UNDERSTANDING AND UNDERMINING
THE BUSINESS OF DDOS BOOTER SERVICES

by

Mohammad Karami
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Information Technology

Committee:

_____	Dr. Songqing Chen, Dissertation Director
_____	Dr. Robert Simon, Committee Member
_____	Dr. Jim Jones, Committee Member
_____	Dr. Dov Gordon, Committee Member
_____	Dr. Damon McCoy, Committee Member
_____	Dr. Stephen Nash, Senior Associate Dean
_____	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering

Date: _____ Summer 2016
George Mason University
Fairfax, VA

Understanding and Undermining the Business of DDoS Booter Services

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

Mohammad Karami
Master of Science
Iran University of Science and Technology, 2009
Bachelor of Science
Applied University of Science and Technology, 2006

Director: Dr. Songqing Chen, Professor
Department of Computer Science

Summer 2016
George Mason University
Fairfax, VA

Dedication

This dissertation is dedicated to my wife Sara, and my parents for their endless love, support and encouragement.

Acknowledgments

I would like to express my deepest gratitude to Dr. Damon McCoy for working so closely with me on my projects and making this dissertation possible. I would also like to thank my dissertation chair Dr. Songqing Chen for his valuable guidance and advice that enabled me to complete this work. In addition, I thank Dr. Robert Simon, Dr. Jim Jones, and Dr. Dov Gordon for serving on my dissertation committee. I really appreciate their flexibility in scheduling my exams. Lastly, I would like to thank my initial advisor Dr. Angelos Stavrou who supported me to obtain my student visa and enabled me to start my studies at George Mason University.

Table of Contents

Page

List of Tables vii

List of Figures ix

Abstract x

1 Introduction 1

1.1 Problem Statement 1

1.2 Thesis Statement 2

1.3 Contributions 3

1.4 Dissertation Organization 6

2 Background 7

2.1 DDoS Attacks 7

2.2 Booter Services 12

2.3 Related Work 16

2.4 Ethical Framework 20

3 Operational Scale of Booter Services 21

3.1 Description of Datasets 21

3.2 Subscribers 23

3.3 Attacks 25

3.3.1 Analysis of Attacks Launched by Users of booter.tw 27

3.4 Revenue 31

3.5 Geography 32

4 Attack Techniques and Infrastructure 34

4.1 Dataset Description 34

4.2 Frontend Servers 36

4.3 Attack Servers 37

4.4 Attack Techniques 38

4.5 Amplifiers 39

4.6 Amplifiers Geolocation 40

4.7 Amplifiers Churn 41

4.8 Bandwidth Amplification Factor 44

4.8.1 Computing Bandwidth Amplification Factor 44
4.8.2 Bandwidth Amplification Factor Measurements 46
4.8.3 Domains Resolved 47
4.9 Attack Power Measurement 49
5 PayPal Intervention 51
5.1 Booter Payment Ecosystem 51
5.2 Usage pattern of PayPal Accounts 52
5.3 Booters' status 55
5.4 Effect of PayPal Unavailability on Attack Levels 57
5.5 Qualitative Assessments 59
5.6 Booters' Responses 60
6 Booter Attack Attribution 62
6.1 Dataset Description 62
6.2 Attack Features Used for Classification 64
6.3 Classification Algorithm 66
6.4 Evaluation Results 67
6.5 Discussion 70
7 Attribution of Economic Denial of Sustainability Attacks 72
7.1 Exploitation of the Utility-based Pricing Model 75
7.2 Related Work 77
7.3 The Proposed Method 80
7.3.1 Hidden Semi-Markov Model and Parameter Estimation 80
7.3.2 HsMM for Detection of Malicious Sources in EDoS Attacks 82
7.4 Experimental Evaluation 84
7.4.1 Dataset Description 84
7.4.2 Attack Scenarios 86
7.4.3 Experimental Results 87
8 Discussion 91
9 Conclusion 93
Bibliography 95

List of Tables

Table Page

3.1 Summary of `asylumstresser.com`, `lizardstesser.su`, and `booter.tw` leaked databases and scraped `vdos-s.com` reported data. † Revenue was converted from Bitcoin to USD. *Revenue is estimated based on subscription cost and number of paying subscribers. 21

3.2 Service usage of the three user groups. 30

3.3 Top country geolocations of booter operators and subscribers based on IP addresses used for logging into their PayPal accounts. 33

4.1 List of booter services we measured, the attack types offered, and the cost of the least expensive one-month subscription. 35

4.2 spoof friendly VPS services tested. 37

4.3 Number of total amplification servers and percentage of overlap with amplification servers used by other booters. 39

4.4 Top country locations and autonomous systems for amplifiers. 40

4.5 Summary of weekly collected attack traces. The numbers within () show the percentage of possible weekly attack traces that we were able to collect. . . 43

4.6 Bandwidth amplification factor across attack types and booters. 46

4.7 Domains resolved by booter services for DNS amplification. 48

4.8 Summary of attack power measurements. 50

5.1 Number of PayPal accounts used by monitored booters before and after the intervention. The numbers within the () are the average lifespan of the accounts used by that booter. Accounts that are active both before and after are counted only in the before and not included when computing the average lifespan. Matching symbols indicate that this set of booters shared at least one PayPal account. These shared accounts might be instances of a third party agreeing to accept payments for these services. 53

6.1 Summary of the self-attacks. 63

6.2 Experimental results for booter amplification attack attribution (E1 and E2). 69

6.3 Experimental results for booter amplification attack attribution (E3). . . . 70
7.1 Amazon EC2 Outgoing Data Transfer pricing as of February 2016. 76
7.2 Summary of the normal experimental dataset. 85
7.3 Mapping of request sizes to relative cost values. 85
7.4 Experimental results for attacks focusing on high number of requests. . . . 89

List of Figures

Figure		Page
2.1	Subscription plans offered by an example booter service.	13
2.2	Structure of booter services.	16
3.1	The number of weekly new paid users of `vdos-s.com`.	24
3.2	The number of weekly attacks launched by users of `vdos-s.com`.	25
3.3	CDF of attack durations.	26
3.4	Distribution of users' contribution to the launched attacks.	28
3.5	`Asylum Stresser` monthly revenue.	32
4.1	An example of a booter front-end.	36
4.2	Churn rate of amplifiers over time.	42
4.3	Overlap of amplifiers seen in weekly attack traces and the initial set of amplifiers.	43
5.1	PayPal account usage over time. Booter domain names are abbreviated to the first three letters. Black asterisks denote a new PayPal account and gaps in the blue line represent PayPal unavailability for that time period. The red vertical line indicates when the reporting of accounts started.	54
5.2	Lifespans of PayPal accounts before and after the intervention.	56
5.3	Status of booters over time.	57
5.4	The impact of PayPal unavailability on daily attack counts.	58
6.1	Jaccard similarity coefficients for amplifier sets abused by example booter services on consecutive dates.	65
7.1	Experimental results for attack strategies focusing on high cost requests.	88

Abstract

UNDERSTANDING AND UNDERMINING THE BUSINESS OF DDOS BOOTER SERVICES

Mohammad Karami, PhD

George Mason University, 2016

Dissertation Director: Dr. Songqing Chen

Distributed Denial of Service (DDoS) attacks are becoming a commoditized service operated by profit-motivated adversaries. While having control over a large number of compromised hosts was traditionally required for an adversary to be able to launch successful DDoS attacks, the emergence of DDoS as a service phenomenon in recent years has made DDoS infrastructure conveniently accessible to a wide range of malicious actors for a minimal cost. This in turn has contributed to the proliferation of DDoS attacks in recent years.

The evolution of underground forums and marketplaces in the last decade has facilitated access to a more robust, effective and easy-to-manage attack infrastructure for the operators of DDoS for hire services. Underground markets offer a diverse range of abusive services and tools for purchase. Among other things, it includes hosting solutions that are IP spoofing friendly and allow malicious DDoS traffic to be transferred, malicious scripts that can be used for initiating DDoS attacks, and lists of publicly accessible misconfigured servers that can be abused to amplify DDoS attack traffic. The dynamics of the modern underground markets have significantly lowered the technical barriers for malicious actors to build DDoS infrastructure and lease it for a small monthly fee, typically ranging from $10-$200.

While we are aware of the existence of an underground economy revolving DDoS for hire, we do not have much insight into the structure of such services and the supporting technical and business infrastructure they rely on. A deeper understanding of the operational internals of these abusive services is the first step towards exploring effective methods for undermining these abusive services.

In this dissertation, we investigate the phenomenon of low cost DDoS as a service better known as booter services in underground markets. We set to understand these DDoS booter services from both a technical and business perspective with the goal of identifying weak points in these services that can be effectively leveraged to undermine them. In the first part of the dissertation, we explore the technical infrastructure of booter services and point out methods of identifying and potentially undermining key pieces.

Research efforts on defending DDoS attacks can be broadly classified to attack prevention or reduction of attacks, identification of attack sources, and detection of attacks as they occur.

During our study, we find booter services to be heavily dependent on convenient payment methods, such as PayPal for selling subscriptions to their customers. While a significant challenge to find effective solutions to completely prevent DDoS attacks launched by booter services, we collaborate with PayPal to conduct a large-scale payment intervention that shows such efforts can be effective in reducing the scale of booter operations and the attacks that they launch.

Next, we build a classifier based on features extracted from a victim's network traces to attribute amplification DDoS attacks to the booter services responsible for launching them. Our experimental results show a promising level of accuracy for attribution of attack instances to booter services.

Due to their effectiveness, volumetric amplification attacks are the primary attack mechanism employed by booter services to deliver their ordered attacks. However, the characteristics of the malicious traffic generated by such attacks is essentially the same whether the attack has been launched by a booter service or not, and detection of amplified volume-based attacks has been extensively studied in the past. We instead consider detection of

a more subtle and recent variation of DDoS attack known as Economical Denial of Sustainability (EDoS). An EDoS attack can be considered as a much more subtle variation of a DDoS attack where the attacker's goal is to disrupt the economical sustainability of a victim cloud consumer by inflicting cost through fraudulent consumption of billable cloud resources. We propose a method for detection of malicious sources engaged in EDoS attacks and experimentally evaluate the performance of the proposed method.

Chapter 1: Introduction

1.1 Problem Statement

DDoS attacks have continuously grown in frequency and traffic volume over the last few years and are becoming a growing threat with high-profile DDoS attacks disrupting many large services. Amplification constitutes a key piece of most modern DDoS attacks. In these attacks, misconfigured UDP-based network services are abused to significantly amplify the amount of malicious DDoS traffic that an adversary is able to generate. The attacker sends small spoofed requests [1] to vulnerable servers and far larger responses are directed from these servers that we refer to them as amplifiers to the victim. To mention a few real-world examples, adversaries targeted *spamhaus.org* with a 300 Gbps DNS amplification attack on March 2013 [2]. CloudFlare announced a nearly 400 Gbps NTP amplification attack on one of the company's customers in February 2014 [3]. Although it may appear that control over a very large number of hosts is necessary to enable an attack of this magnitude, as stated by the CEO of CloudFlare, anyone with access to a 1 Gbps link and a list of a few thousands NTP amplifiers could launch such a devastating attack. As we will show latter in this dissertation through empirical measurements, there are many low-cost DDoS for hire services known as booters in underground forums having access to the infrastructure required for mounting large DDoS attacks. As the last example, Sony PlayStation and Microsoft Xbox networks suffered disrupting DDoS attacks on December 25, 2014. Individuals calling themselves the *Lizard Squad* took responsibility for the attacks and shortly after the incident, the group announced that the attacks were meant to demonstrate the power of `Lizard Stresser`, a booter service they started to offer to users on a subscription basis.

Part of the DDoS attack proliferation in recent years can be attributed to commoditization of DDoS attacks. A large amount of DDoS attacks are being launched by relatively

unsophisticated attackers that have purchased subscriptions to booter services. These services are operated by profit-motivated adversaries that scale up their DDoS infrastructure to meet the growing demand for DDoS attacks that can be used for a large range of abusive activities, such as knocking competing services offline, harassment, and censorship [4, 5].

While imposing a significant security risk, little is known about the technical and business structure of these services and potential weaknesses in these operations that could be used to undermine them. To this end, empirical studies are required to understand the various social, technical and business components involved in the real-world operations of these abusive services. Gaining an in-depth understanding of the internal operations of booter services has the potential for isolating effective intervention methods that can be leveraged to undermine them.

1.2 Thesis Statement

Our hypothesis is that by using empirical methods such as direct interaction with booter services, collaboration with third parties, analysis of leaked datasets, and data collection by web scraping, different aspects of booter operations can be characterized. We hypothesize that this empirical characterization can help us to identify weak points in the ecosystem where intervention efforts could be focused on to best mitigate the threat.

Because of the large number of booter services in operation and the empirical nature of this research, covering all booter services is not feasible. We limit the scope of our research to booter services that were advertised primarily on English speaking underground forums. Many booter services have stability issues and only live for a short period of time. These services are usually operated by individuals with little technical skills and commitment to their paid subscribers. To exclude such unstable services from our analysis, we monitor a larger set of booter services for several weeks to identify and limit our analysis to a subset of more stable booter services.

1.3 Contributions

In this dissertation, we address the gap in our knowledge by undertaking a large-scale measurement study of booter services to understand how they are structured both technologically and economically with the focus of isolating where their potential weaknesses lay. By taking a more holistic view of the problem, we can offer several alternative and possibly more effective methods for undermining these profit-motivated services.

The first challenge tackled was to measure the attack infrastructure of booter services to understand their capability and asses the risk they impose. To this end, we locate and directly engage with 15 identified booter services to understand the technical infrastructure used to launch attacks. Previous work has focused on more professional services that are rented by fellow cybercriminals or smaller scale studies that only looked at one booter. We find that these services are mostly targeted at more casual Internet users, such as gamers or people that want to bully or otherwise harass other Internet users. In addition, we find that the operators of these booters are possibly more risk adverse and business oriented, leading many of them to rent more stable attack servers from hosting services that offer unlimited bandwidth, the ability to spoof packets and permission to launch DDoS attacks at a low-cost. This provides them with a more stable and low risk attack infrastructure than one built using compromised hosts. Also, by directly interacting with these services we can map out the vulnerable amplification servers that are being actively abused by these services. From these measurements, we find that the pool of servers used by booters tend to be a more stable subset of the servers located by scanning the Internet for vulnerable servers.

Based on this finding we recommend that patching and notification efforts be focused on these actively abused servers in an attempt to undermine booters' attack infrastructure. We also find that past patching efforts of NTP hosts has been effective, but enough vulnerable NTP servers remain for booters to launch effective attacks using NTP. We also find that an effort to patch CharGen hosts might eliminate the small remaining pool of these servers that are launching high amplification attacks. Finally, we find that by purchasing hosting

from DDoS friendly hosting services we can isolate their locations and potentially exert pressure on these hosting services.

We further analyze leaked data and scraped data from four booters to understand the scale of these services and the payment methods they use. We find that for the two services that accepted PayPal as a payment method, almost all of their revenue was collected using PayPal. We also find that `Lizard Stresser`, the one service that didn't accept PayPal, has less than 2% conversion rate of registered users to paid subscribers and has a large number of support tickets requesting that they accept PayPal payments. This suggests that booters' more casual subscriber base has a difficult time using alternate payment methods and that by not accepting PayPal or other convenient payment methods this results in a smaller amount of revenue for the service. Aggregate geolocation information provided by PayPal also suggests that the subscribers of these services are largely located in the United States, thus making it more difficult for them to access virtual currencies more commonly used by cybercriminals.

In order to further measure this effect we engage with PayPal to undertake a large-scale payment intervention to disrupt the merchant accounts used by booters to collect payments. We measure PayPal account usage and availability before and during this intervention for 23 stable booters. Our results show that 7 booters ceased operation during the 6 week intervention period and many of the booters experienced payment outages from frozen accounts that resulted in lost revenue from seized funds in the account and opportunity costs. Some of these services directly blamed PayPal as the reason they ceased operation.

Thirdly, we focus on the problem of attributing DDoS attacks to booter services responsible for launching them and show that it is feasible to build a classifier for this purpose based on features extracted from a victim's network traces. In order to validate our proposed method, we subscribe to 23 booter services and generate a ground truth dataset of attack instances to booter service mappings. Our experimental results show a promising level of accuracy for attribution of attack instances to booter services responsible for launching them.

Finally, we consider EDoS attacks as a more subtle and recent variation of DDoS attacks which are very likely to be offered by DDoS for hire services in the near future as the underground economy evolves and more small businesses start to use public clouds to run their operations and services. In DDoS attacks, the attacker's goal is to render a target service unavailable to its intended users by overwhelming victim's resources. In contrast, EDoS attacks are not meant to cause availability issues or noticeable degradation of service quality for the users of a target service. EDoS attacks instead target the financials of public cloud consumers. An EDoS attacker seeks to increase the financial burden of the victim service by making fraudulent requests that result in high consumption of billable resources for which the victim will have to pay the cost. We propose a method based on Hidden Semi Markov Model (HsMM) for detection of malicious sources engaged in EDoS attacks and experimentally evaluate the performance of the proposed method.

To summarize, our contributions are as follows:

- We perform the first large-scale measurement study to understand booter services by combining measurements from direct interaction, leaked data and scrapers that monitored their operations. As a result of these measurements we isolate several potential weaknesses in their technical and business infrastructure.

- We perform a large-scale payment intervention with the help of PayPal and are able to measure the effectiveness of the intervention. Our results show both quantitatively and qualitatively that this intervention had a large impact on the booter ecosystem.

- We present a booter attribution method that allows a DDoS victim to attribute attacks based on features extracted from the victims network traces.

- We propose an anomaly detection scheme based on HsMM to profile the behavior of legitimate users in terms of their resource consumption and to detect malicious sources engaged in fraudulent use of cloud resources as part of an EDoS attack.

- We provide a list of key lessons learned from our study and describe a number of

other potentially effective methods of further understanding and undermining booter services at both a technical and socio-economic level.

1.4 Dissertation Organization

The rest of this dissertation is organized as follows. In chapter 2, we provide a background on booter services and explain the ethical framework for our study. To understand the scale of booter operations, chapter 3 presents our analysis of leaked and scraped data from four booter services. The next chapter presents measurements of booters' attack infrastructure. In chapter 5, we present our analysis of a payment intervention that resulted in revenue disruption for several booters. We present our proposed methodology for attributing amplification DDoS attacks to booter services in chapter 6, and our anomaly detection scheme for identifying malicious sources participating in EDoS attacks in chapter 7. Finally, discussion and conclusion remarks are presented in chapters 8, and 9, respectively.

Chapter 2: Background

In this chapter we present some background on DDoS attacks, explain the high level business and technical structure of booter services, discuss previous work related to our research, and talk about the underlining ethical framework for our study.

2.1 DDoS Attacks

A Denial of Service (DoS) attack is an explicit attempt by a malicious party to render a service unavailable to its intended users [6–8]. A Distributed Denial of Service (DDoS) attack has the same goal as a DoS attack but rather than using a single host to perform the attack, multiple distributed resources are utilized for a more devastating effect [6–8].

DDoS has been known as an issue for a long time and DDoS attack and defense techniques have been studied for close to two decades [6,7,9–17]. There are two major approaches taken by DDoS attacks in an attempt to disrupt access to a target service. The more popular approach relies on the aggressive exhaustion of limited key resources of a victim system such as network bandwidth, memory, or computational resources to interrupt legitimate access to the services of a victim system. Network links are the resource most frequently targeted by DDoS attacks in this category. The other approach relies on software flaws in operating systems, applications or communication protocols to crash a target system with a few malformed packets. The ping of death attack fits in this category. Using this attack, a malicious attacker was able to easily crash a wide range of target systems running early implementations of the TCP/IP stack by sending a single malformed packet [18]. It's relatively easy to defend against such attacks by either eliminating the root cause vulnerability by applying software patches or by adding firewall rules to protect target systems by filtering out malicious packets [19].

Depending on how attack traffic is transmitted from a host initiating the attack to the victim host, DDoS attacks can be classified to direct attacks and reflexive attacks. In a direct attack, the attack traffic is directly sent from the source of attack to the victim. Direct attacks usually employ source IP spoofing and random source ports for the outgoing attack traffic. This generally makes it very difficult for the attack target to distinguish attack traffic from normal legitimate traffic. In reflexive attacks, intermediate hosts referred to as reflectors are used to direct attack traffic to victims [9]. Reflectors are usually misconfigured legitimate services abused by attackers to hide their IP addresses and in most cases to also amplify the amount of traffic which is sent to a victim. The attacker sends request packets to the vulnerable service with the source IP address set to IP address of the victim. As a result, the victim is flooded with response packets from the abused reflector. DNS amplification is a common attack type that fits in this class of attacks. Due to the lack of source port randomization, reflexive attacks are easy to detect and filters can be implemented to drop attack traffic. However, to protect a victim service, an upstream link should have enough available bandwidth to be able to handle the incoming attack traffic. Otherwise, the victim service will be disrupted even though the attack traffic is easily detectable. Because of this, there are few links on the Internet that can stand very large-scale amplification attacks.

According to Peng et. al. [20], the mechanisms proposed to defend against DoS or DDoS attacks can be classified in four broad categories, namely, attack prevention, attack detection, attack source identification, and attack reaction. Next, we will briefly discuss each of these categories of defense mechanisms.

Attack prevention: The aim of attack prevention is to protect attack targets by stoping attack attempts at a point as close as possible to the source of attacks. Ingress/Egress filtering is a major method proposed to prevent attacks employing source IP spoofing. The rationale behind ingress/egress filtering is to make sure that only packets carrying source addresses within an expected IP address range are permitted to enter or leave a network [21]. To be effective, these packet filters need to be deployed globally on network routers. Unfortunately, due to the open and decentralized nature of the Internet, a global

implementation to prevent attacks based on source address spoofing is not practical [1, 22]. Furthermore, this attack prevention method could only be effective for DDoS attacks using source address spoofing. However, botnet based DDoS attacks supported by a large number of compromised hosts don't need to employ source address spoofing to hide the identities of offending hosts.

Attack detection: If an attack can not be prevented from happening, it's beneficial to detect and mitigate an attack in the links close to the source of the attack. This reduces the cumulative network resources wasted for handling the attack traffic. Because of performance degrades resulted from a sever ongoing DDoS attack, it's not usually very difficult to infer that an attack is happening. It's however much more challenging to accurately classify individual packets as malicious or legitimate traffic. Because attack traffic can look very similar to legitimate traffic, detection schemes have to be designed to minimize false positives.

Similar to the more general problem of intrusion detection, DDoS attack detection can be either signature based or anomaly based [23]. None of the two detection approaches is perfect and each one has its own cons and pros. In the signature based approach, previous attacks are analyzed to identify attack signatures and use them to define detection rules. SNORT [24] and Bro [25] are two popular signature based detection systems that match existing rules against incoming traffic to detect attacks. The signature based approach is unable to detect new attacks for which no prior detection rule exists. On the other hand, the anomaly based approach has the potential for detecting unknown attacks but it is also susceptible to high false positive rates.

Attack source identification: source identification is useful for blocking malicious traffic from attacking hosts. However, there is no easy way to trace back traffic to the source of an attack when source IP addresses are spoofed. To address this shortcoming, many schemes have been proposed to support source IP traceability. Most of these schemes require modifications of existing protocols or support for additional functionalities by network routers.

There are a few major approaches for IP traceback. The first approach is characterized by routers actively interfering with detected attack traffic and observing the reaction of attack traffic to identify the source of attack traffic. Backscatter traceback [26], and link-testing traceback [27] are example techniques based on this approach. A major shortcoming of traceback techniques taking the active interaction approach is the substantial control required to coordinate all participating routers. However, the way that the Internet is managed makes the possibility of such control and coordination very unlikely.

Probabilistic IP traceback is the next major approach proposed to identify attack sources. Here the idea is to extend the functionality of routers to probabilistically include partial path information in some of the incoming packets. This additional information can then be used by an attack target to reconstruct packet paths. The probabilistic packet marking (PPM) proposed by Savage et. al. is an example scheme based on the probabilistic IP traceback approach [28]. Probabilistic schemes are based on the assumption that a significant amount of attack traffic will be transmitted on a single path from an attack source to an attack target. However, this assumption doesn't hold for highly distributed attacks with many participating attack sources sending out the attack traffic on a large number of independent paths. For such attacks, the probabilistic marking approach fails to identify attack sources due to the lack of significant attack traffic on independent paths.

To make source IP traceback independent of the attack volume, a third approach based on storing the hash of routed packets on network routers has been proposed. Assuming that the hash of an attack packet has been recorded by all routers on the path from an attack source to an attack target, the target can query its upstream router for the attack packet, this router is able to look up its records to identify the port on which the queried packet was received. This way, the source of attack can be identified by querying upstream routers recursively. The hash-based IP traceback is an example scheme based on this approach [29]. In addition to the overhead of computing and storing the hash of incoming packets, global deployment by network routers is required for this approach to succeed.

In general, the IP traceback schemes are most helpful when there are only a small number

of attack sources. In the case of highly distributed DDoS attacks with a large number of attack sources, most proposed schemes do not scale and even if all attack sources are identified, it's not clear how to take meaningful actions against a large number of identified attack sources.

Attack reaction: To minimize the damage caused by DDoS attacks, a reaction mechanism needs to be deployed to react when an ongoing attack is detected. A reaction mechanism can be deployed at the network where an attack source is located, at the attack target host or the network where the attack target is located, or at the intermediate network where the path from an attack source to an attack target passes through.

A target host can employ various resource management techniques to mitigate the damage of DDoS attacks. For example, a host can configure its operating system to disallow numerous half-open TCP connections and therefore defending against TCP SYN flood attacks. Also, the immediate network of attack targets can deploy a mechanism to detect and filter out malicious attack traffic. Reaction mechanisms at target hosts or routers close to them can be effective and the owner of targeted resources are motivated for deployment of reaction schemes. Most of today's commercial DDoS protection solutions belong to this category of reaction mechanisms.

Alternatively, a reaction mechanism can be deployed at the immediate network where an offending host is located. It's feasible to detect and block abnormal attack traffic at the attack source network. This is an ideal reaction mechanism as it minimizes the amount of collateral damage that can be caused by attack traffic. This approach works best when the attack traffic is generated by a few aggressive attacking hosts. For a highly distributed DDoS attack where each participating attack source only generates a small portion of the whole attack traffic, a router at the attack source network may not be able to detect an block the attack traffic. Finally, as network operators have little financial incentive to deploy reaction mechanisms on their outgoing traffic, this class of reaction mechanisms is not expected to see deployments at large scale.

The last option is to deploy a reaction mechanism at intermediate routers positioned

between an attack source and an attack target. The further the intermediate routers from the victim of a DDoS attack, the more difficult is detection of malicious attack traffic. Because of this, victims must use a communication mechanism to push information on ongoing attacks to upstream routers so that they can start filtering out unwanted traffic. Compared to blocking attack traffic at the target host or network, it's preferential to filter out the attack traffic at the intermediate network to reduce the amount of collateral damage. However, ISPs are less motivated than targeted systems to deploy reaction mechanisms to mitigate DDoS attacks.

2.2 Booter Services

DDoS has been known as an issue for close to two decades and has received extensive attention from research community. However, due to the evolution of cybercrime ecosystem in the last decade, the landscape of DDoS has seen transformative changes. While access to a large number of compromised hosts was traditionally required for launching successful DDoS attacks [30, 31], the evolution of underground cybercrime has resulted in commoditization of DDoS, where the attack infrastructure has been made conveniently accessible to a range of different malicious adversaries for a minimal fee.

Thinly veiled booter services have existed since at least 2005 and primarily operate using a subscription based business model. As part of this subscription model, customers or subscribers [1] can launch an unlimited number of attacks that have a duration typically ranging from 30 seconds to 1-3 hours and are limited to 1-4 concurrent attacks depending on the tier of subscription purchased. The price for a subscription normally ranges from $10-$300 USD per a month depending on the duration and number of concurrent attacks provided. Figure 2.1 shows a screenshot of subscription plans offered by an example[2] booter service. As seen, this particular booter service offers subscription plans ranging from 1 day to 10 years in period. The maximum duration of a single attack ranges from 200 seconds

[1] We use these two terms interchangeably in this dissertation.
[2] These are subscription plans offered by inboot.me on March 2015.

to 7200 seconds for different subscription plans and customers are offered to purchase plans that allow them to initiate up to 3 concurrent attacks. As shown, this booter service is accepting payments using PayPal and Bitcoin. As we will discuss later in more details, PayPal and Bitcoin are the primary fund transfer mechanisms used by booter services for receiving payments from their customers.

PLAN	BOOT TIME	CONCURRENTS	LENGTH	PAYMENT
Trial **$4.99**	200sec	1	1 Days	
1 Month Bronze **$11.99**	600sec	1	1 Months	
1 Month Silver **$15.99**	1200sec	1	1 months	
3 Months Bronze **$19.99**	600sec	1	3 months	
1 Month Gold **$23.99**	2400sec	1	1 months	
1 Month Diamond **$34.99**	3600sec	2	1 months	
Lifetime Bronze **$44.99**	600sec	2	10 years	
3 Months Silver **$45.99**	1200sec	2	3 months	
3 Months Gold **$59.99**	2400sec	2	3 months	
1 Month Paradise **$62.99**	7200sec	2	1 months	
Lifetime Silver **$79.99**	1200sec	2	10 years	
Diamond Lifetime **$300**	7200sec	3	10 years	

Figure 2.1: Subscription plans offered by an example booter service.

In order to maintain a facade of legitimacy, booter services often describe themselves as network stress testing solutions meant to be only used by network operators to stress test their own infrastructure. However, these same services market themselves as DDoS services that "hit hard" on underground forums such as *hackforums.net* and offer a number of add-on services, such as locating a victim's IP address via his Skype ID [32] and a server's real IP address to get around CloudFlare and other anti DDoS services. In practice, booter

services have become synonymous with DDoS for hire and are a growing threat due to the fact that they have commoditized DDoS attacks that reach upwards for 2-3 Gbps. By offering a low-cost shared DDoS attack infrastructure, these criminal support services have attracted thousands of malicious customers and are responsible for hundreds of thousands of DDoS attacks a year as we will show in chapter 3.

Booter services are found advertised in underground forums [33] and by simple web searches for terms, such as "stresser" and "booter". They maintain front-end sites that allow their customers to purchase subscriptions and launch attacks using simple web forms. Also, booter services typically use a database to record details on registered users, payment transactions and attacks launched by users.

The back-end infrastructure of booter services consists of malicious DDoS attack scripts, lists of misconfigured hosts abused for reflecting and amplifying attacks and most rent high-bandwidth Virtual Private Servers (VPS) to perform attacks. On the same underground forums as where booters are advertised, there are advertisements from hosting services that rent servers and are tolerant of launching DDoS attacks. These advertisements and comments from the operators of these booter services indicate that many of them are renting VPSs instead of using compromised servers or large botnets for their attack infrastructure.

For DDoS amplification attacks performed by botnets, each participating bot makes malicious requests to a list of vulnerable reflectors. As the bots are typically geographically distributed, malicious payloads as seen by an abused amplifier contain a diversified set of TTL values in the IP header. However, researchers operating honeypot amplifiers have observed many attack instances for which the attack payloads contain a fixed TTL value [34]. This is the expected behavior when amplification DDoS attacks are performed using rented servers rather than botnet of compromised hosts.

From booters perspective, using rented servers to perform DDoS attacks could be a reasonable business decision. Compared to compromised hosts, rented servers are often much more powerful, they are also more stable and do not have a significant maintenance overhead.

Ironically, booter services depend on DDoS protection services, such as CloudFlare, to protect their front-end and attack infrastructure from attacks launched by rival competing booter services.

Our analysis of victims of DDoS attacks launched by users of booter services shows that they are predominantly residential links and gaming related servers, with a small number of higher profile victims, such as government, journalist and law enforcement sites. Although, as we will show in chapter 4 most booter services have access to the required infrastructure to enable them to launch large attacks on the order of a few hundreds Gbps, they confine themselves from doing so. The attack capacity of booter services is shared among their users and individual users are typically able to initiate attacks that are limited to 2-3 Gbps in traffic volume. This amount of malicious traffic suffices for easily overwhelming almost all residential links as well as many small to medium sized websites. Furthermore, an adversary can easily subscribe to multiple booter services and launch larger attacks by simultaneously directing the attack traffic of each booter service to a single victim. Most booter services find being part of high-profile attacks and therefore bringing attention to their operation to be detrimental to their business. However, some booter services might be willing to offer more powerful attacks to users paying for premium subscriptions.

Figure 2.2 provides a detailed illustration of the infrastructure and process of using a booter service. (①) The customer first locates a booter site and visits their front-end web server, which is normally protected by CloudFlare. (②) The customer must next purchase a subscription using a payment method, such as Bitcoin or PayPal. (③) The customer then uses the front-end interface to request a DDoS attack against a victim. (④) This request is forwarded from the front-end server to one of the back-end attack servers. (⑤) The back-end server then sends spoofed request packets to a set of previously identified vulnerable amplification servers.[3] (⑥) Finally, DDoS traffic in the form of replies is sent

[3]This is only the case for amplification based attacks, for other attacks such as TCP SYN flood, attack traffic is directly sent from back-end servers to an attack target.

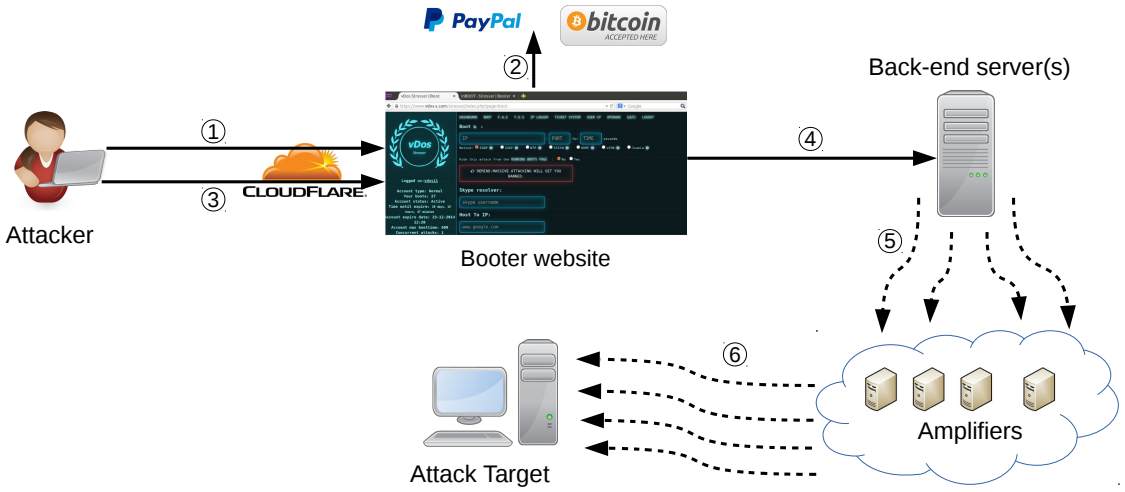

Figure 2.2: Structure of booter services.

to the victim from the abused amplification servers.

2.3 Related Work

One of the most well known empirical studies of DDoS attacks in the wild was performed using backscatter analysis [35] and these measurements were revisited by Wustrow et. al. [36]. The goal of backscatter analysis is to estimate the prevalence of DDoS attacks on the Internet where the source IP addresses of attack packets are randomly spoofed. To hide the source of attack traffic, some DDoS attacks spoof the source IP address. Assuming that the spoofed source IP is randomly selected from the entire IPv4 space, backscatter analysis is able to infer DDoS attacks by measuring unwanted traffic observed on unused IP address blocks. The unwanted traffic seen on unused network addresses are generated by victim hosts when replying to spoofed requests. Besides inferring DDoS attacks, the idea of monitoring unused network addresses is a popular method to study other forms of Internet threats including botnets and worm propagations [37]. Backscatter analysis only works when source IP addresses are randomly spoofed and is irrelevant otherwise. This includes much of modern DDoS attacks including attacks based on abusing reflectors to

amplify attack traffic [38].

More recent works have studied various types of amplifiers that comprise a key component of most modern volume-based DDoS attacks [16, 17, 39]. Other recent studies have focused on in-depth measurement and analysis of specific UDP-based network protocols such as Network Time Protocol (NTP) [40, 41] and Domain Name System (DNS) [42, 43]. NTP and DNS are among the most widely misused protocols in amplification-based DDoS attacks in recent years.

There is a body of work exploring the structure of botnet supported DDoS attacks and methods to mitigate such attacks [31, 44–50]. A recent work in this area closer to our work was done by Welzel et. al. [46]. In this study, victims of DDoS attacks launched by *DirtJumper* and *Yoddos* botnets were monitored to measure the impact of attacks on them. For this measurement study, the researchers infiltrated the two botnets and were able to observe and measure the impact of DDoS attacks on 646 distinct victims by monitoring a total of 14 identified Command and Control (C&C) servers for a total duration of 10 weeks. As we will discuss in section 3.3, booter services and botnets tend to be used in very different ways to perform DDoS attacks.

As pointed to earlier, building and running a botnet of compromised hosts introduces significant overhead. Furthermore, botnets are often susceptible to infiltration [51], take over [52], or take down [53–57] efforts and their operators run a high risk of legal actions being taken against them [54,55]. Finally, compromised workstations misused for launching DDoS attacks are often connected to Internet using low-bandwidth links offering limited abuse potential to botmasters. To avoid the issues of building and running botnets, many operators of modern DDoS for hire services have started using powerful rented or occasionally compromised servers in conjunction with publicly accessible amplifiers to build their attack infrastructure. In this dissertation, we will exclusively focus our efforts on characterizing different aspects of this category of DDoS attack infrastructure which is a more recent phenomenon and has received very little attention from the research community [58, 59].

Contrary to most previous works merely focusing on understanding the attack infrastructure and characterizing the malicious DDoS traffic, we seek a holistic understanding of the whole ecosystem of booter services. This includes customers, the attack infrastructure, victims, operators of DDoS services and economical aspects of their abusive activities.

In this goal, our study is similar to prior work looking at stakeholders and infrastructure of criminal enterprises in other domains, such as abusive advertising [60–63], malicious Bitcoin mining [64], and fake anti-virus [65]. Since booters are a criminal support service rather than the previously studied domain of abusive advertising they operate under a different set of constraints. In this respect our work is more along the lines of studies focused on criminal support services, such as email spam delivery [66,67], malware distribution [68] and fake account creation [69].

Depending on the problem under investigation, the details of how empirical studies exploring criminal enterprises and criminal services supporting their operations are conducted could be fairly specific to the domain. However, a general procedure is followed by most of these studies for conducting the research. First, one or more relevant data collection methodologies are selected to collect an empirical dataset to characterize the malicious activity under study, then this is followed by an analysis phase where the collected data is analyzed to discover interesting details, drawing conclusions and coming up with ideas for mitigating the problem. The following is a brief description of the most frequently employed data collection methodologies for acquiring empirical datasets in the research community [70]:

- **Leaked data**: Occasionally, systems used by cybercriminals to run abusive activities are compromised and the operational data becomes publicly accessible on the Internet. This offers the research community an opportunity to look at ground truth data to learn the details of how the underground economy operates. Additionally, leaked data can be used as a basis for evaluation and validation of results from research studies not based on ground truth data. Here the challenge is to validate the authenticity of the acquired data.

- **Third party collaboration**: Depending on the abusive activity under study, there are usually organizations which have a vantage point for observing some details of malicious activities and it is not uncommon among the researchers in the field to collaborate with such organizations to acquire relevant datasets.

- **Infiltration**: This is basically a form of ethical hacking of the infrastructure used by cybercriminals to learn about their operations. This is usually realized through reverse engineering components of an abusive infrastructure and building benign components mimicking the behavior of malicious components to gain an insider view of the abusive activities.

- **Direct observation and sampling**: In this data collection methodology, researchers directly interact with an abusive service which provides a publicly accessible interface to observe and collect a sample representative dataset characterizing the activity under study. Usually an infrastructure is built to automate the interaction and making longitudinal or large-scale studies feasible. Depending on the specific abusive activity under investigation, the implementation of this methodology can take many different forms. The general idea however, is to collect the dataset by direct observations in the real world where an abusive activity is operating.

- **Service purchase**: Sometimes actively engaging in purchasing abusive services offered by cybercriminals on underground communities and using them provides an opportunity to understand the operational details of the purchased service.

In this dissertation, all of the mentioned data collection methodologies except infiltration are employed to acquire empirical datasets to characterize different aspect of booter DDoS services.

In summary, subscription based booter services are a growing threat but the ecosystem of these abusive services has not been studied in much depth. These services are structured differently from traditional botnet based DDoS services that are rented for a fixed time period in terms of the underlying attack infrastructure, and also the customer base, business

model and payment methods are different for booter services. We believe that understanding the nature of booter services, how they are structured and a detailed characterization of the infrastructure used to deliver ordered attacks, enable us to assess the potential of various intervention choices at the infrastructure level to mitigate the problem. Also, we believe that our PayPal intervention study enables us to gain a better understanding of the effect of payment intervention efforts aiming at disrupting booter services.

2.4 Ethical Framework

As part of the ethical framework for our study, we placed a number of restrictions on the types of booter services we interacted with and what we included in this dissertation. First, we did not engage with any DDoS service that advertised using botnets to perform attacks and ran tests to detect if a service was using botnets. In the single case of `Lizard Stresser`, when we detected a botnet was being used, we immediately abandoned plans to collect active attack measurements from this service and restricted ourselves to passive measurements.

In order to collect some of our measurements, we had to purchase subscriptions from booter services. When purchasing a subscription for a booter service, we always selected the cheapest option to minimize the amount of money given to these services. Based on the guidance of our institution's general counsel, our victim host was connected by a dedicated 1 Gbps network connection that was not shared with any other hosts. We also obtained consent from our ISP before conducting any DDoS attack experiments. We also tried to minimized the self-attack durations and had a protocol in place to end an attack early if it caused a disruption at our ISP.

Finally, we did not disclose customers or operators of these services even when we became aware of their identities and we did not reveal the identities of any victims unless they had been previously publicly disclosed. We received an exemption from our Institutional Review Board (IRB), since our study did not include any personally identifiable information and was based on publicly leaked data and measurements of services that are publicly accessible.

Chapter 3: Operational Scale of Booter Services

In this chapter, using publicly leaked databases of three booter services, `lizardstresser.su`, `asylumstresser.com`, and `booter.tw` and also data collected by frequent crawling of `vdos-s.com`, we present some numbers to better understand the scale of booter services. This includes, the number of users, the number of victims and the number of attacks initiated by the subscribers of theses services.

3.1 Description of Datasets

Our datasets for this chapter are comprised of three leaked back-end databases for `Asylum Stresser` , `Lizard Stresser`, and `Twbooter` along with scrapped data from `vdos-s.com`. Table 3.1 summarizes the datasets that we analyze in this chapter. For leaked datasets, the period is calculated based on the timestamp of the first and the last attack record contained in the dataset. Before presenting our analysis, we will first describe each of these data sets in more detail.

Table 3.1: Summary of `asylumstresser.com`, `lizardstesser.su`, and `booter.tw` leaked databases and scraped `vdos-s.com` reported data. † Revenue was converted from Bitcoin to USD. *Revenue is estimated based on subscription cost and number of paying subscribers.

Booter	Period	All Users	Subscribers	Revenue	Attacks	Targets
asylumstresser.com	11/23/2011 - 03/22/2013	26,075	3,963	$35,381.54	483,373	142,473
lizardstesser.su	12/30/2014 - 01/12/2015	12,935	176	$3,368 †	15,998	3,907
vdos-s.com	12/02/2014 - 02/03/2015	11,975	2,779	$52,773*	138,010	38,539
booter.tw	01/23/2013 - 03/15/2013	312	108	$8,127	48,844	11,174
Total	-	51,297	7,026	$99,649.54	686,225	196,093

vdos-s.com Scraped Data. At the time we started monitoring `vdos-s.com` to measure the scale of its operation in early December 2014, it was one of the top booter services on

underground forums with a high rate of positive reviews. During an eight weeks period ending in early February 2015, we crawled this booter's website every 10 minutes to collect data on users of the service and details of attacks launched by them. We found `vdos-s.com` to be unique in reporting a wealth of public information on their users and attack details. This data includes all users that logged into the service in the past 15 minutes and distinguishes paying subscribers from unpaid users. In addition, it displays a list of all currently running attacks that includes the type of the attack, the target of the attack, duration and the time remained for the attack to be finished. Users can optionally choose to remain anonymous and hide the target of attacks, but the default is for all information to be public. Less than 30% of login records scraped were anonymous and the target was hidden for 39% of all attacks seen.

While we cannot fully vet this self-reported data, we did verify that the data representing our actions were reported accurately. We also validated that all NTP attacks reported for a day were accurate by sending monlist requests in 10-minute intervals to a set of 12 NTP amplifiers known to be abused by `vdos-s.com` and recording the received responses. A total of 44 distinct NTP attacks for which the target was not hidden were reported by `vdos-s.com` for the same 24 hours time period and we were able to find matching records for all 44 targets in the monlist responses collected from the set of monitored NTP servers. This gives us some increased level of confidence that the details of reported attacks and users are accurate.

Asylum Stresser Back-end Database. `Asylum Stresser` was an established booter that was in operation for over two years before it was hacked and the database of users, payments and attack logs was publicly leaked. It ceased operation shortly after the compromise and has not resumed operation. This leaked database has been vetted by many members of the anti DDoS community that have located their own test accounts in the user registration data and is believed to be authentic.

Lizard Stresser Back-end Database. `Lizard Stresser` was launched in late December 2014 by individuals calling themselves the *Lizard Squad*. This same group was responsible

for DDoS attacks on Sony PlayStation and Microsoft Xbox networks on December 25, 2014. These attacks causing access issues for users of these gaming networks gained wide media coverage. Latter, the group announced that the attacks were meant to demonstrate the power of `Lizard Stresser`, a booter service they started to offer to users on a subscription basis. As the attack infrastructure used by this service was backed by hacked home Internet routers, we did not directly interact with this service. However, the front-end server hosting `lizardstesser.su` was hacked around mid January 2015 and the database of users, payments and attack logs was publicly leaked. We had registered with this booter service before the public leak of its database. We were able to locate our registered user account, payment transaction and a few attacks that we launched on our own server to verify that compromised residential routers were misused to build the attack infrastructure. For this database, since all payments were in Bitcoin and the wallet addresses are included we were able to validate that this part of the database is accurate.

booter.tw Back-end Database. Although `booter.tw` is not thought to be among the largest booter services, it attracted attention on March 2013 after being linked to a series of DDoS attacks targeting a popular blog on computer security and cybercrime [71] and the Ars Technica website [72]. For this dataset, we contacted three of the victims and confirmed that the data correlated with attacks that they experienced.

When possible, we have checked for internal consistency within these leaked databases. While we cannot rule out that some of the data has been fabricated, it would take a fair amount of resources to create this high fidelity of a forgery.

3.2 Subscribers

We find that 15% of `Asylum Stresser` users, 23% of `vdos-s.com` users and 35% of `Twbooter` users purchased a subscription. This rate is less than 2% for the users of `Lizard Stresser` [1]. This might be attributed to the fact that `Asylum Stresser`, `vdos-s.com` and `Twbooter` all

[1] Note that Lizard Stresser did not offer free trial accounts.

accepted PayPal payments at least sporadically while `Lizard Stresser` only accepted Bitcoin as the payment method. It is difficult to attribute why the conversion rate of registered users to subscribers is much less for `Lizard Stresser`, since other factors, such as the media coverage, might have also driven many users to sign up out of curiosity. The `Lizard Stresser's` leaked database contains a total of 225 user support tickets. Out of these, 42 are related to user requests for purchasing subscriptions using PayPal. As one potential attacker wrote, "I want to pay via paypal real bad I'm a huge fan of and want to buy this ASAP but I don't have Bitcoins."

Figure 3.1 shows the number of weekly new paid users for `vdos-s.com`. As expected, a large number of new paid users were seen on the first week of scraping this booter service. However from the second week, the number of weekly new paid users converges to around 240 users per week. `vdos-s.com` had a high rate of positive review on underground forums from customers satisfied with the effectiveness of ordered attacks. Also, our frequent scraping of this booter service confirms the high availability rate of the service. This shows that there is constant customer demand for booter services that manage to have high availability rate and deliver effective DDoS attacks.

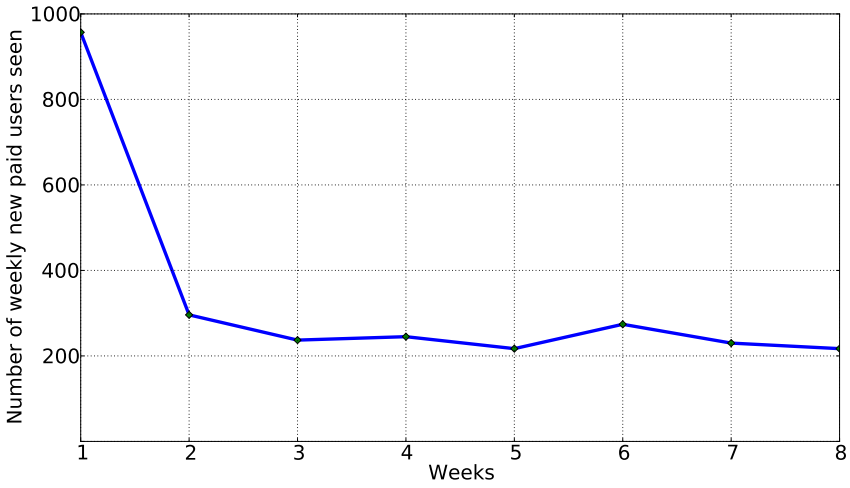

Figure 3.1: The number of weekly new paid users of `vdos-s.com`.

3.3 Attacks

From the leaked data we find that these four booters were responsible for close to 700,000 separate attacks against nearly 200,000 distinct victims. In general, the quantity of attacks launched by booter services tends to be much more than the numbers reported by previous research for botnet supported DDoS attacks. For instance, in [44], the researchers infiltrated *DirtJumper* a botnet supported DDoS for hire platform [73] and were able to observe only less than 2,000 DDoS attacks by monitoring a total of 35 identified Command and Control (C&C) servers for a total duration of four months. To contrast, Figure 3.2 shows the number of weekly attacks seen by scraping `vdos-s.com` for 8 weeks. On average, the users of this booter service launched more than 17,000 attacks per week. This difference in the quantity of attacks has to do with the fact that booter services have a different business model and customer base than botnet supported DDoS services. The target of DDoS attacks originated from *DirtJumper* are almost exclusively websites, while as we will discuss shortly, residential links are the primary target of attacks originated by booter services.

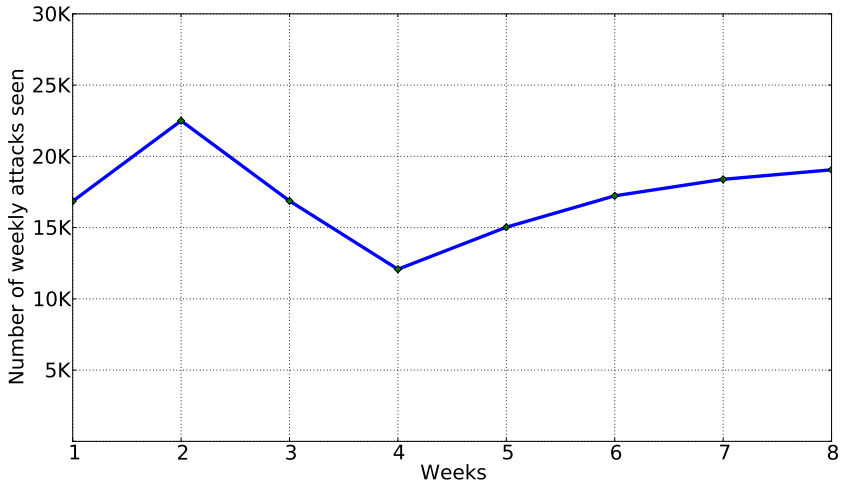

Figure 3.2: The number of weekly attacks launched by users of `vdos-s.com`.

Compared to DDoS attacks launched by botnet supported DDoS services, attacks launched

by booter services tend to have shorter durations. Many users of booter services purchase cheaper subscriptions that allow them to launch attacks that can last up to 10 minutes. Analyzing nearly 20,000 DDoS attacks launched by *DirtJumper*, researcher observed an attack duration of an hour or more for two-thirds of the attacks [44]. Figure 3.3 shows the CDF of attack durations of the four booter services for which we have the details of attacks launched. As evident from the Figure, only a very small percentage of all attacks launched by these booter services lasted for an hour or more. Ignoring `vdos-s.com`, the majority of attacks launched by the other three booter services have lasted less than 15 minutes. A similar distribution of attack durations is expected for `vdos-s.com`, however as we crawled this booter service on 10 minutes intervals to collect the details of running attacks, we are likely to have missed many short-lived attacks starting and ending during a time period between two consecutive crawling.

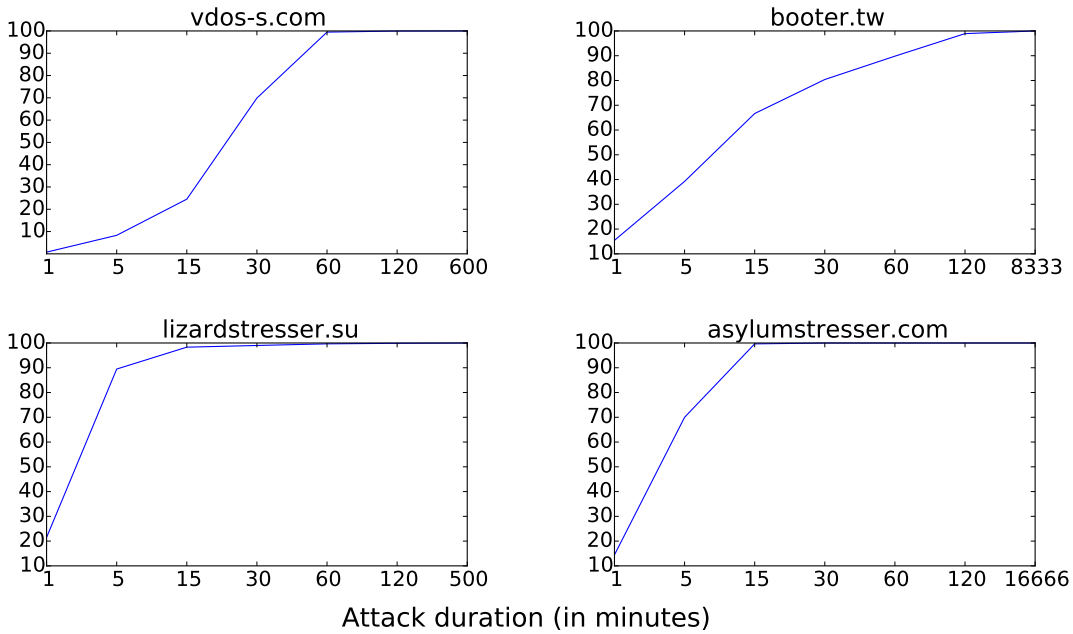

Figure 3.3: CDF of attack durations.

While the average attack duration for booter services is not very long, the data we

presented on DDoS attacks launched by booter services demonstrates the large-scale abuse problems and unwanted traffic generated by these services.

To gain a deeper understanding of the nature of users using booter services and the target of attacks that they launch, we present a more detailed analysis of attacks launched by users of `booter.tw`.

3.3.1 Analysis of Attacks Launched by Users of booter.tw

Online gamers constitute the primary group of customers served by `TwBooter`. However, as we will see there are smaller groups of customers using the service for purposes other than targeting online gamers. At the registration time, the users subscribe to a one month license for launching DDoS attacks. Depending on the amount paid, the subscribers can initiate attacks that can last for a limited maximum amount of time. There are several attack duration options available ranging from one minute to two hours. The users can also pay an additional fee to be able to initiate up to three concurrent attacks. There is no limit on the number of sequential attacks that a user can initiate during a month of subscription.

Like most booter services, `TwBooter` utilizes high bandwidth servers to mount DDoS attacks. Gamers typically use residential Internet connections to play online games. Considering the limited capacity of gamers' links, they can be easily overwhelmed with large amounts of traffic originated from one or more servers for a short period of time. For this reason, the majority of `TwBooter` users comprised of gamers have subscribed for short-lived DDoS attacks. About 65% of users have chosen attack durations of 10 minutes or less and 32% have selected attack durations of more 10 minutes, up to two hours.

Intuitively, the users subscribed for an attack duration of 10 minutes or less are likely gamers and those subscribed for an attack duration of an hour or more (15% of users) are likely users targeting websites. Interestingly, there are a few users who have the privilege to initiate attacks lasting for more than two hours. [2]

In terms of attack concurrency, 74% of users subscribed for only one attack at a time.

[2]Note that this option is not available to ordinary users at registration time.

Again by intuition, most of the users in this group should be gamers since they do not require multiple simultaneous attack sessions to reach their goals. Only 9% of the users have chosen the option of initiating two concurrent attacks and 15% of users with the need for higher capacities have subscribed for three concurrent attacks. Again, there are a few privileged users that are allowed to initiate more than three concurrent attacks.

Figure 3.4 shows how a small percentage of users are responsible for most of the attacks both in terms of number and duration. The top 2% of users (6 users) in terms of attack duration are responsible for about half of the whole attack time in 52 days (28,154 hours). Unsurprisingly, all of the users in the top 2% group are either privileged users or ordinary users subscribed for concurrent attacks of at least one hour. The users of this group have been active for an average of 33.5 days and various websites are their primary attack target. In term of attack count, the top 5% of users (14) are responsible for about 40% of all attacks. The users in this group are a mix of gamers and the website attackers. Ten users of this group have subscribed for an attack duration of half an hour or less and the rest have subscribed for durations of more than an hour. Only three of the users in this group overlap with the members of the top 2% group.

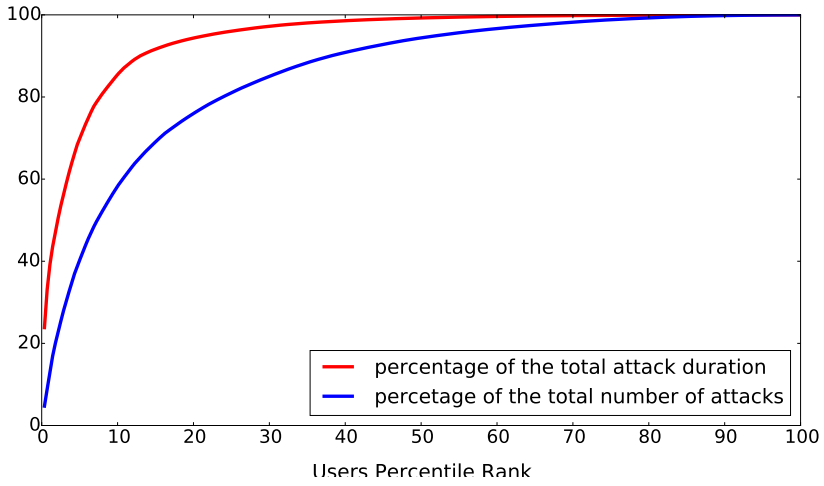

Figure 3.4: Distribution of users' contribution to the launched attacks.

The leaked database contains a table recording IP address and user-agent of the browsers used by users to login to the `booter.tw` website. A brief analysis of this table reveals that a considerable portion of users were concerned with keeping their identities unknown. Anonymizing services such as proxies, VPN services or the Tor network are the most prevalent means used for this purpose. Almost half of the users (137) have initiated at least 50 attack instances. Among those users, 60% (82) have logged into the service with at least 10 different IP addresses. The average number of distinct login IP addresses for this group of users is 34.

In the rest of this section we focus on discussing usage patterns for each of the three distinct groups of users identified: gamers mounting short-lived attacks of no longer than 10 minutes, website attackers with attacks lasting between one and two hours and the privileged users with the right to initiate attacks lasting for more than two hours. Users which could not be easily categorized into any of these groups were excluded from the analysis. The users assigned to one of the three groups account for about 83% of all users.

Table 3.2 summarizes service usage for the three groups of users. As observed, gamers and website attackers exhibit similar behavior in terms of the average number of attacks initiated per day and the number of distinct victims targeted per day. Users in the third group however behave differently. While privileged users tend to target fewer number of distinct victims per day, they initiate more attack instances on those targets. This is probably attributable to the fact that the privileged users are more likely to utilize concurrent attacks.

In terms of the average number of attacks initiated per day, we observe that users in all of the three groups use the service fairly heavily. As expected, the average attack time varies significantly among each of the user groups. While the maximum duration of an attack for gamers and website attackers is limited to 10 minutes and 2 hours respectively, we have attack records for privileged users that last for a few days. Besides the privilege of mounting longer lasting attacks, higher attack concurrency could be another factor contributing to

Table 3.2: Service usage of the three user groups.

	Gamers	Website	Privileged
Number of users	180	41	8
Avg distinct targets per day	3.32	3.46	2.86
Avg attacks per day	13	13	16
Avg attack time per day	59 m	14 h	105 h

the very large average attack time for the group of privileged users.

Victims

For each attack record in the database, the target is specified as either an IP address or a website URL. We identified 689 unique websites and 10,485 unique IP addresses in the attack records.

It is possible for a service subscriber to supply an IP address rather than a website URL when initiating an attack on a website. Consequently, the actual number of websites targeted by `TwBooter` could be higher than the above-mentioned number. To investigate if any of the targeted IP addresses were running web servers, we tried to establish an HTTP(S) connection with each one of them. About 1,200 of IP addresses successfully finished the connection attempt and returned an HTTP status code of 200 (OK) in response to a HEAD request. However, a manual inspection showed that many of these IP addresses resolve to default pages returned by hosting service providers or running web servers serving no content. The mapping of the IP addresses to valid websites was rare and we noticed that most of such websites were already included in the list of 689 targeted websites. Based on our observations, the number of unique targeted websites is not expected to be significantly higher than the number of targeted website we identified initially.

To understand what types of websites were victims of DDoS attacks initiated by `TwBooter's` subscribers, we manually visited the top 100 websites in terms of the overall time being under attack. While the type of targeted websites is quite diverse, ranging from other booters

to governmental agencies, the overwhelming majority of targeted websites were either game servers or game forums.

An observation of interest were two users ordering attacks targeting several different governmental websites. The primary focus was on two Indian websites and the website of Los Angeles police department. Collectively, the three websites were under attack for a total duration of 142 hours by these two users. The two users have subscribed to the service at the same date and have targeted a very similar set of destinations. This observation suggests that booter services are serving various customers with very different intentions.

3.4 Revenue

`Asylum Stresser` earned an average of $2,079 per month. However, as Figure 3.5 shows their revenue started at a modest $500/month and grew to over $3,000 per month towards the end of the leaked data period. Also, it is interesting to note that their revenue from subscription renewals was $16,025.12 and almost equal to the $19,356.42 earned from new subscriptions. The `Lizard Stresser` leak only covers 2 weeks during which they earned $3,368 and `vdos-s.com` earned an estimated $24,737 per month confirming that `vdos-s.com` was operating at a far larger scale. `Asylum Stresser` collected 99.4% ($35,180.14) of its revenue through PayPal payments and only 0.6% ($201.40) of their revenue was collected using their secondary payment method of MoneyBookers. `Lizard Stresser` collected all their revenue through their only supported payment method of Bitcoin and `vdos-s.com` accepted both PayPal and Bitcoin. All of this underscores the fact that revenue from paid subscriptions and renewals is the driving factor for operating these services and expanding them to grow customer bases. They are presumably profitable, but these individual booters do not generate the profits required to pay the upfront capital, fees and potential fines for dedicated credit card merchant processing accounts, which amounted to around $25K-$50K per an account, as was the case with illicit pharmaceutical and fake anti-virus groups that had revenues on the order of millions of USD dollars a month [61, 65].

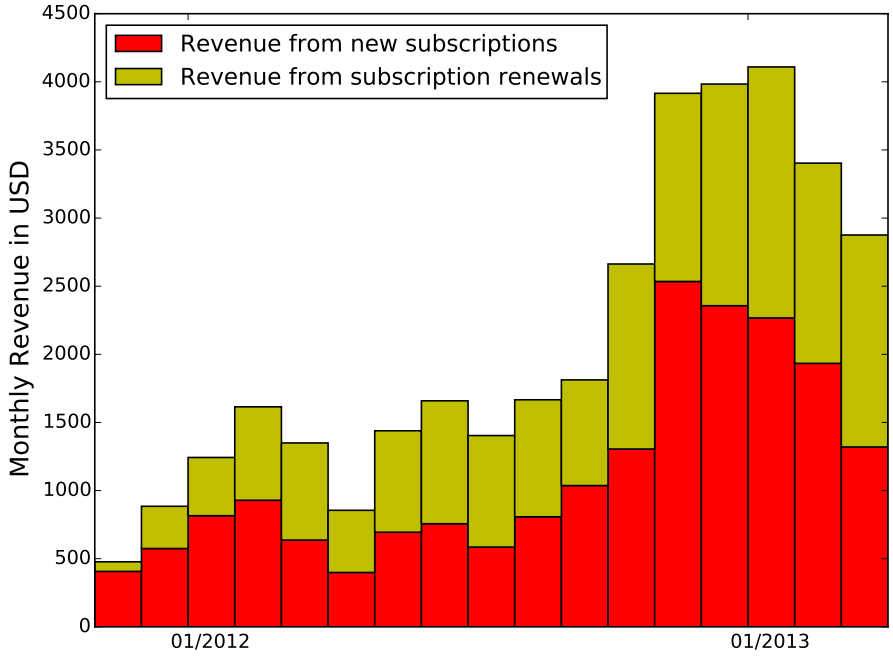

Figure 3.5: `Asylum Stresser` monthly revenue.

3.5 Geography

In order to understand where operators and subscribers are potentially located, we use aggregated data provided to us by PayPal that was computed from all the accounts identified by PayPal as belonging to booter operators and subscribers. This data did not include any scale on the number of booter and subscriber accounts included in the dataset. It was computed by assigning the location for each account to the country from which the majority of their logins occurred and computing the percentage of accounts assigned to each country. In the case that an account did not have a majority of their logins occurring from a single country, it was removed from the dataset. This accounted for 3% of subscriber accounts and none of operator accounts. Also, IP addresses for proxies, VPN services, hosting services and Tor were removed using a database from IP2Location [74].

Based on this information, 44% of the accounts used to accept payments for booters are potentially owned by individuals in the United States as shown in Table 3.3. The subscriber

locations are similar with nearly half of the accounts owned by someone potentially located in the Unitied States and many of the remaining account owners geolocated to Western Europe based on their IP as shown in Table 3.3. There are many inherent limitations of this data which we can not correct or quantify due to the highly aggregated nature of the data. These include the fact that booter services create multiple accounts to replace the ones that are limited by PayPal. Thus it might be the case that a few booter services control a large number of the total accounts and are biasing the location and the same could hold for customers as well. However, these locations match with much of the anecdotal information of this ecosystem including their preference for advertising in English language based underground forums and customers' preference for using PayPal as a payment method over more traditional virtual currencies, such as Webmoney which is more commonly used by Russian cybercriminals [75]. Assuming that our conclusions are accurate, there is the potential for a meaningful undermining of the booter ecosystem by increased law enforcement resources focused on this problem.

Table 3.3: Top country geolocations of booter operators and subscribers based on IP addresses used for logging into their PayPal accounts.

Operator		Subscriber	
CC	**%**	**CC**	**%**
US	44.06%	US	47.58%
PK	15.03%	DE	10.45%
CA	13.99%	GB	5.60%
GB	6.29%	NL	4.87%
AU	3.15%	RU	4.81%

Chapter 4: Attack Techniques and Infrastructure

We begin our analysis of booters with a study of their attack techniques and the infrastructure they rely on to generate DDoS attacks. These measurements are based on direct interaction with the booters and support services to understand what techniques and hosts are being actively misused for supporting attacks. Using this information we provide a better understanding of cost structure and trade-offs of different attack techniques. It also informs defenders as to which ISPs and hosts to focus on for blacklisting, remediation and notification efforts. Our analysis of front-end servers finds a reliance on CloudFlare to protect this infrastructure from takedown and DDoS. In addition, we find that booters gravitate to using more stable infrastructure when possible. This differs from previous studies that scan the Internet for the vulnerable populations of misconfigured amplification servers many of which might be highly transient and not be used for DDoS attacks [39, 76]. We also identify two hosting providers that have been actively courting booter operators and providing stable high bandwidth attack servers that allow spoofing.

This study includes data gathered from a combination of sources including subscribing to booters and launching attacks against our servers, active probing measurements and analysis of hosting providers that rent attack servers.

4.1 Dataset Description

Our first task was to identify booter services for this part of our study. Absent a centralized location for finding booters, we found services via search engines and advertisements on hacker forums such as *hackedforums.net*. We selected 15 stable booter services for our attack infrastructure characterization. We make no claim about the coverage these booters provide of the entire ecosystem. Rather we were looking to provide a sample of stable

services located using common methods that garnered strong reputations on underground forums.

Once we identified our set of booters we created an account on each service to see which attack types were offered and purchased a one month subscription from each of the services which ranged from $2.50-18.99. We chose to measure the most common amplification reflection attack types offered by the booters, which were SSDP, NTP, DNS and CharGen. Table 4.1 shows the set of booters, which of the four attack types each booter offered and the cost of a basic monthly subscription. We protect our identities from the booter services by using multiple PayPal accounts, pre-paid credit cards and Bitcoin as payment methods.

Table 4.1: List of booter services we measured, the attack types offered, and the cost of the least expensive one-month subscription.

Booter	**Attack Types**	**Cost**
anonymous-stresser.net	DNS	$6.60
booter.io	NTP,CharGen	$2.50
crazyamp.me	DNS,SSDP	£10.99
grimbooter.com	NTP,SSDP	$5.00
hornystress.me	NTP,SSDP	$6.99
inboot.me	DNS,NTP,SSDP	$11.99
ipstresser.com	NTP,SSDP,CharGen	$5.00
k-stress.pw	SSDP,CharGen	$3.00
powerstresser.com	SSDP	$14.99
quantumbooter.net	DNS,SSDP	$10.00
restricted-stresser.info	DNS,NTP	$10.00
specialistsservers.tk	DNS,NTP,SSDP,CharGen	$12.00
stresstest.tv	DNS,SSDP	$3.00
vdos-s.com	DNS,NTP,SSDP	$18.99
xr8edstresser.com	DNS	$10.00

Once we verified that our subscriptions were activated, we conducted attacks directed at our target server from December 2014 - January 2015. The goal of these attacks was to measure the set of misconfigured hosts that were being abused by each booter to amplify their reflection attacks. The configuration of the target system used for measuring the attacks was an Intel Xeon 3.3GHz server with 32 GB of RAM and a 1 Gbps network connection running Ubuntu.

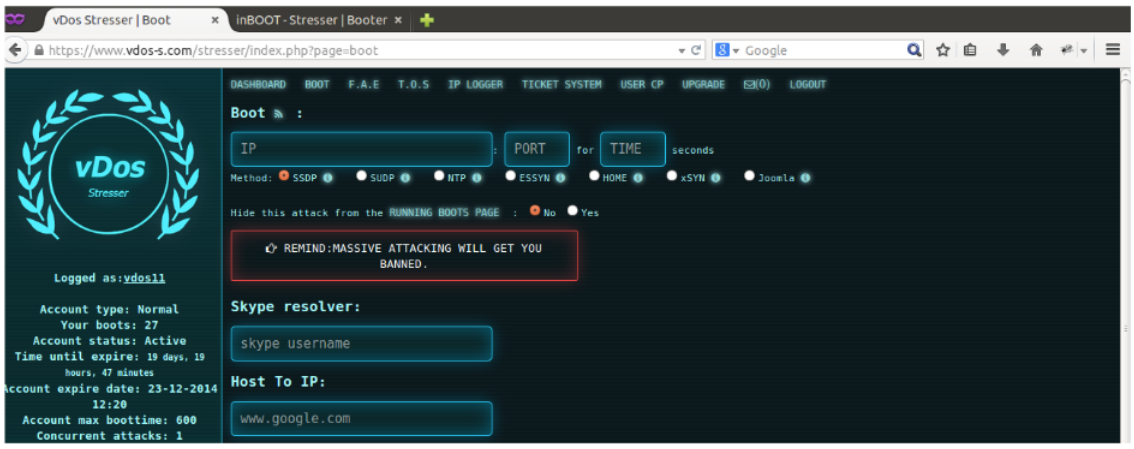

Figure 4.1: An example of a booter front-end.

We used gulp [77] which is a lossless Gigabit packet capture tool to capture the attack traffic. For each covered booter service/attack type, we collected an hour of attack traffic. This was comprised of shorter attack instances lasting for 10 minutes at most, which is the standard time limit for basic booter subscriptions. The reasoning behind the longer attack times was to increase our probability of identifying all the misconfigured reflection hosts used by a booter for each attack type. However, we found that almost all of the amplification servers of a specific type abused by a booter were identified in the first attack. Therefore we limited our attacks to shorter durations for the rest of our experiments that required launching self-attacks.

4.2 Frontend Servers

Booter services maintain a front-end website that allows customers to purchase subscriptions and launch DDoS attacks using simple forms or convenient drop-down menus to specify the attack type, attack duration and victim's IP address or domain name. Figure 4.1 shows an example booter front-end.

These front-end websites commonly come under DDoS attack by rival booters and are subject to abuse complaints from anti DDoS working groups. All 15 booters in our study

used CloudFlare's DDoS protection services to cloak the ISP hosting their front-end servers and to protect them from abuse complaints and DDoS attacks.

As part of this study, we contacted CloudFlare's abuse email on June 21st 2014 to notify them of the abusive nature of these services. As of the time of writing this document we have not received any response to our complaints and booter services continue to use CloudFlare. This supports the notion that at least for our set of booters, CloudFlare is a robust solution to protect their front-end servers. In addition, `crimeflare.com` has a list of over 100 booters that are using CloudFlare's services to protect their front-end servers.

In [78], a methodology to classify a website as a booter service is proposed. The goal is to automatically detect booter websites and generate a blacklist to prevent access to the detected websites. While an interesting idea, in practice, the construction and enforcement of such blacklist would be very difficult. Furthermore, it would be possible for operators of booter services to modify the features of their front-end websites in an attempt to evade being detected and blacklisted.

4.3 Attack Servers

Renting back-end servers to generate attack traffic directed to amplifiers or victims is the primary source of cost for the operators of booter services. We did some research to get a high level sense of the market availability and cost of back-end servers that allow source IP address to be spoofed. Being spoof friendly, fast uplink speed and unmetered bandwidth usage are the key requirements of a server appropriate for supporting the operation of a booter service.

Table 4.2: spoof friendly VPS services tested.

Provider	VPS IP	Uplink speed	Bandwidth	Monthly cost
cavpshost.com	192.210.234.203	3.5 Gbps	Unmetered	$35
sparkservers.eu	96.8.114.146	949 Mbps	10 TB	$60

To this end, we looked for services selling spoof friendly servers on underground forums and purchased VPS from two service providers. Table 4.2 summarizes the services that we purchased. Some service providers let customers to choose the location of the servers, in our case however, we were not given this option. Both of the purchased VPSs were hosted by the same ISP (ColoCrossing) in the US. We tested to see if spoofing was enabled, and the result was positive for both services. We ordered 1 Gbps links for both services and tested them to measure their actual link speeds, one VPS provided around 1 Gbps uplink bandwidth and the other one interestingly provided up to 3.5 Gbps. The servers we purchased were virtual and therefore running on shared hardware. A busy booter service would need to use dedicated servers with more resources to support its operation. The price range for most of dedicated servers with a link speed of 1 Gbps and unlimited bandwidth usage was around $300-$500. Due to budget and time limitations we did not purchase any of these higher end services.

4.4 Attack Techniques

Due to their effectiveness, amplified volume-based attacks are the default attack technique offered by most booter services. We focused our analysis on SSDP (more commonly known as Universal Plug and Play (UPnP)), DNS, NTP and CharGen, which were the most popular attack types offered by the set of booters we selected for measurements. These attacks depend on servers running misconfigured UPnP, DNS resolvers, NTP and CharGen services that enable attackers to amplify attack traffic by sending spoofed packets with the victim's source address in the IP header and having these services respond with a larger amount of traffic directed to the victim.

We have also seen booter services offering reflection-based attacks by misusing popular web Content Management Systems (CMS) such as WordPress and Joomla to generate and direct HTTP requests to target web servers.

In addition, many booters offer direct attacks including TCP SYN [79], and UPD flood where the attacker spoofs the source IP address and directly sends packets to the victim.

Some booters also implement HTTP-based flooding attacks including HTTP POST/GET/HEAD, RUDY (R-U-Dead-Yet) [80] and Slowloris [81].

In the subsequent sections, we characterize the vulnerable hosts misused by booter services to launch amplification attacks based on DNS, NTP, SSDP and CharGen.

4.5 Amplifiers

As part of our measurements we can map out the set of amplifiers that are being abused to magnify the traffic volume of attacks. This sheds light on the population of hosts that are not only vulnerable to amplification attacks, but are actively being abused for launching DDoS attacks.

Table 4.3: Number of total amplification servers and percentage of overlap with amplification servers used by other booters.

	CharGen		**DNS**		**NTP**		**SSDP**	
Booter	(#)	(%)	(#)	(%)	(#)	(%)	(#)	(%)
anonymous-stresser.net	-	-	1,827	73%	-	-	-	-
booter.io	370	65%	-	-	1,764	86%	-	-
crazyamp.me	-	-	43,864	56%	-	-	64,874	46%
grimbooter.com	-	-	-	-	1,701	72%	10,121	60%
hornystress.me	-	-	-	-	8,551	58%	242,397	30%
inboot.me	-	-	38,872	55%	4,538	92%	170,764	54%
ipstresser.com	1,636	44%	-	-	1,669	85%	90,100	29%
k-stress.pw	1,422	30%	-	-	-	-	5,982	76%
powerstresser.com	-	-	-	-	-	-	1,424,099	11%
quantumbooter.net	-	-	10,105	85%	-	-	39,804	67%
restricted-stresser.info	-	-	2,260	82%	27	100%	-	-
specialistsservers.tk	2,358	38%	26,851	61%	6,309	35%	258,648	24%
stresstest.tv	-	-	93,362	53%	-	-	7,126	74%
vdos-s.com	-	-	16,133	82%	6,325	82%	150,756	62%
xr8edstresser.com	-	-	44,976	52%	-	-	-	-
Total	4,565	23.46%	181,298	35.30%	17,599	42.31%	2,145,015	11.84%

Table 4.3 shows that the set of abused CharGen and NTP servers are smaller and more highly shared between two or more services, whereas there is an ample supply of vulnerable DNS and SSDP servers abused as amplifiers. However, the overlap of DNS servers used by two or more booter services is still relatively high suggesting that these DNS resolvers

might be more stable, have higher bandwidth connections and be in more limited supply.

4.6 Amplifiers Geolocation

As demonstrated by Table 4.4, both the geolocation and Autonomous System (AS) of amplifiers abused by booters are fairly diffuse.

Table 4.4: Top country locations and autonomous systems for amplifiers.

CC	%	AS	%
CharGen			
CN	48.78%	4134 (Chinanet)	14.46%
US	12.51%	37963 (Hangzhou Alibaba Advertising)	10.47%
KR	5.50%	4837 (CNCGROUP China169 Backbone)	6.88%
RU	4.58%	17964 (Beijing Dian-Xin-Tong Network)	2.61%
IN	2.56%	7922 (Comcast Cable Communications)	2.61%
DNS			
US	12.38%	4134 (Chinanet)	2.68%
RU	11.58%	3462 (Data Communication Business Group)	2.15%
BR	9.19%	18881 (Global Village Telecom)	1.46%
CN	6.84%	4837 (CNCGROUP China169 Backbone)	1.45%
JP	3.61%	7922 (Comcast Cable Communications)	1.27%
NTP			
US	31.47%	3462 (Data Communication Business Group)	14.01%
TW	15.29%	46690 (Southern New England Telephone)	12.35%
CN	10.68%	7018 (AT&T Services)	4.84%
KR	5.50%	4134 (Chinanet)	3.58%
RU	4.74%	4837 (CNCGROUP China169 Backbone)	2.18%
SSDP			
CN	36.26%	4837 (CNCGROUP China169 Backbone)	18.98%
US	19.37%	4134 (Chinanet)	11.16%
EG	6.83%	8452 (TE Data)	6.61%
AR	5.37%	22927 (Telefonica de Argentina)	5.13%
CA	5.36%	7922 (Comcast Cable Communications)	4.60%

There are a few notable exceptions, such as the concentration of CharGen amplifiers in China with three Chinese ASs connecting 34% of these amplifiers. In addition, there is a slight concentration of abused NTP servers connected to one Taiwanese AS and two United States network operators. This might indicate a potential to focus patching efforts on these networks, given the limited pool of hosts used for CharGen and NTP attacks from

Table 4.3. Feeds of these actively abused servers could also be distributed to these network operators and to DDoS mitigation services.

4.7 Amplifiers Churn

In order to measure the stability of the identified amplifiers, we probed them on a daily basis for 13 weeks to understand how many were still located at the same IP and vulnerable to abuse.

Protocol-specific request packets were used for probing the set of amplifiers of different types. For NTP, we consider a server responding to a probe attempt to be vulnerable only if it replies with an NTP monlist payload. For DNS, we send an A request query and verify that the received DNS response indicates that the name server has not refused the request an it is willing to recursively resolve DNS queries.

All of the four types of amplifiers that we probe are UDP based and therefore packet drops can result in underestimating the number of amplifiers that are accessible and vulnerable. To minimize the effect of dropped packets, we retry the probing up to 5 times for amplifiers for which a response has not been received in previous attempts. The probing requests sent to an amplifier on a day are distributed throughout the day.

As shown in Figure 4.2, the set of DNS resolvers were the most stable with nearly 80% still misconfiged and located at the same IP after one month, and over 60% were still accessible for abuse after 13 weeks. This result is counter to previous results of churn rate measurement based on Internet wide scanning that found a 50-60% churn rate for open DNS resolvers after one week [39]. It indicates that booters have gravitated to using a more stable set of open DNS resolvers and that focusing mitigation efforts on these might cause these DNS attacks to be less efficient and require additional bandwidth and cost. From our measurements, SSDP servers were the least stable with a 46% churn rate after only a single week. This result agrees with the previous Internet wide scanning result and indicates that either booters have not found or there might not exist a set of more stable SSDP servers.

For the first five weeks of the time period that the set of identified amplifiers were

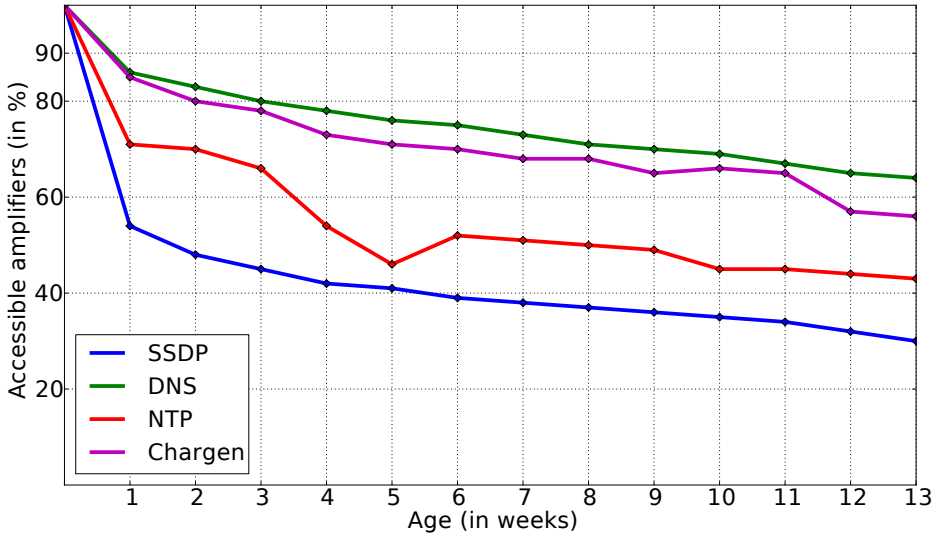

Figure 4.2: Churn rate of amplifiers over time.

probed, we also collected weekly one-minute attack traces for all of the 15 booter services and the attack types of interest offered by each of the booter services. Table 4.5 shows the summary of weekly attack traces that were collected for each of the four attack types. Note that for reasons such a booter service being unable to deliver a requested attack type for several days, we were not able to collect all of the possible attack traces during the 5 weeks time period. However, as shown in the second column of the table 4.5, we were able to collect at least 70% of attack traces for all of the four attack types. The last column of the table shows the number of distinct amplifiers of different types that were seen in the weekly collected attack traces. The relative number of amplifiers seen in weekly attack traces is in line with the number of initially identified amplifiers presented in table 4.3.

To understand if the set of amplifiers that were identified and we learned to be fairly stable over time based on the probing results, were actually misused by booter services over the next 5 weeks for launching DDoS attacks, we computed the overlap between the set of initially identified amplifiers abused by the 15 booter services and the amplifiers seen

Table 4.5: Summary of weekly collected attack traces. The numbers within () show the percentage of possible weekly attack traces that we were able to collect.

Attack type	**# of collected traces**	**# of amplifiers seen**
CharGen	14 (70%)	6,923
DNS	32 (71%)	224,521
NTP	35 (87%)	17,506
SSDP	41 (75%)	3,245675

in weekly collected attack traces. For each attack type, one data point is calculated for each of the 5 weeks for which attack traces were collected. To calculate the overlap for a specific attack type and a specific week, we check all the attack traces of the same attack type collected in that specific week to get a list of all amplifiers seen in the collected attack traces. We then compute the overlap of these amplifiers with the set of initially identified amplifiers of the same type.

Figure 4.3 shows the overlap of amplifiers seen in weekly collected attack traces and the set of initially identified amplifiers. This confirms that the set of initially identified amplifiers that are still accessible are actively abused by booter services to deliver attacks.

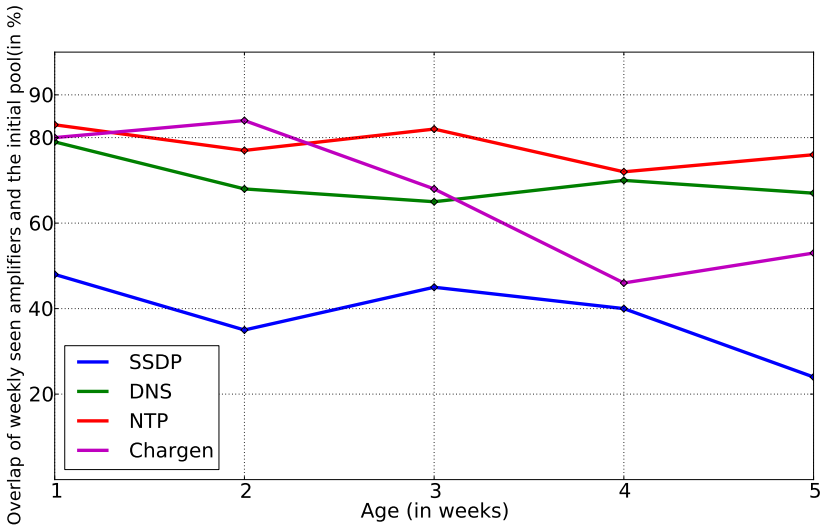

Figure 4.3: Overlap of amplifiers seen in weekly attack traces and the initial set of amplifiers.

A large number of SSDP amplifiers were seen in the weekly collected attack traces and compared to the other types of amplifiers, the SSDP amplifiers seen in the weekly attack traces have the smallest overlap with the set of initially identified amplifiers. This suggests that for attack types such as SSDP for which there is a higher rate of amplifiers churn, booter services have to scan for new vulnerable hosts and update their lists more frequently. However, as far as enough amplifiers are in existence, scanning for amplifiers using very fast scanning tools such as Zmap [82] or Masscan [83] doesn't appear to be a bottleneck for the operators of booter services.

4.8 Bandwidth Amplification Factor

One of the few direct costs incurred for every attack a booter service launches is the bandwidth consumed by their rented attack servers. In order to reduce this cost, amplification attacks are used for volume-based flooding attacks. Some attack methods can potentially produce a larger amplification factor than others, but there are other factors that effect the amplification factor. Based on our measurements we can better understand how effective each attack type is and what effects the amplification factor.

4.8.1 Computing Bandwidth Amplification Factor

For each booter service and attack type, we use the first of the one-minute weekly collected attack traces to measure the amplification factor.

To calculate the Bandwidth Amplification Factor (BAF), we would need to compute the number of requests sent by the attack servers of a booter service to generate the received attack traffic and the size of each request packet. The size of a request packet depends on the underlying protocol misused for launching an attack. All the attack techniques for which we computed the BAF are based on UDP and each UDP packet includes UDP, IPv4 and Ethernet headers for a total size of 42 bytes (8 + 20 + 14). The size of each request packet would be the sum of these 42 bytes and any extra UDP payload required by the higher-layer protocol.

DNS: The DNS request packet format is composed of a fixed-length header of 12 bytes and a variable length section for the domain name to be queried from a name server. We compute the exact length for the variable length fields based on the actual domain name requested as seen in captured attack traffic.

NTP: The NTP monlist feature works on mode 7 (MODE_PRIVATE), a mode defined for data exchange between a client and an NTP server for purposes other than the usual time synchronization. It takes 8 bytes of data to construct an NTP header for a monlist request and thus a total of 50 bytes for an NTP monlist request packet.

SSDP: For amplification attacks based on SSDP, the attacker needs to send an M-SEARCH request meant to be used for service discovery to an UPnP-enabled device. SSDP is a text-based HTTP-like protocol and the payload of the M-SEARCH request is normally 90 bytes. In our calculations, we considered the total size of an SSDP request to be 133 bytes.

CharGen: This is an old protocol originally designed for network troubleshooting and measurement purposes. Upon receiving a UDP datagram on port 19, a CharGen server will reply with a random number of characters depending on the implementation. The request packet is not required to carry any payload and if it does, it will be discarded. So, the length of a CharGen request packet would be 42 bytes, the length of a UDP packet with no payload.

For DNS and CharGen, the number of requests and responses are one to one but for NTP and SSDP one request can result in several response packets.

Given the number of request packets and the size of each request, BAF can be computed using the following equation:

$$BAF = \frac{\text{volume of received traffic}}{\text{number of requests} \times \text{size of each request}}$$

4.8.2 Bandwidth Amplification Factor Measurements

As Table 4.6 shows, NTP and CharGen are the two attack types commonly used by booters that generate the largest amplification factors. However, they are also the two least supported attack types by the 15 booter services we measured. It is also interesting to note the variations in BAF between booters that offer the same attack type. In the case of DNS, the BAF is effected by the domain names being resolved along with the pool of abused open DNS resolvers. For NTP, the amplification factor is effected by the number of hosts returned by the monlist request, which is normally capped at 600 hosts. Based on the average BAF of 603, most of these amplifiers are unfortunately returning close to the maximum number of hosts. However, a clever attacker could try to maximize the BAF of abused NTP servers by adding entries to their NTP monlist responses by making bogus spoofed queries to these servers. SSDP offers the lowest BAF making it the most expensive attack type in terms of attacker bandwidth, but it also has the largest pool of misconfigured hosts to perform this type of attack.

Table 4.6: Bandwidth amplification factor across attack types and booters.

	Amplification Factor			
Booter	**CharGen**	**DNS**	**NTP**	**SSDP**
anonymous-stresser.net	-	42	-	-
booter.io	75	-	585	-
crazyamp.me	-	26	-	29
grimbooter.com	-	-	583	29
hornystress.me	-	-	594	26
inboot.me	-	25	654	24
ipstresser.com	61	-	764	22
k-stress.pw	71	-	-	32
powerstresser.com	-	-	-	21
quantumbooter.net	-	41	-	30
restricted-stresser.info	-	40	638	-
specialistsservers.tk	44	25	379	22
stresstest.tv	-	22	-	33
vdos-s.com	-	32	631	23
xr8edstresser.com	-	19	-	-
Average	63	30	603	26

Given this analysis, it seems that the most potentially effective course of action to raise the attacker's bandwidth costs is to continue to focus patching efforts on NTP and CharGen servers. This would have the effect of driving up attacker's bandwidth costs and forcing booters to either provide less effective attacks or increase the cost of their subscriptions.

As pointed to earlier, booter services are not usually associated with very large-volume attacks similar to the few example cases mentioned in section 1.1. However, based on the number of amplifiers accessible to booter services and the very high amplification rates achievable in the case of NTP-based attacks, booter services are capable of generating very large DDoS attacks. In the case of the 400 Gbps DDoS attack reported by CloudFlare [3], a total of 4,529 NTP servers were abused to launch the attack.

Based on our observations, in theory a booter service with access to a 1 Gbps link, should be able to generate around 600 Gbps of DDoS traffic by abusing vulnerable NTP servers for amplification. By increasing the bandwidth available for sending request packets, it would be also possible to generate large attacks using amplifiers with smaller amplification rates than NTP. However, the business model adopted by most booter services discourage them from delivering such large DDoS attacks to their individual subscribers.

4.8.3 Domains Resolved

For the DNS-based DDoS attacks, open DNS resolvers are abused as attack amplifiers. The maximum size of a DNS response was limited to 512 bytes in the original DNS protocol design. For larger responses, a truncated response was returned and a bit was set in the protocol header to notify the client of a truncated response. The client then could initiate a TCP session to the DNS server to receive the full response. To avoid the overhead of TCP, extension mechanisms for DNS (EDNS) allowing UDP responses of up to 4096 bytes were proposed and latter widely deployed. This allows attackers to send the ANY DNS requests to DNS servers supporting large responses to achieve a high amplification factor.

In our analysis of attack traffic received from open DNS resolvers, we observed that booter services tend to send spoofed ANY requests for a single or a very few domain

names that result in large responses to be returned to victims. Table 4.7 summarizes the domain names resolved by each booter service and the type of response records returned for each domain name. Some of the observed domains like `energystar.gov` are legitimate domains with the authoritative name server implementing DNSSEC and configured with large RRSIG and DNSKEY records that results in a BAF of 41. For some other observed domains like `fkfkfkfz.guru`, it seems that the authoritative name server is maliciously configured to respond with a large number of bogus records. For instance, a query for `fkfkfkfz.guru` resulted in more than 200 A records in the received response and a BAF over 40 for both booters that resolve this name. In the case of these malicious domain names, it is possible to either blacklist these domains or focus efforts on taking them down. In the case of legitimate domain names that return large records, it is difficult to mitigate the threat, but these records could be scrutinized more carefully by upstream networks and dropped when the link becomes congested.

Table 4.7: Domains resolved by booter services for DNS amplification.

Booter	**Resolved domain(s)**	**RR types**
anonymous-stresser.net	fkfkfkfz.guru	A(97%)
crazyamp.me	ohhr.ru	A(97%)
inboot.me	ifortuna.cz	NS(29%), RRSIG(20%), A(15%), DNSKEY(5%)
quantumbooter.net	energystar.gov	DNSKEY(35%), RRSIG(30%)
restricted-stresser.info	fkfkfkfz.guru	A(97%)
specialistsservers.tk	pidarastik.ru	NS(24%), TXT(18%), A(9%)
stresstest.tv	doleta.gov	A(97%)
vdos-s.com	defcon.org(47%), rula.net(43%), uspsoig.gov(10%)	NS(31%), RRSIG(27%), A(13%), DNSKEY(8%)
xr8edstresser.com	gransy.com(85%) defcon.org(15%)	NS(41%) A(25%), RRSIG(9%)

4.9 Attack Power Measurement

To measure the volume of attack traffic generated by attacks initiated through booter services, we used a number of different booter services to direct attacks to one of our own servers. We limited the measurements to the four amplification based attacks discussed earlier in this chapter. All the measurements are for attack instances each lasting for 30 seconds. The target of attacks was a rented VPS running Ubuntu and connected to a 10 Gbps shared link. To measure the actual bandwidth available to our VPS, we used the iperf3 tool [84] to measure the amount of UDP traffic that the VPS was able to transmit to a public iperf server[1] connected to a 40 Gbps link. According to the results reported by iperf3, the VPS was able to transmit around 4.5 Gbps of UDP traffic over the shared link. Unfortunately, we didn't have access to a fast link to use as an iperf client to measure the capacity of the VPS's shared link for receiving incoming traffic. The DSTAT tool [85] was used on the attack target to record the volume of attack traffic and the number of packets received per second.

Table 4.8 summarizes the attack instances that were measured. The numbers reported for attack volumes and the number of received packets are the average over the attack duration and the numbers are rounded for brevity. For booter services reporting the number of currently running attack instances, this data is reported in the third column.

The fact that various attack types such as CharGen and NTP with very different amplification factors are generating similar attack traffic volumes suggests that booter services adjust the rate of malicious requests sent to amplifiers of different type to control the attack volume sent to victims. We have seen a few booter services offering premium subscription plans claimed to deliver 5-6 Gbps of attack traffic. Access to a fast dedicated link is required for measuring higher volume DDoS attacks offered by these booter services.

[1] iperf.scottlinux.com

Table 4.8: Summary of attack power measurements.

Booter Service	Attack Type	Running Attacks	Power	PPS
crazyamp.me	SSDP	6	825 Mbps	337 kpps
crazyamp.me	DNS	-	94 Mbps	52 kpps
hornystress.me	SSDP	15	851 Mbps	348 kpps
hornystress.me	NTP	15	1.53 Gbps	441 kpps
inboot.me	SSDP	27	886 Mbps	360 kpps
inboot.me	NTP	34	139 Mbps	39 kpps
inboot.me	DNS	41	790 Mbps	80 kpps
ipstresser.com	SSDP	-	858 Mbps	343 kpps
ipstresser.com	NTP	-	1.46 Gbps	420 kpps
ipstresser.com	DNS	-	324 Mbps	36 kpps
ipstresser.com	CharGen	-	1.82 Gbps	217 kpps
qantumbooter.net	SSDP	12	938 Mbps	381 kpps
restricted-stresser.info	SSDP	-	617 Mbps	247 kpps
restricted-stresser.info	NTP	-	1.47 Gbps	424 kpps
restricted-stresser.info	DNS	-	922 Mbps	98 kpps
specialistsservers.tk	NTP	-	1.36 Gbps	405 kpps
specialistsservers.tk	DNS	-	838 Mbps	91 kpps
specialistsservers.tk	CharGen	-	1.56 Gbps	199 kpps
vdos-s.com	SSDP	11	844 Mbps	345 kpps
vdos-s.com	NTP	14	59 Mbps	124 kpps
vdos-s.com	DNS	13	927 Mbps	118 kpps

Chapter 5: PayPal Intervention

As part of our study we sought out opportunities to understand and also measure the effectiveness of intervention efforts to undermine DDoS Services. In this chapter, we present our measurements of a payment intervention that was conducted in collaboration with PayPal.

We find that reporting booter payment accounts to responsive payment service providers, such as PayPal, can have the desired effect of limiting their ability and increasing the risk of accepting payments using these payment services. This technique requires constant monitoring of the booters and drives booter services to move to more robust payment methods, such as Bitcoin.

5.1 Booter Payment Ecosystem

At the onset of our study the majority of booter services accepted credit card payments via PayPal as their primary mechanism for receiving funds from their customers. In addition to PayPal, some accepted Bitcoin payments as a secondary payment method largely using third party payment services, such as Coinbase or BitPay, that handles collecting Bitcoin payments on behalf of the merchant. A limited number of booters also accepted credit card payments using Google Wallet[1] or Skrill and virtual currencies, such as WebMoney and Perfect Money.

We identified a total of 60 booter services that initially accepted PayPal and created custom crawlers to monitor their payment methods and merchant accounts for about 6 weeks from April 22, 2014 through June 07, 2014. These booters were located from underground forum advertisements and web searches for terms commonly associated with booter services.

[1]Google has announced that their digital goods payment processing will be phased out in March 2015 `https://support.google.com/wallet/business/answer/6107573`

To minimize the effect of unstable booters on our study, the final set of booters included in our analysis was limited to 23 stable booter services that were able to successfully use PayPal to receive funds for at least half of the time before the PayPal intervention and used at least one PayPal account after the intervention.

After collecting our initial data on the stability of their PayPal merchant accounts, we reported these booters' domains directly to PayPal and they began to monitor merchant accounts linked to these domains and suspending them after an investigation. Note that PayPal will initially limit reported merchant accounts that are found to violate their terms of service by accepting payments for abusive services and perform an investigation of the account. Once an account is limited, the merchants cannot withdraw or spend any of the funds in their account. This will result in the loss of funds in these accounts at the time of freezing and potentially additional losses due to opportunity cost while establishing a new account. In addition, PayPal performed their own investigation to identify additional booter domains and limited accounts linked to these domains as well. This had the affect of a large-scale PayPal payment disruption for the majority of booter services.

In order to further understand the effectiveness of our payment intervention, we monitored underground forums where these booters advertise their services and news feeds from booters we joined to discover qualitative data on the effectiveness of PayPal's payment intervention.

5.2 Usage pattern of PayPal Accounts

Based on our observations, booter services for the most part only use a single PayPal account at a time to receive payments and change their PayPal merchant account when a limit is put on their previous account or they proactively change accounts to reduce the risk of limits on their previous accounts.To receive their payments using PayPal, booter services redirect customers to the PayPal website where existing PayPal users can login and complete a transaction. After logging into PayPal, our crawlers were able to collect the merchant account identifier of the corresponding booter service from the HTML source of

Table 5.1: Number of PayPal accounts used by monitored booters before and after the intervention. The numbers within the () are the average lifespan of the accounts used by that booter. Accounts that are active both before and after are counted only in the before and not included when computing the average lifespan. Matching symbols indicate that this set of booters shared at least one PayPal account. These shared accounts might be instances of a third party agreeing to accept payments for these services.

Booter	**accounts before**	**accounts after**	**Status**
anonymous-stresser.net ⋆	6 (8.2)	7 (2.9)	✓
aurastresser.com	6 (7.2)	6 (2.7)	✗
booter.io •	6 (8.3)	11 (2.7)	✓
critical-stresser.com †	4 (9.0)	1 (2.0)	✗
darkbooter.com ◇	4 (6.0)	5 (4.8)	✓
diamondstresser.org †	3 (15.7)	0 (-)	✗
getsmack.de	2 (14.0)	1 (4.0)	✓
grimbooter.com ⋆	4 (10.5)	1 (6.0)	✓
hazebooter.com ◇	4 (12.2)	5 (5.6)	✓
iddos.net ◇	3 (7.7)	2 (9.0)	✓
ipstresstest.com ◇	3 (7.3)	5 (5.4)	✓
powerstresser.com	5 (4.5)	9 (5.0)	✓
primebooter.com	6 (8.8)	2 (1.0)	✗
quantumbooter.net	11 (4.3)	22 (1.8)	✓
ragebooter.net •	13 (3.9)	4 (2.0)	✓
reboot.re	2 (11.5)	9 (2.3)	✗
restricted-stresser.info ⋆	6 (8.2)	7 (2.9)	✓
snowstresser.com	1 (-)	0 (-)	✗
stagestresser.com	5 (13.0)	3 (5.3)	✓
str3ssed.net	1 (47.0)	1 (4.0)	✓
titaniumstresser.net ◇	12 (5.3)	17 (2.9)	✓
xr8edstresser.com	4 (10.5)	11 (1.6)	✓
xrstresser.net •	8 (5.2)	4 (2.5)	✗
	119 (7.84)	133 (3.07)	

the page without completing a transaction. We used the dataset collected during the initial monitoring period to understand how frequently booter services were changing their PayPal accounts. Note that our age measures are both right and left-censored. For the booters' initial accounts our data is left-censored and for the last account our data is right-censored. However, we believe our age measurements accurately represent the effects of the PayPal intervention based on our interactions with and postings from the booters themselves.

Table 5.1 provides an overview of the PayPal accounts observed by our crawler broken down by each service monitored. As Table 5.1 shows, accounts had an average lifespan of about 8 days before the intervention with `str3ssed.net` and `snowstresser.com` each using

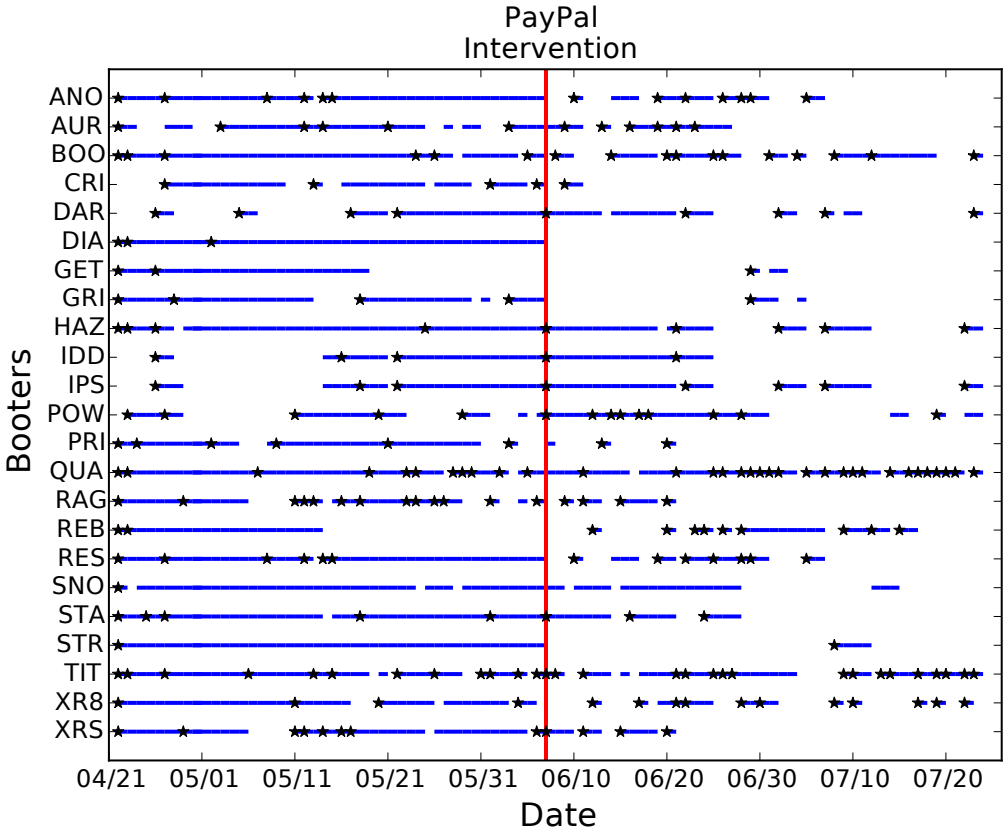

Figure 5.1: PayPal account usage over time. Booter domain names are abbreviated to the first three letters. Black asterisks denote a new PayPal account and gaps in the blue line represent PayPal unavailability for that time period. The red vertical line indicates when the reporting of accounts started.

a single account that remained active during the entire 47-day initial observation period and the `snowstresser.com's` account remaining active for 37 days after the intervention began. On the other end of the spectrum, `quantumbooter.net`, `ragebooter.net` and `titaniumstresser.net` changed accounts every 4-5 days before the intervention. The impact of the intervention can be visually seen in Figure 5.1.

Once the PayPal payment intervention begins, the average lifespan of an account drops to 3.1 days with many booter's PayPal accounts only averaging around two days before they are no longer used again. Figure 5.1 visually shows the impact of the payment intervention on the lifespan of booter's PayPal accounts and provides some indication of the time period

that elapsed between a new PayPal account being actively used to accept payments and when PayPal took action against the account or it was proactively replaced. The length and number of PayPal outages increase after the intervention, with only `quantumbooter.net` and `titaniumstresser.net` avoiding major PayPal outages by resorting to aggressively replacing accounts. Note that this replacement strategy was not fully effective, since our monitoring infrastructure detected and reported these accounts.

We use the Kaplan-Meier estimator to compare the lifespan of PayPal accounts before and after the intervention. The lifespan duration of an account is defined as the time difference in days between the first usage of the account and its last date of usage. The accounts that were first seen before the intervention date and were still in active use on the intervention date are labeled as censored. The same applies to the accounts used by booter services in the time period after the intervention and still in active use at the end of the data collection period (07/24). Also as we don't know the first usage date for accounts already in use by the date we started the monitoring, the accounts seen on the first day of our monitoring are excluded form this analysis.

Figure 5.2 shows that the lifespan decreased after the intervention. The shaded areas represent the 95% confidence bounds. The result of a log-rank test with 99% confidence limits indicates a significant difference between the two survival curves.

5.3 Booters' status

As part of our daily monitoring of the 23 booter services, we recorded if the service could accept PayPal payments and if the site was functional. This enabled us to better understand the impact of the payment intervention on the booters' ability to accept PayPal payments and the operation of the monitored services. For each booter, we placed it in one of the following statuses each day based on the results of our crawl.

Active: The booter is able to successfully use a PayPal account to receive payments from its customers.

Unreachable/Broken: Either the booter's front-end website was not responding to HTTP

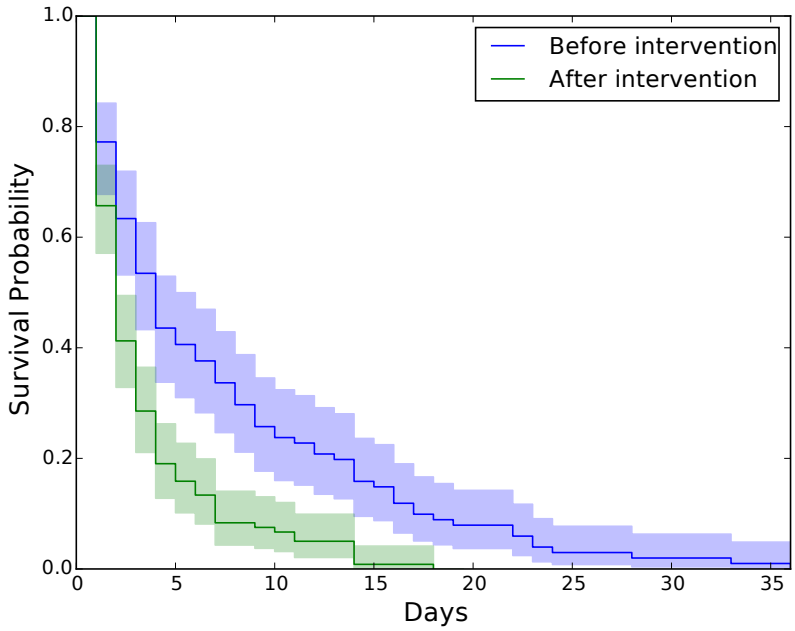

Figure 5.2: Lifespans of PayPal accounts before and after the intervention.

requests, the booter service had closed, or the front-end site was not functional.

PayPal Disabled: The booter's front-end website is active, but the service has either removed PayPal as a payment option, or the PayPal account linked to the booter website is limited and therefore unable to receive payments.

Figure 5.3 shows the status of booter services over time. The vertical line represents the date on which we started sharing our data with PayPal and PayPal started to independently investigate the reported accounts and take action on them. As observed in Figure 5.3, the percentage of active booters quickly drops from 70-80% to around 50% within a day or two following the intervention date and continues to decrease to a low of around 10% and then fluctuating between 10-30%. In addition, we observed 7 booter services in our study shut down their business and most of the remaining services switch to alternative payment methods such as Bitcoin.

However, as we will show in the next subsection, by switching to less convenient payment

methods, such as Bitcoin, the booter services experienced a drop in attacks that likely corresponds to a decrease in subscribers and revenue.

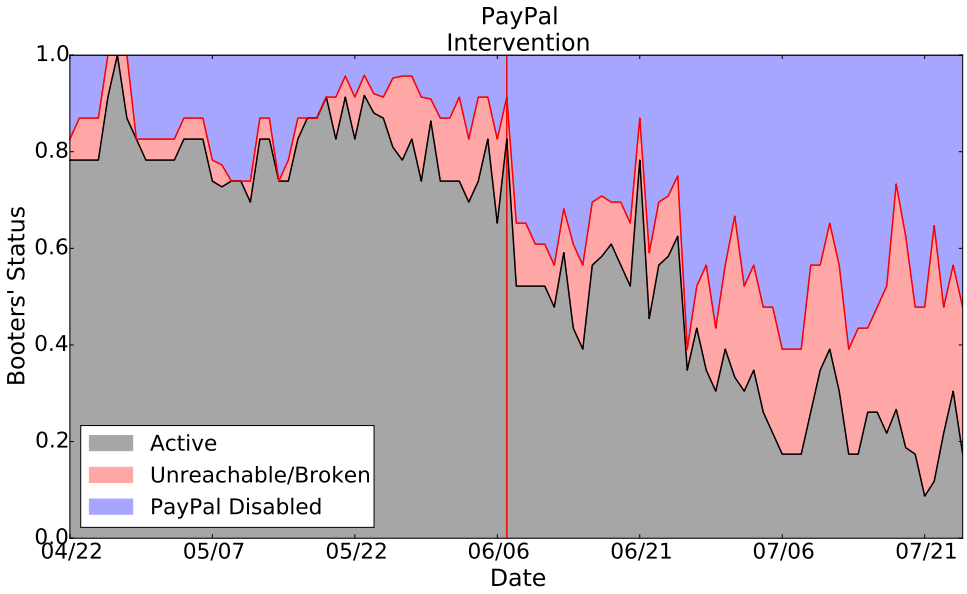

Figure 5.3: Status of booters over time.

5.4 Effect of PayPal Unavailability on Attack Levels

Some booter services report the cumulative number of all attacks launched by their customers. As part of the daily monitoring of booter services, we collected this self-reported data when it was available. Although we manually verified that the reported numbers were correctly updated when we initiated attacks on our own target, it is hard to be confidence that the data is completely accurate. However, assuming that this data can be trusted, the collected self-reported attack numbers combined with our PayPal availability measurements provide us with a metric to evaluate the effectiveness of the intervention on achieving the goal of reducing DDoS attack counts.

Figure 5.4 shows the impact of PayPal unavailability on daily count of attacks for a

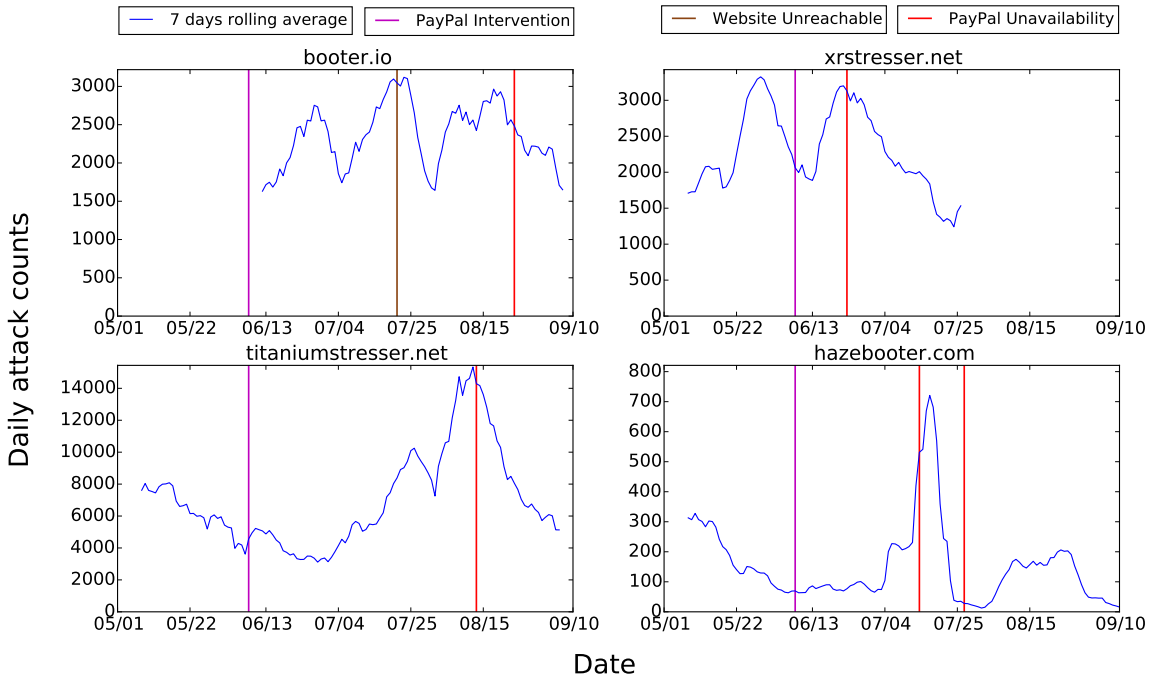

Figure 5.4: The impact of PayPal unavailability on daily attack counts.

number of booter services for which the data required for calculations were available. We consider PayPal to be unavailable to a booter service when the booter has not been able to use PayPal for receiving payments for at least two consecutive weeks. To capture the trend of attack counts over time, the 7 days rolling average of the dataset is plotted. As evident from the graphs, the booter services shown have been able to successfully accept PayPal payments for some time after the intervention was begun.

There are two recurring patterns captured by the graphs. First, the booter services shown have experienced an increase in the number of daily attacks in the period between the start of the intervention and PayPal unavailability. This can be explained by the fact that after the start of the intervention, there was a significant drop in the number of booter services on the market accepting payments using PayPal as the preferred method of payment for users. As a result, the booter services managing to continue accepting PayPal for some time after the start of the intervention attracted more customers and this in turn resulted in an increase in their daily attack counts. The second pattern is the decrease in the number of

daily attacks following the unavailability of PayPal to booter services. This can be explained by the fact that a booter service will not be able to retain its base of paid customers when new customers or existing customers renewing their subscriptions are required to use less convenient payment mechanism such as Bitcoin for payment. This churn in the customer base in turn results in drops in the daily attack counts.

Assuming that the PayPal intervention had an impact on the subscription behavior of users of booter services as we hypothesized, the observed patterns in the attack counts of booter services would make sense. However, many factors can influence the number of attacks launched by a booter service and it would be naive to assume that all observations are influenced by the PayPal intervention. For instance, we are unable to explain the reason why there is a decreasing trend for the number of attacks launched by customers of `titaniumstresser.net` and `xrstresser.net` right before the start of the PayPal intervention. Because of this and the inherent lack of confidence in the trustworthiness of the self-reported number of attacks, the results presented here should be interpreted with skepticism. The impact of PayPal intervention on attack levels could be quantified more confidently if we were able to collect the details of attacks launched by booter services[2]. However, none of the booter services that we were monitoring during the time periods before and after the PayPal intervention reported such details. Another way would be to acquire and study a leaked dataset covering the time period of interest for one or more of the booter services that we have monitored and know their the status over time.

5.5 Qualitative Assessments

From our booter status monitoring we observe that the effect of the payment intervention is more dramatic for less established booters. We believe that one reason for this is that they have not built up enough of a revenue stream and do not have enough reserve capital to weather the losses caused by merchant account terminations. This drove many of the smaller booters to shutdown their services as shown in the increase of unavailable services

[2]Like the case of vdos-s.com

as the intervention continues.

We also have qualitative evidence of PayPal's payment intervention efficacy. By monitoring the underground forums where these services advertise, we can witness the impact of these account limitations. Wrote one booter operator during the intervention, "So until now 5 time my 5 PayPal Accounts got Limited on My stresser is other stresser have same Problem with the f***ing Paypal ? is there any solution what we should do about f***ing Paypal ?" Similarly, customers vented their frustration at being unable to purchase booter subscriptions using PayPal. Wrote one booter customer, "when i go to buy a booter it normally says i can't buy because their PayPal has a problem."

In a number of cases, booters directly link their closures to loss of funds due to PayPal merchant account limitations. This message was posted on the front page of a defunct booter service, "It's a shame PayPal had to shut us down several times causing us to take money out of our own pocket to purchase servers, hosting, and more".

5.6 Booters' Responses

As with any intervention the adversary will respond by adapting to the pressure. In this case, we do not have enough quantitative measures to assess the effectiveness or the full range of responses to the attempt to undermine their payment infrastructure. However, we have identified several common classes of adaptations in response to the intervention.

Alternate payment methods. Most booters have added Bitcoin as an alternate payment method and have posted links to services that allow customers to purchase Bitcoins using credit cards or PayPal. In addition to Bitcoin, some have switched to Google Wallet and others have added the option to pay using virtual currencies, such as Webmoney and Perfect Money. By all accounts, these have resulted in reduced customer bases if the booter cannot directly accept credit card payments.

Offline payment. In some cases booters have posted that customers must open a ticket to pay using PayPal. This method increases the effort to monitor the booter for new accounts, since instead of an automated crawler someone must now interact with the booter service

manually. It also increases the difficulty of PayPal's investigation into the nature of the merchant account. However, this method also requires the booter service to manually active each account and the inconvenience may drives away customers that are seeking automated subscription purchasing systems.

Intermediate domain. Finally, we have noticed that some booters have stopped directly linking to PayPal and are now linking to an intermediary site and then redirecting the customer's browser from this intermediary domain to PayPal's site. This intermediary redirection site is used to hide the booter's real domain name in the referrer field from PayPal. A subset of booters have also started to replace this intermediary domain every time they replace a PayPal account. The benefit of this is that it makes it difficult for PayPal to link accounts using the referrers. This has increased the difficulty of monitoring booter's merchant accounts and required more effort on the part of investigations.

Chapter 6: Booter Attack Attribution

In this chapter, we focus on the problem of attributing DDoS attacks to the booter services responsible for launching them. The goal is to build a classification system to enable a victim to attribute amplification attacks based on features extracted from victim's network traces.

6.1 Dataset Description

The dataset used in this chapter for booter attack attribution was collected as part of a broader collaborative study with our colleagues at Saarland university, Germany for attribution of DDoS amplification attacks [34]. In that study we investigate the feasibility of attributing DDoS amplification attacks based on observations of DDoS attacks from a set of honeypot amplifiers that we operate, as well as attack attribution using features extracted from network traces collected at a victim's network. Similar to the experiments for characterizing the attack infrastructure of booter services described in chapter 4, we subscribe to booter services and direct attack traffic to our own victim server to collect ground truth data for validation of our amplification DDoS attack attribution techniques. In this chapter, we only focus on attack attribution on the victim side for which I was the primary contributor.

Similar to the booter attack infrastructure characterization study, we found booter services via search engines and advertisements on underground forums. We selected a total of 23 services offering amplification attacks based on NTP, DNS, CharGen and SSDP. When selecting these booters, we tried to include services that we speculated to be more stable and have more subscribers based on reviewing user feedbacks on underground forums. To minimize the amount of money we paid to these abusive services, we kept the number of

Table 6.1: Summary of the self-attacks.

Attack Type	**# of booters**	**attacks count**	**attacks observed**	**Avg # of amplifiers per attack**
CharGen	16	608	428	1210
DNS	19	676	455	10350
NTP	22	823	584	2166
SSDP	16	560	360	49646
Total	23	2667	1827	

covered booters relatively small.

We created custom crawlers to automate the task of visiting the websites of covered booters and launching attacks directed at our own target. Using this automation, daily attacks were launched per each covered booter and attack type. A total of 13 booter services were covered within the first week of starting the self-attacks on Dec 09, 2015 and by January 14, 2016 all 23 booters were covered. We choose to limit the length of all self-attacks to 30 seconds to collect valuable insight for attribution of amplification DDoS attacks while minimizing the collateral damage. Our victim server was connected by a dedicated 1 Gbps network connection that was not shared with any other server and the attack traffic generated by daily attacks were captured for further analysis.

Table 6.1 shows a summary of the self-attacks. In total, we launched 2667 attacks, but only around two thirds of these attacks were observed at the victim host. This can be explained by our observation of maintenance issues that some booter websites have. Sometimes booter websites provide the user interface for selecting a particular attack type that has been temporarily dysfunctional. To users, it appears that the attack has been successfully launched, but no actual attack traffic is generated as a result of initiating such attacks. The attack traces collected on the victim host along with the labels (the corresponding booter service) of attack instances constitute the ground truth data that we use to build and validate our classifier for attribution of DDoS amplification attacks.

6.2 Attack Features Used for Classification

In amplification attacks, the attack payloads as seen at the attack targets are generated by the set of amplifiers involved in carrying out the attacks and the attack payloads contain little information directly determined by the booter service responsible for launching the attack. Because of this, features extracted from attack payloads are generally unhelpful for attribution of attacks as observed by a victim.

However, based on the observation that a large number of amplifiers are publicly available on the Internet, and the assumption that many booter services independently scan for discovering amplifiers, we investigate the feasibility of using the set of amplifiers abused during an attack instance as a feature for identifying the corresponding booter service responsible for the attack. To this end, we investigate the similarity of amplifier sets used by the same booter service for the same attack type over time. Our analysis results reveals that for NTP, CharGen, and SSDP attacks, there is usually a significant overlap of amplifiers used by a booter service on consecutive dates. This is reasonable because booter services reuse their lists of amplifiers and update them periodically by rescanning to discover new amplifiers.

To measure the similarity of two sets of amplifiers abused in two attack instances, we use the Jaccard similarity coefficient computed as follows:

$$sim(A, B) = \frac{|\mathbf{IP}_A \cap \mathbf{IP}_B|}{|\mathbf{IP}_A \cup \mathbf{IP}_B|}$$

where A and B denote two attack instances, and $\mathbf{IP}_x$ denotes the set of IP addresses of amplifiers abused in attack instance x. Depending on the portion of amplifiers shared between two attack instances, the similarity coefficient ranges from 0 to 1.

Figure 6.1 shows the Jaccard similarity coefficients for amplifier sets abused by example booter services on consecutive dates. The temporal significant drops in similarities presumably indicates dates on which the booters have replaced their amplifiers lists by rescanning

for discovering new amplifiers.

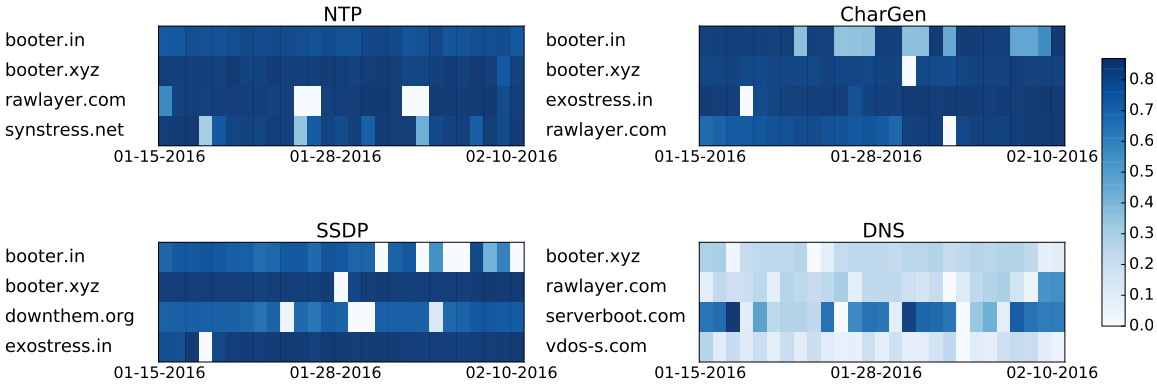

Figure 6.1: Jaccard similarity coefficients for amplifier sets abused by example booter services on consecutive dates.

In our dataset, for NTP, CharGen, and SSDP attacks there is generally a large overlap between the set of amplifiers used by various booters on consecutive dates. This indicates that booter services often continue to abuse a list of discovered amplifiers for some time before the list is updated to compensate for the churn of amplifiers. However, for DNS, there are some booter services for which the overlap of amplifier sets abused on consecutive dates is constantly low (less than 30%). One explanation is that these booters select a random subset of their DNS amplifiers for each launched attack.

As we will show later in this chapter, the set of amplifiers abused to deliver an attack is sufficient to build a classifier that can accurately attribute NTP, SSDP, and CharGen attacks. However as mentioned, for DNS the set of open DNS resolvers abused by individual booter services for DNS amplification is less stable and therefore relying on the set of abused amplifiers as the sole feature for classifying DNS attacks will not provide the same classification performance as it does for the other three attack types. As a result, we need to use an additional feature to improve the classification performance for DNS attacks.

Based on our analysis of DNS attack traces captured at our victim host, we noticed that each booter service tends to send spoofed DNS ANY requests for a single or a very small

number of domain names that result in large responses to be returned to the attack target. This means that in practice we can compensate for the less predictive set of amplifiers feature by using the set of domain names resolved in attacks as an additional feature for classification of DNS attacks.

6.3 Classification Algorithm

We leverage the features described in the previous section to attribute attack traces collected at a victim network to booter services responsible for launching the attacks. To this end, we use the self-attack data as training dataset to build a classifier for attack attribution. For classification, we use the *k-Nearest Neighbor* (k-NN) algorithm that conveniently allows us to use Jaccard similarity coefficients computed over amplifier sets and the set of resolved domain names in the case of DNS-based attacks to classify attack instances. In k-NN, to predict the label of an instance, the set of k neighbors with the least distance from the instance are computed and then each of the k neighbors casts a vote for its own label. The final decision on the instance's label is made by simply taking the majority of the voted labels.

As the similarity metric of our k-NN classifier for NTP, SSDP, and CharGen attacks, we use the Jaccard similarity coefficients computed over the set of amplifiers used by the compared attack instances. For DNS, the Jaccard similarity coefficient is computed for the set of amplifiers, as well as the set of resolved domain names and the similarity score is computed as the mean of the two computed similarity scores.

When using k-NN, the choice of k often has a significant impact on the classification performance. To determine the best value for k, one common approach is to learn the value from the training dataset using *n-fold cross-validation* (CV). In n-fold CV, the training dataset is partitioned into n equally sized sets ,$n-1$ sets are used for training the classifier and the final set is used for validation. This process is repeated n times using each of the n sets as the validation set once. At the end, the overall classification performance is evaluated by averaging the classification performance obtained for each of the n repetitions.

The best value for k can be determined by repeating the n-fold CV for different values of k and selecting the value that results in the best classification performance. As part of the classification, we use the n-fold CV approach to determine the value of $k \in \{1, 3, 5\}$. The value of k is restricted to odd values to avoid ties in the majority voting phase of k-NN.

In our context, the training dataset is not exhaustive (i.e., we don't have labeled attack instances for all booter services in existence.), and this needs to be taken into account when building the classifier. To this end, we apply a cut-off threshold value t to introduce the label "unknown" for attack instances that can not be confidently attributed to any of the booter services covered in the training dataset. That is, when searching for the nearest neighbors in k-NN, only attack instances in the training set for which the similarity score is no less than t are considered and attack instances for which no neighbor can be found are classified as "unknown". We choose a conservative threshold of $t = 0.55$ for CharGen, $t = 0.60$ for DNS, $t = 0.55$ for NTP, and $t = 0.45$ for SSDP. In order to select the threshold value, the score of correct classifications and incorrect classifications were investigated and a reasonably conservative value was selected for each attack type.

As the last note, as booter services are expected to periodically update their amplifiers lists, training attack instances are most relevant when they are not too distant in time form attack instances that we are trying to classify. To reflect this observation, when classifying an input attack instance, we only consider attack instances in the training set that are no more than 7 days apart from the test attack instance. For our dataset, inclusion of training attack instances beyond the 7 days distance does not improve the classification performance.

6.4 Evaluation Results

Although classifying an attack instance with a known booter label to either a different booter service or the label "unknown" is incorrect, we have a strong preference to misclassify attack instances as "unknown" rather than wrongly classifying them as another booter. Embedding this preference into the classifier has the implication that we are likely to classify some attack instances with known labels as "unknown", but on the positive side, we can expect higher

accuracies when attack instances are classified as one of the booter services in the training dataset. To reflect the preference for mislabelling an attack as "unknown" over incorrectly attributing it to the wrong booter, we assign a misclassification cost c_i to each label l_i and then compute the overall cost as $C = \sum c_i \cdot \mathsf{fpr}_i$. In our experiments, we use $c_i = 1$ for all booter labels and $c_{\mathtt{unk}} = \frac{1}{8}$ for misclassifying an attack as "unknown" and select values for the k parameter and the cut-off threshold to minimize the misclassification cost. The fpr for each class is computed as $\mathsf{fpr}_i = \mathsf{fp}_i/(\mathsf{fp}_i + \mathsf{tn}_i)$ where fp_i and tn_i are defined as follows:

- fp_i (false positives): The number of attack instances *incorrectly* classified as the label l_i.
- tn_i (true negatives): The number of attack instances *correctly* classified as a label other than l_i.

We perform several experiments using the labeled self-attack dataset to evaluate the performance of our classifier. For the first experiment, we use 10-fold CV to assess how well the classifier can perform to attribute attacks (E1). As n-fold CV is a randomized process, the reported numbers for this experiment are the mean computed over five runs.

Second, we evaluate the performance of the classifier for attributing attack instances only based on historical data (E2). That is, without relying on labeled attack instances from the *future*. Therefore, in E2, to classify attack instances happened on date d, we only use attack instances collected before this date as the training dataset.

Finally, to estimate how well the classifier can deal with situations when classifying an attack instance from a booter *not* contained in the training dataset, we employ the leave-one-out cross validation approach on the booter level (E3). This means that on each iteration, the attacks from all but one booter constitute the training set, and all attacks from the omitted booter are used for validation, checking if these attacks are correctly classified as "unknown".

To validate the performance of our classifier for attribution of booter DDoS amplification attacks, we perform the three experiments described above. A total of 30 attack

Table 6.2: Experimental results for booter amplification attack attribution (E1 and E2).

	Exp	Avg. Precision (%)	Avg. Recall (%)	Classified as unknown (%)
NTP	E1	98.97	98.93	2.77
	E2	99.25	99.21	12.13
DNS	E1	97.91	97.88	13.27
	E2	97.55	96.98	26.77
CharGen	E1	100	100	3.79
	E2	99.73	99.73	12.33
SSDP	E1	99.70	99.69	8.60
	E2	99.68	99.66	16.24

instances for which the number of abused amplifiers were less than 10 were excluded from the experiments. We assume that these were the result of background network noise or broken attacks that were not successfully launched by booter services. This leaves us with 417 CharGen, 452 DNS, 577 NTP, and 351 SSDP attack instances.

Table 6.2 summarizes the obtained results for the first two experiments. To quantify the performance, we report the percentage of the attack instances classified as "unknown", as well as the average precision and average recall for the remaining attack instances classified as a booter service. In the first two experiments the labels for all attack instances are known and therefore the attacks classified as unknown can be interpreted as the percentage of attack instances that the classifier missed by incorrectly classifying them as "unknown". For instance, looking at the results in Table 6.2 for E1 on CharGen attack instances, the classifier missed about 3.79% of the attack instances, but for the rest of attack instances that were classified as a booter, all attack instances were attributed to the correct booter. In general, in the first two experiments, for the attack instances not classified as "unknown", we obtain a high level of classification accuracy. Compared to E1, a higher percentage of attack instances are incorrectly classified as "unknown" in E2. This is however an expected

behaviour as in this experiment the classifier has to make predictions based on a smaller set of attack instances in the training set.

For E3, all attack instances to be classified are "unknown" and in tbl:selfattacks-e3 we report the percentage of attack instances incorrectly classified as one of the booters for each of the four attack types. As seen, for SSDP, 98.29% of the attack instances are correctly classified as "unknown". However, for the other three attack types, the percentage of attack instances incorrectly classified as a booter ranges from 13.05% to 14.87%. We noticed that in the case of NTP and CharGen, a large number of misclassifications involves only two booters that use an unusually similar set of amplifiers. It is possible that these two booter services share the same operator, but we didn't have enough evidence to confidently treat them as such. Assuming that these two booters are actually operated by the same individual or group, the rate of misclassifications for NTP and CharGen drops to around 3%.

Table 6.3: Experimental results for booter amplification attack attribution (E3).

	Classified as a booter (%)
NTP	13.34
DNS	13.05
CharGen	14.87
SSDP	1.71

In general, for E3 the rate of "unknown" attack instances classified as one of the booter services in the training set is relatively high. In order to compensate for this, we can apply a higher threshold value t, but this will also increase the percentage of attack instances incorrectly classified as "unknown" for the first two experiments.

6.5 Discussion

While the obtained results for attribution of DDoS amplification attacks to booters responsible for launching them are promising, the proposed method may be susceptible to evasion

attempts by booter services.

A booter service could try to evade our attribution methodology by adding noise to the attack traffic. For example, one could randomize the set of amplifiers abused in attacks. This is less practical for attack types such as NTP, and CharGen where only a small pool of amplifiers are available for booter services to abuse. It is not clear whether a classifier could still perform well in the face of such evasion attempts or not.

Also, rather than attempting to evade the proposed attribution methodology, a booter service could try to be attributed as another booter service by trying to use the same set of amplifiers as the other booter for delivering its attacks. Similar to our self-attacks, a booter could subscribe to a victim booter service and launch attacks to a target controlled by itself to learn the amplifiers used by the other booter service. However, using a set of amplifiers abused simultaneously by another booter service can result in a reduced amount of attack traffic that can be generated using these amplifiers. This can result in subscribers dissatisfaction for the booter attempting to masquerade as another booter service and therefore be detrimental to its business. Also, amplifiers aggressively abused by multiple booter services are more likely to be noticed and configured to disallow further abuse.

Chapter 7: Attribution of Economic Denial of Sustainability Attacks

As discussed in chapter 4, volumetric amplification attacks are the primary attack mechanism employed by booter services to deliver their ordered attacks. As in amplification attacks, the abused amplifiers reply with their own IP addresses (no IP spoofing) and there is no source port randomization involved, these attacks are generally easy to detect at the victim. Many different schemes are proposed in the literature for detecting and discarding amplification attack traffic [86]. However, to be able to mitigate a detected amplification attack, a victim needs to have enough bandwidth available to absorb the attack traffic. Otherwise, the victim service will be disrupted even though the attack traffic is easily detectable. As the detection of volumetric amplification DDoS attacks has been extensively studied in the past [86], in this chapter we focus on detection of EDoS attacks as a more subtle and recent variation of DDoS attacks which are likely to be offered by DDoS for hire services in the near future as the underground economy evolves and more small businesses start to use public clouds to run their operations and services.

As a new paradigm, cloud computing is reshaping the entire information technology industry. Cloud service providers enable their consumers to access shared computing resources in a flexible way without the need for upfront investment on infrastructure, platform, and software. Although the adoption of cloud computing has experienced significant growth in recent years, some concerns regarding the unique features of cloud computing environments have hindered its broader adoption. Security and privacy concerns in particular are frequently ranked as one of the top reasons why some organizations are reluctant to adopt cloud computing [87–89].

The understanding and mitigation of security and privacy risks of the public cloud computing model has been an active area of research in recent years. The research efforts

however, have been primarily focused on protecting the confidentiality, and integrity of sensitive data processed in public cloud environments as well as ensuring the continuous availability of cloud services for their intended users [90]. Very little attention has been paid to security threats targeting the cost model of consumers running their services on the public cloud[91].

Services running on public clouds are vulnerable to fraudulent resource consumption attacks aiming at increasing the financial burden of the victim service. This is enabled by exploiting the utility-based pricing model of the cloud where consumers are charged for the actual consumption of computing resources such as CPU cycles, RAM, bandwidth, and storage [92].

An adversary can conveniently rent a botnet [93] consisting of thousands of bot machines to incur artificial cost to a victim service. The target of the attack will have to pay for the cost of fraudulent resource consumption resulted from requests made by bot clients. By keeping the rate of fraudulent requests made by individual bots low to mimic the behavior of legitimate users, and intelligently focusing on requests that are most costly in terms of resource consumption, an attacker can sustain the attack over an extended period of time and maximize the effectiveness of the attack.

In practice, any device with an Internet connection is capable of launching an EDoS attack. The attacker can simply instrument the device to send HTTP GET requests to the victim service at the highest rate possible. This is basically the method used in application layer DDoS attacks where the attacker's goal is to render a targeted service unavailable to its intended users by overwhelming victim's resources. However, this will very quickly result in a significant deviation from the request rate of normal users and this artifact can be used for detecting the offending source and dropping its requests. [94–96].

In this chapter, we focus on an adversarial scenario in which the attacker's goal is to increase the financial burden of the victim. This attack is also refereed to as Fraudulent Resource Consumption (FRC) by some researchers in the literature [91,92]. We assume that the attacker is intelligent in the sense that she makes requests that are resource-intensive

resulting in higher costs for the victim. Also, for the attack to be effective, it needs to remain undetected for an extended period of time. Because of this, not only that malicious requests must not cause any visible degradation in quality of service, but also the number of requests made by malicious sources should not be very different from the number of requests made by legitimate users.

As malicious clients participating in a stealth EDoS attack make requests in a similar rate as legitimate users, this type of attacks can be challenging to detect and mitigate. In this chapter, we present a method for detecting stealth EDoS attacks by directly assigning a cost to each user request in proportion to the resources consumed to serve that request.

The proposed method is based on statistical anomaly detection. First, we process web server logs to identify the sequence of requests made by each individual user over a predefined period of time. Next, according to the amount of resources consumed to serve each request, a relative cost value is assigned to each request. The result is a dataset consisting of a cost sequence of requests for each of the legitimate users in the processed web access logs. The cost sequence of requests for each user is considered as a random or stochastic process and an underlying Hidden semi-Markov Model (HsMM) is used to describe the behavior of users in terms of the cost they incur to a service over time. We use the request cost sequences collected for normal users as training data to estimate the parameters of the HsMM. Once the parameters of the HsMM are estimated, at the detection phase, the abnormality of a newly observed request cost sequence is tested to identify malicious sources participating in an EDoS attack.

We use our department's web access logs of about a month to experimentally evaluate the effectiveness of the proposed method. The experimental results show that our method is very effective in differentiating normal users and malicious users participating in EDoS attacks. While most of previously proposed methods require a malicious source to make significantly more requests than legitimate users to be effective, our method can successfully detect malicious sources that try to remain undetected by making only a few resource-intensive requests.

The remainder of this chapter is structured as follows. We begin by a discussion on the exploitation of the cloud pricing model that motivates this work. Related work is discussed in section 2. Section 3 presents a brief background on Hidden Semi Markov models and our formulation of identifying malicious sources participating in an EDoS attack using HsMM. The details of experiments designed to validate the proposed method and their results are presented in section 4.

7.1 Exploitation of the Utility-based Pricing Model

The cloud computing technology provides many attractive benefits such as avoiding the need for upfront spendings on computing infrastructure, improved manageability, security, and elasticity to businesses of various sizes. While the flexibility of the "pay-as-you-go" pricing model adopted by cloud service providers can be beneficial to cloud consumers, it leaves them vulnerable to financial risks imposed by EDoS attacks [91, 92].

To launch an EDoS attack, all an attacker needs to do is to simply send seemingly legitimate requests to a victim service to make it consume cloud resources for which the victim will have to pay for the cost. If the attacker is able to enforce significant fraudulent resource consumption over an extended period of time, the economical sustainability of the victim service could be threatened.

In an EDoS attack, the attack target can be a website or web applications hosted in a third party public cloud and we assume that attack targets predominantly serve public content accessible to all Internet users.

Unlike Distributed Denial of Service (DDoS) attacks, an EDoS attack is not meant to cause availability issues or noticeable degradation of service quality for the users of a targeted service. To be effective, an EDoS attack needs to be stealthy and remain undetected over an extended period of time (e.g., weeks or months). To remain undetected, a wise attacker will want to keep the rate of fraudulent requests low to blend them into the noise of legitimate requests, but instead focus on making requests resulting in high levels of cloud resource consumption in order to achieve the objective of the attack.

As documented in recent studies, DDoS for hire services can be readily located and rented on underground black markets [97–99]. These abusive services are often supported by botnets consisting of tens of thousands of compromised hosts and offer both network layer and application layer attacks [100]. With the availability of DDoS for hire services, an attacker does not need to be capable of building a supporting attack infrastructure.

The potential impact of an EDoS attack can be best quantified by examining a hypothetical attack on a service hosted on a real public cloud service provider. In the sequel we consider a hypothetical attack on a victim service hosted on Amazon's Elastic Compute Cloud (EC2) platform. Although cloud consumers are billed for various cloud resources including computing, network, and storage resources, for simplicity, this work only focuses on data transferred from the cloud environment to the Internet. Table 7.1 shows the cost of outgoing data transfer for Amazon's EC2 platform [101].

Table 7.1: Amazon EC2 Outgoing Data Transfer pricing as of February 2016.

Data Transfer OUT From Amazon EC2 To Internet	
First 1 GB / month	$0.00 Per GB
Up to 10 TB / month	$0.09 Per GB
Next 40 TB / month	$0.085 Per GB
Next 100 TB / month	$0.07 Per GB
Next 350 TB / month	$0.05 Per GB

According to the HTTP Archive [102], which regularly measures the Alexa top 10,000 websites [103], the average page size was 2,225 KB for the homepage of the top 10,000 websites visited in January 2016. However, many websites host a number of much larger web resources such as videos or large compressed files that an attacker can focus on to maximize the cost of resource consumption for a victim operating on a public cloud. For the purpose of our hypothetical EDoS attack, we assume the average size of web resources requested by malicious bots participating in the attack to be 100 MB.

At the rate of only 100 requests per month which is too low to raise any red flags, a single bot would consume about 10 GB of outgoing bandwidth and the monthly bill will increase by

90 cents. The inflicted cost will grow linearly by increasing the request rate, requesting larger files, or employing more malicious bots. Sending requests with the same characteristics as the single bot scenario from a 1000 bots will approximately cost the victim 900$ per month. As seen from this hypothetical attack, the resource consumption cost accumulated over time can impose an important financial burden to public cloud consumers. As individual bots show no trace of excessive request rates, most of existing detection schemes that look for a large number of requests in a short period of time [95, 104] will not succeed at detecting the described hypothetical attack.

It worth noting that leasing a botnet to carry out an EDoS attack will be a cost factor that an attacker would need to take into consideration. However, due to the fact that only a very small fraction of resources available to a compromised host are actually required to make a few requests at a very low rate, the cost of accessing a botnet can be significantly reduced for an attacker by renting non-dedicated botnets shared with other cybercriminals using the bots for various purposes.

7.2 Related Work

So far there are only a few studies in the literature directly concerned with the issue of EDoS attacks.

Khor and Nakao [105] propose a mitigation mechanism based on cryptographic puzzles to dissuade clients from submitting fraudulent requests. The basic idea of their proposed scheme called self-verifying Proof of Work (sPoW) is to require clients to present a proof of work before a protected service will commit its resources to serve client's requests. When a client first requests a resource, it receives a "crypto-puzzle" from sPoW that mediates all communications between clients and the protected service. The puzzle contains encrypted information necessary to reach the intended service such as the IP address and port number as well as a partial encryption key with k bits concealed. The client will have to spend its resources to discover the encryption key by brute forcing the k concealed bits so that it can decrypt the information necessary to contact the requested service.

However, sPoW or any other solution based on the "crypto-puzzle" approach [106] can be most helpful when malicious sources are sending requests at a high rate to a target service. In an intelligent and stealth EDoS attack, malicious clients can afford to solve the puzzles to submit only a few well-crafted, resource-intensive requests and succeed at adding financial burden to a victim service protected by sPoW.

Sqalli et al. propose a mitigation scheme called EDoS-Shield to address the issue of EDoS attacks in cloud environments [107]. The main idea of EDoS-Shield is to detect whether an incoming request is initiated by a legitimate user or by an automated source. EDoS-Shield depends on CAPTCHA tests to verify the source of requests. The proposed architecture is consisted of virtual firewalls (VF) and verifier nodes (V-Nodes) that are deployed as virtual machines in the cloud. The V-Nodes are responsible for verification of request sources, and VF nodes are implemented to decide if incoming packets should be forwarded or dropped based on the verification results received from the V-nodes. One weakness of the EDoS-Shield mitigation scheme has to do with the cost of additional cloud resources required for deploying the verifier nodes and the virtual firewalls. But, more importantly, this approach requires all users to be verified and research studies suggest that CAPTCHA tests could be annoying for some users and even a certain portion of legitimate users may not be able to solve them [108]. In addition, some existing CAPTCHA tests have been shown to be vulnerable to automated attacks [109], and now inexpensive CAPTCHA solving services that use crowd sourced human labor can be used to effectively defeat the protection purpose of CAPTCHA tests [110].

In [111] the authors use a number of statistical self similarity metrics including Zipf's law, and Spearman's Footrule distance to detect the occurrence of FRC attacks. The proposed detection mechanism only looks at the aggregate pattern of user requests and does not deal with identification of individual malicious sources participating in an attack. In contrast, our proposed method is concerned with identification of malicious sources exhibiting a similar behavior as legitimate users in terms of request rates, but focusing on resource-intensive requests to maximize the cost for the victim service.

Idziorek and Tannian propose a method that attempts to model the behavior of individual users based on the number of requests per session generated by each user over a fixed period of time [91]. A pause of 900 or more seconds between consecutive requests from the same user is used as the criterion to group user requests into web sessions. The premise is that malicious users generating sessions with a random number of requests would be sufficiently different from the profile of normal users, so that an entropy-based detection method could be used to identify malicious sources. This method is based on the assumption of malicious users making more requests/web sessions than legitimate users. However, as mentioned earlier, an intelligent attacker does not necessarily need to make malicious sources to send more requests than legitimate users to succeed. By focusing on web resources that are expensive in terms of resource consumption, malicious sources with similar request rates as legitimate users can be quite effective.

In [112], the authors propose a methodology for identifying malicious sources trying to inflate the utility bill of a victim by making fraudulent requests. The proposed methodology combines four different usage metrics including the number of sessions, the number of requests, and the average number of requests per session. For the last usage metric, the overall request frequency distribution of documents hosted on a website is computed, and the requests made by individual users are compared against this distribution. To evaluate a user, a probability score is computed for each of the four metrics and an overall average probability is computed. The more deviation observed from the normal usage, the higher would be the probability score and the more likely the user would be a malicious client. Again, this model is heavily influenced by the usage volume of individual users, and it will not be effective for detecting malicious users making a small number of high cost requests. As we will show in section 7.4, our proposed method is able to detect both malicious sources making an anomalous number of random requests, as well as more subtle malicious sources with a request rate similar to that of legitimate users but focusing on requests that are more costly for the victim.

7.3 The Proposed Method

In this section we give a brief description on the theory of HsMM and a forward, backward algorithm that we use for estimating the parameters of HsMM [113]. We also describe our formulation of detecting malicious sources participating in an EDoS attack using HsMM.

7.3.1 Hidden Semi-Markov Model and Parameter Estimation

HsMM extends the traditional Hidden Markov Model (HMM) by allowing states to have variable durations [114]. The duration of a state represents the number of observations made while in that state. Consider a HsMM with M states denoted as $S = \{s_1, s_2, ..., s_M\}$. A HsMM can be specified by its parameters as $\lambda = (\{\pi_m\}, \{a_{mn}\}, \{b_m(k)\}, \{p_m(d)\})$ where:

- $\pi_m \equiv \Pr[s_1 = m]$ is the initial state probability distribution. s_t denotes the state taken by the model at time t and $m \in S$. The sum of initial state probabilities adds up to 1($\Sigma_m \pi_m = 1$).

- $a_{mn} \equiv \Pr[s_t = n | s_{t-1} = m]$ is the state transition probability for $m, n \in S$, satisfying $\Sigma_n a_{mn} = 1$.

- $b_m(k) \equiv \Pr[o_t = k | s_t = m]$, for $m \in S, k \in \{1, ..., K\}$ is the state output distribution. The observable output at t is denoted by o_t and k is the index into the observable output set with cardinality K. The output distribution satisfies $\Sigma_k b_m(k) = 1$.

- $p_m(d) \equiv \Pr[\tau_t = d | s_t = m]$ is the state residual time distribution, for $m \in S, d \in 1, ..., D$. D represents the maximum interval between any consecutive state transitions and the residual time distribution satisfies $\Sigma_d p_m(d) = 1$.

Then, if at time t, the pair process (s_t, τ_t) takes on the value (m, d), where $d >= 1$, the semi-markov chain will remain in state m until time $t + d - 1$ and will transit to the next state at time $t + d$. The states themselves are not directly observable. The observables are a sequence of observations $O = (o_1, ..., o_T)$. The notation o_a^b represents

the observation sequence from time a to time b and conditional independence of observed outputs is assumed so that $b_m(o_a^b) = \Pi_{t=a}^b b_m(o_t)$. The model parameters are initially estimated and are then updated as new observations o_t are collected. This process is known as parameter reestimation and it can be done by following the forward and backward algorithm proposed by Yu and Kobayashi [113]. The forward and backward variables are defined as follows:

$$\alpha_t(m,d) \equiv \Pr[o_1^t, (s_t,\tau_t) = (m,d)|\lambda]$$

$$\beta_t(m,d) \equiv \Pr[o_{t+1}^T, (s_t,\tau_t) = (m,d)|\lambda]$$

which can be recursively computed by forward and backward algorithms. Next, the three following joint probabilities are defined that can be expressed and computed in terms of the model parameters and the forward and backward variables defined above. These probabilities are used to readily derive the reestimation formulas to update the model parameters after collecting new observation sequences.

$$\zeta_t(m,n) \equiv \Pr[o_1^T, s_{t-1} = m, s_t = n|\lambda]$$

$$\eta_t(m,d) \equiv \Pr[o_1^T, s_{t-1} \neq m, s_t = m, \tau_t = d|\lambda]$$

$$\gamma_t(m) \equiv \Pr[o_1^T, s_t = m|\lambda]$$

Now, using the joint probabilities defined above, the model parameters can be reestimated by the following formulas:

$$\hat{\pi}_m = \gamma_1(m) \Big/ \sum_{m=1}^{M} \gamma_1(m)$$

$$\hat{a}_{mn} = \sum_{t=1}^{T} \zeta_t(m,n) \Big/ \sum_{t=1}^{T} \sum_{n=1}^{M} \zeta_t(m,n)$$

$$\hat{b}_m(k) = \sum_{t:o_t=k} \gamma_t(m) \Big/ \sum_{k} \sum_{t:o_t=k} \gamma_t(m)$$

$$\hat{p}_m(d) = \sum_{t=1}^{T} \eta_t(m,d) \Big/ \sum_{t=1}^{T} \sum_{d=1}^{D} \eta_t(m,d)$$

The model parameters are reestimated for each input observation sequence and after processing all observation sequences, the trained model can be used to compute the likelihood of a new observation sequence by the following formula:

$$\Pr[o_1^T|\lambda] = \sum_{m} \sum_{d} \Pr[o_1^T, (s_T, \tau_T) = (m,d)|\lambda]$$

$$= \sum_{m} \sum_{d} \alpha_T(m,d)$$

7.3.2 HsMM for Detection of Malicious Sources in EDoS Attacks

In this subsection we apply the HsMM formulation to identify malicious sources participating in an EDoS attack.

Most web requests are for HTML documents that are meant to be rendered, and displayed by a user browser. These requests are typically followed shortly by several subsequent HTTP GET requests to fetch objects such as images, scripts, and CSS files embedded in the main requested document. The requests can also be for downloading objects such as

binary files over HTTP. Web servers can be configured to log the details of all user requests including the IP address of the requesting host, the requested document, the type of request (GET, POST, etc), and the size of data transferred to serve the request. Although all the HTTP request types cause resource consumption on the server side, to simplify our experimentations, we only focus on HTTP GET requests in our work.

Proportional to the amount of data transferred to serve a request, a relative cost value can be calculated and assigned to each request. Based on the data size of various requests, one can decide on a small number of buckets to represent different cost values to be associated with user requests. We will see an example of this in section 7.4 where we use cost values from 1 to 5 for requests in our dataset.

Using the collected web server logs, requests made by each individual user during a specific period of time can be identified and mapped to request cost values. The result would be a sequence of request cost values for each user. We assume that individual users (both legitimate and malicious) can be uniquely identified by their IP addresses. Using browser fingerprinting techniques [115] can be a potential solution for cases where some users can not be reliably identified by their IP addresses.

The sequence of request costs from individual users during a specific period of time can be considered as a random or stochastic process and an underlying HsMM can be used to describe the behavior of users in terms of the cost they incur to a service over time. The request cost values are the observable outputs and the hidden states represent different levels of resource consumption by users. In our implemented model we use 5 hidden states where the model is always initially in the first state. Also, in our model a transition can only happen from a lower state to a higher state.

We use requests made by legitimate users to estimate the parameters of the HsMM and then use the trained model to compute the likelihood of new request cost sequences generated by users. The request cost sequences generated by malicious users would be different from legitimate users and this will result in much smaller likelihood values than those of legitimate users. As we will show in the next section, using the right threshold

likelihood value, legitimate users and malicious users can be effectively distinguished.

7.4 Experimental Evaluation

We conduct experiments to evaluate the effectiveness of the proposed method for detecting malicious sources engaged in fraudulent use of cloud resources. This section provides a description of our experiments and presents the obtained results.

7.4.1 Dataset Description

Our experiments are based on request logs from our department's public web server collected over 32 days from Nov 8, 2015 to Dec 9, 2015. We use the rules below to filter out requests that are irrelevant for our purpose:

- Requests that are not HTTP GET.
- HTTP GET requests with a response code other than 200 (OK).
- Requests with a user agent string indicating access from a non-user entity (e.g., Googlebot, Wget, etc).
- Request sources making requests using 10 or more different user agent strings. This is to remove aggregate request sources such as NAT boxes or web proxies making requests on behalf of their clients. About 90% of all request sources in the dataset only use a single user agent string.
- The access logs are splitted into two 16 days periods. We only include requests from users making at least 3 requests in one of the 16 days period.

A request for an HTML document and the subsequent requests for fetching objects embedded in the same HTML document are combined and treated as a single request. Table 7.2 presents a summary of the normal dataset used for training and testing the proposed model.

Table 7.2: Summary of the normal experimental dataset.

Metric	Train dataset	Normal test dataset
Number of days	16	16
Total number of unique users	4,933	5,252
Total number of requests	36,466	36,474
Avg number of requests per user	7.39	6.94

To generate the normal training dataset, requests in the first half of the logs are grouped based on the request source, and then, proportional to the amount of data transferred to serve the requests, they are mapped to relative cost values. The same process is applied to the requests in the second period of the logs to generate the test dataset representing users with normal resource usage behavior. Based on our observation of the user requests in the dataset, we choose to use the values from 1 to 5 to represent the relative cost of user requests. Thus, the final dataset is a sequence of request costs ranging from 1 to 5 in value for each user. Table 7.3 summarizes the mapping from the request size to relative cost values and the distribution of requests in terms of their cost values in our dataset.

Table 7.3: Mapping of request sizes to relative cost values.

Request size	Relative cost	Percentage of requests
<500KB	1	86.7
$\geq$ 500KB and <5MB	2	11.4
$\geq$ 5MB and <50MB	3	1.9
$\geq$ 50MB and <500MB	4	0.1
$\geq$ 500MB	5	5.4e-03

Although in our experiments we use an observation window of 16 days to profile the behavior of normal users and the same observation period is used for detection of malicious users, the proposed methodology is only sensitive to the resource usage pattern of users and is not restricted to a specific observation period.

7.4.2 Attack Scenarios

To conduct an EDoS attack, an attacker needs to specify the behavior of individual bots by defining the request rate and the requested resources in term of their resource consumption cost. By varying these two parameters, various attack strategies that an attacker is likely to adopt can be constructed and the effectiveness of the proposed detection method can be evaluated for those attack strategies. We first consider attack strategies where the attacker focuses on making requests that result in high levels of resource consumption. For these attack strategies we assume that the attacker has a prior knowledge about the rate of requests made by legitimate users, and uses this knowledge to avoid suspicions by making requests with similar rates as legitimate users.

In sequel, we briefly describe a number of various attack scenarios that we use to generate synthetic malicious request sequences to evaluate the performance of the proposed attribution methodology. These attack scenarios are ordered in an increasing order of detection difficulty and a decreasing order of attack effectiveness. For the attack scenarios listed below, the number of requests made by individual malicious sources is normally distributed with parameters μ=7 and σ=2 which does not significantly deviate from the number of requests made by legitimate users.

- **Scenario 1**: All malicious requests have a cost of 5.
- **Scenario 2**: All malicious requests have a cost of 4 or 5.
- **Scenario 3**: The request cost is 5 for 75% of malicious requests. The cost for the remaining 25% of requests is uniformly distributed between 1 and 4.
- **Scenario 4**: 75% of malicious requests have a cost of 4 or 5. The cost for the remaining 25% of requests is uniformly distributed between 1 and 3.
- **Scenario 5**: The request cost is 5 for 50% of malicious requests. The cost for the remaining 50% of requests is uniformly distributed between 1 and 4.

- **Scenario 6**: 50% of malicious requests have a cost of 4 or 5. The cost for the remaining 50% of requests is uniformly distributed between 1 and 3.

7.4.3 Experimental Results

For each attack scenario, we generate a dataset of malicious requests according to the description of that attack scenario. Each test dataset consists of generated malicious request sequences, combined with the request sequences from the normal test dataset. The normal test dataset is the same for all attack scenarios. Also, in our experiments, each test dataset contains the same number of normal and malicious sources (5,252). False Positive Rate (FPR), and False Negative Rate (FNR) are the metrics used for performance evaluation of the proposed detection method under various attack scenarios. These metrics are briefly described in the following:

- **FPR**: The percentage of request sequences generated by legitimate users classified as malicious. Keeping the FPR under a low threshold is very important. Otherwise, legitimate users will be denied access to the protected service.
- **FNR**: The percentage of malicious request sequences generated by sources participating in an EDoS attack scenario not detected by the proposed method. Unlike an Intrusion Detection System (IDS) where it is very crucial not to miss any intrusions, because a single missed intrusion can result in system compromise, in our context, missed malicious sequences would only cause some billable fraudulent resource consumption.

For each attack scenario, the trained model is used for computing the log likelihood for all request sequences in the test dataset of that attack scenario. In general, the request sequences from legitimate users which are similar to the data used for training the model are expected to receive high log likelihood values. On the other hand, malicious request sequences representing a resource consumption behavior dissimilar to that of legitimate users

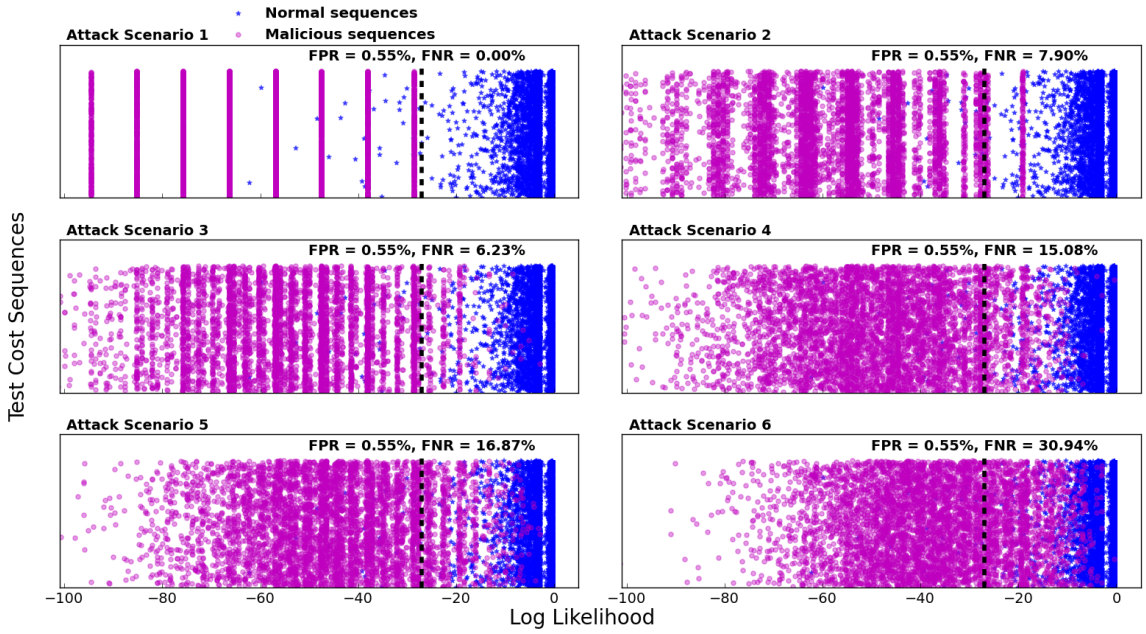

Figure 7.1: Experimental results for attack strategies focusing on high cost requests.

are expected to be assigned lower log likelihood values. Once the log likelihoods are computed for all request sequences, a threshold value can be identified that best distinguishes legitimate, and malicious request sequences.

Figure 7.1 shows the detection performance of the proposed method for the described attack scenarios. We use a fixed threshold value of -28.6 to distinguish legitimate users and malicious sources. This threshold value results in a FPR of 0.55%. The FNR can be improved by choosing a larger value for the threshold, however, this will have the more undesirable result of a higher FPR.

For the first attack scenario, the FNR is 0, meaning that all sources generating malicious requests are successfully detected. Sometimes a given website may have legitimate users that use the website in unusual ways. In our test dataset for legitimate users, there are 29 users that make significantly more requests than the other legitimate users. The average number of requests made by these users is 77 compared to 7 for all legitimate users. The log likelihood values computed for the long request sequences generated by these users are low and this results in some undesirable false positives. If unusual request patterns are

expected from specific users, false positives can be avoided by ignoring requests from these known users. In our experiments, the legitimate users with an abnormally large number of requests are incorrectly classified as malicious.

As the test sequences for legitimate users are shared across all attack scenarios, and the same threshold value is applied for the experiments in Figure 7.1, the FPR remains constant for the various attack scenarios. However, the FNR increases as malicious request sequences become more similar to those of legitimate users. For attack scenario 6 that is the most challenging to detect, still close to 70% of malicious request sequences are successfully detected. It should be noted that the undetected malicious sequences are usually comprised of fewer requests compared to the successfully detected malicious sequences. For instance, for the attack scenario 6, the average number of requests for undetected malicious request sequences is 4.93 versus 7.23 for the detected malicious request sequences. This implies that the undetected malicious request sequences are less effective in terms of fraudulent resource consumption and the more effective malicious request sequences that are more aggressive in nature run higher risk of detection. In our experiments, all malicious sequences containing at least three requests with the request cost of 5, are correctly classified as malicious. This shows that attacks focusing on requests with the highest level of resource consumption will be detected very quickly.

As suggested by the result presented in Figure 7.1, attacks focusing on requests with high resource consumption costs can not go undetected. The alternative for an attacker would be to attempt making requests with a similar distribution of request costs as legitimate users, but in larger quantities to increase the amount of fraudulently consumed resources.

Table 7.4: Experimental results for attacks focusing on high number of requests.

	FNR(# of requests per source)				
FPR	**30**	**40**	**50**	**60**	**70**
0.55%	98.70%	94.00%	76.52%	40.72%	18.16%
1.01%	90.92%	56.74%	24.87%	10.74%	5.03%

Table 7.4 shows the evaluation results for the attack strategy where the attacker focuses on making larger numbers of requests that have the same distribution of request costs as legitimate users. The FNR is reported for various number of requests and two different FPRs. As observed, when malicious clients make 70 requests, about 95% of them are detected, if a FPR of 1.01% is acceptable. For a FPR of 0.55%, still about 82% of malicious sources making 70 requests are successfully detected. It should be noted that from the standpoint of an EDoS attacker, regular requests not causing high levels of resource consumption are not helpful and this attack strategy only makes sense when malicious sources are able to make a significant number of regular requests and manage to remain undetected.

Chapter 8: Discussion

We have gathered a few key points from ours and the communities' efforts to understand and undermine DDoS booter services. Most of these potential strategies involve driving up costs and the risk associated with operating and subscribing to booter services. This might force these booters from operating largely unopposed in the open to more resilient hosting, attack and payment infrastructure for which they will have to pay a premium due to the risk of support services being taken down or blacklisted for being associated with DDoS attackers.

Reducing scale. Limiting access to convenient payment methods, such as PayPal, had an impact on the scale of booter services based on our quantitative and qualitative analysis. However, based on the short duration of the intervention it is unclear if this approach would continue to be effective in the longer term. As a research agenda, it would benefit the anti-cybercrime community to focus on understanding how to improve the effectiveness of these interventions and make them sustainable.

Reducing effectiveness of attacks. One potential course of action for reducing the effectiveness of DDoS attacks launched by booter services is to monitor the amplification servers abused by booters and share the information with existing patching efforts, such as the OpenResolverProject [116] and OpenNTPProject [117]. Our hope is that by focusing mitigation efforts on actively abused amplifiers, we can mitigate the pool of more stable amplifiers and thus reduce the effectiveness of booter attacks as they are forced to use less stable amplifiers. There is some indications that active notification improves patching rates of vulnerable services [118]. Notification efforts could also be helpful for reducing the population of vulnerable servers for newly discovered amplification vectors such as Trivial File Transfer Protocol (TFTP) [119] before these vulnerable servers are abused at large scale by booter services for delivering their ordered attacks.

Increasing costs. This might be achieved with an increased effort to locate and blacklist or de-peer low-cost hosting services that cater to DDoS attacks with providing the ability to send out an unlimited amount of spoofed traffic at high rates. This might force these services to pay a premium for bullet proof hosting attack servers, which would result in reduced profitability or be passed long to subscribers in the form of increased subscription costs. In addition, convincing CloudFlare and other free anti DDoS services to prohibit these booter services would increase their costs by forcing them to build and pay for anti DDoS services that cater to these abusive booters. Admittedly these suggestions will likely not result in large cost increases unless tremendous amounts of pressure were placed on these parts of booter services infrastructure.

Increasing risk to operators. Our analysis of data provided by PayPal suggests that much of this activity is occurring in the United States. If this is the case, there is the potential that increased law enforcement efforts could have a direct impact in arresting key operators of these services and increasing the perceived risk of operating and using these services. Our work for attribution of amplification DDoS attacks to booter services using the data collected at victim's network or honeypot amplifiers [34] can be used by law enforcement to identify and act upon booter services that are responsible for the majority of attacks.

Chapter 9: Conclusion

Unfortunately, there is no silver bullet that will mitigate the threat posed by booter services over night. These booters have grown in scale due to the perceived low-risk nature, their profitability and the increasing demand for DDoS attacks as a method of knocking out competitors, harassment and censorship on the Internet.

In this dissertation we have analyzed the technical infrastructure and business structure of booter DDoS operations. By viewing booters in this light, we have an improved understanding of the full range of support infrastructure that booters depend on in terms of advertising, attack delivery, hosting, and payment. Our investigations via direct interactions with booters and support services highlights potential improvements to ongoing patching efforts to diminish the attack infrastructure. From our measurements, there is a relatively small pool of NTP and CharGen amplifiers that are heavily abused by booters for launching high amplification DDoS attacks, and prioritizing these amplifiers for patching efforts could be potentially helpful in making booter attacks less efficient. We have also demonstrated that payment interventions which undermine the accessibility of convenient payment methods such as PayPal can potentially have an impact on reducing the scale of booter services.

We have also presented a method based on classification to enable victims to attribute amplification DDoS attacks to booter services responsible for launching them. Our proposed classifier mostly relies on the set of amplifiers abused by booter services to deliver their ordered attacks, to attribute attack traces to booter services. The obtained experimental results based on our collected ground truth dataset suggest that the proposed method is effective for identifying booter services responsible for generating the attack traffic seen at the victim network.

Finally, we have considered EDoS attacks as a more subtle and recent variation of

DDoS attacks that could be offered by DDoS for hire services in the near future as the underground economy evolves and more small businesses start to use public clouds to run their operations and services. We designed a method for detecting stealth EDoS attacks by directly assigning a cost to each user request in proportion to the resources consumed to serve that request. We have shown experimentally that our proposed method is able to detect both malicious sources making an anomalous number of random requests, as well as more subtle malicious sources with a request rate similar to that of legitimate users but focusing on high cost requests.

Bibliography

Bibliography

[1] R. Beverly, R. Koga, and K. Claffy, "Initial longitudinal analysis of ip source spoofing capability on the internet," 2013.

[2] M. P. C. Inc., "The ddos that almost broke the internet," https://blog.cloudflare.com/the-ddos-that-almost-broke-the-internet/, March 2013.

[3] ——, "Technical details behind a 400gbps ntp amplification ddos attack," https://blog.cloudflare.com/technical-details-behind-a-400gbps-ntp-amplification-ddos-attack/, Feburary 2014.

[4] J. Nazario, "Ddos attack evolution," *Network Security*, vol. 2008, no. 7, pp. 7–10, 2008.

[5] ——, "Politically motivated denial of service attacks," *The Virtual Battlefield: Perspectives on Cyber Warfare*, pp. 163–181, 2009.

[6] D. Dittrich, J. Mirkovic, P. Reiher, and S. Dietrich, *Internet Denial of Service: Attack and Defense Mechanisms*. oearson Education, 2004.

[7] S. M. Specht and R. B. Lee, "Distributed denial of service: Taxonomies of attacks, tools, and countermeasures," in *Proceedings of the 17th International Conference on Parallel and Distributed Computing Systems*, 2004, pp. 543–550.

[8] J. Mirkovic and P. Reiher, "A taxonomy of ddos attack and ddos defense mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.

[9] V. Paxson, "An analysis of using reflectors for distributed denial-of-service attacks," *SIGCOMM Comput. Commun. Rev.*, vol. 31, no. 3, pp. 38–47, Jul. 2001.

[10] A. Yaar, A. Perrig, and D. Song, "Pi: A path identification mechanism to defend against ddos attacks," in *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, ser. SP '03. IEEE Computer Society, 2003.

[11] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for ip traceback," in *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, ser. SIGCOMM '00. ACM, 2000, pp. 295–306.

[12] H. Wang, C. Jin, and K. G. Shin, "Defense against spoofed ip traffic using hop-count filtering," *IEEE/ACM Trans. Netw.*, vol. 15, no. 1, pp. 40–53, Feb. 2007.

[13] Y. Xiang, K. Li, and W. Zhou, "Low-rate ddos attacks detection and traceback by using new information metrics," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 2, pp. 426–437, June 2011.

[14] J. Ioannidis and S. M. Bellovin, "Implementing pushback: Router-based defense against ddos attacks," in *Proceedings of the Network and Distributed System Security Symposium, NDSS 2002, San Diego, California, USA*, 2002.

[15] S. Kandula, D. Katabi, M. Jacob, and A. Berger, "Botz-4-sale: Surviving organized ddos attacks that mimic flash crowds," in *Proceedings of the 2Nd Conference on Symposium on Networked Systems Design & Implementation - Volume 2*, ser. NSDI'05. USENIX Association, 2005, pp. 287–300.

[16] C. Rossow, "Amplification Hell: Revisiting Network Protocols for DDoS Abuse," in *Proceedings of the 2014 Network and Distributed System Security (NDSS) Symposium*, February 2014.

[17] M. Kührer, T. Hupperich, C. Rossow, and T. Holz, "Hell of a Handshake: Abusing TCP for Reflective Amplification DDoS Attacks," in *Proceedings of the 8th USENIX Workshop on Offensive Technologies (WOOT .14)*, August 2014.

[18] C. L. Schuba, I. V. Krsul, M. G. Kuhn, E. H. Spafford, A. Sundaram, and D. Zamboni, "Analysis of a denial of service attack on tcp," in *Security and Privacy, 1997. Proceedings., 1997 IEEE Symposium on.* IEEE, 1997, pp. 208–223.

[19] J. Mölsä, "Mitigating denial of service attacks: A tutorial," *Journal of computer security*, vol. 13, no. 6, pp. 807–837, 2005.

[20] T. Peng, C. Leckie, and K. Ramamohanarao, "Survey of network-based defense mechanisms countering the dos and ddos problems," *ACM Computing Surveys (CSUR)*, vol. 39, no. 1, p. 3, 2007.

[21] P. Ferguson, "Network ingress filtering: Defeating denial of service attacks which employ ip source address spoofing," 2000.

[22] R. Beverly and S. Bauer, "The spoofer project: Inferring the extent of source address filtering on the internet," in *Usenix Sruti*, vol. 5, 2005, pp. 53–59.

[23] L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred, "Statistical approaches to ddos attack detection and response," in *DARPA Information Survivability Conference and Exposition, 2003. Proceedings*, vol. 1. IEEE, 2003, pp. 303–314.

[24] M. Roesch *et al.*, "Snort: Lightweight intrusion detection for networks." in *LISA*, vol. 99, no. 1, 1999, pp. 229–238.

[25] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer networks*, vol. 31, no. 23, pp. 2435–2463, 1999.

[26] C. Morrow and B. Gemberling, "Blackhole route server and tracking traffic on an ip network," 2001.

[27] H. Burch and B. Cheswick, "Tracing anonymous packets to their approximate source." in *LISA*, 2000, pp. 319–327.

[28] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical network support for ip traceback," in *ACM SIGCOMM Computer Communication Review*, vol. 30, no. 4. ACM, 2000, pp. 295–306.

[29] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer, "Hash-based ip traceback," in *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 4. ACM, 2001, pp. 3–14.

[30] B. McCarty, "Botnets: Big and bigger," *Security & Privacy, IEEE*, vol. 1, no. 4, pp. 87–90, 2003.

[31] V. L. Thing, M. Sloman, and N. Dulay, "A survey of bots used for distributed denial of service attacks," in *New Approaches for Security, Privacy and Trust in Complex Environments*. Springer, 2007, pp. 229–240.

[32] S. Le Blond, C. Zhang, A. Legout, K. Ross, and W. Dabbous, "I know where you are and what you are sharing: exploiting p2p communications to invade users' privacy," in *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. ACM, 2011, pp. 45–60.

[33] "Hack forums / marketplace / premium sellers section / server stress testing," http://www.hackforums.net/forumdisplay.php?fid=232, Jan. 2015.

[34] J. Krupp, M. Karami, M. Backes, C. Rossow, and D. McCoy, "Attributing ddos amplification attacks," 2016, manuscript submitted for publication.

[35] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, "Inferring internet denial-of-service activity," *ACM Trans. Comput. Syst.*, vol. 24, no. 2, pp. 115–139, May 2006.

[36] E. Wustrow, M. Karir, M. Bailey, F. Jahanian, and G. Huston, "Internet background radiation revisited," in *Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '10. ACM, 2010, pp. 62–74.

[37] M. Bailey, E. Cooke, F. Jahanian, A. Myrick, and S. Sinha, "Practical darknet measurement," in *Information Sciences and Systems, 2006 40th Annual Conference on*. IEEE, 2006, pp. 1496–1501.

[38] Z. M. Mao, V. Sekar, O. Spatscheck, J. Van Der Merwe, and R. Vasudevan, "Analyzing large ddos attacks using multiple data sources," in *Proceedings of the 2006 SIGCOMM workshop on Large-scale attack defense*. ACM, 2006, pp. 161–168.

[39] M. Kührer, T. Hupperich, C. Rossow, and T. Holz, "Exit from hell? reducing the impact of amplification ddos attacks," in *Proceedings of the 23rd USENIX Conference on Security Symposium*, ser. SEC'14. USENIX Association, 2014, pp. 111–125.

[40] J. Czyz, M. Kallitsis, M. Gharaibeh, C. Papadopoulos, M. Bailey, and M. Karir, "Taming the 800 pound gorilla: The rise and decline of ntp ddos attacks," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, ser. IMC '14. ACM, 2014, pp. 435–448.

[41] L. Rudman, "Analysis of ntp based amplification ddos attacks," 2014.

[42] D. C. MacFarland, C. A. Shue, and A. J. Kalafut, "Characterizing optimal dns amplification attacks and effective mitigation."

[43] R. van Rijswijk-Deij, A. Sperotto, and A. Pras, "Dnssec and its potential for ddos attacks: A comprehensive measurement study," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, ser. IMC '14. New York, NY, USA: ACM, 2014, pp. 449–460. [Online]. Available: http://doi.acm.org/10.1145/2663716.2663731

[44] A. Büscher and T. Holz, "Tracking ddos attacks: Insights into the business of disrupting the web," in *Proceedings of the 5th USENIX Conference on Large-Scale Exploits and Emergent Threats*, ser. LEET'12. Berkeley, CA, USA: USENIX Association, 2012, pp. 8–8. [Online]. Available: http://dl.acm.org/citation.cfm?id=2228340.2228351

[45] E. Cooke, F. Jahanian, and D. McPherson, "The zombie roundup: Understanding, detecting, and disrupting botnets," in *Proceedings of the Steps to Reducing Unwanted Traffic on the Internet on Steps to Reducing Unwanted Traffic on the Internet Workshop*, ser. SRUTI'05. USENIX Association, 2005.

[46] A. Welzel, C. Rossow, and H. Bos, "On Measuring the Impact of DDoS Botnets," in *Proceedings of the 7th European Workshop on Systems Security (EuroSec 2014)*, April 2014.

[47] F. C. Freiling, T. Holz, and G. Wicherski, *Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks.* Springer, 2005.

[48] E. Alomari, S. Manickam, B. Gupta, S. Karuppayah, and R. Alfaris, "Botnet-based distributed denial of service (ddos) attacks on web servers: classification and art," *arXiv preprint arXiv:1208.0403*, 2012.

[49] J. Nazario, "Blackenergy ddos bot analysis," *Arbor*, 2007.

[50] R. Gummadi, H. Balakrishnan, P. Maniatis, and S. Ratnasamy, "Not-a-bot: Improving service availability in the face of botnet attacks." in *NSDI*, vol. 9, 2009, pp. 307–320.

[51] C. Y. Cho, J. Caballero, C. Grier, V. Paxson, and D. Song, "Insights from the inside: A view of botnet management from infiltration," in *USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2010.

[52] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna, "Your botnet is my botnet: analysis of a botnet takeover," in *Proceedings of the 16th ACM conference on Computer and communications security*. ACM, 2009, pp. 635–647.

[53] P. Lin, "Anatomy of the mega-d takedown," *Network Security*, vol. 2009, no. 12, pp. 4–7, 2009.

[54] M. N. CENTER, "Cracking down on botnets," http://blogs.microsoft.com/blog/2010/02/24/cracking-down-on-botnets/, March 2010.

[55] B. Stone-Gross, T. Holz, G. Stringhini, and G. Vigna, "The underground economy of spam: A botmasters perspective of coordinating large-scale spam campaigns," in *USENIX workshop on large-scale exploits and emergent threats (LEET)*, 2011.

[56] D. Anselmi, R. Boscovich, T. Campana, S. Doerr, M. Lauricella, O. Petrovsky, T. Saade, and H. Stewart, "Battling the rustock threat," *Microsoft Security Intelligence Report, Special edn.(January 2010 through May 2011)*, 2010.

[57] Y. Nadji, M. Antonakakis, R. Perdisci, D. Dagon, and W. Lee, "Beheading hydras: performing effective botnet takedowns," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 121–132.

[58] J. J. Santanna and A. Sperotto, "Characterizing and mitigating the ddos-as-a-service phenomenon," in *Monitoring and Securing Virtualized Networks and Services*. Springer, 2014, pp. 74–78.

[59] M. Karami and D. McCoy, "Understanding the emerging threat of ddos-as-a-service," in *Presented as part of the 6th USENIX Workshop on Large-Scale Exploits and Emergent Threats*. Berkeley, CA: USENIX, 2013. [Online]. Available: https://www.usenix.org/conference/leet13/workshop-program/presentation/Karami

[60] K. Levchenko, A. Pitsillidis, N. Chachra, B. Enright, M. Félegyházi, C. Grier, T. Halvorson, C. Kanich, C. Kreibich, H. Liu, D. McCoy, N. Weaver, V. Paxson, G. M. Voelker, and S. Savage, "Click Trajectories: End-to-End Analysis of the Spam Value Chain," in *Proceedings of the IEEE Symposium and Security and Privacy*, May 2011, pp. 431–446.

[61] D. McCoy, A. Pitsillidis, G. Jordan, N. Weaver, C. Kreibich, B. Krebs, G. M. Voelker, S. Savage, and K. Levchenko, "Pharmaleaks: Understanding the business of online pharmaceutical affiliate programs," in *Proceedings of the 21st USENIX conference on Security symposium*, 2012.

[62] D. Y. Wang, M. Der, M. Karami, L. Saul, D. McCoy, S. Savage, and G. M. Voelker, "Search + seizure: The effectiveness of interventions on seo campaigns," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, ser. IMC '14, 2014, pp. 359–372.

[63] M. Karami, S. Ghaemi, and D. Mccoy, "Folex: An analysis of an herbal and counterfeit luxury goods affiliate program," in *eCrime Researchers Summit (eCRS), 2013*. IEEE, 2013, pp. 1–9.

[64] D. Y. Huang, H. Dharmdasani, S. Meiklejohn, V. Dave, C. Grier, D. McCoy, S. Savage, N. Weaver, A. C. Snoeren, and K. Levchenko, "Botcoin: monetizing stolen cycles," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2014.

[65] B. Stone-Gross, R. Abman, R. Kemmerer, C. Kruegel, D. Steigerwald, and G. Vigna, "The underground economy of fake antivirus software," in *Economics of Information Security and Privacy III*. Springer, 2013, pp. 55–78.

[66] B. Stone-Gross, T. Holz, G. Stringhini, and G. Vigna, "The underground economy of spam: A botmaster's perspective of coordinating large-scale spam campaigns," in *Proceedings of the 4th USENIX Conference on Large-scale Exploits and Emergent Threats*, ser. LEET'11, 2011.

[67] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage, "Spamalytics: An empirical analysis of spam marketing conversion," in *Proceedings of the 15th ACM conference on Computer and communications security*, 2008.

[68] J. Caballero, C. Grier, C. Kreibich, and V. Paxson, "Measuring pay-per-install: The commoditization of malware distribution." in *USENIX Security Symposium*, 2011.

[69] K. Thomas, D. McCoy, C. Grier, A. Kolcz, and V. Paxson, "Trafficking fraudulent accounts: The role of the underground market in twitter spam and abuse," in *Proceedings of the 22nd Usenix Security Symposium*, 2013.

[70] R. Bohme and T. Moore, "Challenges in empirical security research," Tech. rep., Singapoore Management University, Tech. Rep., 2012.

[71] B. Krebs, "The obscurest epoch is today," March 2013. [Online]. Available: http://krebsonsecurity.com/2013/03/the-obscurest-epoch-is-today/

[72] S. Gallagher, "Details on the denial of service attack that targeted ars technica." March 2013. [Online]. Available: http://arstechnica.com/security/2013/03/details-on-the-denial-of-service-attack-that-targeted-ars-technica/

[73] M. M. Andrade and N. Vlajic, "Dirt jumper: A key player in today's botnet-for-ddos market," in *Internet Security (WorldCIS), 2012 World Congress on*. IEEE, 2012, pp. 239–244.

[74] "Ip2proxy ip-country database," http://www.ip2location.com/databases/ip2proxy.

[75] C. Kanich, N. Chachra, D. McCoy, C. Grier, D. Y. Wang, M. Motoyama, K. Levchenko, S. Savage, and G. M. Voelker, "No plan survives contact: Experience with cybercrime measurement." in *CSET*, 2011.

[76] Z. Durumeric, M. Bailey, and J. A. Halderman, "An internet-wide view of internet-wide scanning," in *USENIX Security Symposium*, 2014.

[77] C. Satten, "Lossless gigabit remote packet capture with linux," http://staff.washington.edu/corey/gulp/, 2008.

[78] J. J. Chromik, "Booters (black)list," February 2015. [Online]. Available: http://essay.utwente.nl/66780/

[79] W. M. Eddy, "Tcp syn flooding attacks and common mitigations," 2007.

[80] E. Cambiaso, G. Papaleo, G. Chiola, and M. Aiello, "Slow dos attacks: definition and categorisation," *International Journal of Trust Management in Computing and Communications*, vol. 1, no. 3, pp. 300–319, 2013.

[81] R. Hansen, "slowloris http dos," http://ha.ckers.org/slowloris/, June 2009.

[82] Z. Durumeric, E. Wustrow, and J. A. Halderman, "Zmap: Fast internet-wide scanning and its security applications." in *USENIX Security*. Citeseer, 2013, pp. 605–620.

[83] R. D. Graham, "Masscan: Mass ip port scanner." https://github.com/robertdavidgraham/masscan, Feburary 2013.

[84] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, and K. Gibbs, "iperf: Testing the limits of your network," 2003.

[85] D. Wieers, "Dstat: Versatile resource statistics tool."

[86] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 4, pp. 2046–2069, 2013.

[87] H. Takabi, J. B. Joshi, and G.-J. Ahn, "Security and privacy challenges in cloud computing environments," *IEEE Security & Privacy*, no. 6, pp. 24–31, 2010.

[88] S. Subashini and V. Kavitha, "A survey on security issues in service delivery models of cloud computing," *Journal of network and computer applications*, vol. 34, no. 1, pp. 1–11, 2011.

[89] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[90] M. D. Ryan, "Cloud computing security: The scientific challenge, and a survey of solutions," *Journal of Systems and Software*, vol. 86, no. 9, pp. 2263–2268, 2013.

[91] J. Idziorek and M. Tannian, "Exploiting cloud utility models for profit and ruin," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*. IEEE, 2011, pp. 33–40.

[92] J. Idziorek, M. F. Tannian, and D. Jacobson, "The insecurity of cloud utility models," *IT Professional*, no. 2, pp. 22–27, 2013.

[93] A. K. Sood and R. J. Enbody, "Crimeware-as-a-servicea survey of commoditized crimeware in the underground market," *International Journal of Critical Infrastructure Protection*, vol. 6, no. 1, pp. 28–38, 2013.

[94] S. Wen, W. Jia, W. Zhou, W. Zhou, and C. Xu, "Cald: Surviving various application-layer ddos attacks that mimic flash crowd," in *Network and System Security (NSS), 2010 4th International Conference on*. IEEE, 2010, pp. 247–254.

[95] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: Characterization and implications for cdns and web sites," in *Proceedings of the 11th international conference on World Wide Web*. ACM, 2002, pp. 293–304.

[96] H. Beitollahi and G. Deconinck, "Tackling application-layer ddos attacks," *Procedia Computer Science*, vol. 10, pp. 432–441, 2012.

[97] M. Karami and D. McCoy, "Understanding the emerging threat of ddos-as-a-service," in *Proceedings of the USENIX Workshop on Large-Scale Exploits and Emergent Threats*, 2013.

[98] M. Karami, Y. Park, and D. McCoy, "Stress testing the booters: Understanding and undermining the business of ddos services," in *Proceedings of the World Wide Web Conference (WWW)*, 2016.

[99] E. Alomari, S. Manickam, B. Gupta, S. Karuppayah, and R. Alfaris, "Botnet-based distributed denial of service (ddos) attacks on web servers: classification and art," *arXiv preprint arXiv:1208.0403*, 2012.

[100] V. L. Thing, M. Sloman, and N. Dulay, "A survey of bots used for distributed denial of service attacks," in *New Approaches for Security, Privacy and Trust in Complex Environments*. Springer, 2007, pp. 229–240.

[101] "Amazon ec2 pricing," https://aws.amazon.com/ec2/pricing/, 2016.

[102] "Http archive," http://httparchive.org/interesting.php?a=All&l=Jan%2015%202016.

[103] "Alexa," http://www.alexa.com/.

[104] G. Oikonomou and J. Mirkovic, "Modeling human behavior for defense against flash-crowd attacks," in *Communications, 2009. ICC'09. IEEE International Conference on*. IEEE, 2009, pp. 1–6.

[105] S. H. Khor and A. Nakao, "spow: On-demand cloud-based eddos mitigation mechanism," in *HotDep (Fifth Workshop on Hot Topics in System Dependability)*, 2009.

[106] M. Naresh Kumar, P. Sujatha, V. Kalva, R. Nagori, A. K. Katukojwala, and M. Kumar, "Mitigating economic denial of sustainability (edos) in cloud computing using in-cloud scrubber service," in *Computational Intelligence and Communication Networks (CICN), 2012 Fourth International Conference on*. IEEE, 2012, pp. 535–539.

[107] M. H. Sqalli, F. Al-Haidari, and K. Salah, "Edos-shield-a two-steps mitigation technique against edos attacks in cloud computing," in *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*. IEEE, 2011, pp. 49–56.

[108] E. Bursztein, S. Bethard, C. Fabry, J. C. Mitchell, and D. Jurafsky, "How good are humans at solving captchas? a large scale evaluation," in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 399–413.

[109] E. Bursztein, M. Martin, and J. Mitchell, "Text-based captcha strengths and weaknesses," in *Proceedings of the 18th ACM conference on Computer and communications security*. ACM, 2011, pp. 125–138.

[110] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage, "Re: Captchas-understanding captcha-solving services in an economic context."

[111] J. Idziorek, M. Tannian, and D. Jacobson, "Detecting fraudulent use of cloud resources," in *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*. ACM, 2011, pp. 61–72.

[112] ——, "Attribution of fraudulent resource consumption in the cloud," in *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*. IEEE, 2012, pp. 99–106.

[113] S.-Z. Yu and H. Kobayashi, "An efficient forward-backward algorithm for an explicit-duration hidden markov model," *Signal Processing Letters, IEEE*, vol. 10, no. 1, pp. 11–14, 2003.

[114] S.-Z. Yu, "Hidden semi-markov models," *Artificial Intelligence*, vol. 174, no. 2, pp. 215–243, 2010.

[115] P. Eckersley, "How unique is your web browser?" in *Privacy Enhancing Technologies*. Springer, 2010, pp. 1–18.

[116] "Open resolver projec," http://OpenResolverProject.org/.

[117] "Open ntp scanning project," http://OpenNTPProject.org/.

[118] Z. Durumeric, J. Kasten, D. Adrian, J. A. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer, and V. Paxson, "The matter of heartbleed," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, ser. IMC '14, 2014, pp. 475–488.

[119] B. Sieklik, R. Macfarlane, and W. J. Buchanan, "Evaluation of tftp ddos amplification attack," *Computers & Security*, vol. 57, pp. 67–92, 2016.

Curriculum Vitae

Mohammad Karami started the PhD in Information Technology program at George Mason University in Spring 2012. His current research mainly focuses on empirical measurement of financially-motivated cybercrime. Karami received his MS degree in Information Technology with concentration on Electronic Commerce from Iran University of Science and Technology in 2009 and his BS degree in Software Engineering from Applied University of Science and Technology, Arak, Iran, in 2006.