreads whereas collaboration enables getting knowledge about different points of the network and deciding filters together. The proposed model called ScoreForCore is a statistical mechanism that is inspired from another proactive but individual model. The most distinctive property of our model is the selection of the most appropriate attributes during current attack traffic. We compared our results with the existing model. Our results suggest that the success of system's behavior on legal and attack packets are increased considerably. In addition, most of the attack packets are stonewalled near the source of the attack.

In Chapter 5, we applied our strategy which is the selection of the most appropriate attributes during current attack traffic on SDN environment. We proposed a hybrid mechanism where switches are not simply data forwarders in SDN environment. Instead, they can collect statistics and can decide if DDoS attack is in action. Then they coordinate with the controller and decide on attack packets in cooperation. SDNScore is a statistical and packet-based defense mechanism against DDoS attacks in SDN environment. Since it has a statistical scoring method, it can detect not only known but also unknown attacks. In addition, it does not drop all packets in a flow which includes both attack and legal packets, but rather acts on attack packets using packet-based analysis. During these analysis, since it chooses the most appropriate attributes during current attack traffic it gives high accuracy.

In Chapter 6, we utilized joint entropy calculation for DDoS detection and most importantly for the selection of the most appropriate attributes during current attack traffic on SDN environment. Since there is a trade off between SDN paradigm and “capable” switches in SDNScore, we wanted to propose a more elegant model that evades from this trade off. This model is called Joint Entropy Based Scoring Mechanism in SDN environment (JESS). In JESS, most of the burden is on the controller. Switch only generates a profile periodically for the pair determined as the most appropriate attributes during current attack traffic. Nominal and current profile generations for every pair, joint entropy calculations and comparisons, score calculations and rule generations are handled by the controller. Since it is a statistical model, it is not only successful for known attacks but also unknown ones.

Finally, Chapter 7 concludes the thesis by summarizing our contributions and giving future directions.

2. RELATED WORK

In this chapter, initially we discuss related work of filtering based defense mechanisms against DDoS attacks. Then we present SDN specific defense solutions against DDoS attacks.

2.1. Filtering Based DDoS Defense Mechanisms

In this section, several proposed filtering techniques in the literature are presented.

2.1.1. Ingress/Egress Filtering

RFC 2827 defines ingress and egress filtering mechanisms [1]. Ingress filtering suggests filtering of incoming packets to a network whereas egress filtering suggests filtering of outgoing packets from a network. As shown in Fig. 2.1, a network employs ingress and egress filtering in its edge router. Since the router knows the specified range, i.e. its address space, it does not allow any IP addresses out of 220.25.112.0/24. Similarly, the router is aware of the network topology and has knowledge of all incoming IP address borders. Therefore, it will block any incoming IP packets out of allowed IP address range. This mechanism is proactive since it prevents DDoS attack packets from entering the network. Moreover, it is an individual filtering mechanism since it does not need to cooperate with other routers to decide on filters. It can be an effective countermeasure against IP spoofing [14]. However, it is not a comprehensive solution. Because spoofed IP address can also be in the range of allowed IP addresses. Additionally, various DDoS attacks do not need to use IP spoofing. They may use compromised machines as *zombies* and utilize their valid IP addresses. In addition, despite the fact that it can be a promising solution for leaf networks since they have simple structure, it is not trivial for complex networks as it is not easy to get topological information for IP ranges. Besides, it suffers a performance cost on checking ranges for every packet. Finally, the network which contains the router will not have any direct

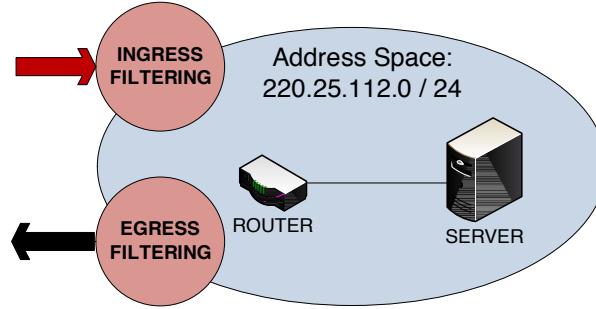


Figure 2.1. Ingress/Egress Filtering [1]. Ingress and egress filters are installed at edge routers. In this example configuration, egress filter does not allow any IP out of 220.25.112.0/24 range. Similarly, since incoming IP address space is known, ingress filters do not allow any “alien” IP.

benefit from egress filtering since it prevents an attack whose victim is in another network.

2.1.2. Route Based Distributed Packet Filtering (RDPF)

In [2], Route Based Distributed Packet Filtering (RDPF) mechanism is proposed. It suggests a filtering model which considers route and network topology information while generating filters. In this model, filters are installed on routers of Autonomous Systems (ASs). ASs are networks that are under control of a single administration mechanism such as universities, large organizations or companies. Each router gets the routing information from BGP routing topology. This mechanism is explained with an example in Fig. 2.2. ASs are depicted as nodes which have their own routers. Edges indicate network communication between these nodes. Suppose that there is an attacker in AS6 which spoofs IP address of AS1. The victim is AS3, which is the destination node of DDoS packets of AS6. If RDPF is installed on AS5, it will block the packets which have source address as AS1 and destination address as AS3. Since AS5 knows the topology, it can conclude that these packets should go on path as AS1-AS2-AS3 and should not pass on AS5. Thus, AS5 prevents the attack targeting AS3. As it is a preventive mechanism, RDPF is proactive. In addition, it is an individual filtering mechanism since it does not need any collaborative communication about filters. Despite the fact that it needs communication for getting information about

routing topology, that activity is specifically for the initialization phase. Routers do not communicate after learning the topology and they do not need to collaborate for filter decisions.

RDPF mechanism can be a promising solution against randomly spoofed IP addresses. However, it cannot prevent intelligently spoofed IP addresses. If IP address of a valid AS is spoofed according to the network topology, RDPF cannot prevent DDoS attack. In addition, this system has a limitation regarding topological changes in dynamic environments such as ad hoc networks. For instance, if AS2 is out of order for some reason, packets from AS1 to AS3 will follow the path as AS1-AS0-AS4-AS6-AS5-AS3. Then in this case, RDPF in AS5 still blocks the packets since it thinks they are attack packets although this time they are legitimate ones. Also, RDPF can only be successful if it is deployed in a significant portion of Internet. In other words, it exhibits the "network effect" where the benefit or utility of a mechanism or technology is exponentially related to the number of adopters or proliferation. Furthermore, this system needs to make some modifications on BGP messages. It is apparent that it is not trivial to change common protocols that are currently in use. Besides, it deteriorates the performance of routers since each packet should be checked. Last but not the least, this mechanism can only block spoofed packets while current DDoS attacks mostly use zombies which do not need IP spoofing.

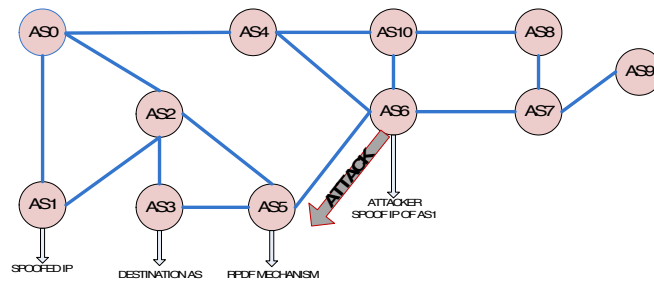


Figure 2.2. Route Based Distributed Packet Filtering (RDPF) [2]. Attacker is at AS6 which spoofs AS1 and Victim is AS3. RDPF installed at AS5 will block packets from AS1 to AS3 since that transmission is not possible in this topology.

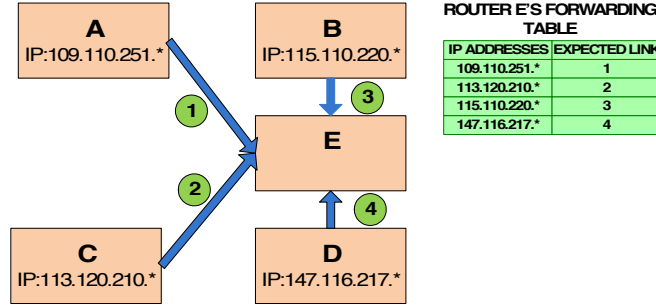


Figure 2.3. Source Address Validity Enforcement (SAVE) [3]. Router E generates forwarding table from SAVE messages. This table contains expected IP ranges from links 1, 2, 3 and 4.

2.1.3. Source Address Validity Enforcement (SAVE)

Li *et al.* propose a defense model called Source Address Validity Enforcement (SAVE) [3] which solves the problem of dynamic changes for routing in RDPF [2]. In this method, source location periodically sends messages with valid source addresses to all destinations. This signaling enables routers to have knowledge of instant and accurate paths, and accordingly IP ranges. As each router knows the expected IP addresses, they prevent packets with addresses out of this range. Routers get information of valid addresses from incoming tables. An example is depicted in Fig. 2.3. In this network, Router E generates its forwarding table according to SAVE messages. Expected ranges of IP addresses from Link 1, 2, 3 and 4 are also shown. If any path is changed, update messages will alter the entries in the table. This model is proactive since it prevents packets with invalid addresses. Also, it is individual as it does not send filters to others or decide filters together. This method is more promising than RDPF due to periodic updates of path information. However, similar to RDPF, it cannot prevent attacks from intelligently spoofed addresses in valid ranges. Also, in order to provide periodic updates, SAVE needs to change existing routing protocols, which incurs high computational and communication costs.

2.1.4. Hop Count Filtering (HCF)

Jin *et al.* propose Hop Count Filtering (HCF) mechanism against DDoS attacks in [15]. This method is based on the idea of considering TTL values of packets. Initial value of TTL is guessed, then current value is decreased from it. As a result, the number of hops that a packet has traveled can be inferred. Guessing initial TTL value is a trivial process since operating systems use a few TTL values which are so far from each other. As hop count can be calculated, a mapping table is created for legitimate IP addresses and their hop numbers. In this method, there are two states: *alert* and *action*. Under normal condition, this mechanism stays in alert state in which TTL behavior is monitored and any packet is discarded. When an attack is detected, HCF mechanism switches to action state in which packets with abnormal hop counts are discarded.

As intervention is carried out after DDoS detection, this mechanism is reactive. In addition, it is an individual filtering mechanism since it does not need to collaborate to decide on filters. HCF is a light-weight, simple and low-storage mechanism. However, it is not an ultimate solution since an attacker can passively monitor and get IP address of a legitimate user and learn about its hop count, and then create a packet with these values. In addition, this model does not deal with dynamic IP addresses which are massively used in today's Internet. Besides, it does not give any solution for NAT (Network Address Translation) devices where several users may get the same IP address. Finally, if the path for a source IP is changed because of congestion or any other reasons, its hop count will change and thus, packets of legitimate users may be rejected.

2.1.5. PacketScore

PacketScore scheme [16] is a statistical filtering mechanism wherein each packet is analyzed according to its attribute values and scores calculated according to these attributes. A packet is announced as *legitimate* if its values are under a dynamic score threshold when they are compared with a baseline profile. This baseline profile

is generated based on Bayesian Theorem [17]. This type of comparison was applied in the detection of DDoS attacks whereas in this method it is utilized for real-time packet filtering against DDoS attacks. It is an individual filtering mechanism since it performs analysis on its own and then determines filters. Also, it is proactive since it blocks according to a scoring approach. This filtering mechanism can differentiate between legitimate and attack packets via statistical analysis. Therefore, it can deal with novel DDoS attack types. Moreover, it works well for non-spoofed attacks since it does not solely have source address attribute, but also other attributes helping to find attack packets. However, it still exhibits some drawbacks. Due to its statistical approach it works well for large volume attacks while it cannot filter low volume attacks. Also, PacketScore scheme needs a baseline profile with no attack phase. But it is not trivial to find a quiet, i.e. attack-free, and sufficiently long period in today's Internet.

2.1.6. Secure Overlay Services (SOS)

According to best of our knowledge, SOS [18] is the first model that is proactive and cooperative. It suggests an onion-like model such that user packets are authenticated in Secure Overlay Access Points which routes the traffic through overlay nodes to beacon nodes. Then, each beacon node forwards the packet to a secret node called Secret Servlet which is known by limited number of entities in SOS architecture. Then Secret Servlet sends the packet to the destination. Destination will only accept the packets that are coming from the Secret Servlet. Otherwise, it will drop them. SOS is a cooperative model since it is distributed over the network and filter decision is obtained cooperatively. It is also proactive since it is constantly active. As far as this mechanism simplifies the rules, it decreases the burden of filtering. However, as it does not use any cryptography, it is easy to find the location of Secret Servlet. When an attacker acquires this information, it can cripple the system easily [7]. Moreover, if a passive attacker listens near the victim, he can easily learn the IP address of Secure Servlet.

2.1.7. Pushback

In [4], Mahajan *et al.* propose a scheme called *Pushback* which rate-limits the aggregated traffic from a congested router to the upstream counterparts. In this scheme, congestion is detected locally at router level. The congestion signature is determined and translated to a router filter. According to the level of this congestion, an appropriate rate limit is determined locally. Then the congested router asks upstream routers to rate-limit the traffic. This pushback operation is propagated to upstream routers. This is the first scheme that proposes collaborative strategy against DDoS attacks. In Fig. 2.4, thick lines show aggregate traffic, whereas dashed lines show pushback request. Each congested router asks only the upstream routers which send large amounts of aggregate traffic to rate-limit. This request will propagate until the source of congestion. Pushback scheme is effective if the attacker traffic traverses a different path from legitimate traffic. Otherwise, legitimate traffic will also be punished since it shares the same link with the attack. Moreover, this technique has high cost since each router along the path between attacker and victim is involved in propagation of information signaling including rate limit. Besides, it does not block all traffic from the attacker. It only limits attack traffic to a fair share. This scheme is cooperative and reactive as it acts cooperatively after traffic congestion.

2.1.8. Active Internet Traffic Filtering (AITF)

Argyaki *et al.* propose a filtering mechanism called *Active Internet Traffic Filtering (AITF)* [5] which uses Route Record scheme to learn the traversing path of each packet. The border routers of each AS participate in recording that path. Filters are generated according to these paths. When a DDoS attack is detected, packets coming from this path are desired to be blocked. Victim does not only think of himself, but also tries to filter out the attack as close to its source as possible. In that way, the attack will be prevented from spreading through the network. In order to provide this structure, collaboration of various routers is necessary.

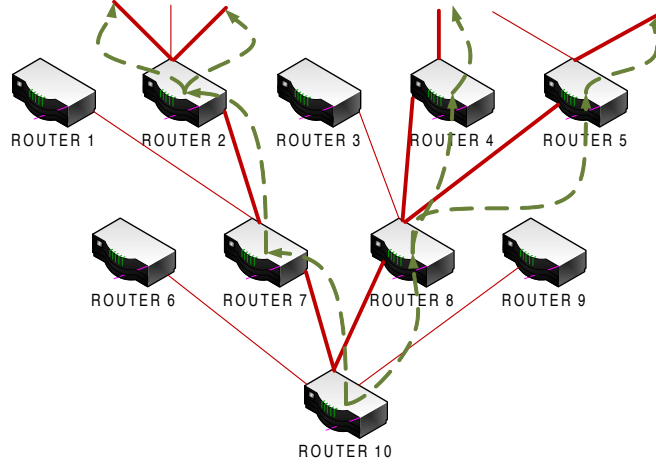


Figure 2.4. Pushback [4]. Router 10 is the congested router. It requests Router 7 and 8 to rate-limit. Then, Router 7 sends request to Router 2 whereas Router 8 sends request to Router 4 and 5. These pushback messages propagate upstream until the source of traffic.



Figure 2.5. Active Internet Traffic Filtering (AITF) [5]. Filter is sent from victim (V) to its gateway (VGW). Then it is propagated to the attacker's gateway (AGW) which requests attacker (A) to stop sending traffic.

If we analyze the network in Fig. 2.5, we can see that a packet sent from A (attacker) to V (victim) will have the record route as $\{A, AGW, VGW, V\}$ where VGW and AGW are border gateways of victim and attacker, respectively. When an attack is detected at victim V, it will generate a filter to stop the packets with this path. This filter request is sent to the gateway of victim (VGW) and VGW blocks these packets temporarily to protect its own AS. Then it forwards this request to the gateway of attacker (AGW) in order to prevent spreading of attack through the network. Accordingly, AGW initially requests the attacker to stop sending packets. If it still continues to send, AGW filters all traffic from the attacker A. The number of accepted filter requests are limited in VGW and AGW since routers have limited capacity for filters. Apparently, this mechanism is cooperative since it requires collaboration for filtering and it is reactive as far as it responds after DDoS detection. Despite the fact that it is

beneficial for preventing DDoS, it still has some drawbacks. First of all, Route Record technique gives some burden in terms of packet length and packet processing. Since packets travel several border gateway routers, they can cause unnecessary fragmentation which result in processing overhead. Additionally, AGW and VGW have limited capacity for filters which can be abused by attackers by sending fake filters. Besides, it requires significant changes in network infrastructure.

2.1.9. StopIt

Liu *et al.* propose a filtering mechanism called StopIt in [6]. This scheme involves a closed control channel which means that each interacting pair knows the identities of each other. Additionally, it allows any destination to stop attack traffic from any source. StopIt system architecture is depicted in Fig. 2.6. In this mechanism, each AS has its own StopIt server that handles filter requests. In a typical flow, victim V installs a filter to block attacker A. V sends a filter request to its access router (Rv). Rv sends this request to its StopIt server. Then, AS3's StopIt server sends this request to AS1's StopIt server. It communicates with the access router of attacker (Ra). Ra installs this filter and sends StopIt request to the attacker. If the attacker does not halt, it will be punished by Ra. Thin arrows in Fig. 2.6 show filter request exchanges between known peers, whereas thick ones show the network traffic flow. After filter installation, the traffic is blocked between A and Ra. This scheme uses Passport system to make secure authentication [19]. Passport puts tokens on packets which allow ASs to verify that source address is valid.

StopIt scheme is a cooperative and reactive approach. The most important advantage of StopIt system is the capability of on-the-fly filter installation during DDoS attacks. Routers do not drop filter requests during attacks since StopIt servers deal with filtering. However, this system requires configuration of routers and StopIt servers to facilitate that they know each other and all hosts before communication. This deployment process is not a trivial task since there are several ASs and each AS may have thousands of nodes.

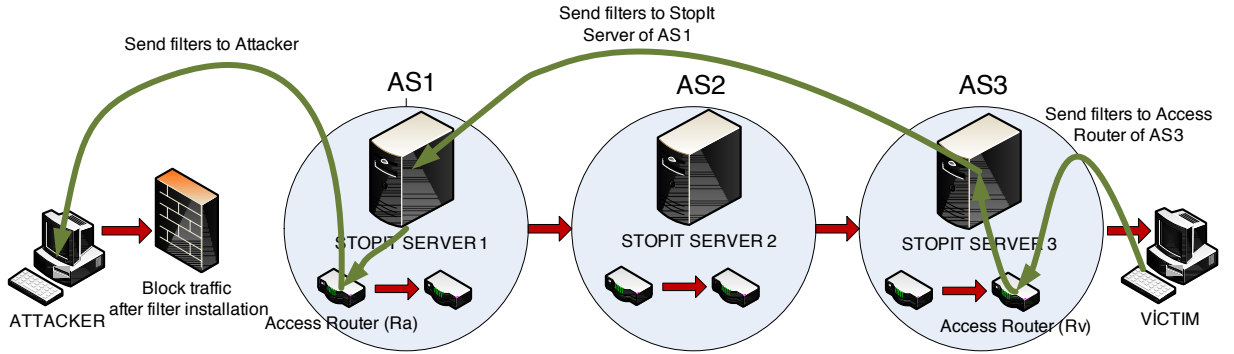


Figure 2.6. StopIt [6]. Victim V sends filter to its access router (Rv) which communicates with its StopIt server. Then StopIt server of AS3 communicates with StopIt server of AS1 in order to reach attacker. StopIt server of AS1 communicates with access router (Ra) which blocks traffic after filter installation.

2.1.10. Probabilistic Filter Scheduling (PFS)

In [20], Seo *et al.* define a filter technique which deals with not only filtering but also its scheduling and how to find best locations for filters. The proposed scheme is called *Probabilistic Filter Scheduling (PFS)*. It has four main phases. In Phase 1, path identification is provided by using Probabilistic Packet Marking technique. A filter router (FR) probabilistically writes its own IP address to IP header of packets. In Phase 2, victim collects all markings and constructs marking values to request a filter. In Phase 3, a filter router which receives several filter requests applies a filter scheduling policy and chooses best k filters. Then, it sends these filters to upstream routers. In Phase 4, as soon as the attack stops, the scores of corresponding filters are decreased and these filters are revoked from filter routers.

PFS phases are depicted in Fig. 2.7(a). FRs 1, 2, 3 and 4 probabilistically mark their addresses as 1, 2, 3 and 4. Large arrow shows more packets which are generated by the attacker. The victim collects 1, 2, 3 and 4, and creates filters by combining these markings. Then it sends filters to FR4. FR4 makes a filter scheduling and chooses best k filters according to scores which are determined by frequency and recency. Filters are propagated from FR4 to FR1 hop by hop. At the end of the attack, FR4 evicts useless filters. PFS scheme propagates filters to optimal location closer to the attack.

Also, it maximizes effectiveness since it chooses best k filters according to frequency and recency of filter requests. These filters are chosen according to the score $S_n(I)$ which is calculated as:

$$S_n(I) = S_{n-1}(I) - \frac{P_{n-t}(I)}{n} + \frac{P_n(I)}{n} - \gamma \quad (2.1)$$

where $S_n(I)$ is the score of the filter I at current time n . Parameter t denotes time window whereas γ is the penalty to decrease filter score in order to be used for filter revocation in Phase 4. $P_n(I)$ is calculated as:

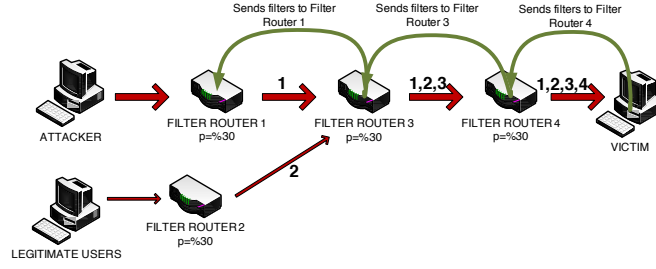
$$P_n(I) = F \cdot m + R \cdot (t_c - t_p) \quad (2.2)$$

where F is the weight of frequency and R is the weight of recency. Parameter m denotes how many times the filter is used while t_c and t_p shows current and previous packet arrival times related to the filter. This equation leads to a dynamic list of filters with different scores, i.e. different suitability. Then best k filters are chosen from this list.

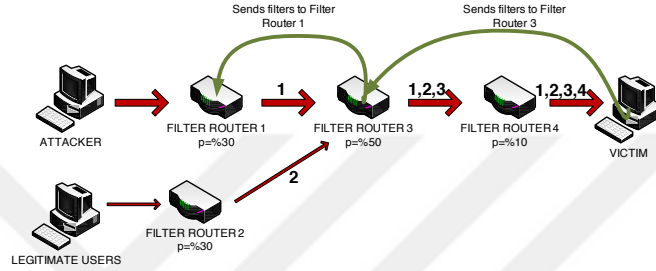
PFS scheme is cooperative since filters are decided and propagated in collaboration. Also, it is reactive since it stops an attack after filter requests which follow the detection of DDoS at victim side. PFS algorithm also induces some burden as overhead. In addition, it requires a security agreement between routers to accept filters. Also, it has marking overwriting since all nodes have fixed probabilistic packet marking. In order to improve this scheme, another approach called *Adaptive Probabilistic Filter Scheduling* is proposed, which is discussed in the following part.

2.1.11. Adaptive Probabilistic Filter Scheduling (APFS)

In [21], Seo *et al.* improve their PFS scheme [20] and propose Adaptive Probabilistic Filter Scheduling (APFS). It addresses filter scheduling problem by adaptive packet marking. As it is mentioned in the previous section, path identification is provided by using probabilistic packet marking. FR probabilistically marks its IP address to IP



(a) Fixed probabilistic filter scheduling (PFS) [20].



(b) Adaptive probabilistic filter scheduling (APFS) [21].

Figure 2.7. Probabilistic filter scheduling schemes.

header of a packet. In fixed marking, all filter routers have same marking probability whereas in adaptive marking all filter routers mark according to their own adaptive probability. This probability is specified by each router based on filtering effectiveness. Filtering effectiveness is determined by three parameters:

- (i) *HOP*: How many hops will it take to reach attacker from this FR?
- (ii) *RES*: How many filters can this FR accept?
- (iii) *DEG*: How many links does this FR have?

According to these factors, the victim will receive more filters from more effective FR(s) and it will choose such FR(s) in filter scheduling phase. In addition, it will propagate the filter to that effective router first which leads to blocking of DDoS attack more rapidly. APFS modifies Phase 1 of PFS to consider the factor of filtering effectiveness. In PFS, most of the markings received by the victim are coming from filter routers that are close to the victim, because victim side routers overwrite markings that are coming from attacker side. Then, this marking overwriting will decrease the effectiveness of defense and cause hop-by-hop filtering. On the other hand, in APFS, as attacker side's marking probability is higher and most of the markings received by

victim are coming from core filter routers. As depicted in Fig. 2.7(b), the most effective filter router will have higher probability which is FR3 in this case. This feature leads to direct propagation of filters to this router which solves hop-by-hop filtering problem. APFS is also a cooperative and reactive scheme similar to PFS.

2.2. Solutions Against DDoS Attacks in SDN Environment

In order to cope with several DDoS scenarios in SDN environment, several solutions are proposed in the literature. In fact, it is a novel research topic that almost all mechanisms are proposed in the last few years. In this section, several solutions are analyzed to examine their properties. According to these properties, the defense mechanisms are classified in several aspects in the next section.

Devoflow [22] is the first example that opposes dumb switches. It suggests that short flows should be handled in switches while only larger flows are sent to the controller. Their results show that Devoflow uses less table space and less communication messages between controller and switch. It achieves less overhead by reducing the need to transfer statistics for short flows and to invoke controller for their setup. The motivation is about improving performance under normal conditions and therefore does not work for DDoS attacks specifically. However, their suggestion is strongly related to defense mechanisms against DDoS attacks.

Avant Guard [23] is a framework to improve the security and resiliency against DDoS and scanning attacks with greater involvement of switches. It introduces two modules on switches: connection migration and actuating triggers. Connection migration proxies TCP SYN requests and classifies them. If these are regarded as legitimate, they are authorized and migrated to the real target. This mechanism provides protection against SYN Flood attacks. Actuating triggers module perceives changes and triggers an event. For instance, if enormous volume of requests is realized, it triggers an event in the controller to insert a flow rule into the flow table. Then, flow rule installation is handled automatically and response time is reduced. The most conspicuous side effect of this mechanism is the performance penalty. Since it utilizes connection

migration, each flow needs to be classified. In addition, this module can only defend against one type of DDoS attack (TCP SYN Flood).

Kokila *et al.* [24] proposes a detection mechanism using Support Vector Machine (SVM) Classifier. Their model deals with the attack scenario whose target is the system resources of the controller. They have also tried several machine learning techniques and their results suggest that SVM outperforms others. However, it takes more time than the other techniques. In addition, this model only deals with detection, it does not give any solution related to mitigation.

The communication between SDN controller and switches can also be the decisive system component enabling attack impact. Scotch [25] is a mitigation method that scales up SDN control plane using VSwitch based overlay. The authors made an experiment to find where the real problem is in case of DDoS attacks or flash crowds. Their results suggest that the bottleneck is the communication from switch to the controller. For this reason, they try to increase the capacity of the communication from switch to the controller. Their model suggests that when physical switch is overloaded, new flows will be tunneled to multiple vSwitches. The most important drawback of this model is that it does not have an actual detection mechanism. It acts same for flash crowds and DDoS attacks. In addition, it drops packets when they exceed a threshold. This means that legitimate packets cannot be preserved in case of flash crowd and DDoS attacks.

Zaalouk *et al.* [26] proposed a detection and mitigation method for attacks. Their model suggests that controller's functions of monitoring and controlling should be decoupled. Communicating with switches for control messages and monitoring traffic on the network at the same time increase the overhead on the controller. They suggest that some type of attacks can be resolved without having access to all packets in the network. These are called "low resolution attacks" such as DDoS attacks, whereas others cannot be detected without accessing all packets such as ARP caching and poisoning. In their architecture, there is an orchestrator who decides which modules should be activated according to the suspicious attack types as low or high resolution.

Then network monitor and controller modules are participated in detection and mitigation according to the commands of the orchestrator. In this model, after an attack is detected it just rate-limits the traffic, it does not completely stop attack traffic. It is not a desirable method for mitigation since the system is still occupied with attack packets.

Another model that suggests splitting the the controller's duties is presented in [27]. Similar to [26], they decouple application monitoring and packet monitoring in the controller. In addition to this separation, they also propose that packet controller should be distributed for security and load balancing reasons. In case of a failure of the main controller, the secondary controller takes charge and plays the same role as the main one. They also suggest another hierarchical model in which controller is separated in two levels consisting of a delegator and lower level controllers that perform specific tasks that are assigned by the delegator. They do not mention about a detection module and their model considers flash crowd and DDoS attack in the same manner which is not desirable.

A feasible method is proposed in [28] for DDoS detection and mitigation in SDN. This work suggests to add a table to the controller. This table stores source IP, number of arrived packets coming from this IP and time-out. They determine nominal values for number of connections and number of packets per connections from traffics of frequent users. Then they suggest that DDoS attackers have less connections and less number of packets than the nominal values. As far as the system detects a malicious traffic, it blocks this traffic. This method does not work for high amounts of traffic since their assumption is that less traffic comes from attackers. If an attacker uses botnets with high traffic amount it cannot cope with it.

Hsu *et al.* proposed a hash-based mechanism that operates in controller to increase scalability of the network [29]. This work performs a hash based round-robin scheduling in assigning the incoming packets in controller. In this model, controller can still serve the switch even it has high amount of traffic due to flash crowd or DDoS attack. It is provided by the hash algorithm which distributes the traffic coming from

crowded switch to several queues in round robin. This model does not have a detection mechanism. It acts same as flash crowd and DDoS attacks. This results in pointless serve for DDoS attack packets in SDN controller.

Another model for controller protection against DDoS attack is suggested by Lim . [30]. Since in SDN environment when the controller is incapacitated, the entire SDN will be unavailable, their model suggests that the most essential aim of a defense mechanism is providing controller's work continuity in case of an attack. They leverage a scheduling based scheme that contains most of the attack traffic at attack ingress switches so that SDN network as a whole can continue normal operation. If one switch is infected by DDoS, normally controller cannot serve to other users. In order to prevent this problem, they create different queues for each switch. Actually this model realizes the opposite idea that was proposed in [29]. The aim of Hsu *et al.* is providing scalability for the controller, whereas [30] aims the continuity of the controller's work. The conspicuous drawback of this work is that it considers flash crowd and DDoS attack in the same manner. It does not have an actual detection mechanism. In addition, since it does not have packet based analysis, it treats same legal and attack packets coming from the same switch.

FlowFence mechanism against DDoS attacks is proposed in [31]. In this model, switches monitor traffic and detects congestion condition by monitoring the bandwidth usage. When a congestion occurs, the switch notifies the controller. Then, the controller requests statistics from every switch that sends flow to the congested link. It classifies the flows and determines badly behaved ones which use more bandwidth. Then, it sends commands to switches to rate-limit bad flows. Their results suggest that it is a fast and simple solution that prevents starvation. However, this mechanism is especially a rate-limit mechanism that does not stop an attack completely.

There are several works that utilized entropy in traditional networks, however there are a few works in SDN. One of them is [32] that proposes an early detection of DDoS Attacks against SDN Controllers. It runs on the controller. It has three main processes for collecting flow statistics, anomaly detection and attack report. Flow

properties are gathered in flow statistics process. Then, statistics are utilized in entropy calculation and anomaly detection is provided by considering the entropy. In this mechanism, entropy is calculated for destination IP address. If entropy decreases under a threshold, DDoS is detected. It enables to determine the victim, but it is not possible to dissociate the legal packets from the attack ones. Their work reduces the frequency of data gathering process and communication overhead between controller and switches. Their results demonstrate that attack detection can be provided in early stages with high accuracy. However, this mechanism only provides detection not mitigation. Another similar model is EBS in [33] that proposes an entropy-based lightweight DDoS flooding attack detection model running in the OF edge switch. This achieves a distributed anomaly detection in SDN and reduces the flow collection overload on the controller. However, there is a trade-off of SDN paradigm vs. capable switches in this model.

In [34], Dharma *et al.* presents their method for DDoS detection and mitigation. Their method considers the time characteristics of DDoS attacks for detection and investigates the pattern of time during DDoS attacks for prevention. Every packet coming to the controller is inspected according to its destination address. If it is not valid or unknown, it is analyzed by flow controller module that is loaded on the controller. It stores that non-valid packet for further inspection. When the number of non-valid packets increases significantly within a certain time, flow collector warns the controller. Then the controller sends new rule to the switches to forward all non valid packets to the flow controller. Flow collector clusters time pattern of DDoS attacks and uses these patterns for prevention. They proposed this model but they did not implement it yet. The problem of this work is about making their decisions by just considering destination IP. If the attacker sends packets to a valid IP, their module will not be activated and DDoS attack detection and prevention cannot be provided.

Katta *et al.* [35] presents a solution for the attack scenario related to the memory of switches. Switches can allow limited number of entries in tables because of memory scarcity. Update policy of these entries is essential in terms of DDoS attacks since attack packets are also dropped or forward according to these entries. This work

proposes a rule update mechanism for switch tables. Their idea is not proposed for the aim of DDoS attacks but their idea is valuable for DDoS mitigation.

In [36], Tri *et al.* presented their work that shows the impact of a DDoS attack on SDN. Their results highlight the importance of managing the flow tables. They propose some suggestions for this problem. Table entry replacement policies should use multiple parameters such as number of packets of a flow entry, generation date of a flow entry and utilization of a flow entry, rather than using just one parameter as earliest expiration time. In addition, controller should have an intermediate module which stores the flow entries temporarily and manages the replacement of flow entries. These suggestions can be utilized as DDoS mitigation methods.

3. ANALYSIS OF DDOS DEFENSE FILTERING MECHANISMS

In Denial of Service attacks, all resources are wasted by the attacker and the use of system services/resources to legitimate users is denied. If this attack is realized by multiple sources, it is called *Distributed Denial of Service (DDoS)* attack. The first documented DoS attack occurred in 1999 at University of Minnesota [37]. It blocked the computer system for more than two days. In 2000, DDoS attacks occurred against major Internet and media companies such as Yahoo, CNN, eBay and Amazon [38]. For these ICT systems, only brief inaccessibility for a short period of time results in huge financial and business losses [39]. For instance, in 2010 Mastercard, PayPal, Visa and PostFinance's websites were shut down with DDoS attacks due to their involvement in banning of WikiLeaks donations [40]. DDoS is not only used for the aim of financial and reputational damage, but also for political matters. There was a DDoS attack on the White House Website in 2002 [41] and in 2009 media web sites were attacked in Belarus because of a political issue between Georgia and Russia [42]. Similarly, in 2009 during Israel and Palestinian fight in Gaza, both sides attacked their websites by utilizing DDoS [43]. Likewise, in 2013 during the Gezi Park resistance in Turkey, many governmental websites were put out of service with DDoS attacks.

3.1. Research areas about DDoS

Due to the increasing size and widespread occurrence of DDoS attacks, DDoS is a major research topic for academicians and practitioners. A potent network security strategy needs to consider DDoS attacks for supporting security efforts. The initial phase of these efforts has focused on how to define and convey features and types of DDoS [44–48]. There are also various works which focus on features, advantages/disadvantages, and classification of the attack tools [45, 48]. DDoS attacks use several tools for depleting the sources of victim systems. Some examples of these tools are Trinoo [49], TFN [50], TFN2K [51], Stacheldraht [52], mstream [53], Shaft [54],

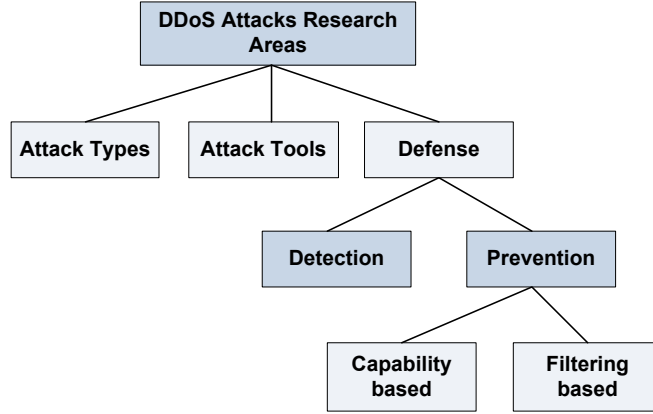


Figure 3.1. Classification of DDoS attack studies.

Trinity [55], and Knight. In consideration of all these efforts, the main aim of DDoS research is to facilitate defense capabilities and devise elements of a security infrastructure which can overcome DDoS attacks. In that regard, the basic defense toolkit against DDoS includes the capability of detection and prevention. In order to see the general view of research studies related to DDoS attacks, a classification scheme is illustrated in Figure 3.1.

3.1.1. DDoS Detection Mechanisms

DDoS attacks generally use packets very similar to the packets of legitimate users which makes it non-trivial to distinguish and detect DDoS attacks. Also, indistinguishability results in false alarms which are termed as *false positives*. Accordingly, a good detection technique should realize DDoS attacks quickly and have a low false positive rate. Since a DDoS attack is a type of intrusion, early classification about DDoS detection techniques was conceived under the topic of Intrusion Detection Systems (IDS). IDSs rely on two types of detection approaches, namely *anomaly-based* and *misuse (signature) detection based* [48]. In anomaly-based detection, a normal behavior is learned from a sufficiently long training period and then abnormal deviation is detected during operation. On the other hand, several exploits known a priori are examined and pattern or signature of these exploits is determined in misuse detection. If any similar pattern is detected, it is marked as an attack. However, it is known that it is difficult to determine a pattern or signature since attackers change type and

Table 3.1. Classification of DDoS Detection Mechanisms [7]

	Properties	Models
Activity Profiling (Statistical techniques)	Profile of a traffic flow is provided by packet header information. Entropy, Chi-Square and Kolmogorov-Smirnov techniques are utilized to obtain distribution.	Entropy and Chi-Square Based Statistical Detection [56], DDoS Detection Method Using Clustering Analysis [57], Kolmogorov-Smirnov Based Statistical Detection [58]
Wavelet Analysis	Anomalies are detected by analyzing signal in terms of spectral components.	Signal Analysis of Network Traffic Anomalies [59], An Improved Wavelet Analysis Method for Detecting DDoS Attacks [60]
Sequential Change-Point Detection	Traffic is analyzed as time series and time statistics show the change-point caused by an attack. CUSUM scheme is utilized to find the change point.	Collaborative Detection of DDoS Attacks over Multiple Network Domains [61], Change-Point Monitoring for the Detection of DoS Attacks [62]
Neural Networks	Machine learning algorithms and visualization techniques are used together. For training, traffic attributes are utilized. Then visualization techniques are employed to show an attack.	Radial Basis Function Based DDoS Detection [63], Learning Vector Quantization based DDoS Detection [64]

content [65, 66]. For this reason, it is generally accepted that it is not efficient to use misuse detection for DDoS attacks [14].

In addition to the basic duality of anomaly vs. signature-based DDoS, other classifications are also proposed for DDoS detection techniques. According to [67], classification can be based on the idea of utilizing IP attributes of packets. *IP-Attributes based DDoS Detection* can use attributes such as source IP address [68], Time-To-Live (TTL) [69, 70], distance [67] and combination of multiple attributes [71] such as protocol type, packet size and server port number. Another type is called *Traffic Volume based DDoS Detection* which analyzes traffic structure and tries to find anomalies according to deviation from normal traffic volume. Several works such as [72] and [73] can be considered in that body of work.

In [74] and [75], DDoS detection techniques are classified based on algorithms that are used for detection. According to this taxonomy, there are three groups: *activity profiling* (statistical techniques), *sequential change-point* and *wavelet analysis*. This classification is extended by Beitollahi *et al.* in [7] to four groups with the addition of *neural networks*. It is summarized in Table 3.1. In activity profiling, packet header information is utilized to construct the profile of a traffic flow. The elapsed time between similar consecutive packets having the same address and port numbers provides average packet rate or so called *activity level*. Then, in order to detect an attack, this reference model is compared with current traffic via statistical techniques. If the traffic behaves differently than this reference model, an attack is detected. In [56], entropy and Chi-Square statistical techniques are used to obtain the statistical distribution of features for flows. Based on these calculations, abnormal behavior of a flow is detected by comparison to normal traffic statistics. In [57], after entropy calculation, Lee *et al.* use clustering techniques to identify anomalies in traffic with respect to normal flows. Considering features of traffic, several variables are determined. These variables are normalized to eliminate the effect of difference between their scales. Normalization is

applied with this formula:

$$z = \frac{x - \bar{x}}{s} \quad (3.1)$$

where x is the value of each variable, \bar{x} is the mean of sample data set and s is the sample standard deviation. Also, in order to measure dissimilarities between clusters, Euclidean distance is utilized, which is:

$$D(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (3.2)$$

where x and y are two records that will be clustered and n is the number of variables. After calculating distance measures, cluster numbers are determined with Cubic Clustering Criterion method [76]. In another work, instead of clustering, Kolmogorov-Smirnov (KS) technique is utilized for computation of relevant statistical distributions [58].

In wavelet analysis methods [59, 60], physical layer is enrolled into analysis as the physical layer signal is handled in terms of spectral components. Anomalies are detected by analyzing the energy of the spectral window. In contrast, traffic is analyzed as time series in sequential change-point methods. Initially, traffic data is filtered in terms of address, port and protocol followed by a representation in time domain. Then traffic statistics pinpoint changes caused by an attack. If the attack started at time λ , the change in time domain will be at λ or greater than λ . In [61] and [62], a cumulative sum (CUSUM) scheme is utilized to find that change-point. It monitors the deviation of short-term behavior from the incumbent long-term behavior. If the cumulative difference is larger than a threshold, attack is detected. Finally, in neural network based methods, visualization techniques are combined with machine learning algorithms in order to detect DDoS. Traffic attributes are used to train neural networks and then visualization techniques are used to show existence of possible attacks. Several neural network algorithms such as Radial Basis Function (RBF) [63] and Learning Vector Quantization (LVQ) [64] are employed for these purposes.

3.1.2. DDoS Prevention Mechanisms

The main functions of intrusion prevention include identifying malicious activities, logging of information about them, actively preventing/blocking them, and reporting these incidents to security administrators. Intrusion prevention systems are placed in-line and can take actions such as sending an alarm, dropping the malicious packets, resetting the connection and/or blocking the traffic from the offending IP address [77]. Prevention mechanisms can be classified into two types as *capability based* and *filtering based* [78]. In capability based mechanisms, sender must obtain explicit authorization from receiver before he sends significant amount of traffic [79]. This authorization is called capability. These mechanisms are analyzed in [79]. On the other hand, filter based mechanisms use traffic filtering which is a very effective and widely employed mechanism in intrusion prevention for network security. In this work, we deal with filtering based methods since there is no comprehensive treatment that focuses on this topic in the literature to the best of our knowledge. Our main goal is to construct an exhaustive categorization of existing filtering methods and provide a detailed comparison. Our work is also instrumental to understand these mechanisms and in choosing the appropriate one for specific contexts and circumstances. In addition, it establishes a baseline view to propose a new filtering method for prevention against DDoS attacks. Firstly, analysis of filtering techniques are provided for traditional networks. Then, it is provided for SDN environment.

3.2. Analysis of Filtering Techniques in Traditional Networks

Detection of DDoS (or any kind of network attack) is supposed to be succeeded by a defense mechanism (countermeasure) for any concrete benefit. To this end, filtering and elimination of “hostile” packets and flows is a very effective DDoS countermeasure to prevent system damage. A filter is essentially a rule which allows or prevents a packet to enter the system [20]. They are generally installed on routers since they allow or block packets before they enter a domain. These mechanisms are vital since they intercept an attack posed to give harm to a large number of machines. In this work, we analyze and classify filtering mechanisms to understand and compare pros

and cons of these methods.

3.2.1. Classification of Filtering Techniques

(i) Collaboration Based Classification: Filtering mechanisms can be classified according to the degree of collaboration. In some circumstances, machines or nodes need to cooperate in order to learn and decide about filters. This type of filtering is called *cooperative filtering* whereas others are called *individual filtering*.

- Cooperative Filtering: Cooperative filtering requires a trusted communication mechanism between collaborative machines. In this type, the most important criterion is to communicate and decide on filters synchronously during defense phase. Communication during initialization phases which is utilized to get information about routing and network topology is not considered as an inclusion factor for this class. In other words, in cooperative filtering mechanisms, machines need to communicate during filtering phase not just for some preparatory information exchange such as Border Gateway Protocol (BGP), routing and IP tabling. These mechanisms interfere with the attack quickly since it has dynamic communication on filtering conditions and rapidly prevents the spreading of the attack in the network.
- Individual Filtering: Individual filtering supports a stand-alone network device. In other words, the filtering mechanism is installed and can be realized on a single machine which decides and creates its own filters. It is easy to deploy since it does not need to cooperate, communicate and trust other machines. In addition, response time against an attack is shorter since it does not suffer latency and communication overhead for cooperation. However, it may be inadequate for large-scale multi-source-to-multi-point attacks.

(ii) Response Time Based Classification: Filtering mechanisms can also be classified according to point-in-time of reaction. A filtering defense mechanism can be active before or after DDoS attack starts. Similar classification is proposed by Hakem *et al.* in [7]. However, it does not focus on filtering mechanisms. Instead, it has classified all countermeasures against DDoS attacks in terms of reaction time.

Table 3.2. Classification of Defense Mechanisms Against DDoS Attacks

	INDIVIDUAL	COOPERATIVE
PROACTIVE	Ingress/Egress Filtering [1], Route Based Packet Filtering (RDPF) [2], Source Address Validity Enforcement (SAVE) [3], PacketScore [16]	SOS [18]
REACTIVE	Hop Count Filtering [15]	Pushback [4], Active Internet Traffic Filtering (AITF) [5], StopIt [6], Probabilistic Filter Scheduling (PFS) [20], Adaptive Probabilistic Filter Scheduling (APFS) [21]

From the filtering perspective, we can have proactive and reactive mechanisms.

- **Proactive Filtering:** Proactive filtering is a preventive mechanism which is enrolled in action before DDoS attack starts. Since it is proactive, it needs to be successful in estimation of incoming malicious packets. It may cause some burden since it always has to be active for prevention. Thus, there is a tradeoff between the burden of prevention mechanism and the impact of DDoS attack.
- **Reactive Filtering:** Reactive filtering is enrolled in action after DDoS attack is started. In this class, DDoS attack detection is the first step in order to trigger the reactive system. Subsequently, reactive filtering participates in the action(s) and prevents DDoS packets to spread through the network. It does not incur extra burden like proactive schemes since it is only active after attack detection.

According to the analysis of several techniques in Chapter 2, all models have their own cons and pros, it is not possible to state that one of these mechanisms is a superior solution for all DDoS attacks. For this reason, network administrators need to choose

appropriate one(s) according to their requirements. In order to provide an easier way to understand and decide, a fundamental classification of these methods is provided in Table 3.2. According to this matrix, there are four main types of filtering based defense mechanisms: individual+proactive filtering, cooperative+proactive filtering, individual+reactive filtering and cooperative+reactive filtering.

Individual proactive filtering provides easy deployment and quick intervention which interferes DDoS attacks before it impairs the network operation. However, it is always active which results in extra burden on performance. If these beneficial properties are more important, Ingress/Egress Filtering [1], Route Based Packet Filtering (RDPF) [2], Source Address Validity Enforcement (SAVE) [3] and PacketScore [16] mechanisms can be adopted. On the other hand, cooperative proactive filtering provides collaborative and preventive mechanisms. When this mechanism is deployed in the network, it gives an opportunity to block a DDoS attack near the source before it expands and affects the entire network. Also, it is more accurate than individual mechanisms since it decides on filters with more visibility and knowledge about the network. However, according to the best of our knowledge, there is only one mechanism SOS [18] that is both cooperative and proactive in the literature. This scenario is challenging since cooperation should be accepted by all peers while no attack is in effect yet. Therefore, this topic deserves more efforts and contributions from security research community.

Individual reactive filtering is active after DDoS attack detection and it provides easy deployment. If we need a mechanism which should be active only for short duration, since we cannot tolerate extra burden and we cannot deploy it through the network, then this type is preferable. Hop Count Filtering [15] provides such properties. On the other hand, cooperative reactive filtering provides collaborative decision after DDoS attack is detected. Pushback [4], Active Internet Traffic Filtering (AITF) [5], StopIt [6], Probabilistic Filter Scheduling (PFS) [20] and Adaptive Probabilistic Filter Scheduling (APFS) [21] are active after detection and filters are propagated through communicated machines. This topic is extensively explored in the literature for practical implementations.

3.2.2. Attribute-based Comparison of Filtering Mechanisms

In this subsection, filtering mechanisms are compared according to four key dimensions, namely their deployment difficulty, communication overhead, scalability and attack prevention efficiency. This information is illustrated in Table 3.3.

Filtering mechanisms are deployed on routers which are not easy to access and require a very high level of availability. Thus, more effort is needed in order to deploy these mechanisms. Deployment difficulty can be an important criterion in filtering mechanism selection. It is relatively easy to deploy the following mechanisms: Ingress/Egress Filtering [52], RDPF [2], SAVE [3], Hop Count Filtering [15] and PacketScore [16]; since only one machine is involved in the system. On the other hand, it is hard to deploy SOS [18], Pushback [4], AITF [5], StopIt [6], PFS [20] and APFS [21] since multiple network nodes and machines are involved in the process.

Communication overhead is another issue that needs to be considered while deciding on appropriate filtering mechanisms. If a filter is constructed via communication of several machines, it is more accurate at the expense of communication overhead. SOS [18], Pushback [4], AITF [5], StopIt [6], PFS [20] and APFS [21] mechanisms decide on filters cooperatively and have higher communication overhead whereas others do not suffer from this burden.

Scalability is the ability of the network to grow in size and handle increasing traffic volumes while still performing at an adequate level of service quality. SOS [18], Pushback [4], AITF [5], StopIt [6], PFS [20] and APFS [21] are more scalable since they can respond to growing number of users and handle them since several machines are utilized in the system. It would be more difficult to tackle this situation with a single machine as in the case for other mechanisms. The distributed nature of these systems allows for different mitigation techniques against high traffic loads.

Finally, attack prevention efficiency (APE) of a defense mechanism is another important feature that needs to be evaluated by network security experts. Attack packets occupy network infrastructure and waste system resources. Especially in DDoS attacks, they prevent the system to work efficiently since the number of packets is very large. If they are stonewalled early in the network, the system performance will improve. Thus, APE metric measures how early a network can get rid of attack packets. It is formally illustrated as in (4.9) where AP is the total number of attack packets, dis_i shows the discard hop of attack packet i and p_i shows the path length of attack packet i .

$$APE = 1 - \frac{\sum_{i=1}^{AP} \frac{dis_i}{p_i}}{AP} \quad (3.3)$$

According to the APE definition and performance results, SOS [18], Pushback [4], AITF [5], StopIt [6], PFS [20] and APFS [21] have high APE since they can stonewall an attack near the source. This outcome is expected since they are more sophisticated mechanisms at the expense of deployment complexity and communication overhead.

3.3. Analysis of Filtering Techniques in SDN Environment

The number of network devices connected to Internet is growing rapidly. It is expected that there will be 1.4 devices per person in 2018 globally [13]. Not only the usage of mobile devices but also newly emerging technologies such as Internet of Things will also increase the connected network devices. Growing network sizes will result in more complex networks and more challenging demands. Existing network technology cannot manage to handle such complex networks.

In order to design future networks that can satisfy the needs of such complex networks, several works have already been proposed and Software Defined Networking (SDN) is considered as the most promising solution among others. SDN's distinctive property is the decoupling of data and control plane in network devices. In traditional networks, routers apply all high level routing algorithms and decides where data packets

Table 3.3. Key Attributes and Comparison of Defense Mechanisms

MECHANISM	DEPLOYMENT DIFFICULTY		COMMUNICATION OVERHEAD		SCALABILITY		ATTACK PREVENTION EFFICIENCY	
	Easy	Hard	Low	High	Low	High	Low	High
Ingress/Egress Filtering [1]	✓		✓		✓		✓	
Route Based Packet Filtering (RDPF) [2]	✓		✓		✓		✓	
Source Address Validity Enforcement (SAVE) [3]	✓		✓		✓		✓	
Hop Count Filtering [15]	✓		✓		✓		✓	
PacketScore [16]	✓		✓		✓		✓	
SOS [18]		✓		✓		✓		✓
Pushback [4]		✓		✓		✓		✓
Active Internet Traffic Filtering (AITF) [5]		✓		✓		✓		✓
StopIt [6]		✓		✓		✓		✓
Probabilistic Filter Scheduling (PFS) [20]		✓		✓		✓		✓
Adaptive Probabilistic Filter Scheduling (APFS) [21]		✓		✓		✓		✓

should be forwarded. In SDN, decision and forwarding functions are separated. Hard decision process is provided by SDN controller whereas data forwarding is handled by switches. Since decision algorithms do not run on network devices, more simple devices such as switches can be utilized rather than routers. The simplification of network devices and central management are the most attractive properties for vendors. Since they have thousands of network devices, they will have enormous reduction in their costs. In addition, in traditional networks, each router has its own security, link failure and forwarding mechanisms. If any of these mechanisms needs to be updated, each network device should handle these tasks individually. However, in SDN you can manage all these problems from a center. What used to take 18 months in traditional networks, takes minutes in SDN [80].

Despite the fact that SDN has its obvious advantages as simplification and flexibility in the network, there are some important challenges that worth consideration [81] which can be listed such as reliability, scalability, latency and controller placement. Despite the fact that SDN is an open door for new threats, there is a limited industry and research community discussion on security issues of SDN [82].

SDN's vulnerability for malicious users are derived from two inherent properties of SDN: the ability to control the network by means of software and the centralization of network intelligence in the controller [82]. These features results in several trust issues and single-point management failures. In order to provide trust between applications and controllers, and between controllers and network devices, authorization policies and authentication mechanisms should be applied. One point management failure can be provided by harming availability of SDN controller. One of the most common ways that can be utilized for such attack is Distributed Denial of Service Attacks (DDoS).

The simple strategy behind a DoS attack is to deny the use of the system services/resources to legitimate users and degrade the system availability. The fundamental mechanism for DoS attack execution is to send a flood of superfluous network traffic to the target so that it cannot respond to genuine requests for services or information. If multiple sources are used by the attacker(s), that type is called as Distributed De-

nial of Service (DDoS) attack which is much more catastrophic than DoS [83]. DDoS attacks are also viable for SDN.

One of the vulnerabilities of SDN architecture regarding DDoS attacks is the limited passive capabilities of switches. Since they send all packets with unknown IPs to the controller, their medium becomes attractive for DDoS attacks. Moreover, they do not possess sufficient resources for very large volumes of traffic. On the controller side, because of central management property, DDoS attack can cause catastrophic results if the controller is saturated with attack traffic.

In the literature, the initial relationships between SDN and DDoS were constructed for providing defense mechanisms against DDoS attacks by utilizing SDN [84]. However, it was realized that SDN has its own vulnerabilities that can attract DDoS attacks and it needs security also for itself. For this reason, DDoS and SDN related works can be categorized as *SDN for security* and *security for SDN*. Latter category especially security against DDoS attacks in SDN environment is a more recent area. There are some works proposed for DDoS defense in SDN. However, it is still an immature area since there is no outstanding solution and all solutions have some drawbacks. According to the best of our knowledge, this will be the first comprehensive survey that focuses on security for SDN in terms of DDoS attacks and classifies these works in several aspects. Our main goal is to construct an illustrative categorization of existing defense mechanisms in SDN and make it easier for network experts to choose a mechanism for specific contexts and circumstances. In addition, we establish a baseline view in order to propose a new defense mechanism against DDoS attacks on SDN.

3.3.1. DDoS attack scenarios on SDN

All types of DDoS attacks such as UDP , ICMP or TCP /SYN flooding attacks, NTP amplification or ping of dead attacks are also viable in SDN environment. Since SDN infrastructure suggests a centralized management for network flows, SDN is very attractive for DDoS attackers. Network flow policy of SDN suggests that when a packet comes from an unknown IP to a switch, it is forwarded to the controller. Then, the

controller sends a flow rule to the switch for this IP. If attackers send a large number of packets from several IPs, each packet will be forwarded to the controller. Then a huge number of attack packets will use all available resources and make the system unavailable for legal users. A simple topology for SDN is illustrated in Figure 3.2. Some DDoS attack scenarios for SDN are scripted as follows.

- S1: Controller can be the target for the attack. Attacker can generate traffic with spoofed IP addresses. In this type, attacker(s) are under the switch SA who needs to send all packets to the controller as they are coming from unknown IP. Then the link between SA and the controller is congested. (Target: $L1$, Attacker: $A1$)
- S2: Controller's system resources can be the target for attackers. This time attackers are under different switches managed by the same controller. Since the attack traffic is coming from several switches attack load is divided and it will be more difficult to detect it. (Target: C , Attacker: $A1$ and $A2$)
- S3: Switch memory can be the target for attackers. A switch have limited memory and it needs to store a new entry for each flow from different IPs. When attacker generates new flows, controller sends new entries. If attacker uses all available table entries, the legal users' traffic from new IPs cannot be served. (Target: SA , Attacker: $A1$)
- S4: A legal user under a switch can be the victim for an attacker (i.e. a server). Attacker can be in the same switch or another switch. If the controller or the switch cannot detect it, the server's resources will be depleted. (Target: $U1$, Attacker: $A1$ or $A2$)

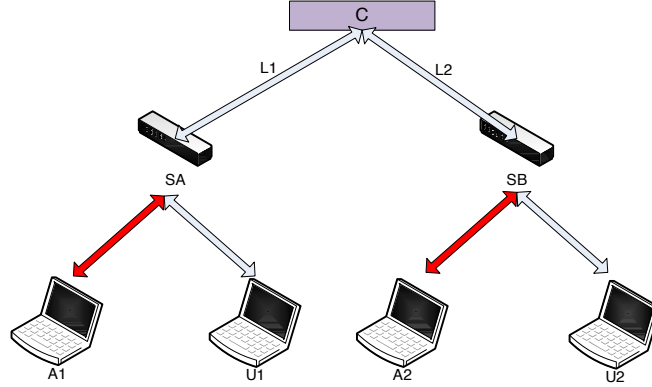


Figure 3.2. A sample SDN topology with DDoS attackers

Compared to traditional networks, software-defined networks can become more prone to DDoS attacks due to three inherent dynamics of attacks as seen in these scenarios:

- Propagation of attacks: Simpler processing pipelines in switches compared to conventional routers render SDN more prone to attack propagation. The controller has to step up to hinder such dynamics.
- Aggregation of attacks: Attack traffic in terms of payload and control traffic aggregates towards SDN controller.
- Widespread impact of attacks: A DDoS attack can rapidly affect the entire network by crippling controller(s). This phenomenon is boosted via two characteristics described above.

While these aspects pose SDN DDoS attacks more challenging, they also provide opportunities for potential defense approaches which are listed and described in the next section.

3.3.2. Classifications of Solutions Against DDoS Attacks in SDN

Several solutions and their properties are displayed in the previous section. Since all models have their own cons and pros, it is not possible to state that one of these mechanisms is a superior solution for DDoS attacks. For this reason, network administrators need to choose appropriate one(s) according to their requirements. In order to

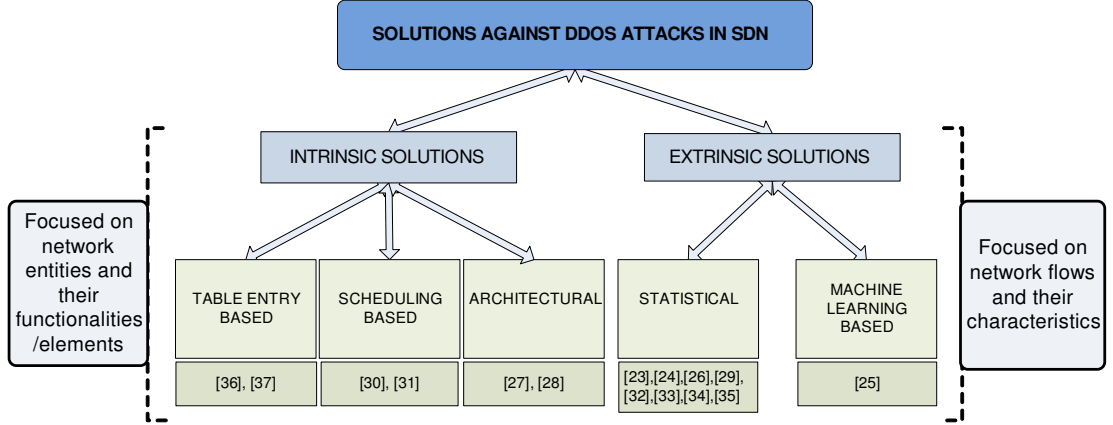


Figure 3.3. Classification of solutions against DDoS Attacks in SDN

provide an easier way to analyse and decide, classifications of these methods in terms of several aspects are provided in this section.

Solutions in the literature can be classified as if they are intrinsic or extrinsic. A property that is inherited and essential are named as intrinsic whereas a property that varies depending on exterior factors are called extrinsic. In our case, some solutions are related to structural property of SDN environment, whereas others are mostly related to the properties of the flows. For this reason, we propose to classify these mechanisms as intrinsic and extrinsic solutions. This classification is illustrated in Fig 3.3.

Intrinsic solutions can be further categorized as table-entry based solutions, scheduling based solutions and architectural solutions. Table-entry based models propose solutions related to the limited table size of switches. Each unknown IP address needs a new entry in switch memory. This becomes a bottleneck during a DDoS attack which has packets with different IP addresses. [35] and [36] suggests some solutions for this problem.

Scheduling based solutions are implemented on the controller. These models suggest that it is essential to protect controller since it is the core of the system in SDN. In order to provide this property, their models deals with scheduling assignment of tasks from switches. The approach in [29] provides scalability whereas [30] provides isolation of the attack traffic with the help of scheduling algorithms.

Table 3.4. Classification of Solutions According to Defense Functionality and Switch Intelligence

Solution Property	Intelligence	
	Capable Switches	Dumb Switches
Detection	[33]	[24] , [32]
Mitigation	–	[25], [27], [29], [30], [35], [36]
Detection & Mitigation	[23], [31]	[26], [28], [34]

Architectural solutions are related to the hierarchies and roles of network elements for solutions against DDoS attacks. [26] and [27] suggest decoupling monitoring and controlling properties of the controller. They also propose a master that manages these modules of the controller.

Extrinsic solutions do not deal with network modules of SDN. Instead, their solutions are related to the properties of the flows. Some researchers focus on identifying the malicious flows in order to defend against DDoS attacks. These solutions can be examined in two categories as statistical based solutions and machine learning based solutions. [22], [23], [25], [28], [31], [32], [33], and [34] models are all statistical based models which generate baseline profiles with some statistical values that are collected at attack free period and attack packets are eliminated by comparing with these profiles. Similarly, the machine learning based solution in [24] trains the mechanism with attack free packets, then classifies attack packets with the help of machine learning algorithms.

Another classification of solutions is illustrated in Table 3.4. Some of these solutions are dealing with detection, whereas some of them are only focused on mitigation, whereas others provide both mitigation and detection. Network moderators can determine their needs and then choose any of these models in the corresponding category. If they have enough resources, a system will be recommended to have both detection and mitigation. Also administrators can make combination of detection and mitigation models if their needs cannot be provided with an existing detection and mitigation model. Columns of Table 3.4 shows intelligence property of the switches in proposed solutions. Some of the works in the literature suggest to bring some intelligence to SDN

switches. The motivation for this approach is based on the idea of keeping flows in the data plane as much as possible. When more flows are forwarded to the controller, it is more prone to attacks by malicious users. If switches become more capable on flow decisions, it will be safer for the controller. Facilitating switches with some minor intelligence features does not compromise the main paradigm of SDN. However, the system designer should be cautious for this trade off of SDN paradigm vs. "capable" switches.

3.4. Chapter Summary

In this work, we present a comprehensive treatment on filtering based DDoS defense mechanisms. First, we explain some properties of DDoS detection mechanisms. Then we give overall description of several DDoS defense techniques and their analysis. This work specifically proposes a classification approach for filtering techniques. They can be classified according to their timing and collaborative properties. Regarding temporal characteristics, they can be proactive or reactive according to defense action time. Also, they can be individual or cooperative. Several filtering techniques are analyzed in details and their trade-offs are presented. According to this classification, it is realized that there is not much work which is both proactive and collaborative. This type is important since it will prevent attack to expand near the source. In addition, it will have more accurate filters, since it decides on filters cooperatively in consideration of instant information of several parts of the network.

Moreover, we present a comprehensive treatment on solutions against DDoS on SDN environment. First, we explain some DDoS attack scenarios that can be viable in SDN environment. Then we give overall description of several DDoS defense techniques and their analysis. This work specifically proposes classification of these solutions in terms of several aspects. Initially, these solutions are categorized as intrinsic and extrinsic. Intrinsic solutions have also sub categories as table-entry based solutions, scheduling based solutions and architectural based solutions. Extrinsic solutions have also sub categories as statistical based solutions and machine learning based solutions. According to this classification, it is realized that there is not much work in machine

learning based solutions. Since machine learning is a broad area, its techniques can work up for SDN environment. Second classification focused on solutions as if they provide detection, mitigation or both. The last classification presents which types of switches are utilized in their model. In order to keep flows as much as possible in data plane, some models suggest to give some intelligence to switches. These types use capable switches whereas others use dumb switches.



4. A STATISTICAL FILTERING DEFENSE

MECHANISM: ScoreForCore

In our previous work, we have analyzed and classified several filtering mechanisms. Our classification is based on reaction time and collaboration. A mechanism can be reactive or proactive in terms of reaction time where as it can be individual or cooperative in terms of collaboration feature. Individual reactive filtering is active after DDoS attack detection and it provides easy deployment. If we need a mechanism which should be active only a small amount of time, since we cannot tolerate extra burden and we cannot deploy it through the network, then this type will be preferable. Hop Count Filtering [15] provides such properties. On the other hand, cooperative reactive filtering provides collaborative decision after DDoS attack is detected. Pushback [4], Active Internet Traffic Filtering (AITF) [5], StopIt [6], Probabilistic Filter Scheduling (PFS) [20] and Adaptive Probabilistic Filter Scheduling (APFS) [21] are active after DDoS attack is detected and filters are propagated through the communicated machines. Individual and proactive filtering provides easy deployment and quick intervention which interferes DDoS attacks before it impairs the network operation. However, since it is always active, it results in extra burden on performance. Ingress/Egress Filtering [1], Route Based Packet Filtering (RDPF) [2], Source Address Validity Enforcement (SAVE) [3], PacketScore [16] are some examples of individual and proactive filtering. Cooperative proactive filtering provides collaborative and preventive mechanisms. It provides an ability of forestalling the attack packets before they expand. Also, it gives more accurate decisions since they create filters cooperatively with more knowledge of the network. Nevertheless, according to the best of our knowledge there are a few mechanisms that are both cooperative and proactive in the literature. It can be challenging since cooperation of all peers are needed while there is no attack. However, if it is possible to make it efficiently and increase the accuracy significantly, peers will cooperate willingly. Thus, we think that this type of filtering should be analyzed in details and more models should be proposed. Our model ScoreForCore [85] is a cooperative and proactive model that increases accuracy considerably.

Table 4.1. Pair Nominal Profile (*OPNP*) Example

TTL Value	Destination Port Number	Number of Packets
48	25	125
48	53	175
48	80	19
50	80	42
...

4.1. ScoreForCore Mechanism

As we deal with distributed systems while we are proposing a cooperative filtering mechanism, we decided to work on an existing and promising individual+proactive model called PacketScore [16]. This model suggests a scoring method based on Bayesian Theorem [17]. Each packet's score is calculated by considering its attributes. If its score is less than a threshold it is regarded as attack packet and discarded, otherwise it is forwarded. In our model, we utilize the same scoring method. Nevertheless, in our model routers create profiles with the most appropriate attributes for current attack which results in much more accuracy. In that sense, we need to consider the following issues: profiling, comparison, collaboration, score calculation, threshold calculation and selective discarding. These issues are explained in the following subsections.

4.1.1. Profiling

Each router generates nominal profiles during an attack-free period. It counts number of packets that have the same attribute value. Following properties are considered as attributes: *IP address*, *port number*, *protocol type*, *packet size*, *TTL value* and *TCP flag*. In order to create a pair attribute nominal profile, router chooses random two attribute pairs named as *OwnPairs* to generate *OwnPairList*. At the end of this period, each router has single nominal profiles *SingleNP* for each attribute and its own pair nominal profiles *OPNPs* created considering *OwnPairList*. An example of a nominal profile *OPNP* for a pair of attributes is illustrated in Table 4.1.

Table 4.2. Single Nominal ($SingleNP_{protocol}$) and Single Current $SingleCP_{protocol}$

Profiles of Protocol Attribute			
Protocol Name	Number of Packets	Protocol Name	Number of Packets
TCP	85	TCP	185
DNS	10	DNS	17
UDP	4	UDP	8
NTP	1	NTP	1
...

Table 4.3. Single Nominal and Single Current Profiles of Packet Size Attribute

PacketSize	Number of Packets	Packet Size	Number of Packets
66	38	66	42
60	29	60	35
83	3	83	12
159	1	159	7
90	3	90	3
...

In an attack period, when a congestion is detected, packet based analysis are activated. For each attribute, current number of packets is calculated and single current profiles $SingleCPs$ are generated.

4.1.2. Comparison

In order to calculate the score of a packet, current pair profile PCP and nominal pair profile PNP are needed. In a router, the pair that will be utilized in current profiling is called $ScorePair$ whereas the current profile that is generated from $ScorePair$ is called PCP . The router should determine $ScorePair$ in order to generate PCP . The router's aim is to determine $ScorePair$ as the most appropriate attribute pair for current attack. For this aim, the router compares single nominal profiles $SingleNPs$ with single current profiles $SingleCPs$ that are generated in the previous step. The examples of $SingleNP$ and $SingleCPs$ of protocol type and packet size attributes are illustrated in Table 4.2 and 4.3.

In this step, the main aim is to determine the two specific attributes that have the most deviation from the nominal profiles. Maximum deviation Dev_A of an attribute A can be determined by finding the maximum deviation of all values for A . For instance an attribute A takes $\{a_1, a_2, a_3, \dots, a_n\}$ values, then initially all deviations should be determined for all values and maximum of these deviations can be found as follows $Dev_A = MAX\{dev_{a1}, dev_{a2}, dev_{a3}, \dots, dev_{an}\}$. dev_{an} is the difference between the number of packets with $A = a_n$ in current profile, $np_{cp_{an}}$, and the number of packets with $A = a_n$ in nominal profile, $np_{np_{an}}$. It can be formulated as $dev_{an} = np_{cp_{an}} - np_{np_{an}}$.

After maximum deviation Dev_A is determined for each attribute, then the attribute that have the largest deviation is chosen as $Dev_{MAX1} = MAX\{Dev_A, Dev_B, Dev_C, \dots, Dev_M\}$ in which M is the number of attributes, in our case it is 6. Subsequently, second largest deviation is chosen as $Dev_{MAX2} = MAX_2\{Dev_A, Dev_B, Dev_C, \dots, Dev_M\}$. And lastly, the router generates suspicious pair from the attributes that gives Dev_{MAX1} and Dev_{MAX2} . This pair is chosen as the most probable signs for ongoing attacks. Thus, it is called as *SuspiciousPair*.

An example of *SingleNP* and *SingleCP* of an attribute protocol type is illustrated in Table 4.2. The most deviation of protocol attribute is provided by TCP value and $Dev_{Protocol} = 100$. Similarly, Table 4.3 shows single profiles of packet size. The most deviation of packet size is provided by 83 and $Dev_{PacketSize} = 9$. All remaining attributes are analyzed in a similar way and all deviations as Dev_{ttl} , $Dev_{portNumber}$, $Dev_{IPnumber}$ and $Dev_{TCPFlag}$ are determined. Then, the two attributes that have the most deviations regarded as *SuspiciousPair*. The router should find a way to access suspicious pair nominal profile named as *SPNP* if its *OwnPairList* does not contain the *SuspiciousPair*.

4.1.3. Collaboration

After *SuspiciousPair* is determined by the router, it checks if it has this pair. If, this is the case, then it uses its *OwnPair* as *ScorePair* and *OPNP* as *PNP*. Also it generates *PCP* according to this pair. Otherwise it needs to communicate with its

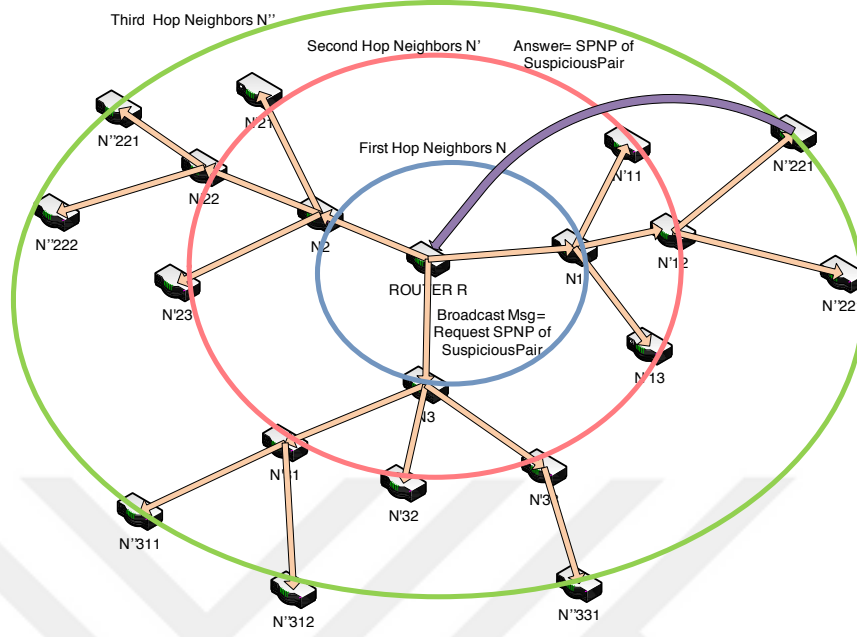


Figure 4.1. Collaboration in ScoreForCore

neighbors. It will ask its neighbors for *SuspiciousPair*'s nominal profile *SPNP*. The owner of the queried pair sends *SPNP* to the router, and then it is utilized as *PNP*. Also, the router determines *ScorePair* as *SuspiciousPair*. Otherwise, the neighbors ask their neighbors. In other words, second hop routers are utilized in order to access *SPNP* of *SuspiciousPair*. If second hop neighbors does not have this pair, then third hop routers are utilized. If *SuspiciousPair* cannot be accessed even in third hop neighbors then the router have to use its *OPNP* as *PNP* and determine *ScorePair* as its *OwnPair*. A sample braodcast messaging is illustrated in Figure 4.1. Router *R*'s *SuspiciousPair* is owned by a third hop neighbor. These operations on Router *R* can be expressed as in the following pseudo code.

Ensure: *SuspiciousPair* is determined by *R*

if *SuspiciousPair* \in *OwnPairList_R* **then**

ScorePair_R = *OwnPair_R* and *PNP_R* = *OPNP_R*

else if *SuspiciousPair* \in *OwnPairList_{NR}* for all *NR* \in *N* **then**

ScorePair_R = *SuspiciousPair* and *PNP_R* = *OPNP_{NR}*

else if *SuspiciousPair* \in *OwnPairList_{NR'}* for all *NR'* \in *N'* **then**

ScorePair_R = *SuspiciousPair* and *PNP_R* = *OPNP_{NR'}*

else if *SuspiciousPair* \in *OwnPairList_{NR''}* for all *NR''* \in *N''* **then**

ScorePair_R = *SuspiciousPair* and *PNP_R* = *OPNP_{NR''}*

else

ScorePair_R = *OwnPair_R* and *PNP_R* = *OPNP_R*

end if

Figure 4.2. Pseudo Code for Collaboration of ScoreForCore

In this pseudocode, subscript in each definition shows the router that owns this property. For instance *ScorePair_R*, *OwnPair_R*, *PNP_R* are all belong to Router *R*. *OPNP_R*, *OPNP_{NR}*, *OPNP_{NR'}* and *OPNP_{NR''}* are the own pair nominal profiles of Router *R*, *NR*, *NR'* and *NR''*. Similarly, *OwnPair_R*, *OwnPair_{NR}*, *OwnPair_{NR'}* and *OwnPair_{NR''}* are the own pairs of *R*, *NR*, *NR'* and *NR''*. *NR* is a router in *R*'s neighbor list *N* whereas *NR'* is a router in second hop neighbor list *N'*. Similarly, *NR''* is a router in third hop neighbor list *N''*.

At first glance, collaboration of the neighbors appears like pointless, since it gives a communication burden for neighbor routers. But indeed, it is a win win negotiation, since attack traffic will be blocked near the source and neighbors' resources will be protected. For this reason, neighbors also want to collaborate with the router in trouble.

After *ScorePair* is determined by collaboration, the router starts to generate its current pair profile *PCP* according to *ScorePair*.

4.1.4. Score calculation

After the collaboration period, the routers have PNP and PCP tables according to $ScorePair$. Each packet's score is calculated considering PNP 's corresponding value. If $ScorePair$ is determined as A and B , then packet p with the attributes $A = a_p$ and $B = b_p$ will have the score S_p as follows:

$$S_p = \frac{PCP_{(A=a_p, B=b_p)}/TPCP}{PNP_{(A=a_p, B=b_p, \dots)}/TPNP} \quad (4.1)$$

In (6.7), $PCP_{(A=a_p, B=b_p)}$ corresponds to the number of packets in current profile that have the property of a_p for attribute A and b_p for attribute B . $TPCP$ is the total number of packets in current profile. Similarly, $PNP_{(A=a_p, B=b_p)}$ is the number of packets in the nominal profile that have the property of a_p for attribute A and b_p for attribute B . $TPNP$ is the total number of packets in nominal profile.

4.1.5. Threshold calculation

The score of a packet needs to be compared with a threshold Th . All scores are stored in a $ScoreList$ and the threshold value Th is determined according to the cumulative distribution of scores by using load shedding algorithm [86]. It is shown as symbolically $CDF(Th) = \Phi$ whereas Φ is the ratio of traffic that should be dropped. The fraction of traffic permitted to pass is $1 - \Phi = \frac{\phi}{\psi}$ whereas ϕ acceptable traffic and ψ is the total current incoming traffic.

4.1.6. Selective discarding

Each packet's score value is compared with the threshold. If it exceeds the threshold, this packet is supposed to be malicious and discarded. Otherwise, it is forwarded to the destination.

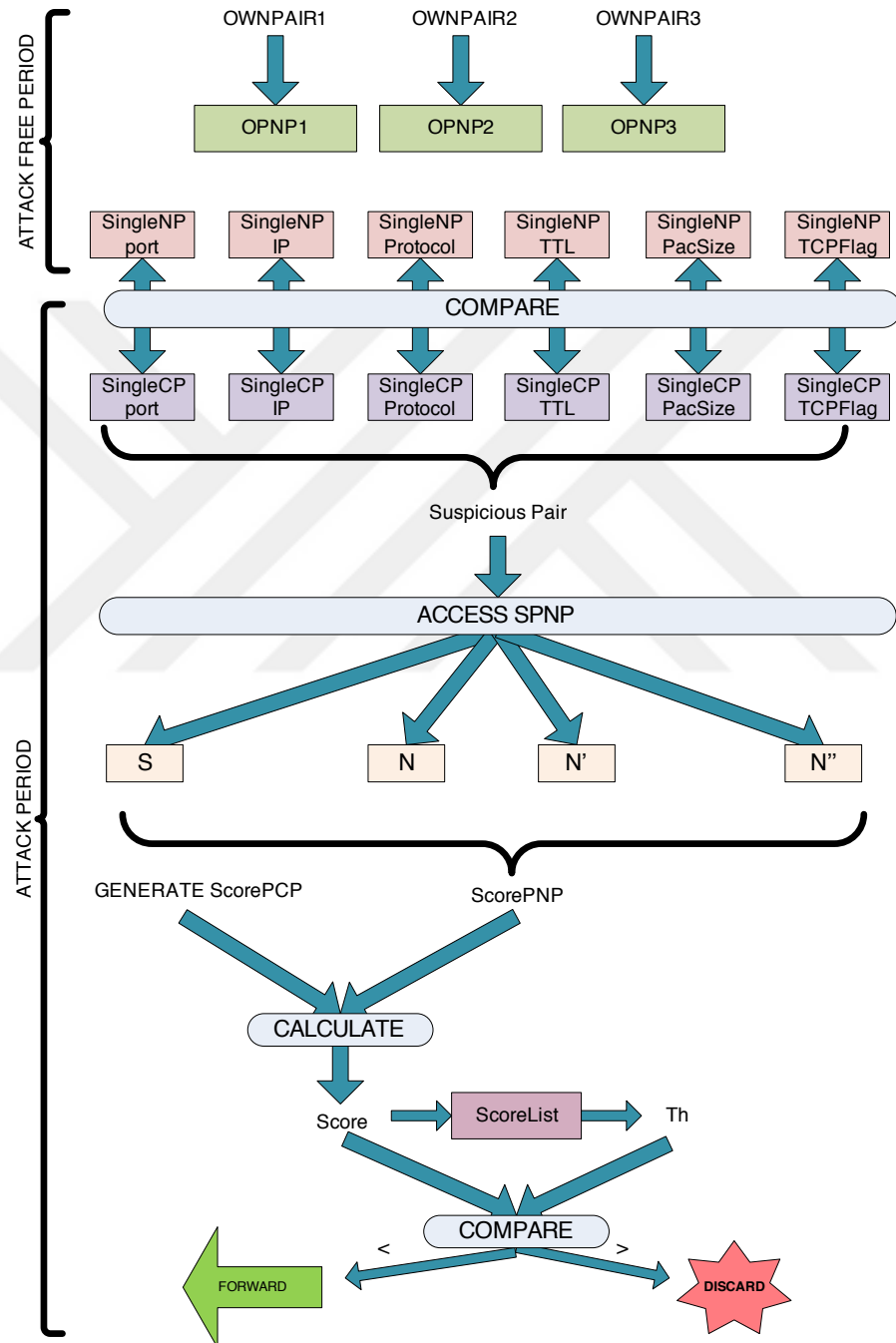


Figure 4.3. Operations in ScoreForCore

To sum up, all these operations are demonstrated in Figure 4.3. In this figure, each router has three OPNPs and *SuspiciousPair* is requested from until third hop neighbors. All the operations in each router can be itemized as follows:

- In an attack free period, *SingleNPs* are created for each attribute whereas *OPNPs* are generated by considering randomly chosen *OwnPairs*.
- After congestion detected, *SingleCPs* are generated for each attribute.
- *SingleCPs* and *SingleNPs* are compared for each attribute and two attributes that have the most deviation from nominal profiles are determined as *Suspicious Pair*.
- The router itself S , its neighbors N , second hop neighbors N' and third hop neighbors N'' are queried for *SuspiciousPair*. If any of the routers in neighbor list have the queried pair, it sends corresponding pair nominal profile to the router. Then, *ScorePair* and *PNP* are determined by collaboration.
- *PCP* is generated according to *ScorePair*.
- Each packet's score is calculated by considering *PCP* and *PNP*.
- If the score is under the threshold, it is forwarded otherwise it is discarded.

4.2. Dataset and Simulation Details

In this section, all the details about the dataset and simulations are presented.

4.2.1. Dataset

In our work, we need real data of Internet traffic. We use real dataset from MAWI Working Group Traffic Archive [87]. MAWILab works on traffic measurement analysis in long-term on global Internet. It was started in 2002 and it is still collecting data from Internet. The part of the data that we have used in our simulations are collected on Jan 12, 2014. This data is utilized to generate nominal profile.

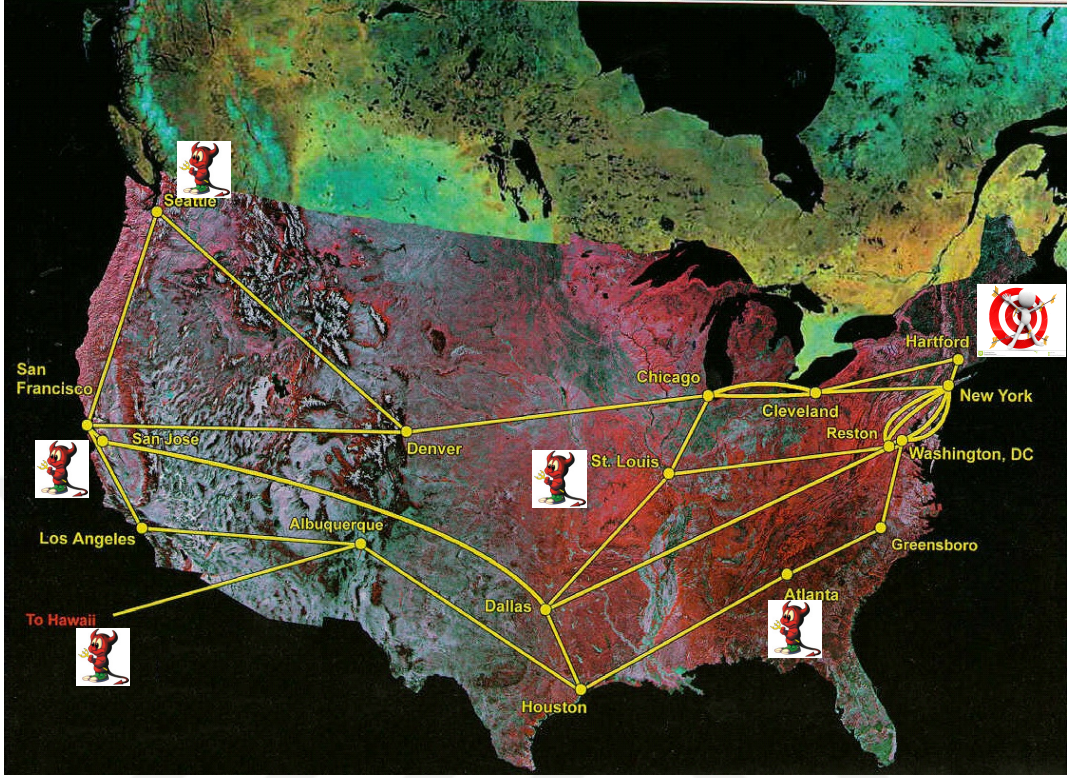


Figure 4.4. Network Topology utilized in ScoreForCore simulations

4.2.2. Simulation Environment

We simulated our model ScoreForCore and existing model PacketScore [16], in order to compare their performances. The simulation programs are written in C++. Test environment is a 3.3 GHz Intel Core i5 processor with 4 GB memory.

4.2.3. Network Generation

In order to perform our simulation, we need to generate a network that is made up of routers. Firstly, we have utilized a real topology that shows the backbones in United States [88]. It is demonstrated in Figure 4.4. It is made up of 18 nodes and average number of neighbors for each node is three.

4.2.4. Attack Generation

According to the network topology in Figure 4.4, attack sources and the victim can be scattered. Suppose that backbone router at Hartford is exposed to a DDoS attack. Botnets are attacking from Seattle, St. Louis, Hawaii, San Jose and Atlanta. Other backbone routers are also sending some legitimate traffic to this victim. Legitimate routers generate legitimate traffic that have similar properties and similar amount of traffic to nominal profile packets. Nominal profile packets are created according to the dataset. Attack nodes generate both legitimate and attack traffic. In PacketScore scheme [16], they made comparison in terms of time based vs. packet based period determination. Their results suggest that packet number based intervals are more suitable. Because time-based window may not allow creating meaningful profiles since it may not have as many as packets. In our simulations, for each period, 5000 legal, 50000 attack packets are generated. We perform attacks by creating new packets that are similar to non-attack period's traffic. We simulate the following attacks:

- *Generic Attack*: All attributes are selected randomly in their ranges.
- *TCP SYN Flood Attack*: The protocol type of an attack packet is TCP and TCP flag is set to SYN flag. Other attributes are randomized.
- *SQL Slammer Worm Attack*: The protocol type of all attack packets is UDP and destination port is set to 1434. Also, packet size is between 371- 400 bytes. Other attributes are randomized.
- *DNS Amplification Attack*: The protocol type of attack packets is DNS and destination port is set to 53. Also, attack packet size is 60 bytes. Other attributes are randomized.
- *NTP Attack*: The protocol type of attack packets is NTP and destination port is set to 123. Also, attack packet size is 90 bytes. Other attributes are randomized.

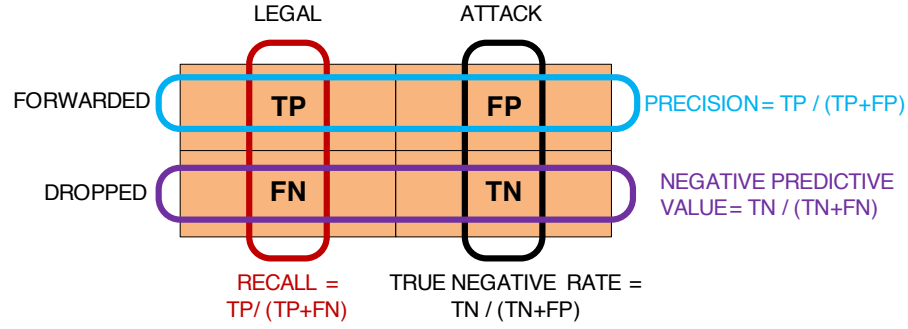


Figure 4.5. Metrics utilized in ScoreForCore evaluation

4.3. Performance Evaluation

In this section we evaluate the performance of ScoreForCore. Firstly, we explain our performance metrics, secondly analyze attribute distribution in the network topology, then compare the results of ScoreForCore and PacketScore. At the end, technical and empirical storage analysis are provided.

4.3.1. Performance Metrics

In pattern recognition and information retrieval, true positive (TP), true negative (TN), false positive (FP) and false negative (FN) based binary classification metrics are instrumental to measure system performance. Since DDoS attack packet identification is also a binary classification problem, we utilize these metrics.

In our case, TP is the number of legal packets that are identified correctly by the system and reach the destination safely. TN is the number of attack packets dropped in the network and stonewalled to prevent reaching the destination. In that vein, FN is the number of legal packets that are falsely discarded whereas FP is the number of attack packets that are falsely forwarded to the destination. Following metrics calculated via these parameters are utilized in performance measurement:

- Precision (PN): What percentage of forwarded packets were legal?

$$PN = \frac{TP}{TP + FP} \quad (4.2)$$

- Recall (RL): What percentage of legal packets were forwarded to destination?

$$RL = \frac{TP}{TP + FN} \quad (4.3)$$

- True Negative Rate (TNR): What percentage of attack packets were dropped?

$$TNR = \frac{TN}{TN + FP} \quad (4.4)$$

- Negative Predictive Value (NPV): What percentage of dropped packets were attack packets?

$$NPV = \frac{TN}{TN + FN} \quad (4.5)$$

- Accuracy (ACC): What percentage of the decisions were correct?

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \quad (4.6)$$

- F-measure (FM): This metric combines precision and recall. In other words, it deals with the system's success regarding legal packets. It is the harmonic mean of precision and recall:

$$FM = \frac{2 \times PN \times RL}{PN + RL} \quad (4.7)$$

- F-measure Complement (FMC): As opposed to F-measure, this is dealing with system's success in attack packets. This metric combines true negative rate and negative predictive value. It is the harmonic mean of these metrics.

$$FMC = \frac{2 * TNR * NPV}{TNR + NPV} \quad (4.8)$$

- **Attack Prevention Efficiency (APE):** Attack packets are occupying network in vain. Especially in DDoS attacks, since number of packets are huge, they prevent the system to work efficiently. If they are stonewalled early in a network, efficiency of the network will increase. Thus, attack prevention efficiency measures how early network can get rid off attack packets. It is formally illustrated as in 4.9. In this equation, AP is the total number of attack packets, whereas dis_i shows the discard hop of attack packet i and p_i shows the path length of attack packet i .

$$APE = 1 - \frac{\sum_{i=1}^{AP} \frac{dis_i}{p_i}}{AP} \quad (4.9)$$

4.3.2. Attribute Distribution Analysis in Network Topology

Since our model suggests selection of the most appropriate attribute pair, it is important to be sure that most of the routers reach *SuspiciousPair* during an attack traffic. For this reason, before we start our simulations, we made some experiments in order to see how many of the routers reach desired pair in how many hops. These are illustrated in Table 4.4. In order to see all aspects, number of *OPNPs* for each router is also varied. If the number of nominal profiles increases for each router, the probability of reaching desired pair will also increase. This is strictly related to memory issue. Adding a pair nominal profile is affordable for a core router since we are not dealing with tuple nominal profiles. These issues are explained in details in Section 4.3.4.

In this table, S shows the number of routers that have *SuspiciousPair* in their *OwnPairList* whereas N stands for the number of routers that reach *SuspiciousPair* in their first neighbors. Similarly, N' and N'' is used for the number of routers that reach *SuspiciousPair* in their second and third hop neighbors. NF shows the number of routers that cannot find *SuspiciousPair* in its neighbors. In MEMX-HOPY experiments, each router has X number of nominal profiles and it is allowed to request *SuspiciousPair* from at most Y hop neighbors. These results suggest that for the topology in Figure 4.4, in MEM3-HOP3 experiment, all routers can reach *SuspiciousPair* since $NF = 0$. In our simulations, in order to be sure that, all routers are selecting

Table 4.4. Number of Routers reaching the *SuspiciousPair*

	S	N	N'	N''	NF
MEM1-HOP1	1	1	0	0	16
MEM1-HOP2	1	4	6	0	7
MEM1-HOP3	1	4	5	4	4
MEM2-HOP1	2	3	0	0	13
MEM2-HOP2	2	4	5	0	7
MEM2-HOP3	2	5	5	4	2
MEM3-HOP1	3	8	0	0	7
MEM3-HOP2	3	5	4	0	6
MEM3-HOP3	3	10	4	1	0

the most appropriate pair for their analysis, we let routers to request the desired pair profile from at most 3-hops neighbors and provide three *OwnPair* and *OPNP* tables for each router.

4.3.3. Results

In this section, we demonstrate our performance results according to the metrics explained above. In our simulations, PacketScore and ScoreForCore models are both run on all routers in the network. In the following tables, SFC stands for ScoreForCore model whereas PS stands for PackstScore model. Table 4.5 shows precision, recall and F-measure results for all attack types. Table 4.6 illustrates true negative rate, negative predictive value and F-measure complement results. Table 4.7 demonstrates accuracy and attack prevention efficiency values for PacketScore and ScoreForCore.

F-measure value shows the success of the system's behavior on legal packets. According to Table 4.5, for all attack types, ScoreForCore's F-measure result is much better than PacketScore's F-measure result. Success of the system's behavior on legal packets can be interpreted as follows: If there is an attack, PacketScore is mostly inclined to punish legal packets in addition to attack packets, whereas ScoreForCore lets legal packets to reach to destination in massive attack environment. If we analyze

Table 4.5. Precision, Recall and F-Measure Results of ScoreForCore

Attack Types	Model	PN	RL	FM
Generic Attack	PS: SFC:	0.59 0.72	0.35 0.94	0.44 0.81
TCP-SYN Flood Attack	PS: SFC:	0.73 1	0.49 0.99	0.59 0.99
SQL Slammer Worm Attack	PS: SFC:	0.98 1	0.86 1	0.92 1
DNS Amplification Attack	PS: SFC:	0.84 1	0.55 0.98	0.67 0.99
NTP Attack	PS: SFC:	0.99 1	0.82 0.99	0.90 0.99

Table 4.6. True Negative Rate , Negative Predictive Value and F-Measure

Complement Results of ScoreForCore				
Attack Types	Model	TNR	NPV	FMC
Generic Attack	PS: SFC:	0.84 0.76	0.67 0.95	0.75 0.85
TCP-SYN Flood Attack	PS: SFC:	0.89 1	0.73 0.99	0.80 0.99
SQL Slammer Worm Attack	PS: SFC:	0.98 1	0.92 1	0.95 1
DNS Amplification Attack	PS: SFC:	0.93 1	0.76 0.98	0.84 0.99
NTP Attack	PS: SFC:	0.99 1	0.89 0.99	0.94 0.99

values in Table 4.5, we can remark that SFC gives perfect results for the known attacks. Even for the unknown attack, named as Generic Attack, SFC overwhelmingly beats PS since SFC chooses the most appropriate attribute whereas PS uses random attributes for analysis.

Table 4.6 illustrates true negative rate, negative predictive value and F-measure complement results for all attack types. F-measure complement value shows the success of the system's behavior on attack packets. For all attack types, ScoreForCore's F-measure complement value is better than PacketScore's F-measure complement. For known attacks SFC gives perfect results whereas for the unknown attack it manages to give 85 % success on attack packets.

Table 4.7. Accuracy and Attack Prevention Efficiency Results of ScoreForCore

Attack Types	Model	ACC	APE
Generic Attack	PS:	0.65	0.61
	SFC:	0.83	0.65
TCP-SYN Flood Attack	PS:	0.73	0.84
	SFC:	0.99	0.97
SQL Slammer Worm Attack	PS:	0.93	0.78
	SFC:	1	0.97
DNS Amplification Attack	PS:	0.78	0.84
	SFC:	0.99	0.97
NTP Attack	PS:	0.92	0.84
	SFC:	0.99	0.97

In Table 4.7, accuracy and attack prevention efficiency results are demonstrated. Accuracy values show the system's accurate decisions on all packets. As it is obvious, for all attack types ScoreForCore outperforms PacketScore. Since the most appropriate attributes are considered in ScoreForCore, accurate decision number of the system increased. Accuracy is perfect for known attacks in SFC, whereas it performs 83% success for an unknown attack. In addition, attack prevention efficiency results for SFC is perfect for known attacks which means that almost all of the attack packets are stonewalled near the source of the attack.

4.3.4. Storage Analysis

Since several profiles are generated in ScoreForCore, it is important to make analysis of its storage requirement. Theoretical and empirical analysis are provided in the following subsections.

- (i) Theoretical: Storage requirement SR for a nominal profile can be formalized as follows: $SR = ES * NE$ in which ES is the size of an entry, NE is the number of entries. Also ES can be formalized as $ES = (NA + 1) \times 32$ bits. NA demonstrates the number of attributes in a profile. An entry in a single attribute profile consists of an attribute value and the number of packets that have these attributes. Thus for each entry 32-bit is used for the attribute value whereas the other 32-bit is used for the corresponding number of packets. Then, each entry in a single attribute profile needs 64 bits in other words $ES = 64$. Similarly, in pair

attribute profile each entry needs $ES = 96$ bits. In order to find a theoretical bound for SR , all packets are supposed to have different values for each packet, then the number of entries are equal to $NE = NPP^{NA}$ whereas NPP is the number of packets. Then, for a single attribute profile storage requirement is $SR = 64 * NPP$ bits whereas it is $SR = 96 * NPP^2$ bits for a pair attribute profile. If the number of packets is considered as 5000, a single attribute profile needs 40 KB, pair attribute profile needs 300 MB. These values are far more than real storage since the number of entries are regarded as maximal.

During the above calculations we did not take into account that the attribute can be an IP address and this can be an IPv6 address. In that case, a value would not take 32bits, instead it will be 128 bits. If an 32-bit is used for the corresponding number of packets, then $ES = 128 + 32 = 160$ bits for single profile and $ES = 128 + 32 + 32 = 192$ bits for a pair profile that have IP addresses for both attributes. Then $SR = 160 * NPP$ bits for a single profile where as it takes $SR = 192 * NPP^2$ bits for a pair profile. If the number of packets is considered as 5000, a single attribute profile needs 100 KB, pair attribute profile needs 600 MB.

Table 4.8. The Number of Different Values for Each Attribute

	SourceIP	DestPort	Protocol Type	TCP Flag	TTL	Packet Size
The Number of Different Values	324	80	7	8	59	67

- (ii) Empirical: In order to show more realistic values, empirical results are provided in this section. The real dataset that we have utilized in our simulations is analyzed in terms of number of different values for each attribute. These values are displayed in Table 4.8:

According to these values, the largest single attribute nominal profile needs $SR = 64 * 324$ which means approximately 2.5 KB whereas the largest pair attribute profile requires $SR = 96 * 324 * 80$ in other words 311 KBs.

If IPv6 is used for all IP packets, then for a single profile $324 * 160$ bits which is approximately 6.8 KBs is needed. Similarly, a pair profile needs $324 * 80 * (128 + 32 + 32)$ bits which is approximately 622 KB. These values are much less than the theoretical results and can be affordable easily by core routers. In our model SFC, routers need to reach the most appropriate attribute pair for score calculation. In order to increase the probability of access to *SuspiciousPair*, nominal profile tables for routers can be increased. Each table increase corresponds to 311 KB memory consumption. In our case, we have utilized three tables for each router.

4.4. Discussion

In order to provide collaboration, packet marking or communication techniques can be utilized. Initially, we focused on packet marking techniques and generated a cumulative scoring model. However, we noticed that for core routers the cost of marking packets is huge, since each packet requires checksum recalculations. In addition, cumulative scoring does not always give accurate results since irrelevant attributes are considered and they mislead the statistical analysis. Then, we decide to focus on finding the most appropriate attribute pair and utilize communication for providing collaboration.

In ScoreForCore, each router needs to access the most appropriate attribute pair. However, this condition depends on the number of neighbors for each router in the topology. In order to increase the probability of access to the desired pair, request hop and number of attribute pair for a router can be increased. Increasing request hop results in increase in communication overhead whereas increasing number of attribute pair for a router requires more storage space. Thus, ScoreForCore has memory and communication overhead issues that needs to be considered. As it is illustrated in Section 4.3.2, in order to be sure that most of the routers can access the most appropriate pairs this model should be deployed after some analysis in terms of memory and hop. As it is mentioned in previous section, since pair attributes are utilized instead of tuples, storage requirements is in KBs which can be afforded easily by core routers. Besides, it does not give much burden on communication since additional hop means an additional broadcast message and an answer, which is a piece of cake for a router.

4.5. Chapter Summary

In this work, we propose a proactive and cooperative filtering model against DDoS attacks. This type of filtering provides an ability of preventing the attack packets before they expand and gives more accurate decisions since they create filters together with more knowledge of the network. The proposed model ScoreForCore is a statistical based model that utilizes several attributes. It can protect against botnets since not only IP address but also several attributes are considered. It can also protect against unknown attacks since it makes statistical analysis and compares the current traffic with the nominal profile. The most distinctive property of this model is the selection of the most appropriate attributes for current attack traffic. This provides considerable improvement in accuracy. We evaluate ScoreForCore according to precision, recall, f-measure, f-measure complement, accuracy and attack prevention efficiency metrics. The results suggest that it gives perfect accuracy and attack prevention efficiency for several known attacks. Also, almost all known attack traffic is stonewalled near the source of the attack. Besides, it performs 80% accuracy for an unknown attack called as *generic attack*. Our results are good enough to encourage other researchers to work

on proactive and collaborative filtering mechanisms against DDoS attacks.

As a future work, we plan to make experiments in a real test-bed. This will make our results more realistic. In addition, secure communication protocols can be provided for collaboration of backbones. Besides, choosing the most appropriate attribute strategy can be applied for other statistical defense mechanisms. In addition, this collaborative model can be applied to emerging Software Defined Networking (SDN) technology. As it is stated in [89], SDN has a logical centralized controller that has network-wide knowledge of the system and can analyze the traffic patterns easily. Also, it can update forwarding rules dynamically. For these reasons, SDN brings new opportunities to defeat against DDoS attacks in cloud computing environment. Besides, according to [84, 90], a cooperative defense mechanism can be very effective against attacks.

5. DEFENSE MECHANISM IN SDN ENVIRONMENT: SDNScore

Software Defined Networking (SDN) is a recent emerging technology which defines a new design and management approach for networking [12]. The main property of this paradigm is the separation of control and data planes. In traditional networks, routers apply high level routing algorithms and decide where data packets should be forwarded. In SDN, decision and forwarding functionalities are separated. Decision process is provided by SDN controller whereas data forwarding is handled by switches. Since decision algorithms do not run on network devices, simpler network devices can be utilized rather than complicated routers [13]. Moreover, in traditional networks, each router has its own security, link failure and forwarding mechanisms. If any of these mechanisms needs to be updated, each network device should be handled individually. However, one can manage all these issues at a central point in SDN architecture.

Despite its advantages, SDN has several inherited challenges such as reliability, scalability, latency, and controller placement [81]. Security can be counted as one of the most vital problems [91]. In that regard, Denial of Service (DoS) attacks provide a favorable way for attackers to damage security of these systems. The main agenda of DoS attackers is to make network-resident services unavailable for legal users. An attacker generates enormous number of attack packets and makes the system busy such that it cannot serve the requests of legal users. If several machines participate in this attack, it is called Distributed Denial of Service (DDoS) attack. It can be created easily whereas it can be detected hardly since malicious packets show up as legal but with very large quantities.

SDN environment is favorable for DDoS attacks since it is inherently managed by the centralized controller [85]. When a packet comes to a switch from an unknown IP, it is forwarded to the controller. Then, the controller sends a flow rule to the switch for this IP. If attackers send a large number of packets from several IPs, each packet will

be forwarded to the controller. Then a huge number of attack packets will consume all available resources of the controller and make the system unavailable for legal users. Besides, the same attack can also cripple the system by exploiting the table capacity of switches. When a huge number of spoofed packets are received by the switch, its memory will be totally occupied. Similarly, the link between switch-controller can become unavailable because of the congestion by malicious traffic. All these issues pose SDN vulnerable for DDoS attacks and thus DDoS defense as a critical research topic for SDN.

A critical characteristic of SDN architecture that nondeliberately serves DDoS attacks is the limited passive capabilities of switches. Since they send all packets with unknown IP addresses to the controller, their medium becomes attractive for DDoS attacks. In addition, they do not have enough resources for very large volumes of traffic. In order to solve these problems, security-oriented intelligence can be integrated to switches. That will help to keep traffic in data plane as much as possible. Several works have suggested this approach such as [22,23,31–33]. In our SDNScore model, we also utilize switches with relevant processing and intelligence capabilities for security. We then rationale behind SDN which is simpler and lower-cost network equipment employed as the forwarding plane elements.

In the literature, there are some works proposed for DDoS defense in SDN. However, it is still an immature area since there is no dominant solution and all models have some drawbacks. In this work, we propose a packet-based statistical defense mechanism against DDoS attacks for SDN. We have been inspired by another packet based model, PacketScore [16], for traditional networks. Our proposal can cope with not only traditional attacks but also unknown new attacks. In order to make statistical analysis, some properties are utilized in packet scoring. The main difference of our proposal from PacketScore is the selection of appropriate attributes for each attack. This crucial property is provided by the support of the controller. Our results suggest that it is an effective method for DDoS defense in SDN.

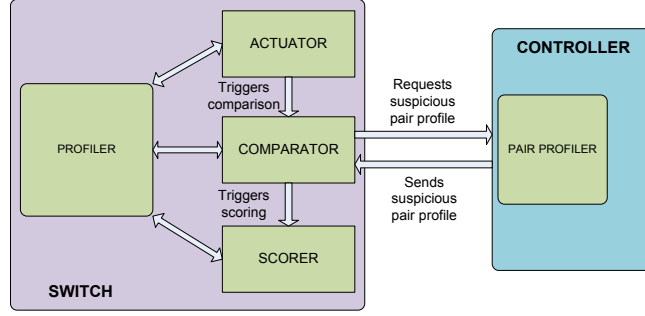


Figure 5.1. SDNScore architecture: switch and controller modules.

5.1. SDNScore Mechanism

SDNScore consists of four modules, *profiler*, *actuator*, *comparator* and *scorer*, to be loaded on the switches and another module, *PairProfiler*, for the controller. These components cooperate for DDoS detection and mitigation in SDN environment.

5.1.1. SDNScore Modules for Switches

Profiler generates nominal profiles in an attack-free period whereas *actuator* inspects traffic and starts packet-based inspection if an attack is detected. *Comparator* finds most appropriate pair for packet attributes and *Scorer* analyzes incoming packets and makes selective discarding according to that. These modules are illustrated in Figure 5.1 and explained in the following subsections.

- (i) **Profiler:** Each switch generates a nominal profile during an attack-free period. It counts the number of packets that have the same attribute value. Following properties are considered as attributes: *source IP*, *destination IP*, *source port*, *destination port*, *protocol type*, *packet size*, *TTL value* and *TCP flag*. During nominal profiling period, the profiler sends headers of all packets to the controller. Accordingly, the controller generates pair nominal profiles for each combination. At the end of this period, it has each pair nominal profile whereas the switches have single nominal profiles. Pair profiles are not stored in the switches because it would occupy a large amount of memory. A switch needs to store eight tables with above mentioned attributes whereas the controller needs to store $C(8, 2) = 28$

Table 5.1. Pair Nominal Profile Example

TTL Value	Destination Port Number	Number of Packets
48	25	125
48	53	175
48	80	19
50	80	42
...

tables. An example of nominal profile for a pair is presented in Table 6.1.

In addition to nominal profile generation, all profiling instructions that will be held in packet inspection period are also provided by the *profiler*. When the *actuator* detects a congestion, it also informs *profiler* and starts to generate current profiles. A *current profile* is generated during an attack period. Pair profiles are needed during current traffic analyses whereas single profiles are only used for choosing the most appropriate attribute selection. Pair profiles are utilized during all attack periods since they provide more detailed information about the traffic and enables more accurate decisions. The generated profiles are used by the other modules.

- (ii) *Actuator*: In normal conditions, flow-based network monitoring is provided in switches. *Actuator* inspects bandwidth usage to determine surges. When traffic exceeds a bandwidth threshold, system monitoring is switched to packet-based inspection, and comparator and profiler modules are activated. Packet based inspection provides generation of new tables as single current profiles in *profiler*. Since *actuator* monitors congestion consistently, when traffic returns to normal density and drops below the threshold, it informs other modules to switch to stand-by state. Then *Actuator* stops packet inspection and the system continues on flow-based monitoring.
- (iii) *Comparator*: After congestion is detected, *actuator* activates this module. Since current single profiles are generated by *profiler*, this module compares single nominal profiles with single current profiles. It determines the two specific attributes that have the most deviation from nominal profiles. These pairs are chosen as the most probable signs for ongoing attacks. Thus, it is called as *SuspiciousPair*.

Then *comparator* requests nominal profile of this pair from the controller. For instance, if it detects that protocol type and destination port have more deviation from nominal profile, the controller sends the nominal profile comprised of protocol type and destination port values and corresponding number of packets. Then it triggers *scorer* in order to start its work.

(iv) Scorer: This module is activated after all tables are generated. Scorer's three main responsibilities are as follows:

- Score calculation: Each packet's score is calculated considering *Suspicious Pair*'s corresponding value. If *SuspiciousPair* is determined as A and B , then packet p with the attributes $A = a_p$ and $B = b_p$ will have the score S_p as follows:

$$S_p = \frac{PNP_{(A=a_p, B=b_p)}/TPNP}{PCP_{(A=a_p, B=b_p, \dots)}/TPCP} \quad (5.1)$$

In (6.7), $PNP_{(A=a_p, B=b_p)}$ corresponds to the number of packets in nominal profile that have the property of a_p for attribute A and b_p for attribute B . Similarly, $PCP_{(A=a_p, B=b_p)}$ is the number of packets in the current profile that have the property of a_p for attribute A and b_p for attribute B . $TPCP$ and $TPNP$ are the total number of packets in the current profile and in the nominal profile, respectively.

- Threshold calculation: The score of a packet needs to be compared with a threshold Th . This threshold value is determined according to the cumulative distribution of scores by using load shedding algorithm [86]. It is shown symbolically as $CDF(Th) = \Phi$ whereas Φ is the ratio of traffic that should be dropped. The fraction of traffic permitted to pass is $1 - \Phi = \frac{\phi}{\psi}$ whereas ϕ acceptable traffic and ψ is the total current incoming traffic.
- Selective discarding: Each packet's score value is compared with the threshold. If it exceeds the threshold, this packet is supposed to be malicious and discarded. Otherwise, it is forwarded to the destination.

5.1.2. SDNScore Module for Controller: PairProfiler

The controller employs a module called *PairProfiler* in order to provide necessary capabilities for SDNScore. In normal conditions, while there is no congestion, switches send the packet flows that do not have an entry for these IP addresses to the controller. The controller decides how it should be forwarded by executing several routing algorithms. Then, it sends these rules to each switch and writes entries for unknown IPs. In our proposal, it has additional duties before and after the attack detection. *PairProfiler*'s primary work is to create pair nominal profiles. While single profiles are generated by the *profiler*, all the packets' headers are also sent to the controller. This can be a burden but it is needed only once in an attack-free period. Period is determined by the number of packets arrived. In PacketScore scheme [16], they made comparison in terms of time based vs. packet based period determination. Their results suggest that packet number based intervals are more suitable. Because time-based window may not allow creating meaningful profiles since it may not have as many as packets. Since our model utilized the similar scoring technique in [16], we choose packet based period determination. We also have utilized 5000 legal packets per period similar to [16]. A switch needs to send TCP and IP header of the each packet. If the controller and the switch communicates by using IPV6 on TCP, an empty packet will need 24 bytes for Ethernet header, 40 bytes for IPV6 header and 40 bytes for TCP header then it needs 104 bytes without the information. Since the switch also needs to send all TCP and IP header information of the incoming packets, we need to add this information length. If the incoming packet is an IPV6 and TCP packet then it will be $40\text{bytes} + 40\text{bytes} = 80\text{bytes}$. Then totally, the length of a packet that contains header information is 184 bytes. Since there are 5000 packets in a period, then it needs $184 * 5000 = 920KB$ as a communication overhead. Since it is needed in an attack free period it is a piece of cake.

As far as, eight attributes are utilized in profiling and each pair combination is generated as pair nominal profiles, $C(8, 2) = 28$ profiles are generated in the controller.

After a congestion is detected by a switch and the *comparator* determines the *SuspiciousPair*, it requests this pair nominal profile from the controller. Then, the controller responds this request with the expected information.

5.2. Simulations and Performance Evaluation

We simulated our model SDNScore and existing model An Entropy-Based Distributed DDoS Detection Mechanism in Software-Defined Networking (EBS) [33], in order to compare their performances. EBS model calculates entropy for destination IP and detects the DDoS attack on edge switches. Since this is a statistical model that runs on switches in SDN, we compared our proposal with this model. Network topology and dataset, attack types, performance metrics and simulation results are discussed in the following subsections.

5.2.1. Network Topology and Dataset

We simulated our SDNScore mechanism and EBS mechanism in C++. Test environment is a 3.3 GHz Intel Core i5 processor with 4 GB memory. Since packet based periods are utilized, 5000 packets are active in nominal profile generation whereas during the DDoS attacks, 10 times of nominal traffic is generated per period. Figure 6.2 depicts the topology used for experiments. There are two switches managed by the controller. One switch has five legal users (L1- L5) and five attacker users (A1- A5), whereas the other switch has the victim and two more hosts (H1,H2). These five attackers are assumed to legal users in the past. Then they are attacked by a malicious user and they generate a botnet to facilitate DDoS attacks. A real dataset from MAWI Working Group Traffic Archive [87] is utilized in traffic generation. It is a popular dataset used in literature [92–94]. MAWILab works on traffic measurement analysis in long-term on global Internet. It was started in 2002 and it is still collecting data from Internet. The part of data that we have used in our simulations was collected on Jan 12, 2014. This data is utilized to generate nominal profiles.

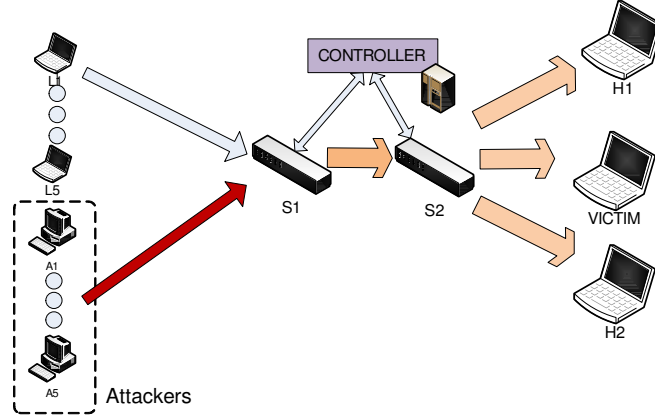


Figure 5.2. Network topology used in SDNScore experiments.

5.2.2. Attack Types

In this topology, attack and legitimate nodes are connected to a switch. Legitimate nodes generate legitimate traffic that have similar properties and similar amount of traffic to nominal profile packets. Nominal profile packets are created according to the dataset. Attack nodes generate both legitimate and attack traffic. In our simulation, we perform attacks by creating new packets that are similar to non-attack period's traffic. We simulate the following attacks:

- *TCP SYN Flood Attack*: The protocol type of an attack packet is TCP and TCP flag is set to SYN Flag. Other attributes are randomized.
- *SQL Slammer Worm Attack*: The protocol type of all attack packets is UDP and destination port is set to 1434. Also, packet size is between 371- 400 bytes. Other attributes are randomized.
- *DNS Amplification Attack*: The protocol type of attack packets is DNS and destination port is set to 53. Also, attack packet size is 60 bytes. Other attributes are randomized.
- *NTP Attack*: The protocol type of attack packets is NTP and destination port is set to 123. Also, attack packet size is 90 bytes. Other attributes are randomized.

In addition to these known attacks, other attack types are analyzed to compare the performance of our model and EBS for unknown attacks. These attacks are setup as follows:

- *Generic Attack*: In this attack, attacker generates attack packets with the attribute values that are selected randomly in their respective ranges.
- *Generic Attack with Determined Attributes*: Attacker chooses several attributes and gives most common values for them whereas he gives random values for other attributes. Choosing the most common values for determined attributes gives the attacker ability to make attack packets similar to legal packets. This approach makes the attacker more powerful and gives the ability to get through the filtering mechanism of the victim. If the attacker chooses two attributes it is named as *Generic Attack with Determined Two Attributes (GAD-2A)*. For instance, he chooses protocol type “TCP” and destination port “80”. Then, the attack packets with these attributes can be mixed up with the legal packets. Similarly, if the attacker chooses three and four attributes, it is denoted as *GAD-3A* and *GAD-4A*, respectively.

5.2.3. Performance Metrics

In pattern recognition and information retrieval, true positive (TP), true negative (TN), false positive (FP) and false negative (FN) based metrics for binary classification are instrumental to measure system performance. Since DDoS attack packet identification is also a binary classification problem, we utilize these metrics.

In our case, TP is the number of legal packets that are identified correctly by the system and reach the destination safely. TN is the number of attack packets dropped in the network and stonewalled to prevent reaching the destination. In that vein, FP is the number of legal packets that are falsely discarded whereas FN is the number of attack packets that are falsely forwarded to the destination. Following metrics calculated via these parameters are utilized in performance measurement:

- *Precision (PN)*: What percentage of the forwarded packets corresponds to legal packets?

$$PN = \frac{TP}{TP + FP} \quad (5.2)$$

- *Recall (RL)*: What percentage of legal packets were forwarded to destination?

$$RL = \frac{TP}{TP + FN} \quad (5.3)$$

- *Accuracy (ACC)*: What percentage of the decisions were correct?

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \quad (5.4)$$

- *F-measure (FM)*: This metric combines precision and recall. In other words, it deals with the system's success regarding legal packets. It is the harmonic mean of precision and recall:

$$FM = \frac{2 \times PN \times RL}{PN + RL} \quad (5.5)$$

5.2.4. Experimental Results

In this section, we demonstrate experimental results according to the metrics explained in Section 6.2.6. Table 5.2 shows precision, recall, accuracy and F-measure values. As far as EBS system determines the attacker according to the destination IP attribute, when it finds the victim it drops all packets to this victim. Then, it will drop all the attack packets including the legal ones. According to the results in Table 5.2, SDNScore outperforms EBS for most of the metrics. For each attack type, different attributes need to be considered. For TCP SYN Flood attack, protocol type and TCP flag are the arbiters, whereas protocol type, destination port and packet size are identifiers for DNS and NTP attacks. Protocol type and destination port are the identifier pair for SQL Slammer Worm attack. Since SDNScore has the same pair for

Table 5.2. Precision, Recall, Accuracy and F-Measure Results of SDNScore

Attack Type	Model	PN	RL	AY	FM
TCP-SYN Flood Attack	EBS: SDNScore:	0.57 0.98	1 1	0.75 0.99	0.73 0.99
SQL Slammer Worm Attack	EBS: SDNScore:	0.58 1	1 1	0.79 1	0.72 1
DNS Attack	EBS: SDNScore:	0.57 0.99	1 1	0.94 0.99	0.74 0.99
NTP Attack	EBS: SDNScore:	0.56 0.99	1 1	0.93 0.99	0.75 0.99

DNS and NTP attack, their results are same for these attacks.

Precision and recall metrics represent the system performance for legal traffic. Since F-Measure is the harmonic mean of them, it also deals with the success of the system on legal packets. Thus, SDNScore outperforms EBS as far as it does not care about the legal packets that goes to the victim. The results suggest that F-Measure values are about 70% for EBS whereas it is 99% for SDNScore. Accuracy metric considers not only legal but also attack packets. The results suggest that SDNScore's decisions are nearly perfect for the attacks whereas EBS cannot catch SDNScore.

In order to compare the performance of our model and EBS for unknown attacks, generic attack and generic attacks with deterministic attributes are performed. Precision, recall, F-Measure and accuracy results are shown in Table 5.3. In generic attack, attacker chooses random values for each attribute. SDNScore mechanism gives moderate results 84% accuracy as seen in Table 5.3. It gives 85% success on legal packets whereas it lets some attack packets pass. On the other hand, EBS gives 84% accuracy whereas it performs about 75% success on legal packets. In SDNScore, since all attributes are chosen randomly, some packets' corresponding values for *SuspiciousPair* attribute is exactly same as legal packets and scores of these packets do not exceed the threshold since there is not high amount of packets with these properties. Therefore, these packets can manage to get over SDNScore.

If an attacker becomes more intelligent and tries to generate attack packets more similar to the legal ones, he or she can choose two attributes and give most popular values for them. Other values are chosen randomly in that case. This corresponds to the GAD-2A case. For instance, it makes attack packets with protocol type = “TCP” and destination port = “80”. SDNScore results in perfect decision on attack packets (FN =0) whereas it drops some legal packets and thus TN rate increases. Accordingly, its precision value decreases considerably. In this case, SDNScore cannot cope with the legal packets that have the same value with the attack packets. The legal TCP packets that are sent to port 80 are given higher scores since there is high amount of packets with these properties. As their score exceeds the threshold, they are marked as attack packets.

If the attacker increases the determined number of attributes to three (GAD-3A) and gives most popular values for these attributes, he tries to make his packets more similar to legal ones. For instance, it generates attack packets with protocol type = “TCP”, destination port = “80” and packet size = “60”. In this case, SDNScore gives perfect results for attack packets whereas it increases the accurate decision performance for legal packets. As the determined number of attributes increases, the number of legal packets who have the determined properties decreases. Thus, the number of legal packets who give higher scores than threshold decreases, accordingly F-measure value increases.

If the number of determined attributes is increased to four, SDNScore gives exactly the same result with three attributes. This is because the number of legal packets who have the same values with attack packets for four attributes is almost equal to the number of legal packets who have the same values with attack packets for three attributes. For instance the number of legal packets with protocol type = “TCP”, destination port = “80”, packet size = “60” and TCP flag = “SYN” are the same as the number of legal packets with protocol type = ”TCP”, destination port = “80” and packet size = “60”. That situation corresponds to the case of saturation where information gain does not contribute to the performance gain anymore.

Table 5.3. Precision, Recall, Accuracy and F-Measure Results of SDNScore for

Unknown Attacks					
Attack Type	Model	PN	RL	AY	FM
Generic Attack	EBS: SDNScore:	0.57 0.99	1 0.75	0.75 0.84	0.73 0.85
GAD-2A	EBS: SDNScore:	0.57 0.55	1 1	0.74 0.80	0.73 0.71
GAD-3A	EBS: SDNScore:	0.57 0.81	1 1	0.75 0.98	0.73 0.90
GAD-4A	EBS: SDNScore:	0.57 0.81	1 1	0.75 0.98	0.73 0.90

All these results suggest that SDNScore performs elegantly for unknown attacks. Even in the conditions that the attacker generates packets that are very similar to legal packets, our mechanism gives favorable results and increases the accuracy whereas EBS results does not change and fall behind SDNScore.

5.3. Chapter Summary

In this work, a statistical defense mechanism is proposed against DDoS attacks in SDN environment. Some intelligence is deployed on SDN switches to enable this mechanism. SDNScore suggests to add four modules, namely Profiler, Actuator, Comparator and Scorer, to switches and one module named as PairProfiler to the controller. It is a defense mechanism including detection and mitigation against DDoS attacks in SDN environment. It is effective against unknown attacks since it utilizes statistical analysis and makes comparison with nominal traffic. It determines most appropriate attributes for current traffic and provides considerable improvement in accuracy. We compare SDNScore with an existing entropy based model (EBS) [33] according to precision, recall, f-measure and accuracy metrics. The results suggest that SDNScore outperforms EBS in all types of attacks. For known attacks, SDNScore gives perfect results. Besides, it performs 84% accuracy for an unknown attack named as *generic attack*. If the attacker generates more similar attack packets and tries to get over the defense mechanism, SDNScore behaves elegantly and gives nearly perfect results as high as 98% accuracy rate.

As future work, we plan to perform experiments in a real setup. In addition, the controller's role can be enhanced for prevention. It can coordinate several switches for preventing future attacks. Besides, it can combine statistics from several switches to provide more accurate scoring.



6. JOINT ENTROPY BASED SCORING MECHANISM AGAINST DDOS ATTACKS IN SDN ENVIRONMENT:JESS

Security can be counted as one of the most vital problems [91] in SDN environment. In that regard, Denial of Service (DoS) attacks provide a favorable way for attackers to damage security of these systems. SDN environment is favorable for DDoS attacks since it is inherently managed by the centralized controller. When a packet comes from an unknown IP to a switch, it is forwarded to the controller. Then, the controller sends a flow rule to the switch for this IP. If the attackers send a large number of packets from several IPs, each packet will be forwarded to the controller. Then a huge number of attack packets will consume all available resources of the controller and make the system unavailable for legal users. Besides, the same attack can also cripple the system by exploiting the table capacity of switches. When a huge number of spoofed packets are received by the switch, its memory will be totally occupied. Similarly, the link between switch-controller can become unavailable because of the congestion by malicious traffic. All these issues pose SDN vulnerable for DDoS attacks and thus DDoS defense as a critical research topic for SDN.

Several works are proposed in the literature for DDoS Defense in SDN. Our previous work, SDNScore is a packet-based statistical model that needs capable switches. In this mechanism, each packet is analyzed according to its attribute values and then scores are calculated according to them similar to the work in PacketScore model [16]. In SDNScore, switches are not simply data forwarders. Instead, they can collect statistics and decide if DDoS attack is in action. Then they coordinate with the controller and decide on attack packets in cooperation. This model suggests to add four modules, namely profiler, actuator, comparator and scorer, to the switches and one module named as pairprofiler to the controller. Facilitating switches with some minor intelligence features does not compromise the main paradigm of SDN. However, the system designer should be cautious for this tradeoff of SDN paradigm vs. “capable” switches.

In order to get rid off this trade-off problem, we proposed a Joint-Entropy based defense mechanism that carries all burden to the controller and does not need capable switches.

Entropy shows the randomness in a data set. Since similar type of packets are sent and occupied most of the traffic during a DDoS attack, randomness decreases. For this reason, entropy calculation is a good tool for DDoS detection. Since it has statistical calculation, it does not give huge burden on performance unlike machine learning algorithms. There are several works that utilized entropy in traditional networks such as [95–97], however there are a few works in SDN. One of them is [32] that proposes an early detection of DDoS Attacks against SDN Controllers. It runs on the controller. In this mechanism, entropy is calculated for destination IP address. If entropy decreases under a threshold, DDoS is detected. It enables to determine the victim, but it is not possible to dissociate the legal packets from the attack ones. Another similar model is [33] that proposes an entropy-based lightweight DDoS flooding attack detection model running in the OF edge switch. This achieves a distributed anomaly detection in SDN and reduces the flow collection overload on the controller. However, there is a trade-off of SDN paradigm vs. “capable” switches in this model.

Another work that utilizes entropy calculation is [98], that they proposes a detection and mitigation method. They reduced data gathering with sampling and utilized entropy for anomaly detection. Then provide anomaly mitigation using OF. Their model runs on the controller. Their mitigation strategy is to cut-off all the flows to the host under attack. However, this does not enable to protect the legal packets.

In our model called Joint Entropy based Scoring on SDN (JESS), we utilized joint-entropy for both detection and mitigation. We did not only focused on destination IP entropy but also all combinations of IP and TCP layer attributes. Several hosts under a switch can be the victims at the same time, thus destination IP address does not always give correct results. Thus, there is a need for analysis of different attributes in differentiating unknown attacks. For each type of attack, different attribute needs to be considered. For instance in TCP SYN Flood attack, protocol type and TCP flag

are the arbiters, whereas protocol type, destination port and packet size are identifiers for SQL Slammer Worm, DNS and NTP attacks. In order to be ready for an unknown attack, the model should not stick to an attribute. Appropriate attribute pair should be selected for the current attack. For this reason, in our model joint-entropy is utilized during the appropriate pair selection.

6.1. JESS Mechanism

Joint Entropy based Scoring on SDN mechanism has three main phases: nominal stage, infant stage and lifelong stage. JESS mechanism creates nominal information in a non-attack period named as nominal stage. When the bandwidth exceeds the threshold, second phase called infant stage is come into play. In this phase, DDoS attack is detected and some parameters are determined. Then the last phase named as lifelong stage becomes active which defends the system during the attack. All these stages are shown in Figure 6.1. The details are explained in the following subsections.

6.1.1. Nominal Stage

This stage consists of the preparation steps before the attack. JESS generates the necessary data that will be used in an attack period. Nominal information is gathered in an attack free period in order to be compared in the attack period to detect the anomalies. The pair nominal profiles and joint entropies for each pair are the needed information. Following subsections explain how these information are obtained.

- (i) Nominal Pair Profile Generation: Each switch generates nominal profiles for each pair attribute during an attack-free period. It counts the number of packets that have the same attribute value. Following properties are considered as attributes: *source IP*, *destination IP*, *source port*, *destination port*, *protocol type*, *packet size*, *TTL value* and *TCP flag*. During this profiling period, the switch sends headers of all the packets to the controller. Accordingly, the controller generates pair

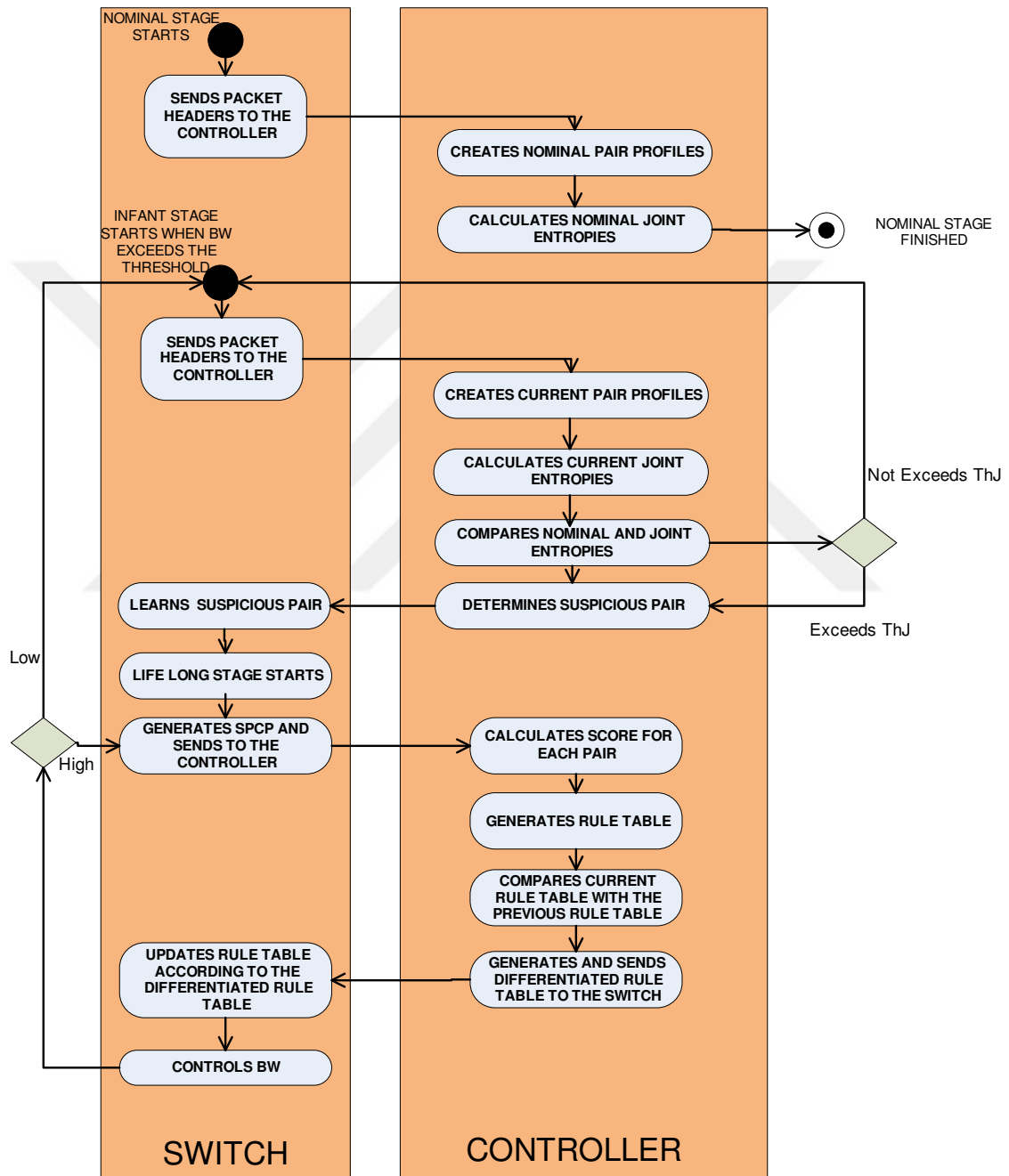


Figure 6.1. Protocol of JESS Mechanism

Table 6.1. An Example of a Pair Nominal Profile for TTL Value & Destination Port

Attributes		
TTL Value	Destination Port Number	Number of Packets
48	25	125
48	53	175
48	80	19
50	80	42
...

nominal profiles for each combination. An example of a nominal profile denoted as NP for a pair is presented in Table 6.1.

- (ii) Joint Entropy Calculation for Nominal Profiles: JESS utilizes joint-entropy in DDoS detection and appropriate pair attribute selection. Since entropy shows randomness in a data set, it is a good sign for DDoS detection. As far as similar packets are sent in DDoS, entropy will decrease and DDoS can be detected. Joint-entropy shows the amount of randomness we get when we observe more than one attributes at the same time. In JESS, joint-entropy $JENP$ is calculated for each nominal pair profile and it is utilized in detection first. When the joint entropies of the current profiles are compared with the joint entropies of the nominal profiles, if the difference of any of the pair exceeds a threshold, DDoS will be detected. Then, the most differentiated pair will be utilized as the appropriate pair for the current attack.

A nominal profile for a pair attribute such as A and B is denoted as $NP_{A,B}$. Joint entropy of this nominal profile is calculated as follows:

$$JENP_{A,B} = - \sum_a \sum_b PN(A = a, B = b) \log[PN(A = a, B = b)] \quad (6.1)$$

$PN(A = a, B = b)$ shows the probability of a packet to have the property of $A = a$ and $B = b$. This can be calculated as follows:

$$PN(A = a, B = b) = PNP_{(A=a, B=b)} / TPNP \quad (6.2)$$

In this equation $PNP_{(A=a,B=b)}$ is the number of packets in a nominal profile that have the property of a for attribute A and b for attribute B . Also $TPNP$ corresponds to the total number of packets in the nominal profile $NP_{A,B}$.

At the end of this stage, nominal profiles and joint-entropies are ready to use in case of an attack.

6.1.2. Infant Stage

When the bandwidth exceeds its nominal value, infant stage is activated. This stage detects the attack and determines the most appropriate pair for profiling in lifelong periods of the attack. Following steps are provided in this stage.

- (i) Current Pair Profile Generation: Each switch generates current profiles for each pair attribute during the attack period. It counts the number of packets that have the same attribute value. During this profiling period, the switch sends the headers of all the packets to the controller. Accordingly, the controller generates pair nominal profiles for each combination.
- (ii) Joint Entropy Calculation for Current Profiles: Joint entropy $JPCP$ is calculated for each pair in order to provide the comparison with $JENP$. Similar to the $JENP$, $JECP$ is calculated as follows:

$$JECP_{A,B} = - \sum_a \sum_b PC(A = a, B = b) \log[PC(A = a, B = b)] \quad (6.3)$$

where $PC(A = a, B = b)$ is determined as follows:

$$PC(A = a, B = b) = PCP_{(A=a,B=b)} / TPCP \quad (6.4)$$

In this equation, $PCP_{(A=a,B=b)}$ is the number of packets in a current profile that have the property of a for attribute A and b for attribute B . Also $TPCP$ corresponds to the total number of packets in the current profile $CP_{A,B}$. In order

to have the entropy value in between $[0, 1]$, we normalized its value as follows:

$$NorJENP_{A,B} = JENP_{A,B} / \log[TPNP] \quad (6.5)$$

In this thesis, all entropy values are normalized, so when we used joint entropy JE , it means that it is normalized, in other words we did not need to add Nor for indicating normalization.

- (iii) Comparison: Whenever the joint entropies of current profiles are ready, they are compared and $DJ_{A,B}$ for each pair is calculated. If any of the difference exceeds a threshold ThJ , then DDoS is detected.

$$DJ_{A,B} = JEC_{A,B} - JENP_{A,B} \quad (6.6)$$

After the DDoS detection, the maximum difference

$$DJ_{MAX} = MAX\{DJ_{A,B}, DJ_{A,C}, DJ_{A,D}, \dots, DJ_{B,C}, DJ_{B,D}, \dots, DJ_{C,D}, \dots, DJ_{Y,Z}\}$$

is determined and the pair who have the maximum difference is detected as the *SuspiciousPair* who are the most appropriate pair for mitigation of the current attack traffic. Controller sends a message to the switch to inform it about the suspicious pair *SuspiciousPair*.

6.1.3. Lifelong Stage

After the *SuspiciousPair* is determined by the controller and the switch is informed about it, infant stage is completed and the lifelong stage is initialized. Attack traffic is mitigated by dropping the attack packets whereas legal packets are protected. This stage is active during the attack and it continues to mitigate until bandwidth decreases to its nominal values. Following steps are realised in this stage.

- (i) SuspiciousPair Profile Generation: Header's of all packets are sent from switch to the controller in previous two stages. This takes high amount of communication burden on a switch. This burden is decreased in this phase. Since the switch

knows the *SuspiciousPair*, it starts to generate current profile *SPCP* for the *SuspiciousPair*. It does not send all packets' headers, instead switch sends the current profile *SPCP* to the controller at the end of each period.

Period is determined by the number of packets arrived. In PacketScore scheme [16], they made comparison in terms of time based vs. packet based period determination. Their results suggest that packet number based intervals are more suitable. Because time-based window may not allow creating meaningful profiles since it may not have as many as packets. Since our model utilized the similar scoring technique in [16], we choose packet based period determination.

- (ii) Score Calculation: After the collaboration period, the controller have the nominal profile *SPNP* and current profile *SPCP* of the *SuspiciousPair*. Each entry of *CPNP* is taken and its score S is calculated by considering its corresponding value in *SPNP*. If *SuspiciousPair* is determined as A and B , then the entry with attributes $A = a$ and $B = b$ will have the score S as follows:

$$S = \frac{PCP_{(A=a,B=b)}/TPCP}{PNP_{(A=a,B=b,...)}/TPNP} \quad (6.7)$$

In (6.7), $PCP_{(A=a,B=b)}$ corresponds to the number of packets in current profile that have the property of a for attribute A and b for attribute B . $TPCP$ is the total number of packets in current profile. Similarly, $PNP_{(A=a,B=b)}$ is the number of packets in the nominal profile that have the property of a for attribute A and b for attribute B . $TPNP$ is the total number of packets in nominal profile. All the scores are stored in a score table ST .

- (iii) Threshold Determination: The score of a packet needs to be compared with a threshold Th . All scores are stored in a score table ST and the current threshold value Th_C is determined according to the cumulative distribution of scores by using load shedding algorithm [86]. It is shown as symbolically $CDF(Th) = \Phi$ whereas Φ is the ratio of traffic that should be dropped. The fraction of traffic permitted to pass is $1 - \Phi = \frac{\phi}{\psi}$ whereas ϕ acceptable traffic and ψ is the total current incoming traffic. As far as it will be a good property to decide on the

Table 6.2. An Example of a Rule Table (RT_C) for the *SuspiciousPair* TTL Value & Destination Port Attributes

TTL Value	Destination Port Number	Number of Packets
48	25	PASS
48	53	PASS
48	80	DROP
50	80	PASS
...

threshold Th with not only the current but also the previous period's Th_P , Th is calculated as follows:

$$Th = (Th_C + Th_P)/2 \quad (6.8)$$

- (iv) Rule Generation: After the controller calculates the scores and the threshold, the system is ready to generate the rules. Since the score of each entry in $SPCP$ is calculated, decision for DROP or PASS can be provided by comparison with the threshold. If the score value exceeds the threshold, it is supposed to be malicious and discarded. Otherwise, it is forwarded to the destination. A current rule table RT_C is generated for each entry in $SPCP$. An example of a rule table is illustrated in Table 6.2
- (v) Differentiated Rule Determination: Differentiated rules are determined by comparing the current rule table RT_C and the previous rule table RT_P . Same rules that have the same decisions will not sent to the switch in vain. For instance, RT_P in Table 6.3 is compared with RT_C in Table 6.2 and only the last two rules of RT_C are determined as differentiated rules $DifRT$, since they are different from the previous ones. Then, $DifRT$ is sent to the controller.

Table 6.3. RT_P for the *SuspiciousPair* TTL Value & Destination Port Attributes

TTL Value	Destination Port Number	Number of Packets
48	25	PASS
48	53	PASS
48	80	PASS
50	80	DROP
...

To sum up, all these operations are demonstrated in Figure 6.1. In this figure, there are three main stages:

- In an attack free period called Nominal Stage, a baseline information is generated by creating nominal pair profiles for each attribute pair. During this period, switch sends headers of all packets to the controller. At the end of the period, when the nominal profiles are ready, their joint entropies of each pair are calculated.
- After a congestion is detected, infant stage is started. The switch starts to send the headers of packets to the controller. Controller generates the current pair profiles for each attribute pair and calculates the joint entropies. If any of the difference exceeds Th_J , DDoS attack is detected. Then, the pair who has the maximum difference is determined as the suspicious pair *SuspiciousPair*. This information is sent to the switch.
- As far as the *SuspiciousPair* is determined, Lifelong stage is started. This stage continues until the congestion is decreased. During a period in this stage, switch creates current profile *SPCP* of *SuspiciousPair* and at the end of each period it sends *SPNP* to the controller. Controller calculates score for each entry in the profile and generates a score table. Then, a threshold Th_C is determined by load-shedding algorithm. Then Th is calculated by averaging Th_C and previous period's threshold Th_P . After that, a current rule table RT_C is generated by comparing the scores of each pair with Th . If the score is under the threshold, it is forwarded otherwise it is discarded. Then, the current rule table RT_C is compared with the previous rule table RT_P . Only the differentiated rules *DifRT* are sent to the switch since they have changed after the previous period. Then

the switch updates its rule table according to *DifRT*.

6.2. Simulations and Performance Evaluation

We perform experiments via simulations for performance evaluation of JESS. Dataset, simulation environment, network topology, attack types, performance metrics and simulation results are discussed in the following subsections.

6.2.1. Dataset

In our work, we need a real data of Internet traffic. We use real dataset from MAWI Working Group Traffic Archive [87]. MAWILab works on traffic measurement analysis in long-term on global Internet. It was started in 2002 and it is still collecting data from Internet. The part of the data that we have used in our simulations are collected on Jan 12, 2014. This data is utilized to generate nominal profile.

6.2.2. Simulation Environment

In order to evaluate our work, we utilized Mininet [99] as a network simulator. It enables to create a realistic network topology and it is convenient for SDN environment. It is compatible with OpenFlow and a controller Ryu [100]. The test environment is Ubuntu 14.04.

6.2.3. Network Generation

Figure 6.2 depicts the topology used for experiments. There is a switch managed by the controller. The switch has three hosts. One of the hosts is attacked by a malicious user and it starts to generate DDoS attacks to the other host named as Victim under the same switch. There is also one more host that is neither a victim nor an attacker who generates legal communication.

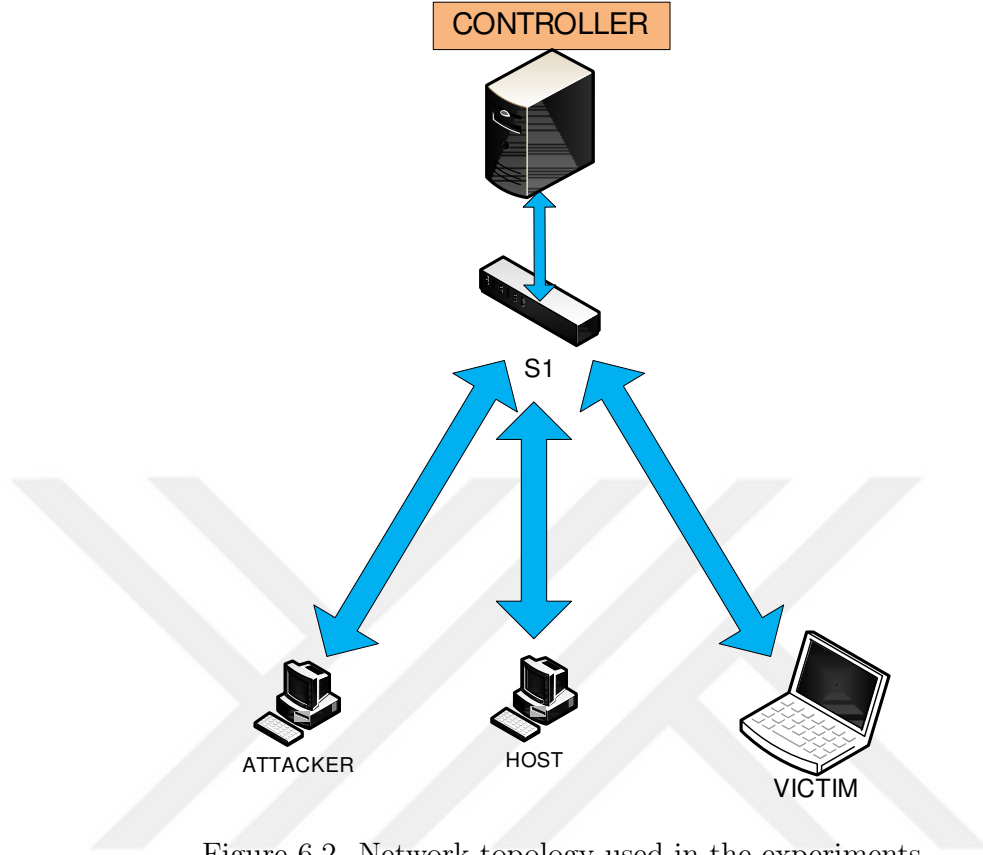


Figure 6.2. Network topology used in the experiments.

6.2.4. Packet Analysis in Mininet

In order to reach the attribute values for each packet we need to know the corresponding values in each header. Source IP and destination IP addresses, TTL value, packet size and protocol type information is obtained from IP header. Destination port, source port and TCP flag attributes are obtained from the TCP header. The corresponding names in headers for the attributes are provided in the following Table 6.4.

6.2.5. Attack Types

In this topology, attacker and legitimate nodes are connected to a switch. Legitimate nodes generate legitimate traffic that have similar properties and similar amount of traffic to the nominal profile packets. Nominal profile packets are created according to the dataset. Attacker generate both legitimate and attack traffic. In our simulation,

Table 6.4. Attributes in Headers

Header	Attribute	Corresponding Name in header
IPv6	packet size	payload_length
IPv6	TTL	hop_limit
IPv6	protocol type	next_header
IPv6	source IP	src
IPv6	destination IP	dst
IPv4	packet size	total_length
IPv4	TTL	ttl
IPv4	protocol type	proto
IPv4	source IP	src
IPv4	destination IP	dst
TCP	source port	src_port
TCP	destination port	dst_port
TCP	TCP flag	bits

we perform attacks by creating the new packets that are similar to the nominal traffic. We simulate the following attacks:

- *TCP SYN Flood Attack*: The protocol type of an attack packet is TCP and TCP flag is set to SYN Flag. Other attributes are randomized.
- *DNS Amplification Attack*: The protocol type of attack packets is DNS and destination port is set to 53. Also, attack packet size is 60 bytes. Other attributes are randomized.
- *NTP Attack*: The protocol type of attack packets is NTP and destination port is set to 123. Also, attack packet size is 90 bytes. Other attributes are randomized.
- *Generic Attack*: All attributes are selected randomly in their ranges. This can be stated as an unknown type of attack.
- *Mixed Attack*: TCP SYN Flood Attack, SQL Slammer Worm Attack, DNS Amplification Attack and NTP Attack are applied at the same time to a victim.

6.2.6. Performance Metrics

In pattern recognition and information retrieval, true positive (TP), true negative (TN), false positive (FP) and false negative (FN) based metrics for binary classification are instrumental to measure system performance. Since DDoS attack packet identification is also a binary classification problem, we utilize these metrics.

In our case, TP is the number of attack packets dropped in the network and stonewalled to prevent reaching the destination whereas TN is the number of legal packets that are identified correctly by the system and reach the destination safely. In that vein, FP is the number of legal packets that are falsely discarded whereas FN is the the number of attack packets that are falsely forwarded to the destination. Following metrics calculated via these parameters are utilized in performance measurement:

$$FPR = FP / (FP + TN) \quad (6.9)$$

False Positive Rate metric FPR deals with the success of the system on legal packets and tries to calculate what percentage of the legal packets are dropped.

$$ACC = (TP + TN) / (TP + TN + FP + FN) \quad (6.10)$$

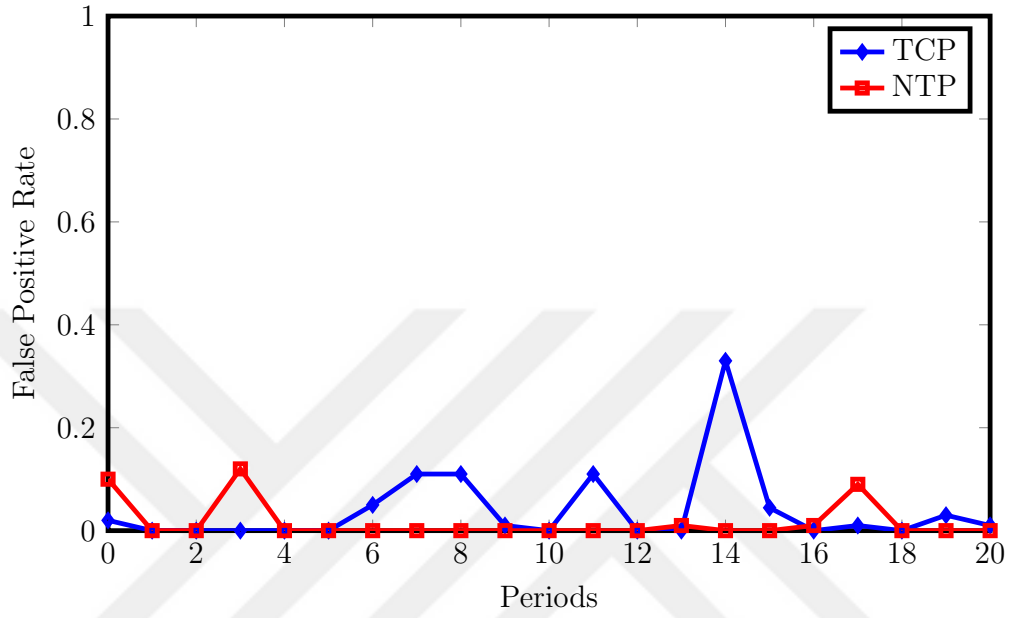
Accuracy metric ACC deals with the success of the system and tries to calculate what percentage of the decisions are correct.

6.2.7. Experimental Results

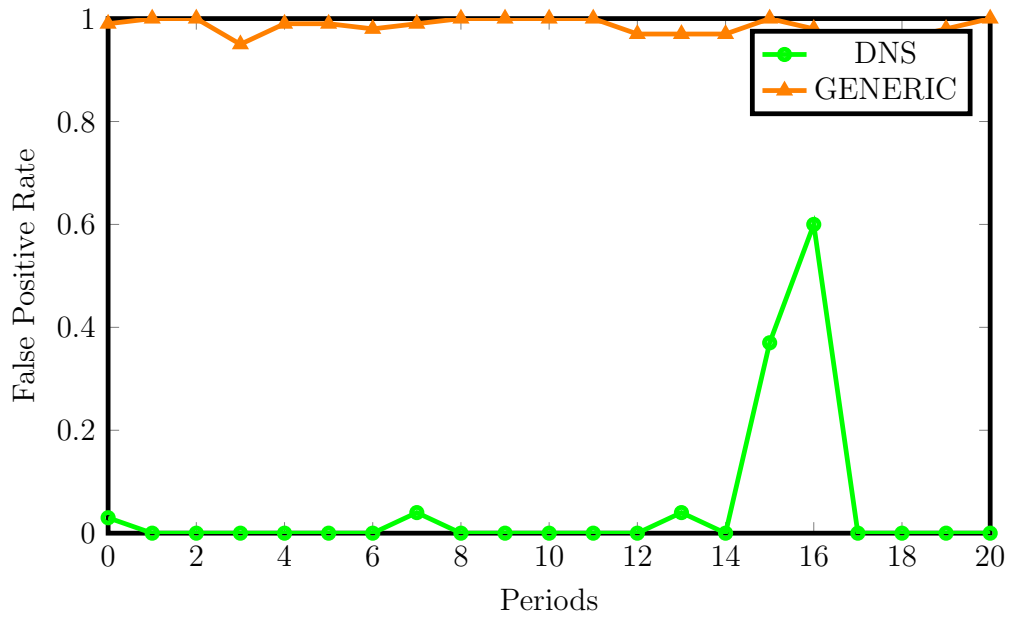
- (i) During Different DDoS Attack Types: In this section, we demonstrate our performance results according to the metrics explained above. In order to see the performance of the system on legal packets FPR results are demonstrated in Figure 6.3. The lower values for FPR is better since it shows the ratio of the legal packets that are discarded erroneously. In Figure 6.3(a), JESS's performance is measured against TCP SYN Flood and NTP attacks during 20 periods. Similarly,

in Figure 6.3(b) JESS's performance is measured against DNS Amplification and Generic attacks. The results does not include the warm-up periods which consists of nominal and infant stages. During these stages switches does not have the updated rules yet, so it cannot start to filter according to JESS's results. Our mechanism starts to filter in life-long stage. Our results suggest that while there is TCP SYN Flood attack, nearly all packets can reach to the victim safely. There are a few small rises but they falls immediately which shows that legal packets can be identified successfully by JESS. In only one period in TCP, it rises to %40, the reason is that there are more packets who have the same property with the attack packets. Then, the legal ones are also dropped. Similarly during NTP and DNS attacks, in most of the periods all legal packets can reach to the victim safely. During DNS attack, in one place, FPR increase rapidly, when we analyze the legal packets of this period we see that there are more packets who have the same properties with the attack packets. For instance, JESS selects packet Size and destination port pair as *SuspiciousPair*. DNS attack packets have the properties as $destination_{port} = 53$ and $packetSize = 60$. Then all the packets with this properties will be dropped. If there are more legal packets with these properties in this period then, FPR increases. In order to decrease this probability sliding window periods are utilized. It is explained in the following subsection. The most distinctive result is obtained during Generic Attack. This is a difficult type for JESS, since it needs at least two fixed properties of an attack in order to determine *SuspiciousPair*. However in generic attack, all attributes are randomized and thus, JESS's FPR rates are high during this attack.

Figure 6.4 shows the success of the system on both legal and attack packets. As it can be interpreted that accuracy of JESS is perfect during TCP SYN Flood, NTP attack and DNS Amplification attacks. JESS gives about 80% accuracy during Generic attack. This shows that JESS performs 80% success even on unknown attacks. Its performance on legal packets is not so great but it can differentiate attack packets easily and can drop it.



(a) During TCP SYN Flood and NTP attacks



(b) During DNS Amplification and Generic attacks

Figure 6.3. FPR performance of JESS during DDoS attacks

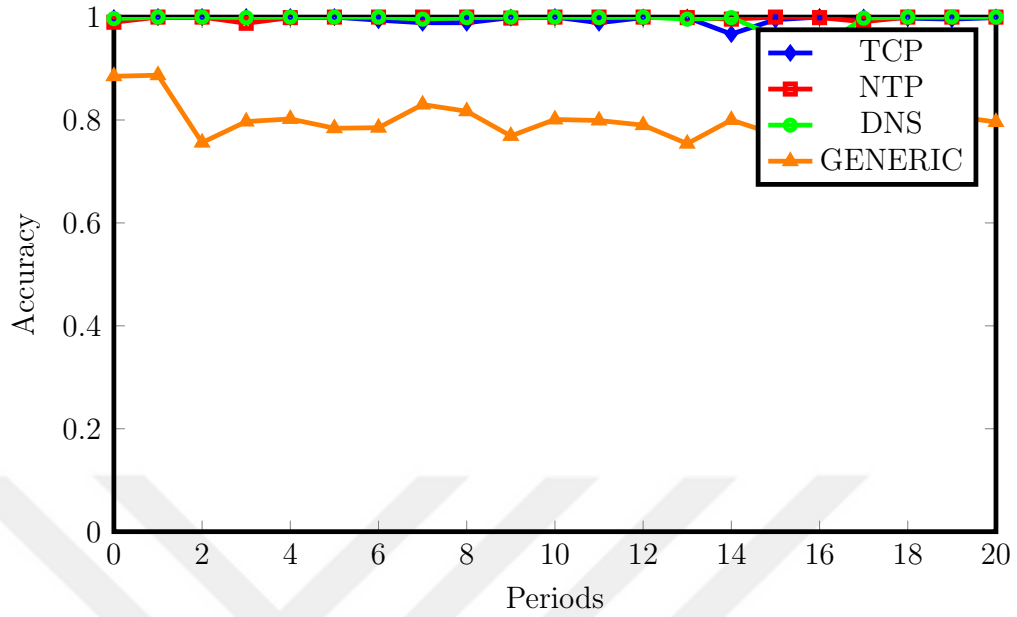


Figure 6.4. Accuracy performance of JESS during TCP SYN Flood, NTP attack, DNS Amplification and Generic attacks

Another attack type called Mixed attack is performed on different topology to show it more clearly. Network topology in Figure 6.5 is used for mixed attack experiment. All the attacker hosts apply different attacks on the victim. TCP SYN Flood, DNS Amplification and NTP attack packets are send to the victim simultaneously. Also the legal host tries to communicate with the victim. This is one of the most difficult type of attacks. The results are illustrated in Figure 6.6. During 14 periods FPR is under 10% . This means that more than 90% of legal packets can manage to reach the victim. There are several peaks but it is an expected result since the packet types are differentiated. In addition, it is a promising result since the system recovers itself after at most three periods. In addition average ACC is 67% which is a decent performance against such a difficult type of attack.

- (ii) With Sliding Window Periods: In JESS, packet based discrete periods are utilized since time-based window may not allow creating meaningful profiles as reason of it may not have as many as packets. The results in Figure 6.3(a) and 6.3(b), there are some peaks that are more than % 20. Especially for TCP attack, it may result in problems since the legal users cannot start a communication. In

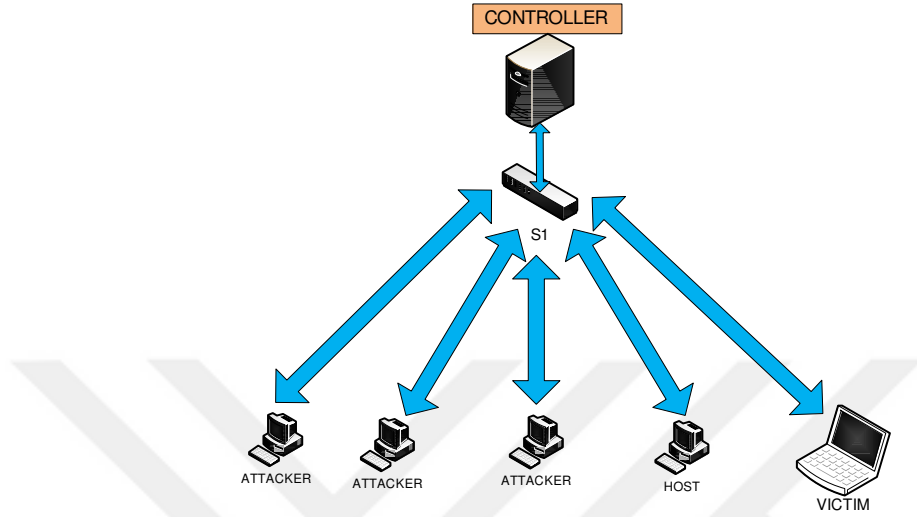


Figure 6.5. Network topology used in the mixed attack experiments of JESS

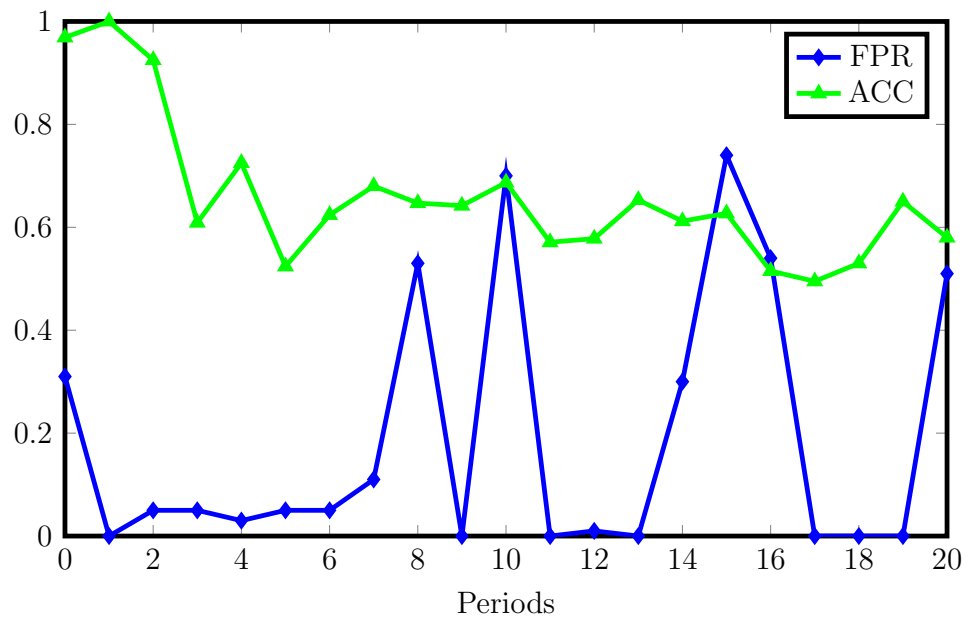


Figure 6.6. Accuracy and FPR performance of JESS during Mixed Attack

addition, as far as there is a slow start mechanism in TCP, when the legal user's machine cannot get an acknowledgement message, the system pulls the brake at initial stage and will not manage to start connection. In order to get rid off this problem, we decided to use sliding window periods. This will decrease the peaks, since they consider not only the current but also the previous period's packets. However, in this case, JESS needs to store the headers of the packets. In discrete periods, we did not need to store them. Because the properties are processed to the nominal profile *SPNP* of the switch and it sends this profile to the controller at the end of each period. But this time in sliding window periods, we need to use some packets that came in the previous period, and we could not update the profile since we do not know which packets properties belong to which entry. For this reason, we need to store the headers of each packet until they are not used anymore. At the end of each period, *SPNP* is created by considering W number of headers of packets where W is the size of the sliding window. Since in our simulations, each period has 1000 packets, our window size is $W = 1000$. If the incoming packet is an IPV6 and TCP packet then it will have $40bytes + 40bytes = 80bytes$ header size. Then the needed extra storage for a switch is $1000 * 80 = 80KBs$. It is affordable for current switches. In addition, sliding number *sld* should be determined, to express how much of the new packets are involved in the new period. In our simulations $sld = 500$. The results of FPR with sliding windows during different attacks are illustrated in Figure 6.7. The peaks of TCP and DNS is about 0.1 mostly, whereas the peaks are also below 0.15 for NTP. These results suggest that JESS gives elegant results with sliding windows with the expense of 80KBs more in the switches.

- (iii) During Different Attack Intensities: Performance results of JESS is analyzed against different attack intensities. Figure 6.8(a) shows the FPR of JESS under different rates of DDoS traffic. The results shows that it gives about 10% FPR during 10 times of nominal traffic. However, it gives worse results for 5 and 2 times of nominal traffic. It is easy to determine the legal packets when the ratio of the attack traffic to the nominal traffic increases. It would be more difficult to differentiate legal packets when the number of legal packets are equal to the

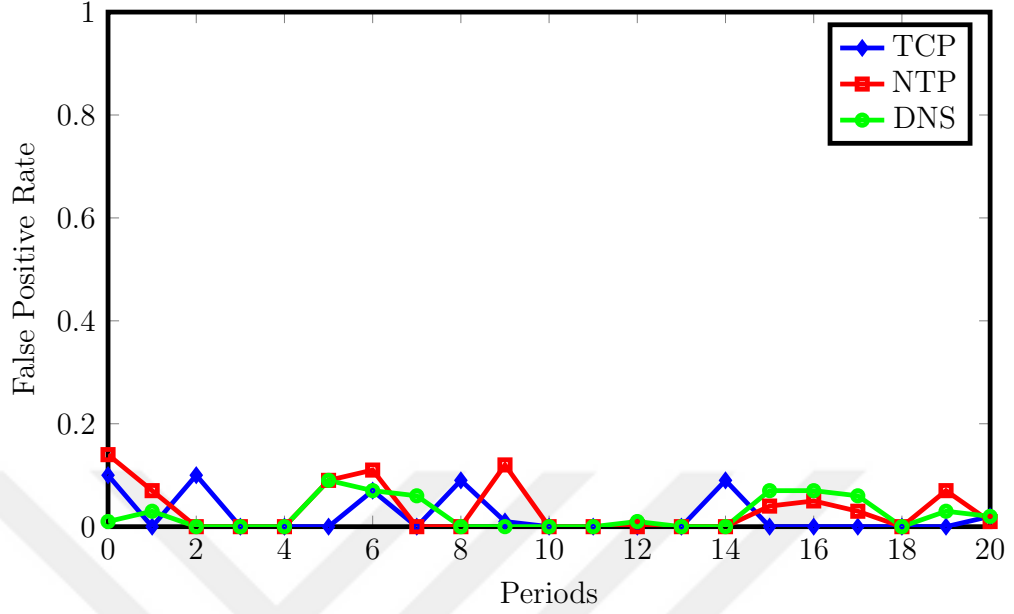


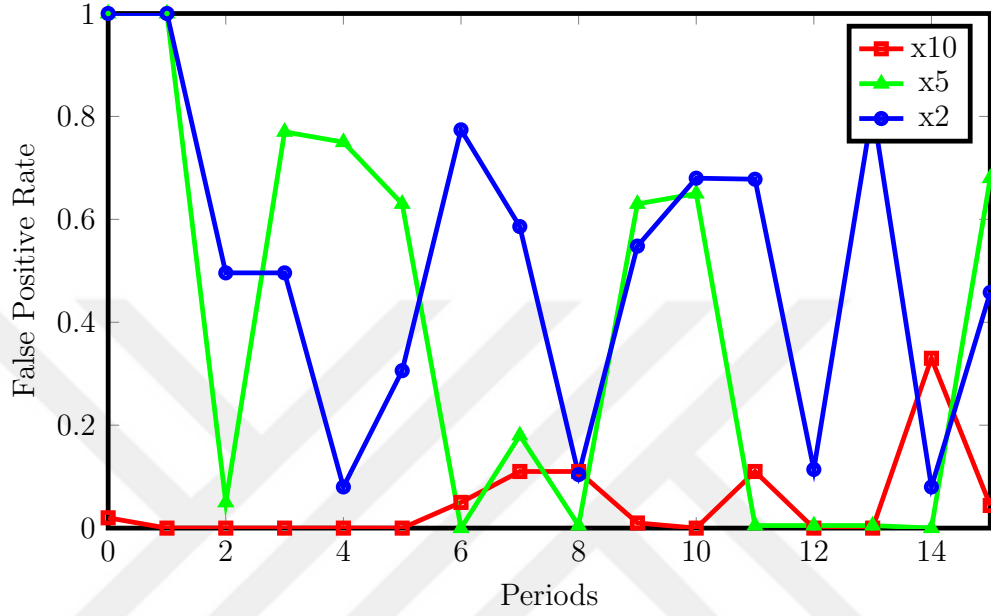
Figure 6.7. FPR during TCP SYN Flood, NTP attack and DNS Amplification with Sliding Windows

number of attack packets as in the case of $x2$ in Figure 6.8(a). However our model JESS still gives promising ACC results for low densities. As it is illustrated in Figure 6.8(b), it gives perfect result for 10 times. For $x5$ accuracy rate is about 90% percent whereas its average falls to % 70s for $x2$. Decrease in accuracy while the attack traffic rate increases is an expected solution. It is elegant that it gives 70% percent accuracy even for low densities. Detecting low volumes of attack is a difficult task, since it is similar to the normal traffic. But, our results suggest that JESS gives decent solutions even for low volumes.

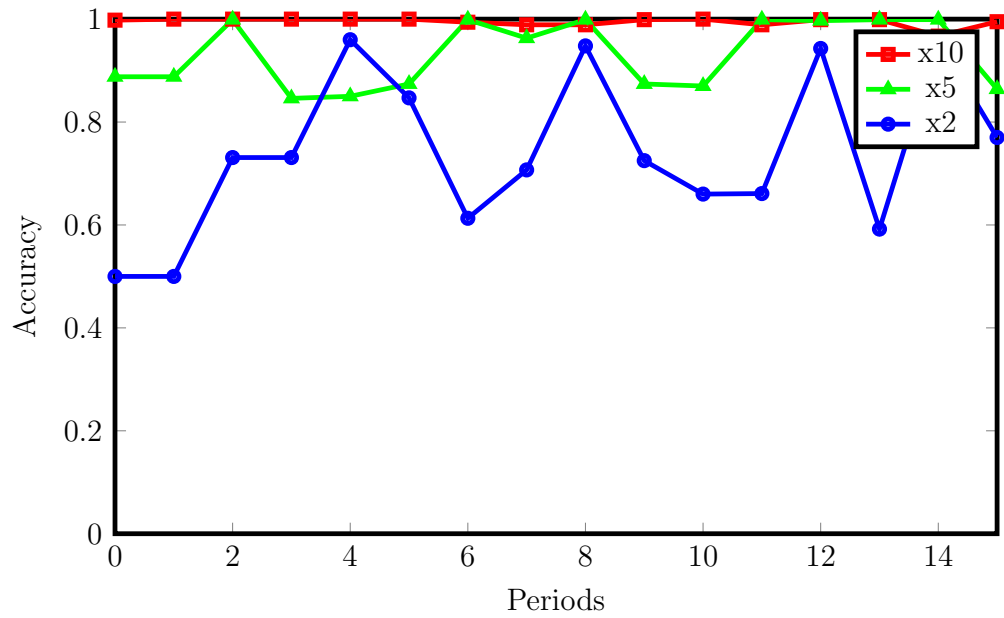
6.2.8. Time Analysis

- (i) Complexity: All the steps in JESS for controller and switch and their corresponding time complexities are shown in Table 6.5.

According to the table all profile generations have $O(N^2)$ complexity because these are generated from pair attributes and they have two dimensions. Similarly rule table generation and comparison operations have $O(N^2)$ complexities because of their two dimensions. Switch has a few operations but as far



(a) FPR during different attack intensities



(b) ACC during different attack intensities

Figure 6.8. Performance of JESS during different attack intensities

Table 6.5. Time Complexity Analysis for steps in JESS

Place	Operation	Complexity
Controller	Nominal Profiles Generation	$O(N^2)$
Controller	Joint Entropy Calculation of Nominal Profiles	$O(N)$
Controller	Current Profiles Generation	$O(N^2)$
Controller	Joint Entropy Calculation of Current Profiles	$O(N)$
Controller	Compare Joint Entropies	$O(N)$
Switch	Current Suspicious Pair Profile Generation	$O(N^2)$
Controller	Score Table Generation	$O(N)$
Controller	Rule Table Generation	$O(N^2)$
Controller	Rule Table Comparison	$O(N^2)$
Switch	Update Rule Table	$O(N)$

as *SuspiciousPair*'s current profile *SPCP* is generated in the switch, the time complexity of our algorithm on both switch and controller is $O(N^2)$. This can be improved by utilizing different types of data structures like dictionaries or hash trees in order to decrease this complexity.

- (ii) Processing: In our simulations we utilized packet based periods. Thus, the mechanism does not consider the time during the analysis. In this section, we provide analysis if JESS's periods are meaningful in terms of timing. How long does a period correspond to and can a switch handle the packets processing or do they need to wait?

For this analysis we utilized from the MAWI dataset. The average packet size is about 500 bytes. Then, each period has $500 * 8 * 1000 = 4Mb$ if a period is 1000 packets long. An average switch's speed is 600 Mbps, then a period's packets processing takes 0.06 seconds. If we analyse the dataset and look at the duration between the first and 1000th packets arrival, this duration is 0.182 seconds. As 0.06 is much less than 0.18, the packets does not need to wait.

6.2.9. Storage Analysis

Since most of the storage is provided by the controller, it is not a vital problem. Because high performance computers are utilized as the controller. Nevertheless, we provide storage analysis for both the switch and the controller. Theoretical and empirical analysis are provided in the previous chapter Section 4.3.4. Since similar profiles are utilized in both models same values can be used. According to the empirical results $311KB$ is needed for a pair profile. As far as there are 6 attributes, there are $\binom{6}{2} = 15$ profiles for each current and nominal operations. Thus there is a need for $311KB * 15 * 2 = 9,3MB$. If we consider IPv6 for destination and IP addresses, then the pairs who have IP addresses will have $622KB$ size, as it is mentioned in 4.3.4. The number of profiles who have $622KB$ will be 5 whereas the remaining have $311KB$ profiles for each current and nominal operations. Then there is a need for $((311KB * 10) + (622KB * 5)) * 2$ is approximately 12.4 MB. This is a few burden for a controller. Also, switch only needs a place for one profile of *SuspiciousPair* which is $311KB$ or $622KB$ if IPv6 is used. These are also too small values even for a basic switch.

6.2.10. Communication Analysis

Communication between the switch and the controller can be analyzed in terms of stages. In Nominal Stage, switch needs to send TCP and IP header of each packet. If the controller and the switch communicates by using IPV6 on TCP, an empty packet will need 24 bytes for Ethernet header, 40 bytes for IPV6 header and 40 bytes for TCP header then it needs 104 bytes without the information. Since the switch also needs to send all TCP and IP header information of the incoming packets, we need to add this information length. If the incoming packet is an IPV6 and TCP packet then it will be $40bytes + 40bytes = 80bytes$. Then totally, the length of a packet that contains header information is 184 bytes. In our simulations, 100 packets were utilized in nominal stage, then it takes $184bytes * 100 = 18,4KB$. Similarly, during the infant stage, all header information is needed to sent to the controller. This time number of packets increased ten times since DDoS attack is started. Then, it takes $184 * 1000 = 184KB$

communication during the infant stage. In the lifelong stage, at the end of each period current profile of the *SuspiciousPair* should be sent to the controller. As far as a pair profile takes about 311KB and its header lengths will be in bytes thus it takes about 311KBs. All the above analysis are done for IPV6 and TCP packets since they have larger header sizes. If IPV4 is utilized header size will be 20 bytes instead of 40 bytes. Similarly, for UDP packets header size will be 8 bytes instead of 40 bytes.

That is to say, all the calculated values for each stage are in KBs. Since the Internet speed is in Mbps in an average place of the world, it is not even a burden for the communication of a switch and the controller.

6.3. Discussions

An important issue is the need of different nominal profiles for different times of day. This is also analyzed in [16]. Their analysis suggests that the traffic profiles are most similar among the traffic at the same day, at the same time. In addition, a traffic profile is still very similar for a different time or day. They compared morning and evening profiles and they suggest that it may be enough to keep one profile. But still, it is possible to compare the stability of the profiles and if there is too much difference, it may be better to keep more than one profile for different times of a day. However, it is obvious that maintaining different profiles per site and per time of day would increase managerial overhead. Thus, there is a trade-off between more accurate profiling and managerial overhead. In our simulations, we suppose that profiles are very similar during the day and one profile is utilized for the nominal traffic.

Another issue is about period lengths. As far as we have utilized packet-based periods, we need to determine how many packets there will be in a period. There should be enough number of packets in a profile in order to have accurate information about the network. But then, in the lifelong stage of JESS at the end of each period current profile of *SuspiciousPair* is sent to the controller. For this reason, it should be in affordable sizes for a communication link which is under attack. In our experiments, we have utilized from 1000 packets since they give enough information and its profile

needs KBs while it is sent to the controller.

There is also another issue about attribute independency. We did not consider dependent attributes while we are choosing the most appropriate attribute pair. But, Some attributes are dependent. For instance a TCP will most probably have port 80. Thus, our model chooses protocol and port number together as the most appropriate pair. However, it would give better solution, if it considers dependency and does not waste attributes in *SuspiciousPairs* for obviously dependent properties.

6.4. Chapter Summary

In this work, a statistical defense mechanism is proposed against DDoS attacks in SDN environment. This model is the first model that utilizes joint entropy in a defense mechanism against DDoS attacks. There are three stages in this model called: nominal stage, infant stage and lifelong stage. In the nominal stage, nominal pair profiles are generated and their joint entropies are calculated by the controller. When the bandwidth exceeds the threshold, the infant stage is started. In this stage, current profiles are generated and their joint entropies are calculated. Then, they are compared with the nominal ones in order to detect the attack. If it is detected, then it determines the *SuspiciousPair*. In the lifelong stage, switch generates current profile of the *SuspiciousPair* and sends it to the controller at the end of each period. Then the controller calculates scores and generates rules. Then the controller sends the differentiated rules to the switch. Since this is a statistical model, it is effective against not only known but also unknown attacks. The results suggest that, it gives perfect results for several known attacks. Besides, it performs 80% success for an unknown attack denoted as *generic attack*. It also performs 67% success for a mixed attack.

As future work, we plan to perform experiments in a real setup. In addition, the controller's role can be enhanced by combining statistics from several switches to provide more accurate scoring. It can use these statistics for preventing the future attacks.

7. CONCLUSION

In this thesis, filtering based defense mechanisms are focused for traditional networks and SDN environment. Initially, a comprehensive survey about filtering techniques are presented. In order to ease choosing the most appropriate mechanism for the needs, we specifically proposed a classification approach that classifies filtering mechanisms according to their timing and collaborative properties. They can be proactive or reactive in terms of timing, whereas they can be individual or cooperative in terms of collaboration property. Then, we focused on cooperative and proactive filtering and proposed a mechanism called ScoreForCore. This model suggests the selection of the most appropriate attributes during current attack traffic. We implemented our model and our results suggest that the success of system's behavior on legal and attack packets are increased considerably. In addition, most of the attack packets are stonewalled near the source of the attack. As our strategy is a flexible solution that can be applicable for not only traditional networks but also future networks, we decided to apply our strategy on SDN environment. Initially, some DDoS scenarios that can be viable in SDN environment are presented. Also, analysis of defense mechanisms in SDN environment are presented. Then, our work proposes a classification of solutions against DDoS attacks in SDN environment.

We applied our strategy which is the selection of the most appropriate attributes during current attack traffic in SDN environment. SDNScore mechanism is a hybrid model in which switches are not simply data forwarders but also can decide on attack packets in cooperation. It is a statistical and packet-based defense mechanism which can detect not only known but also unknown attacks. In addition, it only drops attack packets by preserving legal packets during an attack. Since this model uses capable switches, it poses a trade-off between SDN paradigm and capable switches. In order to get rid off this problem, we proposed another mechanism called Joint Entropy Based Scoring for SDN (JESS) that transfers switch's burden to the controller. Switch only generates a current profile of a pair during the attack. This pair is determined by comparing joint entropies of current and nominal profiles. Then rules are generated

by the controller at the end of the calculation of scores for each value of the pair. The experiments are simulated for several known and unknown attacks. The results suggest that JESS mechanism performs great for several known attacks. Even for an unknown attack type called generic attack it gives % 80 accuracy and % 70 accuracy for a mixed attack.

As a future work, this model can be applied on real SDN environment in order to see more realistic results. Besides, the strategy of the selection of the most appropriate attributes can be applied on other statistical models in the literature. In addition, the controller's role can be enhanced for prevention. Controller can provide prevention by combining statistics from several switches.

REFERENCES

1. Ferguson, P. and D. Senie, “Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing”, *Technical Report RFC 2267*, 1997.
2. Park, K. and H. Lee, “On the Effectiveness of Route-based Packet Filtering for Distributed DoS Attack Prevention in Power-law Internets”, *SIGCOMM Computer Communications Review*, Vol. 31, No. 4, pp. 15–26, August 2001.
3. Jelena, J. L., J. Li, J. Mirkovic, M. Wang, P. Reiher and L. Zhang, “SAVE: Source Address Validity Enforcement Protocol”, *In Proceedings of IEEE INFOCOM '2002*, pp. 1557–1566, 2001.
4. Mahajan, R., S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson and S. Shenker, “Controlling high bandwidth aggregates in the network”, *ACM SIGCOMM Computer Communication Review*, Vol. 32, No. 3, pp. 62–73, 2002.
5. Argyraki, K. and D. R. Cheriton, “Active Internet Traffic Filtering: Real-time Response to Denial-of-service Attacks”, *Proceedings of the Annual Conference on USENIX Annual Technical Conference*, ATEC '05, pp. 10–10, 2005.
6. Liu, X., X. Yang and Y. Lu, “To Filter or to Authorize: Network-Layer DoS Defense Against Multimillion-node Botnets”, *In ACM SIGCOMM*, 2008.
7. Beitollahi, H. and G. Deconinck, “Analyzing well-known countermeasures against distributed denial of service attacks”, *Computer Communications*, Vol. 35, No. 11, pp. 1312 – 1332, 2012.
8. Gligor, V. D., “A Note on Denial-of-service in Operating Systems”, *IEEE Transactions on Software Engineering*, Vol. 10, No. 3, pp. 320–324, May 1984.

9. Gür, G., S. Bahtiyar and F. Alagöz, “Security Analysis of Computer Networks: Key Concepts and Methodologies”, F. Z. M.S. Obaidat, P. Nicopolitidis (Editor), *Modeling And Simulation of Computer Networks And Systems: Methodologies and Applications*, 2014.
10. Geers, K., *Strategic Cyber Security*, CCD COE Publications, 2011.
11. “Prolexic Report: 2014-Q1 Global DDoS Global Attack Report”, <http://www.prolexic.com/knowledge-center/prolexic-download/prolexic-quarterly-global-ddos-attack-report-q114.html>, accessed at October 2016.
12. Selvi, H., S. Güner, G. Gür and F. Alagöz, “The Controller Placement Problem in Software Defined Mobile Networks (SDMN)”, *Software Defined Mobile Networks (SDMN): Beyond LTE Network Architecture*, pp. 129–147, 2015.
13. Xia, W., Y. Wen, C. H. Foh, D. Niyato and H. Xie, “A Survey on Software-Defined Networking”, *Communications Surveys Tutorials, IEEE*, Vol. 17, No. 1, pp. 27–51, Firstquarter 2015.
14. Peng, T., C. Leckie and K. Ramamohanarao, “Survey of Network-based Defense Mechanisms Countering the DoS and DDoS Problems”, *ACM Computer Surveys*, Vol. 39, No. 1, pp. 3–45, April 2007.
15. Jin, C., H. Wang and K. G. Shin, “Hop-count Filtering: An Effective Defense Against Spoofed DDoS Traffic”, *Proceedings of the 10th ACM Conference on Computer and Communications Security, CCS '03*, pp. 30–41, 2003.
16. Kim, Y., W. C. Lau, M. C. Chuah and H. J. Chao, “PacketScore: a statistics-based packet filtering scheme against distributed denial-of-service attacks”, *IEEE Transactions on Dependable and Secure Computing*, Vol. 3, No. 2, p. 141, 2006.
17. Sivia, D. S., *Data Analysis, A Bayesian Tutorial*, Oxford, 1996.

18. Keromytis, A. D., V. Misra and D. Rubenstein, “SOS: An architecture for mitigating DDoS attacks”, *IEEE JSAC*, Vol. 22, No. 1, pp. 176–188, 2004.
19. Liu, X., A. Li, X. Yang and D. Wetherall, “Passport: Secure and Adoptable Source Authentication”, *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, NSDI’08, pp. 365–378, 2008.
20. Seo, D., H. Lee and A. Perrig, “PFS: Probabilistic filter scheduling against distributed denial-of-service attacks”, *Local Computer Networks (LCN), 2011 IEEE 36th Conference on*, pp. 9–17, October 2011.
21. Seo, D., H. Lee and A. Perrig, “APFS: Adaptive Probabilistic Filter Scheduling against distributed denial-of-service attacks”, *Computers & Security*, Vol. 39, pp. 366–385, 2013.
22. Curtis, A. R., J. C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma and S. Banerjee, “DevoFlow: Scaling flow management for high-performance networks”, *ACM SIGCOMM Computer Communication Review*, Vol. 41, pp. 254–265, ACM, 2011.
23. Shin, S., V. Yegneswaran, P. Porras and G. Gu, “Avant-guard: Scalable and vigilant switch flow management in software-defined networks”, *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pp. 413–424, ACM, 2013.
24. Kokila, R., S. Thamarai Selvi and K. Govindarajan, “DDoS detection and analysis in SDN-based environment using support vector machine classifier”, *Advanced Computing (ICoAC), 2014 Sixth International Conference on*, pp. 205–210, IEEE, 2014.
25. Wang, A., Y. Guo, F. Hao, T. Lakshman and S. Chen, “Scotch: Elastically scaling up SDN control-plane using vswitch based overlay”, *Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies*, pp. 403–414, ACM, 2014.

26. Zaalouk, A., R. Khondoker, R. Marx and K. Bayarou, "OrchSec: An orchestrator-based architecture for enhancing network-security using Network Monitoring and SDN Control functions", *Network Operations and Management Symposium (NOMS), 2014 IEEE*, pp. 1–9, IEEE, 2014.
27. Chourishi, D., A. Miri, M. Milic and S. Ismaeel, "Role-based multiple controllers for load balancing and security in SDN", *Humanitarian Technology Conference (IHTC2015), 2015 IEEE Canada International*, pp. 1–4, IEEE, 2015.
28. Dao, N.-N., J. Park, M. Park and S. Cho, "A feasible method to combat against DDoS attack in SDN network", *Information Networking (ICOIN), 2015 International Conference on*, pp. 309–311, IEEE, 2015.
29. Hsu, S.-W., T.-Y. Chen, Y.-C. Chang, S.-H. Chen, H.-C. Chao, T.-Y. Lin and W.-K. Shih, "Design a Hash-Based Control Mechanism in vSwitch for Software-Defined Networking Environment", *Cluster Computing (CLUSTER), 2015 IEEE International Conference on*, pp. 498–499, IEEE, 2015.
30. Lim, S., S. Yang, Y. Kim, S. Yang and H. Kim, "Controller scheduling for continued SDN operation under DDoS attacks", *Electronics Letters*, Vol. 51, No. 16, pp. 1259–1261, 2015.
31. Piedrahita, A. F. M., S. Rueda, D. M. Mattos and O. C. M. Duarte, "FlowFence: a denial of service defense system for software defined networking", *Global Information Infrastructure and Networking Symposium (GIIS), 2015*, pp. 1–6, IEEE, 2015.
32. Mousavi, S. M. and M. St-Hilaire, "Early detection of DDoS attacks against SDN controllers", *Computing, Networking and Communications (ICNC), 2015 International Conference on*, pp. 77–81, IEEE, 2015.
33. Wang, R., Z. Jia and L. Ju, "An Entropy-Based Distributed DDoS Detection Mechanism in Software-Defined Networking", *Trustcom/BigDataSE/ISPA, 2015*

- IEEE*, Vol. 1, pp. 310–317, IEEE, 2015.
34. Gde Dharma, N., M. F. Muthohar, J. Prayuda, K. Priagung and D. Choi, “Time-based DDoS detection and mitigation for SDN controller”, *Network Operations and Management Symposium (APNOMS), 2015 17th Asia-Pacific*, pp. 550–553, IEEE, 2015.
 35. Katta, N. P., J. Rexford and D. Walker, “Incremental consistent updates”, *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pp. 49–54, ACM, 2013.
 36. Tri, N., T. Hiep and K. Kim, “Assessing the impact of resource attack in Software Defined Network”, *Information Networking (ICOIN), 2015 International Conference on*, pp. 420–425, IEEE, 2015.
 37. Kessler, G. C., “Defenses against distributed denial of service attacks”, *SANS Institute*, Vol. 2002, 2000.
 38. Garber, L., “Denial-of-service attacks RIP the Internet”, *Computer*, Vol. 33, No. 4, pp. 12–17, Apr 2000.
 39. Bahtiyar, S., G. Gür and L. Altay, “Security Assessment of Payment Systems under PCI DSS Incompatibilities”, N. Cuppens-Boulahia, F. Cuppens, S. Jajodia, A. Abou El Kalam and T. Sans (Editors), *ICT Systems Security and Privacy Protection*, Vol. 428 of *IFIP Advances in Information and Communication Technology*, pp. 395–402, Springer Berlin Heidelberg, 2014.
 40. “Operation Payback cripples MasterCard site in revenge for WikiLeaks ban”, <http://www.theguardian.com/media/2010/dec/08/operation-payback-mastercard-website-wikileaks>, accessed at October 2016.
 41. Evans, D. and D. Larochelle, “Improving Security Using Extensible Lightweight

- Static Analysis”, *IEEE Software*, Vol. 19, No. 1, pp. 42–51, Jan. 2002.
42. Nazario, J., “DDoS Floods in Belarus: Political Motivations”, <http://asert.arbornetworks.com/2009/06/ddos-floods-in-belarus-political-motivations/>, accessed at October 2016.
 43. Nazario, J., “The Effects of War: Gaza and Israel”, <http://asert.arbornetworks.com/2009/01/the-effects-of-war-gaza-and-israel/>, accessed at October 2016.
 44. Karig, D. and R. Lee, “Remote Denial of Service Attacks and Countermeasures”, *Technical Report of Department of Electrical Engineering, Princeton University*, 2001.
 45. Specht, S. M., “Distributed denial of service: taxonomies of attacks, tools and countermeasures”, *Proceedings of the International Workshop on Security in Parallel and Distributed Systems, 2004*, pp. 543–550, 2004.
 46. Mirkovic, J. and P. Reiher, “A Taxonomy of DDoS Attack and DDoS Defense Mechanisms”, *SIGCOMM Computer Communication Review*, Vol. 34, No. 2, pp. 39–53, April 2004.
 47. Specht, S. and R. Lee, “Taxonomies of Distributed Denial of Service networks, attacks, tools and countermeasures”, *Technical Report CE-L2003-03, Princeton University*, 2003.
 48. Mitrokotsa, A. and C. Douligeris, “DDoS Attacks and Defense Mechanisms: Classification and State-of-the-art”, *Computer Networks*, Vol. 44, No. 44, pp. 643–666, April 2004.
 49. Criscuolo, P. J., “Distributed Denial of Service TrinOO”, Tribe Flood Network 2000, and Stacheldraht CIAC- 2319, Department of Energy Computer Incident

- Advisory (CIAC), UCRL-ID-136939, Lawrence Livermore National Laboratory, 2000.
50. Dittrich, D., “The Tribe Flood Network Distributed Denial of Service attack tool”, Technical Report of University of Washington, 1999.
 51. Barlow, J. and W. Thrower, “TFN2K—an analysis”, *Axent Security Team*, 2000.
 52. Dittrich, D., “The Stacheldraht Distributed Denial of Service attack tool”, University of Washington, 1999.
 53. D. Dittrich, S. D., G. Weaver and N. Long, “The mstream Distributed Denial of Service attack tool”, University of Washington, 2000.
 54. Dietrich, S., N. Long and D. Dittrich, “Analyzing Distributed Denial of Service Tools: The Shaft Case”, *Proceedings of the 14th USENIX Conference on System Administration*, LISA '00, pp. 329–340, 2000.
 55. Hancock, B., “Trinity v3, a DDoS Tool, Hits the Streets”, *Computers & Security*, Vol. 19, No. 7, p. 574, 2000.
 56. Feinstein, L., D. Schnackenberg, R. Balupari and D. Kindred, “Statistical approaches to DDoS attack detection and response”, *DARPA Information Survivability Conference and Exposition, 2003. Proceedings*, Vol. 1, pp. 303–314 vol.1, April 2003.
 57. Lee, K., J. Kim, K. H. Kwon, Y. Han and S. Kim, “DDoS Attack Detection Method Using Cluster Analysis”, *Expert Systems with Applications*, Vol. 34, No. 3, pp. 1659–1665, April 2008.
 58. Toledo, A. and X. Wang, “Robust Detection of MAC Layer Denial-of-Service Attacks in CSMA/CA Wireless Networks”, *Information Forensics and Security, IEEE Transactions on*, Vol. 3, No. 3, pp. 347–358, September 2008.

59. Barford, P., J. Kline, D. Plonka and A. Ron, "A Signal Analysis of Network Traffic Anomalies", *Proceedings of the 2Nd ACM SIGCOMM Workshop on Internet Measurment*, IMW '02, pp. 71–82, 2002.
60. Lu, L. F., M. L. Huang, M. Orgun and J.-W. Zhang, "An Improved Wavelet Analysis Method for Detecting DDoS Attacks", *Network and System Security (NSS), 2010 4th International Conference on*, pp. 318–322, September 2010.
61. Chen, Y., K. Hwang and W.-S. Ku, "Collaborative Detection of DDoS Attacks over Multiple Network Domains", *Parallel and Distributed Systems, IEEE Transactions on*, Vol. 18, No. 12, pp. 1649–1662, December 2007.
62. Wang, H., D. Zhang and K. Shin, "Change-point monitoring for the detection of DoS attacks", *Dependable and Secure Computing, IEEE Transactions on*, Vol. 1, No. 4, pp. 193–208, October 2004.
63. Karimazad, R. and A. Faraahi, "An anomaly-based method for DDoS attacks detection using RBF neural networks", *2011 International Conference on Network and Electronics Engineering, IPCSIT*, Vol. 11, 2011.
64. Li, J., Y. Liu and L. Gu, "DDoS attack detection based on neural network", *Aware Computing (ISAC), 2010 2nd International Symposium on*, pp. 196–199, November 2010.
65. Kompella, R. R., S. Singh and G. Varghese, "On Scalable Attack Detection in the Network", *4th ACM SIGCOMM Conference on Internet Measurement*, pp. 187–200, 2004.
66. Cheng, G., "Malware FAQ: Analysis on DDoS tool Stacheldraht v1. 666", <http://www.sans.org/resources/malwarefaq/stacheldraht.php>, accessed at 2016.
67. You, Y., M. Zulkernine and A. Haque, "Detecting Flooding-Based DDoS Attacks", *Communications, 2007. ICC '07. IEEE International Conference on*, pp.

- 1229–1234, June 2007.
68. Peng, T., C. Leckie and K. Ramamohanarao, “Proactively detecting distributed denial of service attacks using source IP address monitoring”, *International Conference on Research in Networking*, pp. 771–782, Springer, 2004.
 69. Talpade, R., G. Kim and S. Khurana, “NOMAD: traffic-based network monitoring framework for anomaly detection”, *Computers and Communications, 1999. Proceedings. IEEE International Symposium on*, pp. 442–451, 1999.
 70. Jung, J., A. W. Berger and H. Balakrishnan, “Modeling TTL-based Internet Caches”, *IEEE Infocom 2003*, San Francisco, CA, April 2003.
 71. Kim, Y., J.-Y. Jo and K. K. Suh, “Baseline Profile Stability for Network Anomaly Detection”, *Proceedings of the Third International Conference on Information Technology: New Generations*, ITNG '06, pp. 720–725, 2006.
 72. Gil, T. M. and M. Poletto, “MULTOPS: a data-structure for bandwidth attack detection”, *In Proceedings of 10th Usenix Security Symposium*, pp. 23–38, 2001.
 73. Lee, S., H. Kim, J. Na and J. Jang, “Abnormal traffic detection and its implementation”, *Advanced Communication Technology, 2005, ICACT 2005. The 7th International Conference on*, Vol. 1, pp. 246–250, IEEE, 2005.
 74. Carl, G., G. Kesidis, R. R. Brooks and S. Rai, “Denial-of-service attack-detection techniques”, *Internet Computing, IEEE*, Vol. 10, No. 1, pp. 82–89, 2006.
 75. Alenezi, M. and M. Reed, “Methodologies for detecting DoS/DDoS attacks against network servers”, *ICSNC 2012, The Seventh International Conference on Systems and Networks Communications*, pp. 92–98, 2012.
 76. Nemeth, M. A., “Applied Multivariate Methods for Data Analysis”, *Technometrics*, Vol. 42, No. 2, pp. 211–211, 2000.

77. Scarfone, K. A. and P. M. Mell, “SP 800-94. Guide to Intrusion Detection and Prevention Systems (IDPS)”, *Technical Report National Institute of Standards & Technology*, 2007.
78. Mishra, S. and R. Pateriya, “A Comparative Study on Capability v/s. Filtering based Defense Mechanisms”, *International Journal of Computer Applications*, Vol. 93, No. 11, pp. 29–35, 2014.
79. Kambhampati, V., C. Papadopoulos and D. Massey, “A taxonomy of capabilities based DDoS defense architectures”, *Computer Systems and Applications (AICCSA), 2011 IEEE/ACS Ninth International Conference on*, pp. 157–164, 2011.
80. Gryta, T., “AT&T Targets Flexibility, Cost Savings With New Network Design”, <http://www.wsj.com/articles/SB10001424052702303426304579402953146294792>, accessed at October 2016.
81. Jammal, M., T. Singh, A. Shami, R. Asal and Y. Li, “Software defined networking: State of the art and research challenges”, *Computer Networks*, Vol. 72, pp. 74–98, 2014.
82. Kreutz, D., F. Ramos and P. Verissimo, “Towards secure and dependable software-defined networks”, *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pp. 55–60, ACM, 2013.
83. Kalkan, K., G. Gür and F. Alagöz, “Filtering-Based Defense Mechanisms Against DDoS Attacks: A Survey”, *IEEE Systems Journal*.
84. Yan, Q., F. R. Yu, Q. Gong and J. Li, “Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges”, *IEEE Communications Surveys & Tutorials*, Vol. 18, No. 1, pp. 602–622, 2016.

85. Kalkan, K. and F. Alagöz, “A distributed filtering mechanism against DDoS attacks: ScoreForCore”, *Computer Networks*, Vol. 108, pp. 199–209, 2016.
86. Kasera, S., J. Pinheiro, C. Loader, M. Karaul, A. Hari and T. LaPorta, “Fast and robust signaling overload control”, *Network Protocols, 2001. Ninth International Conference on*, pp. 323–331, IEEE, 2001.
87. “MAWI Traffic Archive”, <http://mawi.wide.ad.jp/mawi/>, accessed at October 2016.
88. “The Internet Topology Zoo”, <http://www.topology-zoo.org/>, accessed at October 2016.
89. Foundation, O. N., “Software-defined networking: The new norm for networks”, *ONF White Paper*, 2012.
90. Chin, T., X. Mountroudou, X. Li and K. Xiong, “An SDN-supported collaborative approach for DDoS flooding detection and containment”, *Military Communications Conference, MILCOM 2015-2015 IEEE*, pp. 659–664, IEEE, 2015.
91. Shu, Z., J. Wan, D. Li, J. Lin, A. V. Vasilakos and M. Imran, “Security in Software-Defined Networking: Threats and Countermeasures”, *Mobile Networks and Applications*, pp. 1–13, 2016.
92. Jiang, H., S. Chen, H. Hu and M. Zhang, “Superpoint-based detection against distributed denial of service (DDoS) flooding attacks”, *Local and Metropolitan Area Networks (LANMAN), 2015 IEEE International Workshop on*, pp. 1–6, April 2015.
93. Chen, Q., W. Lin, W. Dou and S. Yu, “CBF: A Packet Filtering Method for DDoS Attack Defense in Cloud Environment”, *Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*, pp. 427–434, December 2011.

94. Juszczyszyn, K. and G. Kołaczek, “Motif-based attack detection in network communication graphs”, *Communications and Multimedia Security*, pp. 206–213, Springer, 2011.
95. Wagner, A. and B. Plattner, “Entropy based worm and anomaly detection in fast IP networks”, *14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise (WETICE’05)*, pp. 172–177, IEEE, 2005.
96. Yu, S., W. Zhou, R. Doss and W. Jia, “Traceback of DDoS attacks using entropy variations”, *IEEE Transactions on Parallel and Distributed Systems*, Vol. 22, No. 3, pp. 412–425, 2011.
97. Kumar, K., R. Joshi and K. Singh, “A distributed approach using entropy to detect DDoS attacks in ISP domain”, *2007 International Conference on Signal Processing, Communications and Networking*, pp. 331–337, IEEE, 2007.
98. Giotis, K., C. Argyropoulos, G. Androulidakis, D. Kalogeras and V. Maglaris, “Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments”, *Computer Networks*, Vol. 62, pp. 122–136, 2014.
99. “Mininet”, <http://mininet.org/>, accessed at October 2016.
100. “Ryu”, <https://osrg.github.io/ryu/>, accessed at October 2016.