Title
Scalable Real-Time DDoS Traffic Monitoring and Characterization

Author
Huyn, Joojay

Publication Date
2018

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Scalable Real-Time DDoS Traffic Monitoring and Characterization

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Computer Science

by

Joojay Huyn

2018

ABSTRACT OF THE THESIS

Scalable Real-Time DDoS Traffic
Monitoring and Characterization

by

Joojay Huyn
Master of Science in Computer Science
University of California, Los Angeles, 2018
Professor Songwu Lu, Co-Chair
Professor Peter L. Reiher, Co-Chair

High volume DDoS attacks continue to cause serious financial losses and damage to company reputations, despite years of research in preventing and mitigating them. Many proposed techniques for handling these attacks assume that the attack has already been detected and its traffic properly characterized; yet, existing methods of detecting and characterizing such attacks have not been widely adopted, for various reasons. We describe a scalable real-time DDoS monitoring system that leverages modern big data technologies to effectively analyze high volume DDoS attacks. Evaluated on multiple large-scale traffic datasets that capture recent real-world DDoS attacks and synthetic traffic based on sophisticated attack characteristics, our approach detects and characterizes these attacks quickly and accurately. Furthermore, we show that our monitoring system 1) clearly justifies its decisions resulting from explainable analysis of input traffic volume metrics, thus increasing monitoring transparency and facilitating the diagnosis and debugging of monitoring performance for network security teams 2) leverages identified attack characteristics to separate benign from malicious traffic and send helpful defense recommendations, the identified attack characteristics and malicious traffic traces, to downstream DDoS traffic filtering systems.

The thesis of Joojay Huyn is approved.

Carlo Zaniolo

George Varghese

Peter L. Reiher, Committee Co-Chair

Songwu Lu, Committee Co-Chair

University of California, Los Angeles

2018

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This research is the result of funding provided by the Science and Technology Directorate of the United States Department of Homeland Security under contract number D15PC00204. The views and conclusions contained herein are those of the authors and should not be interpreted necessarily representing the official policies or endorsements, either expressed or implied, of the Department of Homeland Security or the US Government.

CHAPTER 1

Introduction

DDoS attacks that generate unmanageably large volumes of traffic directed at the victim continue to cause serious problems. The Mirai botnet performed such an attack on October 21, 2016, resulting in a widespread Internet outage throughout America and Europe [1, 2]. Perhaps the most serious of the many DDoS attacks launched since 1999 [3], this volumetric DDoS attack generated malicious traffic at a jaw-dropping 1.2 terabits per second [1], successfully taking down major websites such as Twitter, Netflix, Paypal, and CNN [2]. In network-level or transport-level volumetric DDoS attacks, a flood of traffic overwhelms the victim server's bandwidth, router processing capacity, or other network resources, thus rendering the victim unresponsive to legitimate users [4]. Figure 1.1 illustrates a volumetric DDoS attack targeting a victim server (labeled "Victim") in a hypothetical company A's enterprise network.

Because such attacks disrupt online services, damage reputations, strain customer relationships, and wreak tremendous financial losses for organizations and companies [3], researchers have proposed many defense systems for volumetric DDoS attacks to mitigate and filter malicious DDoS traffic [5–12]. Despite these defense approaches, volumetric DDoS attacks continue to increase in severity, growing in duration and average peak traffic sizes over the last several years [13–18], as demonstrated by the Mirai botnet attacks. Several factors contribute to this continued growth:

1. Volumetric DDoS attacks are increasing in sophistication
2. Current DDoS defense approaches fail at sufficiently high attack volumes
3. Monitoring, detecting, and characterizing volumetric DDoS attack traffic is challenging,

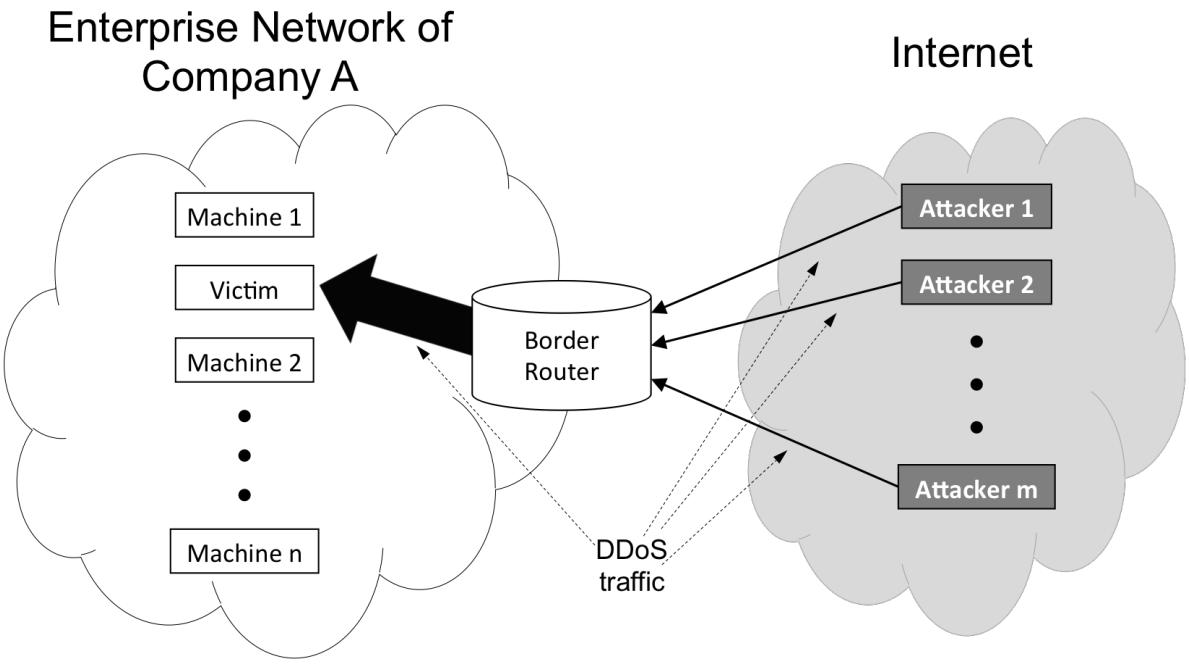

Figure 1.1: Volumetric DDoS Attack.

as is separating benign from malicious attack traffic

4. Existing DDoS monitoring solutions are a black box that do not clearly explain why an attack was or was not detected nor thoroughly characterize a detected attack, thus decreasing transparency for downstream intervention

Typically, DDoS defense approaches include a monitoring system that analyzes network traffic and sends the results in real-time to a traffic filtering system, as illustrated in Figure 1.2. Thus, DDoS monitoring systems play a significant role in successfully defending against volumetric DDoS attacks. To construct an effective volumetric DDoS monitoring system, we have identified important objectives in Table 1.1. Many works regarding such systems do not thoroughly address all the objectives mentioned in Table 1.1 simultaneously. Other DDoS protection services offered by commercial companies advertise to detect DDoS attacks and filter malicious traffic in a scalable and real-time manner but do not discuss in detail the technical implementation of their advertised DDoS monitoring services [19–27]. Thus, we will not reference these commercial services in the related work chapter. Because such monitoring analysis provides necessary input to traffic filtering systems, this paper fo-

cuses on the development of a volumetric DDoS monitoring system intended to complement traffic filtering systems. In fact, the proposed monitoring system of this paper is currently being integrated with DrawBridge, an advanced traffic filtering system [12].

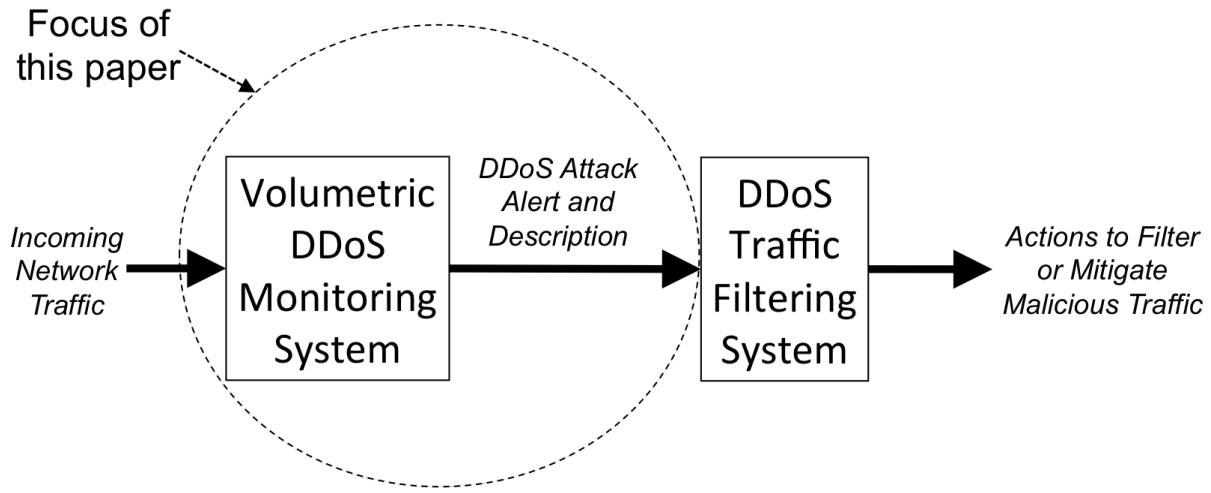

Figure 1.2: Two Essential Components of a DDoS Defense System.

We propose an effective volumetric DDoS monitoring system that leverages data mining and modern big data technologies to:

- Analyze high volume and high velocity network traffic offline and in real-time
- Optimize the volumetric DDoS attack detection and characterization model parameters from large-scale network traffic traces offline
- Accurately detect and characterize attacks from large-scale network traffic traces in real-time to provide defense recommendations, the identified attack characteristics and malicious traffic traces, to downstream traffic filtering systems
- Clearly explain how the system arrived at detection and characterization decisions resulting from network traffic analysis

To expand on explainability, we argue that a volumetric DDoS monitoring system that can explain its decisions not only increases transparency of network traffic behavior from a network operator's perspective but also offers helpful insights that operators can leverage to

Table 1.1: Important Objectives For an Effective Volumetric DDoS Monitoring System

Objective	**Importance of Objective**
Provide timely and precise information for attack intervention	Accurately detecting a volumetric DDoS attack in real-time is the first goal of defense. Providing characterizations of detected attacks and defense recommendations to downstream intervention systems helps contain the rapidly growing damage of the attack.
Process big data in real-time	Network traffic data has a high volume and high velocity nature.
Justify detection and characterization decisions	Clearly explaining decisions resulting from analysis of traffic events empowers network security teams to take more effective actions.
Enable continuous improvement of volumetric DDoS attack detection and characterization models	Attacks constantly evolve, so only continuous improvement of detection and characterization models helps to effectively keep up with them.

take further actions in assessing the overall state of the network and mitigating DDoS attacks. For example, our proposed monitoring system applies a set of tests that examines easily understandable and interpretable network traffic volume statistics (taking time-dimension into account as well) of traffic destined for the monitored servers. Upon detection and characterization of a volumetric DDoS attack, our system shows which tests failed to explain how it reached various conclusions, such as time of detected attack, relevant attack traffic protocol, targeted victim servers, and attack sources. Hence, network operators not only know why and how our proposed system arrived at such final decisions but can also leverage specifics of this explanation to take further interventions and conduct better directed investigations in mitigating the attack and diagnosing possible causes of the attack respectively. In case the monitoring system has raised a false alarm, network operators can review the system's explanation of the attack alert and characterizations. If the provided explanation does not seem convincing, network operators may override the monitoring system's alert to avoid a false alarm. As discussed in the next chapter, other proposed volumetric DDoS monitoring systems mostly provide detection alerts, with some generating very limited characterizations and explanations of the detected attack.

From a big data perspective, volumetric DDoS monitoring systems require modern big data technologies to process and analyze internet-scale network traffic offline and in real-time, as discussed in Chapter 4. The computational resources required to process traffic data of such scale can easily overwhelm single machines, including large powerful ones, but modern big data technologies only require commodity hardware and software resources to process internet-scale network traffic in a fault tolerant, distributed, horizontally scalable, and parallelized manner. Furthermore, many modern big data technologies seamlessly interface with many other useful systems (e.g. data storage and warehousing systems, databases, data visualization dashboards, high-speed message buses, streaming platforms) and come packaged with advanced data mining tools that leverage and combine a wide variety of powerful artificial intelligence, machine learning, and statistical techniques to quickly assemble sophisticated algorithms for detecting, analyzing, and explaining anomalous and rare events hidden in large-scale data, such as volumetric DDoS attacks and other cyber attacks. We

show how clearly structured data mining techniques combined with necessary big data technologies create a scalable real-time volumetric DDoS monitoring system that is effective and explainable.

This paper is organized as follows. Chapter 2 surveys the state-of-the-art volumetric DDoS monitoring systems. Chapter 3 describes the volumetric DDoS attack detection and characterization model used in our proposed monitoring system, and Chapter 4 discusses how to leverage big data technologies to process large-scale network traffic data. Chapter 5 presents our evaluation results, and in Chapter 6, we discuss future research and conclude.

CHAPTER 2

Related Work

Many works regarding volumetric DDoS monitoring systems do not thoroughly address all of the following important aspects simultaneously:

1. Evaluation of detection models on recent real-world volumetric DDoS attacks
2. Characterization of attacks upon detection to send defense recommendations, the identified attack characteristics and malicious traffic traces, to downstream intervention systems
3. Real-time and offline analysis of large-scale traces
4. Horizontal scalability of monitoring systems for big data analytics
5. Ability of the monitoring system to explain how it arrived at detection and characterization decisions

[28, 30, 34, 35, 37, 38] evaluate the accuracy of their detection models on synthesized attack traces, and [32] does not clearly explain the nature of the traces used for evaluation. [29, 31, 33, 36] evaluate the accuracy of their detection models on real world DDoS attack traces; however, these attack traces (except the ones in [33]) refer to well-studied, highly characterized, and old attack datasets: CAIDA DDoS Attack 2007 Dataset [39], 1999 and 2000 DARPA DDoS Attack Datasets [40]. [33] does not clearly describe or provide references to their 2005 DDoS attack dataset gathered from a large tier-1 ISP. Furthermore, these old attack datasets may not have important characteristics present in modern volumetric DDoS attacks.

[28, 30, 36, 38] propose algorithms that focus on mere attack detection, while [29, 31–33, 37] propose algorithms with very limited attack characterization abilities beyond detection, mostly victim identification ([37]'s algorithm focuses on characterizing traffic events as high rate attacks, low rate attacks, or flash crowds). [35] labels packets as malicious but does not clearly discuss how their detection modules return specific attack characterizations to label packets. Detection without thorough characterization of DDoS attacks makes it difficult to take effective interventions and mitigate attack traffic.

[28–31,34,35,37] present algorithms that can perform in real-time, but these works mostly focus on developing a computationally efficient algorithm on a single node, which may have optimized hardware. They do not address leveraging modern big data technologies to process large-scale network traffic data (in which the volume would overwhelm a single machine) in real-time and in parallel for DDoS detection. [32,33,36,38] did not implement their algorithm to run in real-time. Overall, few works address how to leverage modern big data technologies to construct DDoS attack detection and characterization models offline from and analyze large-scale network traffic data in real-time. [34] leverages the Hadoop ecosystem, a popular open source implementation of the MapReduce framework proposed by Google [41, 42], to parallelize model execution in near "real-time" on a computer cluster, and [32, 34] leverage Hadoop to build detection models from large-scale network traffic data offline. However, the Hadoop ecosystem was designed as a batch processing system, not a real-time streaming system [41–43]. Furthermore, Hadoop's MapReduce framework makes it extremely difficult to implement data mining algorithms.

Overall, [28–33, 35–38] present detection algorithms that cannot easily explain why it detected (or did not detect) an attack for various reasons, which include:

- Proposing approaches that derive many statistical metrics from traffic traces, where such metrics are not immediately understandable nor identify specific areas of the network to investigate
- Feeding many statistical metrics into a black-box algorithm, a chain of complex procedural algorithms, or statistical models that do not easily explain how they arrived

at a detection decision nor identify which input metrics influenced the decision

Of the above related works, we further analyze three recently proposed volumetric DDoS detection systems [29, 32, 34] that seem interesting or most similar to ours. [29] proposes to leverage software-defined networking (SDN) and the OpenFlow protocol to implement a complete system, capable of both volumetric DDoS attack detection and mitigation. Upon detecting the attack and identifying the victim (through an anomaly detection algorithm that utilizes a bidirectional count sketch algorithm), this system exploits network programmability that OpenFlow offers to enhance Remote-Triggered Black-Hole (RTBH) filtering for mitigating the attack. While some may view this system's completeness (i.e. capable of both detection and mitigation tasks) as a strength, others may argue that this completeness could lead to inflexibility when deciding upon a technology stack for volumetric DDoS monitoring and defense. Although [29] briefly discusses their system's modular architecture, they do not explicitly address how network operators may easily switch out their proposed defense system in favor of other preferred defense systems, while still using their proposed detection system in [29]. Furthermore, [29] does not clearly explain their system's scalability properties with respect to the OpenFlow controller when processing large-scale network traffic from many attached OpenFlow-enabled devices. While these devices leverage packet sampling techniques to avoid possible resource depletion of the controller during high traffic rates, the OpenFlow controller may still become overwhelmed when it has to request flow table statistics from a large number of attached devices while separating benign from malicious traffic through many distinct controller applications.

Both [32, 34] leverage the Hadoop ecosystem to implement a scalable volumetric DDoS detection system that can process large-scale network traffic traces. More specifically, both works implicitly or explicitly mention that they leverage Hadoop to mine and extract, from a large historical dataset of network traffic traces, parameters required by their proposed detection algorithms. After obtaining these necessary parameters, both show that their Hadoop implementation of a volumetric DDoS detection system can horizontally scale in order to detect a volumetric DDoS attack given large volumes of incoming network traffic traces, as illustrated through the results of their scalability experiments. However, as

previously mentioned in this chapter, the Hadoop ecosystem was designed as a batch processing system, not a real-time streaming system [41–43]. Hence, volumetric DDoS detection systems leveraging Hadoop may struggle to achieve real-time analysis on the magnitude of seconds and lower. Furthermore, both implement a simple detection algorithm consisting of a short sequence of steps or computations, the counter-based algorithm, using Hadoop. However, when network operators elect to use more sophisticated and powerful detection algorithms, this may affect usability, maintainability, and speed of the detection system. As mentioned previously in this chapter, the interface offered by the Hadoop MapReduce framework makes it difficult to implement more sophisticated data mining algorithms requiring many steps or computations. Internally, after each computation of the sequence, Hadoop writes the resulting output to disk; then, the next computation in the sequence reads as input this previous output from disk [41–43]. While a few disk reads and writes may not have a large effect on speed, a long sequence of computations requiring many disk reads and writes will significantly impact the system's ability to execute real-time data processing and analytics. In contrast, our proposed scalable real-time volumetric DDoS monitoring system leverages Apache Spark [44, 45], a modern big data processing system that can not only execute entire sequences of computations in memory to achieve real-time analytics [41, 43, 44, 46] but also compress and optimize long sequences of computations into much shorter (but still conceptually equivalent) sequences of computations [46, 47].

Furthermore, all the above works do not address the importance of diagnosing and debugging a volumetric DDoS monitoring system. If the monitoring system does not detect and characterize an attack well, which aspects of the monitoring system require improvement? Which input metrics require adjustment? Does each metric provide clear enough semantics to identify specific areas of the network to investigate? How exactly does each metric and the internal components of the model influence the final detection and characterization decision and identification of benign and malicious traces? Due to the limited accessibility to real-world DDoS attack datasets, lack of gold standard guidelines for synthesizing sophisticated volumetric DDoS attacks, and constantly evolving attacks, it becomes difficult to determine whether a monitoring system that performs well on a small collection of real or synthesized

DDoS attack traces will actually yield the same performance against modern volumetric DDoS attacks in the wild. Hence, monitoring systems deployed in defense systems must inevitably undergo constant diagnosis and debugging to maintain a strong performance against attacks in the wild. Our work presents an accurate and explainable volumetric DDoS monitoring system that facilitates diagnosis and debugging operations. Furthermore, we show how to leverage modern big data technologies to create a horizontally scalable and real-time monitoring system that sends defense recommendations, the identified attack characteristics and malicious traffic traces, to downstream traffic filtering systems.

CHAPTER 3

DDoS Detection and Characterization

The proposed volumetric DDoS monitoring system includes the following components:

1. A volumetric DDoS attack detection and characterization model that learns the normal behavior of a network's traffic volume offline
2. A real-time network traffic processing system that computes metrics from incoming network traffic, signals an attack when the model (previously computed offline) detects analyzed metrics indicative of abnormal and high traffic volume behavior deviating from normal behavior, characterizes the attack, and sends defense recommendations (identified attack characteristics and malicious traffic traces) to downstream traffic filtering systems
3. A real-time attack source identification scheme

The following sections describe the conceptual steps behind the volumetric DDoS monitoring system.

3.1 Training Phase: Constructing the DDoS Detection Model

The training phase refers to learning parameters of multiple volumetric DDoS detection models offline from:

1. A large enough training dataset containing enough benign network traces to thoroughly capture normal traffic volume behavior

Table 3.1: Trace Data Schema

Trace Data Field	**Data Type**
trace timestamp	timestamp
source IP address	string
source port	integer
destination IP address	string
destination port	integer
protocol	string
packets	integer
bytes	integer

2. A list of IP addresses corresponding to n server machines to monitor, where network operators have identified these machines as potential targets and requested protection from volumetric DDoS attacks

Each trace in the dataset should contain the following values: trace timestamp, source IP address, source port, destination IP address, destination port, protocol (ICMP, TCP, UDP), packets (indicator of traffic volume), bytes (another indicator of traffic volume). Table 3.1 shows the data type of each field in a trace.

To begin this phase, divide the training dataset into n subsets, where each subset contains traffic traces destined only for a unique monitored server on the list, based on simplifying assumptions that traffic events of different servers are independent of each other. Then, construct n models offline, one model for each monitored server given its corresponding subset. Consider the following steps to learn the parameters of the model for monitored server A given its corresponding subset *traces_dst_A* (which contains only traces destined for A):

1. Because traffic protocols play a core part in the identity and nature of traffic events, divide *traces_dst_A* into 3 traffic protocol groups and analyze each group separately. *ICMP_traces_dst_A* contains only ICMP protocol traffic traces of *traces_dst_A* destined for A. Similar definitions apply to groups *TCP_traces_dst_A* and *UDP_traces_dst_A*.

2. For each traffic protocol group p of *traces_dst_A*, compute the following 4 metrics that approximate normal traffic volume behavior of traces of protocol p destined for A in an easily comprehensible manner:

   - $MaxPkts(A, p)$ = the maximum total # of packets of protocol p that A receives per unit of time u
   - $MaxBytes(A, p)$ = the maximum total # of bytes of protocol p that A receives per unit of time u
   - $MaxPktsFromSrc(A, p)$ = the maximum total # of packets of protocol p that any source IP address sent to A per unit of time u
   - $MaxBytesFromSrc(A, p)$ = the maximum total # of bytes of protocol p that any source IP address sent to A per unit of time u

Our experiments offline suggest that setting $u = 1$ minute is a reasonable choice. After the training phase, each model learns 12 parameters (4 metrics computed for each of the 3 traffic protocol groups) from the initial training dataset. As discussed in the next section, parameters $MaxPkts(A, p)$ and $MaxBytes(A, p)$ serve as normalization constants applied to incoming traffic volumes (in packets and bytes) of traffic protocol p destined for A, and parameters $MaxPktsFromSrc(A, p)$ and $MaxBytesFromSrc(A, p)$ help to identify potential attack sources targeting A with traffic of protocol p. Figure 3.1 shows a conceptual picture of the training phase.

3.2 Real-time DDoS Detection and Characterization

To analyze incoming raw network traffic in real-time, a network traffic processing system, every unit of time u, divides the traffic received during the current unit of time i and the pre-

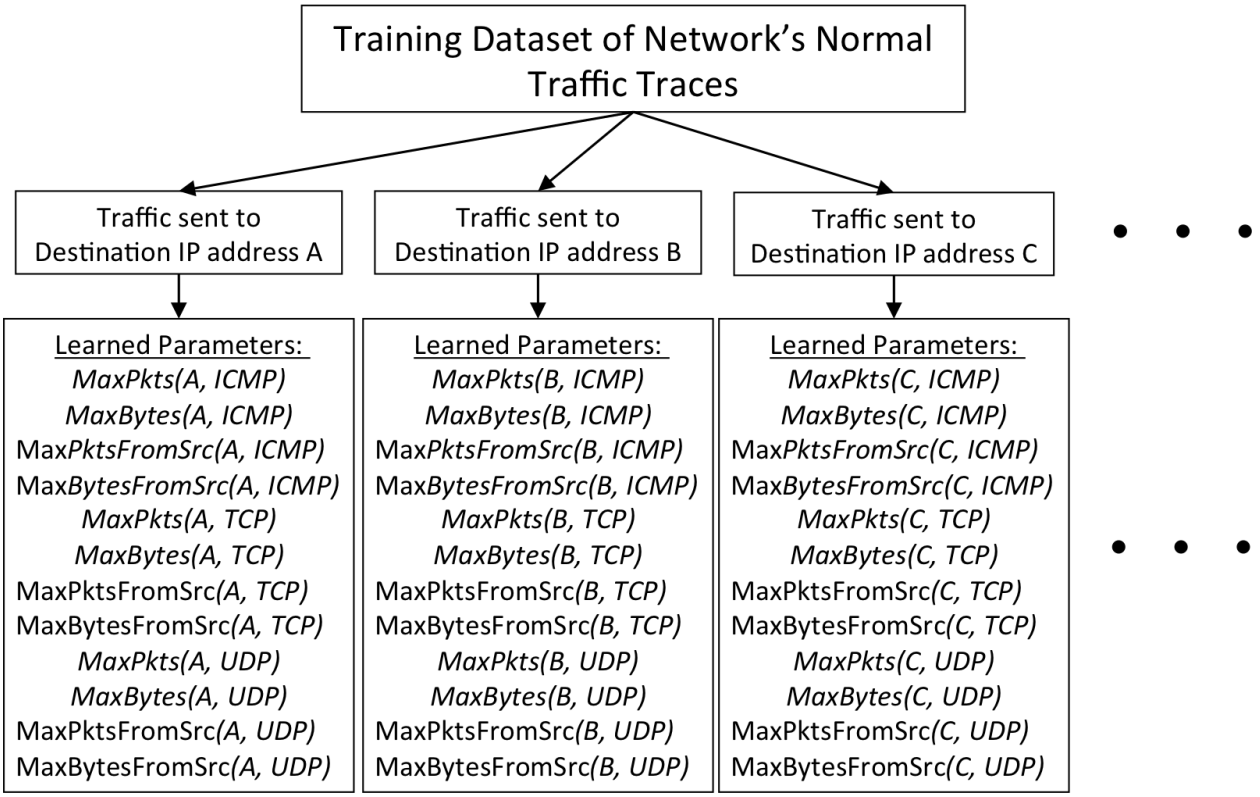

Figure 3.1: Learning Parameters of DDoS Detection Models.

vious $k-1$ units of time into n subsets, where each subset contains traffic traces destined only for a unique monitored server on the list. Consider the subset of traces *current_traces_dst_A* destined for A during this window of k units of time. From *current_traces_dst_A*, the system computes the following 4 metrics for each traffic protocol p:

- $NormalizedPkts(A, p, i) = \#$ of packets of protocol p A receives in minute i / $MaxPkts(A, p)$
- $NormalizedPktsMovingAvg(A, p, i) = \sum_{j=0}^{k-1} NormalizedPkts(A, p, i-j)/k$
- $NormalizedBytes(A, p, i) = \#$ of bytes of protocol p A receives in minute i / $MaxBytes(A, p)$
- $NormalizedBytesMovingAvg(A, p, i) = \sum_{j=0}^{k-1} NormalizedBytes(A, p, i-j)/k$

Note that our experiments offline suggest that:

1. Setting u and i to 1 minute is a reasonable choice, just as it was for the training phase

2. Setting the sliding window size $k = 5$ is a reasonable choice

3. $MaxPkts(A, p)$ and $MaxBytes(A, p)$ refer to parameters computed during the training phase for the model of A

After computing all 12 metrics (4 metrics for each of the 3 traffic protocols) from *current_traces_dst_A* in real-time, we apply Decision Rule 1 to these 12 metrics to determine whether A is under attack at minute i from traffic of protocol p, where Decision Rule 1 analyzes 4 metrics of the same protocol at a time from these 12 metrics.

Decision Rule 1 DDoS Attack Detection Rule

---

if ($NormalizedPkts(A, p, i) > \theta_1$ **and**

$NormalizedPktsMovingAvg(A, p, i) > \theta_2$) **or**

($NormalizedBytes(A, p, i) > \theta_1$ **and**

$NormalizedBytesMovingAvg(A, p, i) > \theta_2$)

then

Signal that a volumetric DDoS attack with traffic

of protocol p targeting A has occurred at minute i

Overall, the system computes a set of 12 metrics for each monitored server A, B, C, ... and applies Decision Rule 1 to each set. Our experiments offline suggest setting $\theta_1 = \theta_2 = 10$ is a reasonable choice, and these thresholds remain the same for any traffic protocol and any monitored server. Furthermore, the above metrics and decision rule clearly show that our detection model characterizes an attack by providing the time of the attack, victim, and volume statistics and protocol of attack traffic. Figure 3.2 illustrates a conceptual understanding of Decision Rule 1, structured as a decision tree, that the model applies to traffic volumes (in packets and bytes) of protocol p destined for A at minute i.

Upon detecting an attack of traffic protocol p targeting victim server A during minute i, the real-time attack source identification scheme obtains a list of all distinct source IP addresses that sent traffic of protocol p to A during minute i and computes the following metrics for each distinct source IP address S on this list:

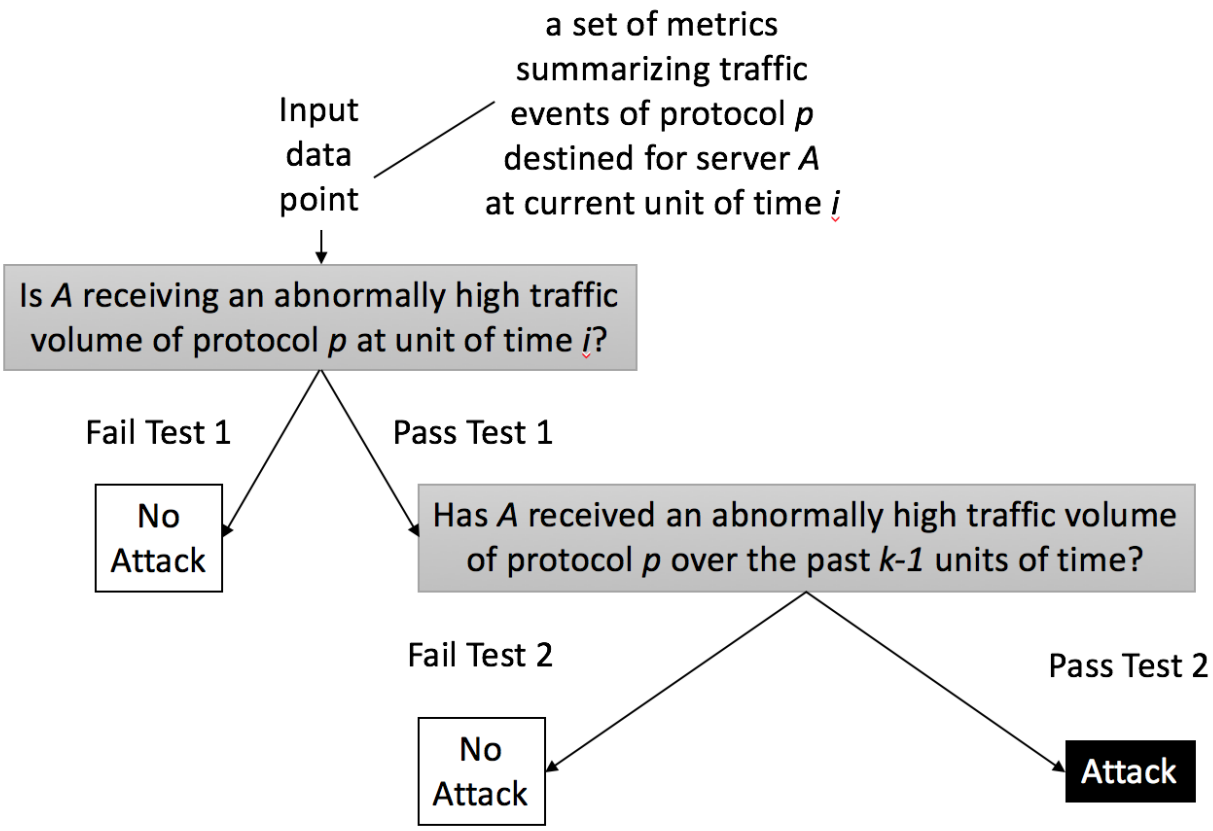

Figure 3.2: Decision Rules Used In DDoS Attack Detection.

- $PktsFromSrc(S, A, p, i) = \#$ of packets of protocol p source S sent to A during minute i

- $BytesFromSrc(S, A, p, i) = \#$ of bytes of protocol p source S sent to A during minute i

To determine whether S should be labeled as an attack source of A, apply Decision Rule 2.

Decision Rule 2 DDoS Attack Source Identification Rule

if $\dfrac{PktsFromSrc(S, A, p, i)}{MaxPktsFromSrc(A, p)} > \theta_3$ **or**
$\dfrac{BytesFromSrc(S, A, p, i)}{MaxBytesFromSrc(A, p)} > \theta_3$
then

Label S as an attack source

Note that $MaxPktsFromSrc(A, p)$ and $MaxBytesFromSrc(A, p)$ refer to parameters computed during the training phase for the model of A. Our experiments offline suggest setting $\theta_3 = 10$ is a reasonable choice, and this threshold remains the same for any traffic protocol, monitored server, and source IP. Decision Rule 2 shows how our detection and characterization model determines whether a source IP address S is malicious, further characterizing the attack.

Finally, the system leverages the following identified attack characteristics

- time of attack at minute i
- traffic protocol p of attack
- victim server A
- attacker S

in a simple pattern matching scheme to identify malicious traces. In other words, among the incoming traffic traces of the current minute i, malicious traces have fields that contain the above values. These identified attack characteristics and malicious traces form important defense recommendations that the system sends in real-time to downstream traffic filtering systems.

Note that the number of computations for this identification scheme is maximally bounded by the number of traffic traces received during minute i, an arguably significantly lower number than the possible address space of IPv4 and IPv6 addresses most of the time. For example, if the monitoring system were to receive 10 million traces at minute i, the identification scheme only needs to analyze the traffic volumes sent from at most 10 million distinct sources. While it is possible for the number of distinct sources present in received traces to equal the size of IPv4 and IPv6 address spaces, we argue that 1) most attacks have not reached such volumes 2) such traffic volumes may crash upstream traffic collection systems forwarding traffic to our monitoring system, a problem beyond the scope of this paper. In scenarios where all IP addresses in the address space target the network and upstream traffic collection systems do not crash, we may take a random sample of the incoming traffic traces

and aggregate distinct source IP addresses into IP address blocks based on a desired IP prefix scheme to bound the number of computations to a manageable number.

Finally, many DDoS attacks use IP spoofing to conceal their true sources and otherwise confuse defenders, but in many other cases these attacks do not use IP spoofing. The current system specifically targets non-spoofed attacks, with continuing research on addressing spoofed attacks.

3.3 Explainability of DDoS Detection and Characterization Model

The parameters computed during the training phase and metrics computing during real-time analysis of incoming traffic clearly summarize the behavior of traffic volume, empowering network security teams to understand the exact semantics of these parameters and metrics. To enable the monitoring system to explain its detection and characterization decisions, we designed Decision Rule 1 to follow the paradigm of a simple decision tree structure, as illustrated in Figure 3.2, which contains easily comprehensible tests at various levels of the tree for the following reasons:

- Because the definition of a volumetric DDoS attack implies that a victim server must have received an inundation of traffic, Test 1 quickly eliminates monitored servers that have received normal traffic volumes of a certain protocol during the current minute i
- Because non-malicious events that send a high volume of traffic of a certain protocol toward a server for an extremely short amount of time may frequently occur, Test 2 at the next level of the decision tree filters out such noisy events to decrease the false alarm rate of the monitoring system. We argue that a system consisting of one simple threshold test that examines the received traffic volume only at the current time will result in many false alarms.
- Structuring a sequence of tests in a simple tree structure enables the monitoring system to: 1) clearly explain how it arrived at certain decisions at the tree's leaf nodes 2) show which input metrics at what level of the tree influenced a decision, empowering network

security teams to take more effective actions 3) easily improve by allowing the addition of future tests through adding more branches to the tree at desired levels 4) facilitate diagnosis and debugging of the monitoring system by showing which input metric or which test at what level made the wrong decision and needs further adjustment

For example, suppose the model detected an attack at minute i and characterized this attack as having traffic of protocol p targeting A at minute i. Based on the decision tree, A has received an abnormally high volume of network traffic of protocol type p during minute i, as indicated by metrics $NormalizedPkts(A, p, i)$ or $NormalizedBytes(A, p, i)$ and Test 1. This event at minute i is not a false alarm because this spike in traffic destined for A has consistently remained high over recent history, the past $k-1 = 4$ units of time, as indicated by metrics $NormalizedPktsMovingAvg(A, p, i)$ or $NormalizedBytesMovingAvg(A, p, i)$ and Test 2. This explanation not only helps the monitoring system detect and characterize DDoS attacks but also identifies how certain metrics and tests influenced the final decision, facilitating the inevitable tasks of diagnosing and debugging the monitoring system in cases where it does not perform well against never-before-seen DDoS attacks in the wild. For example, if Test 1 performs poorly, perhaps the input metrics to this test must improve, the threshold θ_1 should increase for more conservative decisions, or another test should be added to this same level to yield more insight.

Upon attack detection, the real-time attack source identification scheme provides additional characterization of the attack, showing how it identifies source IPs as malicious attackers and how metrics $PktsFromSrc(S, A, p, i)$, $MaxPktsFromSrc(A, p)$, $BytesFromSrc(S, A, p, i)$, $MaxBytesFromSrc(A, p)$ influenced this decision in Decision Rule 2, thus facilitating future diagnosis and debugging operations in case the computed metrics or tests of the scheme need to be adjusted to identify attack sources with better accuracy.

CHAPTER 4

Big Data Technologies for Effective DDoS Monitoring

Given large volumes of network traffic, an effective volumetric DDoS monitoring system must be able to perform the following offline tasks with reasonable efficiency:

- Store and quickly retrieve large training and test datasets of traffic traces
- Mine insightful metrics indicating normal and attack traffic from large-scale traffic datasets
- Construct the attack detection and characterization model from the large training dataset
- Tune models by searching the large parameter space for the optimal parameter values (in our case, these parameters include unit of time u and i, window size of recent history k, thresholds θ_1, θ_2, θ_3, and other parameters referenced in Chapter 3)

Table 4.1 displays the sizes of some of the datasets [48, 49, 51] used to construct and evaluate the proposed volumetric DDoS attack detection and characterization model. These datasets represent traffic traces gathered throughout one day. Note that some of the sizes of the RADB_DDOS dataset [51] are estimated because we are currently processing this dataset. These large datasets suggest that single machines do not provide enough storage nor computational power to analyze such large datasets.

However, a large enough computer cluster consisting of many small, cheap, and commodity machines provides enough storage to contain such large datasets and can efficiently parallelize expensive computations to conduct complex offline analysis to construct models in a reasonable amount of time. To achieve this, we suggest using a high performance

Table 4.1: Size of Sample Trace Datasets

Trace Dataset	**Size In Compressed Format**	**Size In CSV Format**	**Corresp. Database Table Size**	**Num. Traces In Database Table**
DDOS_DNS_AMPL	33.5 GB	307 GB	320 GB	3.14 Billion
DDOS_CHARGEN	67.1 GB	628 GB	659 GB	6.47 Billion
RADB_DDOS	448 GB	4 TB est.	4 TB est.	40 Billion est.

and affordable MPP (massively parallel processing) database [54], such as the Greenplum database [55], Apache Impala on Apache Kudu [56–58], and the Amazon Redshift database [59], to develop attack detection and characterization models and tune model parameters from large volumes of stored network traffic. An MPP relational database not only stores data in a meaningful and structured way but also leverages many machines in the cluster to facilitate and accelerate, through horizontally scaling, the process of executing complex database queries in a parallelized and distributed manner (via the expressive power of SQL) for interactive data exploration and mining insightful features from the stored structured data. By leveraging the MPP relational database, data scientists and network security operators can start building an optimized model, a process which may require multiple iterations of similar computations to search for the best model parameters, from a large dataset of network traffic traces, a task difficult to achieve using a standalone single node relational database.

Given that network traffic software (deployed at observation points) gathers and forwards incoming network traffic data in real-time, an effective volumetric DDoS monitoring system must also 1) detect and characterize volumetric DDoS attacks from large volumes of incoming traffic traces and 2) deliver the analysis results and defense recommendations to downstream systems for further processing, all in real-time. We define real-time as processing an updated dataset (spanning a sliding window of k units of time) every stride length u, as illustrated

in Figure 4.1. Our experiments suggest setting the stride length u =each minute and the sliding window size $k = 5$ in minutes, as discussed in Chapter 3. To achieve such real-time analysis, we leverage the Apache Kafka distributed streaming platform [60] to gather network traffic data into an intermediate staging area. Then, we utilize the Apache Spark distributed computing system as a streaming application [43–46] to perform the following each unit of time u, or each minute in our case:

1. Ingest network traffic data from Apache Kafka and compute metrics from traffic data

2. Analyze these metrics with a model (constructed offline) to detect whether a volumetric DDoS attack has occurred. If so, characterize the attack, identify attack sources and malicious traffic traces, and forward these attack characteristics and malicious traces (the defense recommendations) to downstream traffic filtering systems, all in real-time

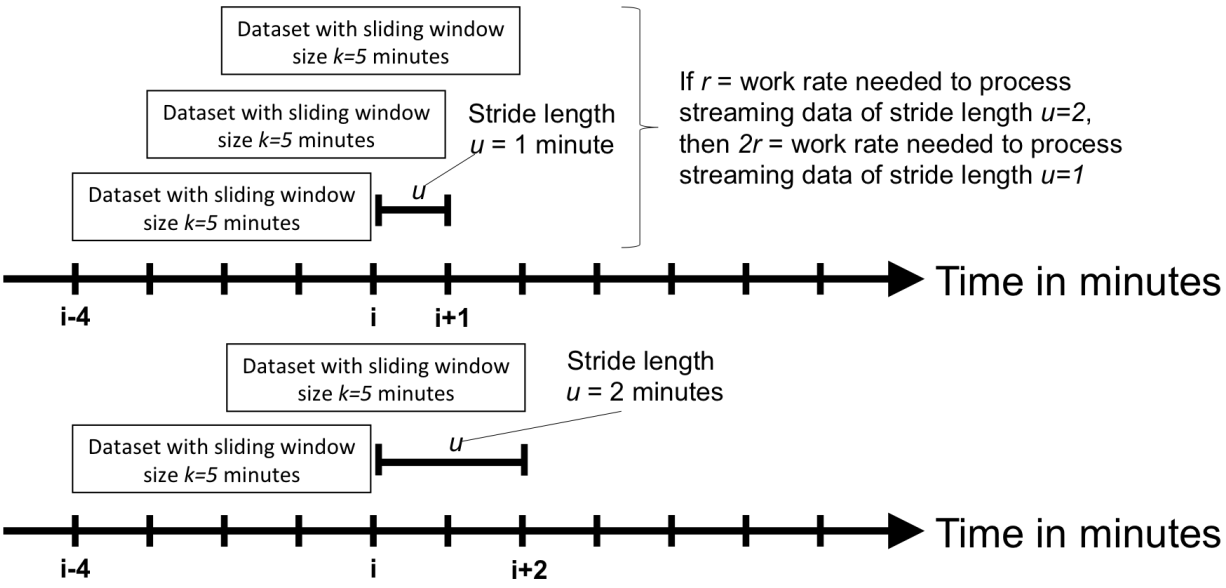

Figure 4.1: Higher Work Rates With Shorter Strides.

Figure 4.2 illustrates the proposed large-scale real-time volumetric DDoS traffic monitoring and characterization system. To facilitate the rapid ingestion of high volume and high velocity data, Kafka partitions incoming data records across machines in its cluster with a partitioning function based on the data record's corresponding key, provided by the

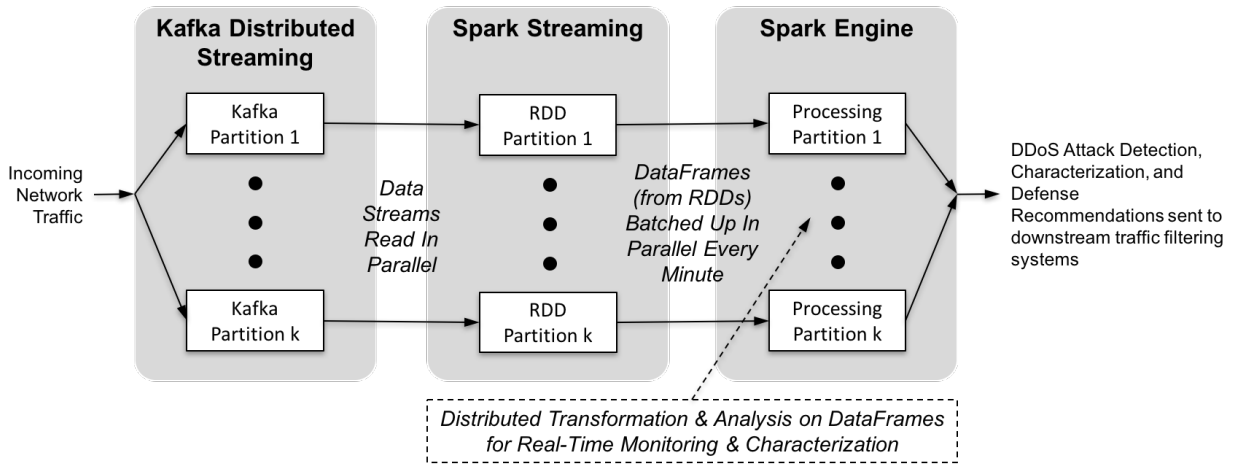

Figure 4.2: Leveraging a Parallel Streaming Platform for Real-Time Monitoring.

entity sending data to Kafka. This mechanism ensures that data records are distributed fairly amongst processing nodes in a cluster, thus fully utilizing a cluster's computational resources. Furthermore, many streaming applications, such as Apache Spark, can receiving incoming data from this source in real-time. More specifically, we leverage Apache Spark's streaming module (Spark Streaming [43]) to perform the following in real time using SQL:

1. Process network traffic data from Kafka in a structured relational schema format
2. Extract features from raw network traffic data
3. Apply Decision Rule 1 (see Chapter 3 Section 2) to these features to detect a DDoS attack
4. Apply Decision Rule 2 (see Chapter 3 Section 2) to identify attack sources and query raw network traffic data to identify malicious traffic traces based on currently identified attack characteristics

Specifically, the Spark DataFrame is a fault-tolerant and distributed data structure that offers a SQL interface to query (streaming) data in a structured format. Spark's ability to efficiently perform SQL computations on the DataFrame include the following reasons:

- Computations performed in main memory

- Spark Catalyst Optimizer performs SQL query optimization [47]

Suppose future testing on more datasets suggests that halving the stride length u yields a more accurate detection rate. This reduction in stride length doubles the minimum required work rate to process streaming datasets in real-time, as illustrated in Figure 4.1. If the initial amount of computational resources does not provide enough capacity for the Kafka distributed streaming platform and Spark streaming application to analyze the incoming datasets in real-time given a stride length u, these components can utilize their horizontal scalability properties to easily gain more computing power. In other words, because these streaming technologies leverage a computer cluster to parallelize expensive computations in real-time, adding more nodes to the cluster increases their ability to carry out more computations in a faster parallelized manner.

To illustrate how the proposed volumetric DDoS monitoring system leverages the previously mentioned big data technologies to process large-scale network traffic data, we present our scalable real-time end-to-end framework for the monitoring system as illustrated in Figure 4.3. When the MPP database has accumulated enough normal traffic data, the monitoring system begins mining insightful features and metrics and learning the attack detection and characterization model parameters from these features offline. For real-time analytics, the Spark streaming application receives incoming network traffic from Kafka, computes metrics (similar as offline) from the traffic, and applies the optimized detection model (created offline) to these metrics. Upon detecting an attack, the model characterizes the attack by returning the time of attack, victim of attack, attack traffic type, and attack sources; these attack characteristics help identify malicious traffic traces. Then, defense recommendations (the identified attack characteristics and malicious traces) are sent to downstream traffic filtering systems.

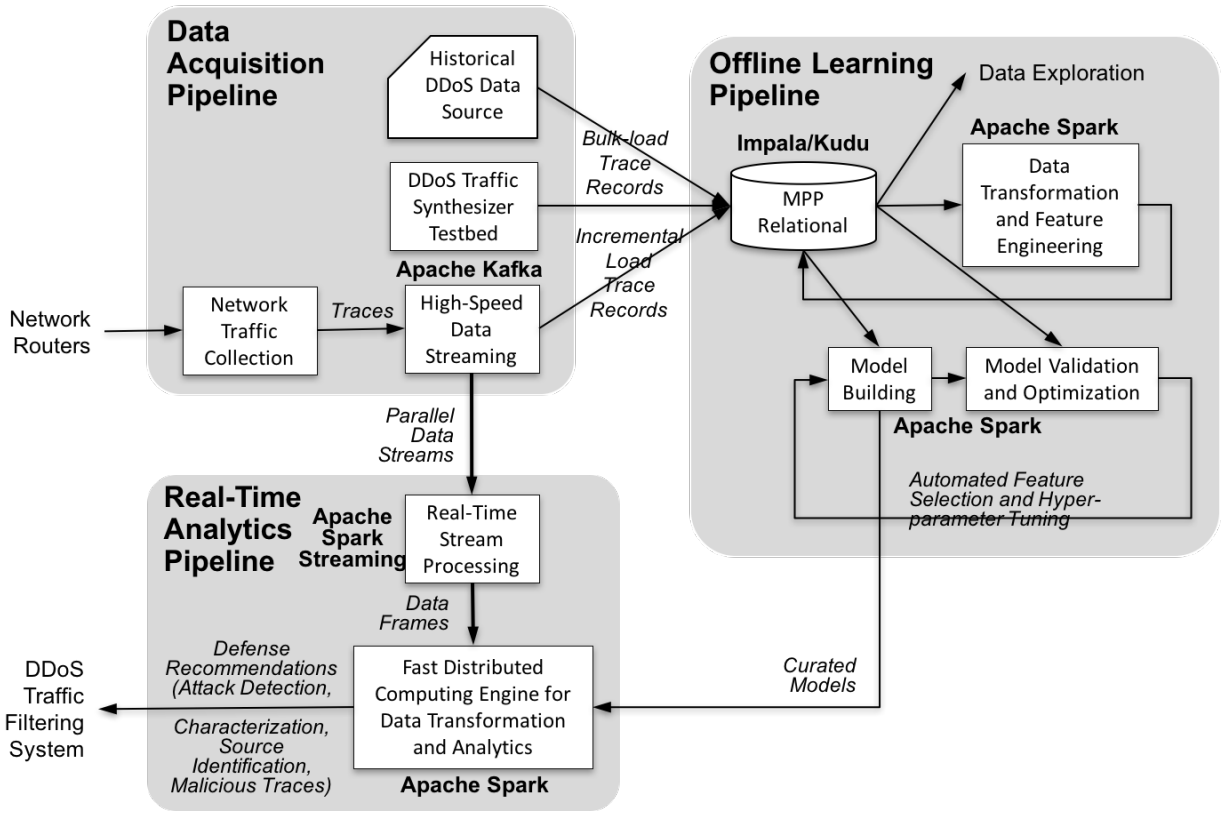

Figure 4.3: End-to-End Architecture For Scalable Real-Time DDoS Monitoring.

CHAPTER 5

Evaluation of DDoS Monitoring System

We evaluated the effectiveness of the proposed volumetric DDoS monitoring system on three recent and real-world volumetric DDoS attack trace datasets (DDOS_DNS_AMPL [48], DDOS_CHARGEN [49], and SYN_FLOOD_ATTACK [50]) supplied by the IMPACT repository [52] and three synthesized volumetric DDoS attack trace datasets (Deter Experiment 1, Deter Experiment 2, and Deter Experiment 3) gathered by running 3 DDoS attack experiments on the state-of-the-art DeterLab testbed [53]. All 6 datasets contain at least the following:

- Traces collected at a border router, which represents the gateway between an enterprise network and the Internet
- A consecutive sequence of normal traffic traces, used as a training set to learn the DDoS attack detection model parameters
- IP addresses within the network to monitor
- A consecutive sequence of traffic traces containing episodes of only normal traffic traces and episodes of both attack and normal traces mixed together, used as a testing set for evaluation

Note that the 3 synthesized datasets each contain the following:

- Periods of benign traffic, generated by the DeterLab HTTP web client modules from benign client nodes of the network topology

- 3 waves of volumetric DDoS attacks, where each wave contains a mix of benign and malicious traffic. The malicious traffic was generated by the DeterLab flooder modules from attacker nodes of the network topology

For Deter Experiment 1, the same 6 attackers participate in every attack wave of a constant bit rate attack. However, for Deter Experiments 2 and 3, different non-overlapping pairs of attackers launch each attack wave. More specifically, each attack wave of Deter Experiment 2 is a constant bit rate attack; however, each attack wave of Deter Experiment 3 has a pulsing nature that varies the attack's bit rate in an attempt to cause confusion. Figure 5.1 illustrates the realistic network topology created on the DeterLab testbed to run the 3 DDoS attack experiments. In this network topology, all links are 100 Mbps duplex links that connect 2 nodes.

Table 5.1 summarizes the results of the proposed volumetric DDoS monitoring system on all 6 datasets. Note that for the 3 synthesized volumetric DDoS attack trace datasets gathered, the provided attack descriptions, which include:

- Attack start and end times
- Brief notes on the attack nature
- Victim and attack sources

are the gold standard truths. However, as discussed later in this chapter, for the 3 recent and real-world volumetric DDoS attack trace datasets, the attack descriptions provided by network analysts are considered as rough approximations and sometimes incomplete, especially regarding the attack start and end times for the DDOS_DNS_AMPL dataset. Furthermore, network analysts did not provide any identified attack sources, and only the indicated victim is considered as a gold standard truth.

For the 3 synthesized datasets, our monitoring system correctly identified the attack's presence, characteristics in all the attack waves (including the targeted victim IP, the attack traffic protocol, and all the corresponding attackers of each attack wave), and malicious

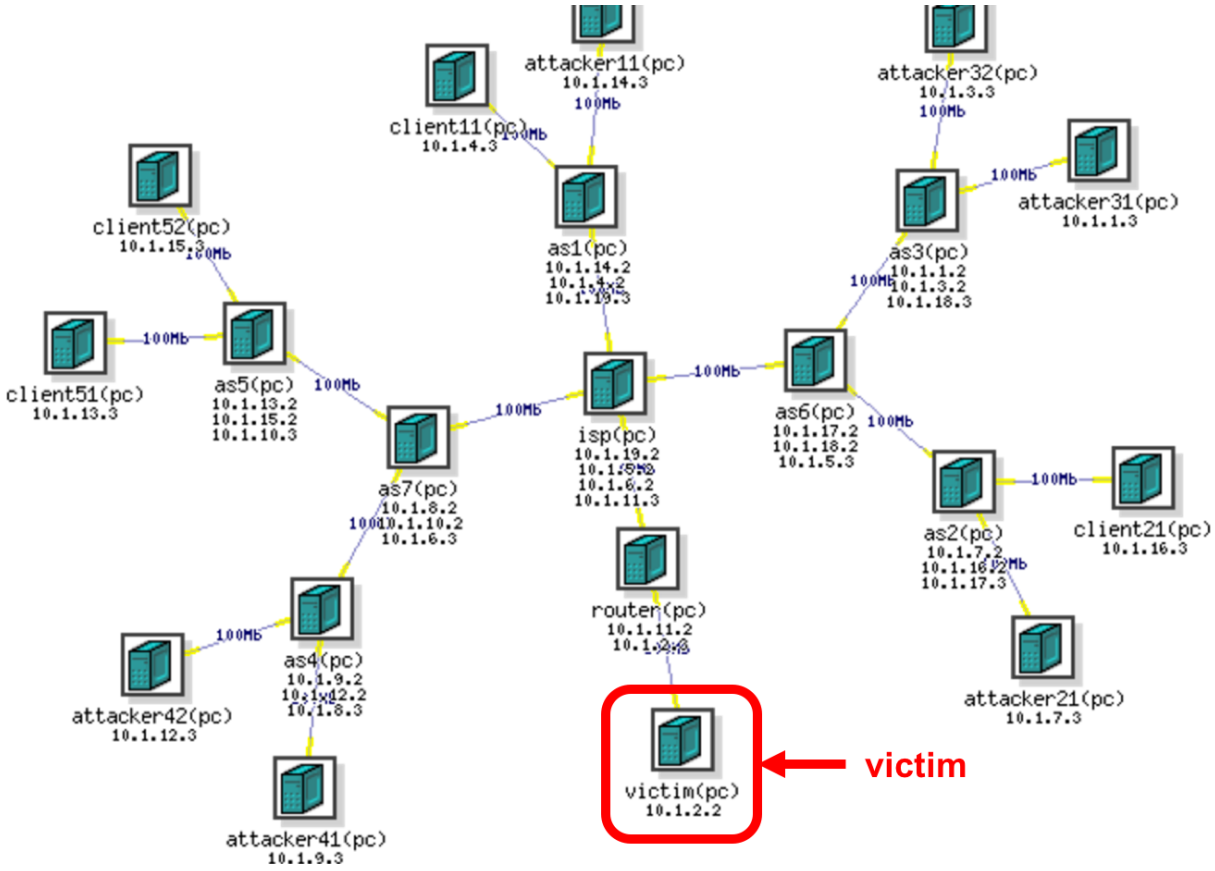

Figure 5.1: Network Topology Created on DeterLab.

traces from attackers with 100% accuracy. Figure 5.2 illustrates the normalized traffic volume in packets received by the designated victim 10.1.2.2 in Deter Experiment 3, the most sophisticated synthesized DDoS traffic experiment in this paper, where:

- $NormalizedPkts$ refers to the blue line, the number of TCP packets 10.1.2.2 receives during minute i divided by $MaxPkts = 274$ ($MaxPkts$ refers to the maximum total number of TCP packets 10.1.2.2 received during a minute in the training set)
- $NormalizedPktsMovingAvg$ refers to the green line, the moving average of $NormalizedPkts$ computed over a sliding window of $k = 5$ minutes spanning the current minute and previous 4 minutes

The red line near the bottom of Figure 5.2 refers to the thresholds $\theta_1 = 10$ and $\theta_2 = 10$, indicating that when both $NormalizedPkts$ and $NormalizedPktsMovingAvg$ rise above

Table 5.1: Results of DDoS Attack Detection And Characterization Algorithm On 6 Datasets

Trace Dataset	**Attack Start and End**	**Attack Description**	**Detection: Attack Time, Traffic Type, Identified Attack Sources**
DDOS_DNS_AMPL	After 11:00 am – Before 12:00 pm	Reflection and amplification DDoS attack on 7/22/2015	10:59 am – 11:13 am, ICMP, N/A 11:28 am – 11:32 am, ICMP, N/A 11:01 am – 11:13 am, UDP, N/A 11:30 am – 11:32 am, UDP, N/A
DDOS_CHARGEN	˜11:59 am – N/A	Reflection and amplification DDoS attack based on CHARGEN protocol over UDP on 11/25/2016	11:55 am – 12:00 pm, ICMP, N/A 11:54 am – 12:00 pm, UDP, N/A
SYN_FLOOD_ATTACK	˜4:18 pm – ˜4:28 pm	TCP SYN Flood attack on 3/4/2011	4:17 pm – 4:21 pm, ICMP, N/A 4:17 pm – 4:23 pm, TCP, N/A
Deter Experiment 1	Wave 1: Min 11 – Min 23 Wave 2: Min 35 – Min 47 Wave 3: Min 59 – Min 71	TCP Flood attack in 3 waves, each wave exhibits constant bit rate attack and involves all 6 attackers	Min. 11 – Min. 23, TCP, 6 attackers Min. 35 – Min. 47, TCP, 6 attackers Min. 59 – Min. 71, TCP, 6 attackers
Deter Experiment 2	Wave 1: Min 11 – Min 23 Wave 2: Min 35 – Min 47 Wave 3: Min 59 – Min 71	TCP Flood attack in 3 waves, each wave exhibits constant bit rate attack and involves 2 different attackers	Min. 11 – Min. 23, TCP, 2 attackers Min. 35 – Min. 47, TCP, 2 attackers Min. 59 – Min. 71, TCP, 2 attackers
Deter Experiment 3	Wave 1: Min 11 – Min 23 Wave 2: Min 35 – Min 47 Wave 3: Min 59 – Min 71	TCP Flood attack in 3 waves, each wave exhibits attack of pulsing nature and involves 2 different attackers	Min. 11 – Min. 23, TCP, 2 attackers Min. 35 – Min. 47, TCP, 2 attackers Min. 59 – Min. 71, TCP, 2 attackers

this threshold for any minute, the monitoring system signals an attack alert. See Chapter 3 for a detailed discussion of the above metrics.

Figure 5.2 illustrates that the first wave of the attack is detected between minutes 11 and 23, the second wave between minutes 35 and 47, and the third wave between minutes 59 and 71, a 100% accurate detection rate. Because all the plots (6 total) of the TCP flooding attack's traffic volume metrics in packets and bytes for Deter Experiment 1, 2, and 3 look

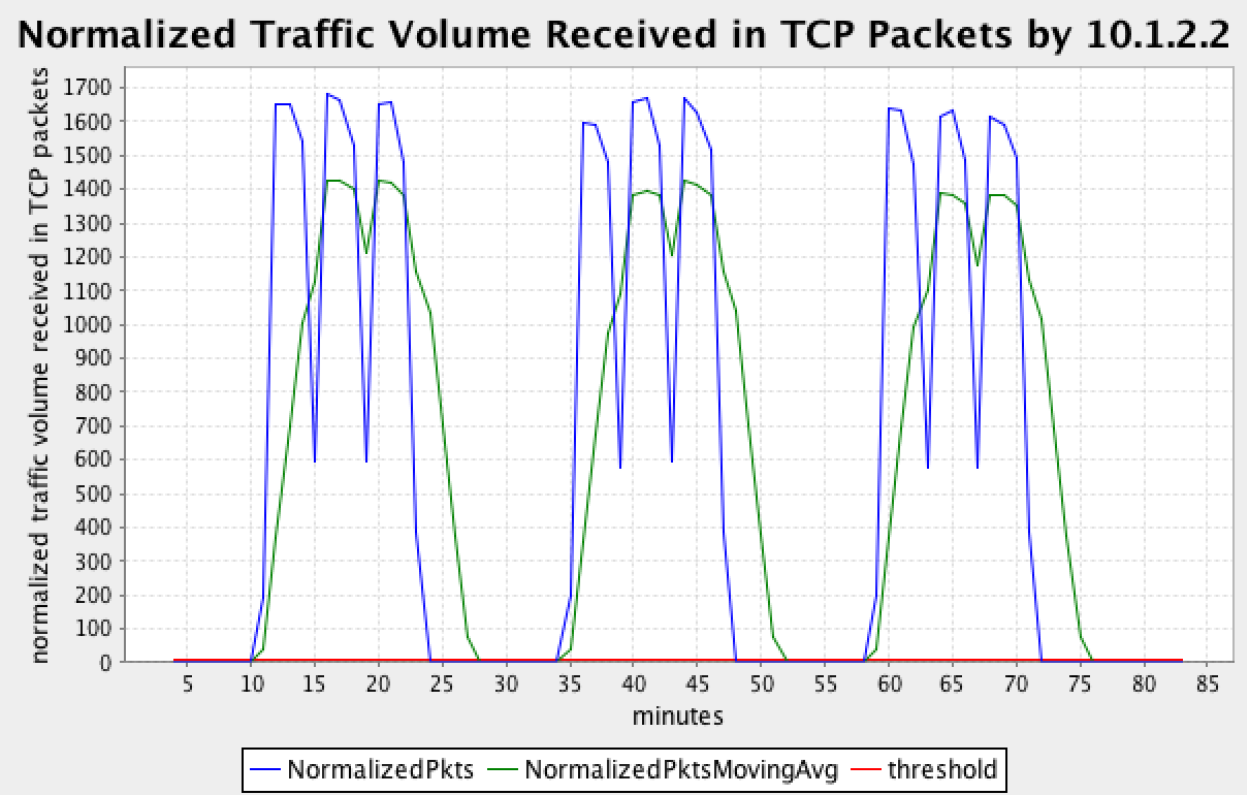

Figure 5.2: TCP Network Traffic Volume Plot in Packets of Deter Experiment 3 Dataset.

roughly similar, we show only 1 of these 6 plots. See appendix A to view all plots for Deter Experiment 1, 2, and 3.

Evaluating the results on the 3 real datasets raises complications because such attacks in the wild do not come with a set of exact ground truths fully describing the attack or labeling traces as benign or malicious. Network security analysts examined these traces to establish the victim IP but provided few ground truths describing the complete and exact nature of the attack. For example, they roughly identified the attack's start and end times but not specific attack sources. Hence, the value "N/A" in the right most column of Table 5.1 indicates that it is not possible to validate the correctness of the attack sources identified by the monitoring system; however, because our real-time attack source identification scheme performed well on the synthesized datasets, we argue that this same scheme would have identified many of the attack sources behind the real world DDoS attack traces. Furthermore, due to the lack of exact ground truths (and therefore a degree of uncertainty in the attack descriptions

provided by network analysts), we consider the following acceptable:

- Detection of an attack's start and end time within 5 to 10 minutes of its alleged start and end time provided by network analysts
- Detection of an attack throughout large portions of its alleged duration
- Identification of attack traffic protocols containing at least one of the alleged attack traffic protocols (if this detail was supplied in the attack trace description)

Thus, given that:

- Network analysts provided descriptions, with a degree of uncertainty, of the identified volumetric DDoS attacks in the 3 real datasets, as shown in Table 5.1
- The unknown detection and characterization ground truths (i.e. exact attack start and end time, traffic protocols relevant to attack, exact attack sources, victims) of the 3 real datasets may not exactly align with the approximate descriptions provided by the network analysts
- The detection and characterization results of our proposed volumetric DDoS monitoring system is consistent with the approximate descriptions provided by network analysts

we argue that the detection and characterization results of our monitoring system, as shown in Table 5.1, may match the unknown ground truths of the real-world DDoS attacks better than the approximate descriptions provided by the analysts. In fact, we can even argue that our results closely match, with near 100% accuracy, these unknown ground truths. The remainder of this chapter and Appendix A justifies this argument through analysis of traffic behavior as it relates to volumetric DDoS attacks.

To visualize traffic volume behavior of the traffic protocols (ICMP and UDP as shown in Table 5.1) identified as relevant to the detected attack in the DDOS_DNS_AMPL trace dataset, Figures 5.3 and 5.4 illustrate the normalized traffic volume in ICMP and UDP

packets received each minute by victim 204.38.0.0 of the DDOS_DNS_AMPL trace dataset, where minute 23959380 in both figures refers to 11:00 AM on 7/22/2015. The metrics plotted in these 2 figures have similar semantics as those plotted in Figure 5.2. Because the traffic behavior of the volumetric DDoS attack in the DDOS_DNS_AMPL trace dataset seems most interesting, this chapter focuses on analyzing potential discrepancies between our systems detection and monitoring results and the rough approximations provided by network analysts (and how they relate to the unknown ground truths of the attack). For a more comprehensive analysis on all 18 plots (6 plots per dataset) of the 3 real-world volumetric DDoS attack trace datasets (DDOS_DNS_AMPL, DDOS_CHARGEN, and SYN_FLOOD_ATTACK), see Appendix A. As illustrated in Appendix A, we argue that our detection and characterization results for the DDOS_CHARGEN and SYN_FLOOD_ATTACK datasets match well with approximate descriptions provided by network security analysts. For the DDOS_DNS_AMPL dataset, we next discuss our results in depth to address complications that arise when attempting to compare our detection and characterization results with the approximate descriptions provided by network security analysts.

While Figures 5.3 and 5.4 show that our monitoring system signaled the end of the attack at around minute 23959420 (11:40 AM), about 20 minutes earlier than the alleged end time of the attack provided by network analysts (minute 23959440 or 12:00 PM), we argue that this alleged end time is merely an approximation. Furthermore, our data analysis suggests that the DDoS attack came in 2 waves, with the 2nd wave ending earlier than 12:00 PM. Perhaps network analysts decided to wait a while before marking the end of the attack to ensure that they do not miss out on a 3rd wave. Based on Figures 5.3 and 5.4, we argue that the attack really ended around 11:40 AM and that our detection results are near 100% accurate. Similar, but minor, complications arise in the other 2 recent real-world DDoS attack datasets; however, the detection and characterization results of our volumetric DDoS monitoring system match the approximate descriptions given by network analysts relatively well.

Overall, our proposed volumetric DDoS monitoring system detects and characterizes volumetric DDoS attacks simulated in the Deter experiments with 100% accuracy. Due to

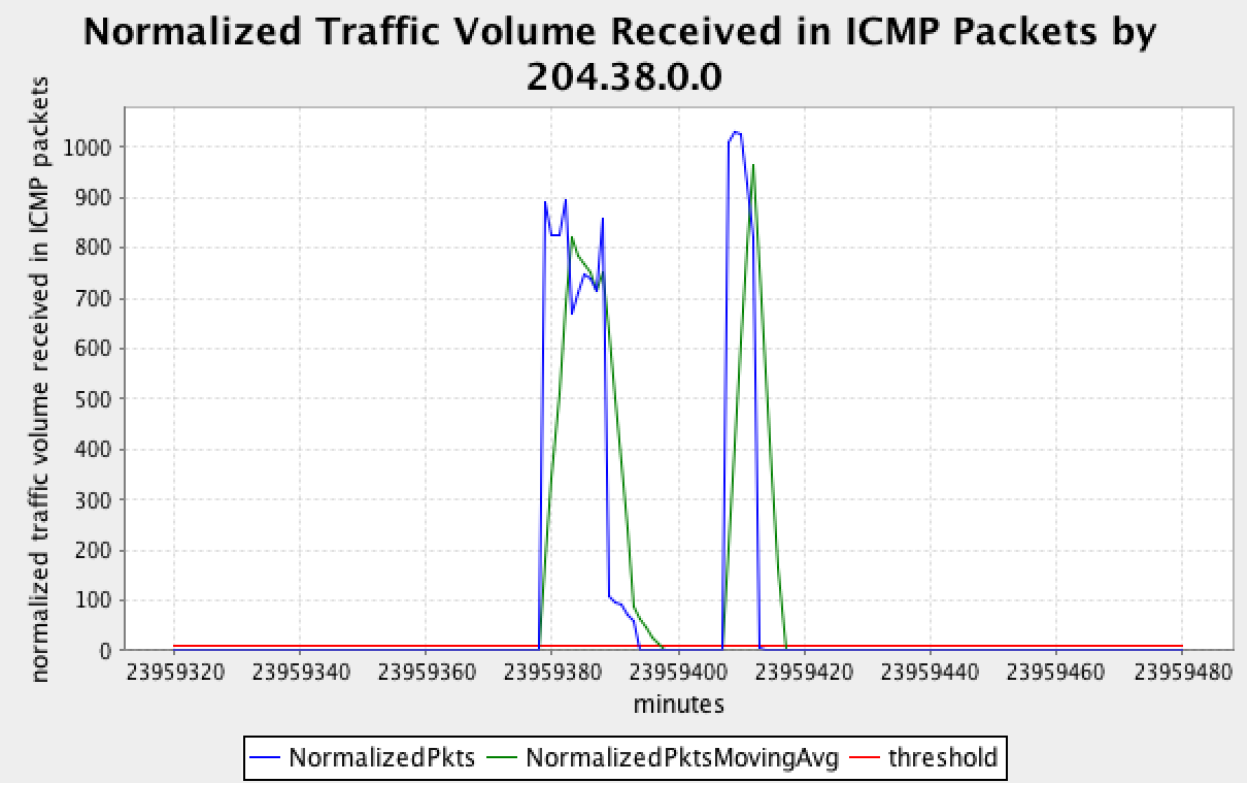

Figure 5.3: ICMP Network Traffic Volume in Packets of DDOS_DNS_AMPL Trace Dataset.

the lack of exact ground truths, approximate descriptions provided by analysts, and absence of important details such as exact attack sources, we cannot provide a specific quantitative measure that rates our system's detection and characterization performance on the recent real-world world volumetric DDoS attacks. However, we argue that our system detects and characterizes these attacks with near 100% accuracy, based on our justifications in previous paragraphs and the system's strong baseline performance on the synthesized traces.

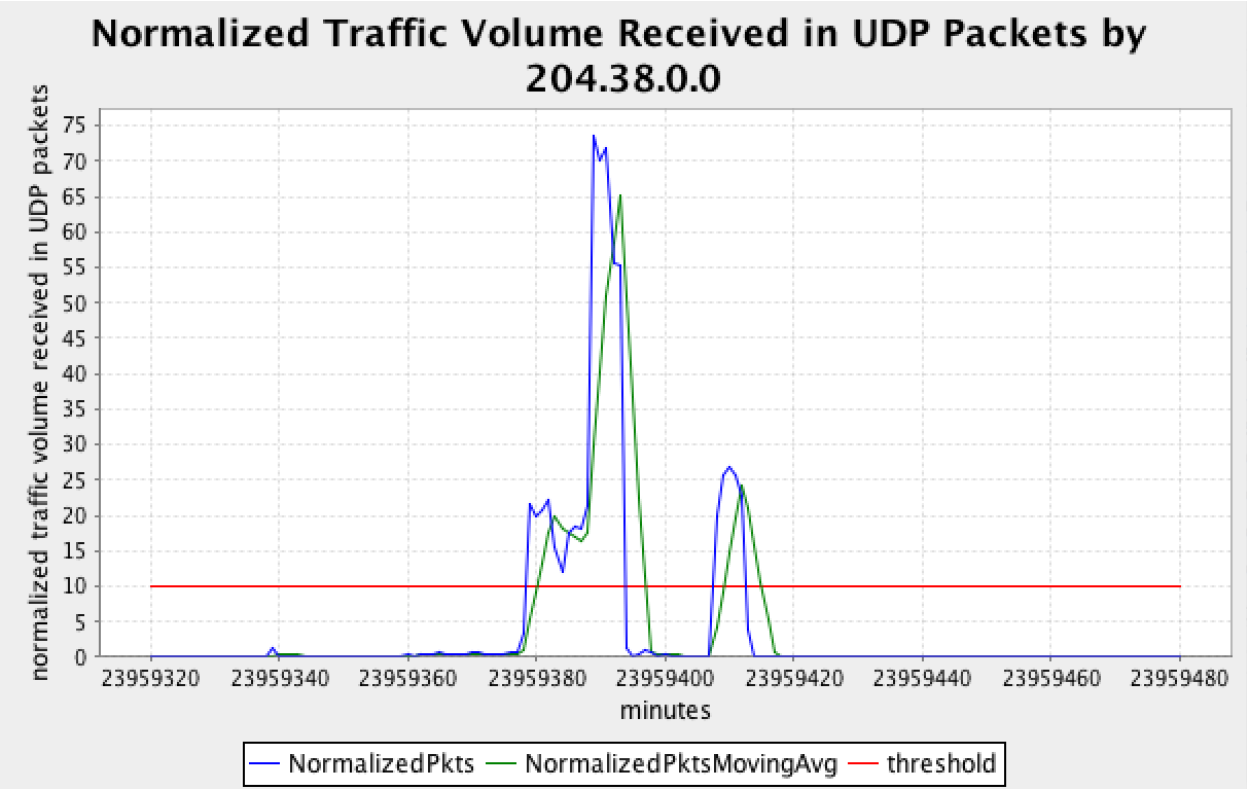

Figure 5.4: UDP Network Traffic Volume in Packets of DDOS_DNS_AMPL Trace Dataset.

CHAPTER 6

Future Work and Conclusion

From a network security perspective, future research includes creating a more robust and sophisticated attack detection and characterization model that:

- Analyzes more network traffic metrics indicative of a volumetric DDoS attack, such as entropy [35], chi-square [35], information gain [38], bidirectional count [29, 31]
- Investigates the importance of other standard network traffic trace fields in DDoS detection and characterization, such as port number
- Contains more tests to identify stealthier volumetric DDoS attacks, especially spoofed attacks, and separate out attacks from other traffic events, such as flash crowds
- Employs more advanced tests to identify attack sources

Furthermore, validating the performance of the proposed volumetric DDoS monitoring system on traffic traces of larger-scale, better, and diverse quality will indicate how to improve current monitoring strategies of sophisticated DDoS attacks. Such real world and synthesized traces should contain sophisticated attacks (e.g. spoofed attacks), more identified attack sources, better descriptions, and other anomalous events similar in nature to DDoS attacks (e.g. flash crowds). Traffic experiments generating synthesized traces should occur on testbeds capable of supporting larger internet-scale network topologies.

From a big data perspective, future research includes running many scalability experiments on the offline model building and real-time DDoS monitoring pipelines. In both pipelines, discovering the best partitioning strategy to process network traffic data with MPP databases and distributed streaming platforms ensures the maximum utilization of a

cluster's computational resources. From a deployment perspective, it is important to be able to formulate the minimum size of the monitoring cluster (e.g., number of nodes, number of CPUs, memory size, amount of disk storage) required to meet a given offline big data analysis task and real-time DDoS attack monitoring service-level agreement (e.g., stride length u as described in Chapters 3 and 4) under a given problem size and scope (e.g., attack size s, number k of servers to monitor across n networks). At a more detailed level, we would like to understand, for example, the number and size of partitions required to achieve optimal performance and how to achieve linear scalability for the monitoring cluster.

In conclusion, we propose an explainable volumetric DDoS monitoring system that:

1. Leverages modern big data technologies to accurately detect and characterize both recent real-world and synthesized DDoS attacks in real-time from large-scale traffic traces

2. Sends defense recommendations, the identified attack characteristics and malicious traces, to downstream traffic filtering systems

Constructing an explainable monitoring system that shows how input metrics and internal rules influence its decisions increases transparency of the approach and facilitates debugging operations, especially when the system does not perform well against DDoS attacks with new behaviors. We hope this research and discussion not only motivates the use of modern big data technologies as a core part of DDoS monitoring and defense solutions but also encourages others to develop explainable monitoring systems. We hope these changes will result in a wider deployment of monitoring solutions to combat the rapid growth and evolution of DDoS attacks.

Appendix A

Network Traffic Volume Plots for Trace Datasets

This appendix illustrates normalized network traffic volume plots of traffic destined for the victim for all 6 datasets explored in this paper. These datasets include 3 recent and real-world volumetric DDoS attack trace datasets (DDOS_DNS_AMPL [48], DDOS_CHARGEN [49], and SYN_FLOOD_ATTACK [50]) supplied by the IMPACT repository [52] and 3 synthesized volumetric DDoS attack trace datasets (Deter Experiment 1, Deter Experiment 2, and Deter Experiment 3) gathered by running 3 DDoS attack experiments on the state-of-the-art DeterLab testbed [53]. See chapter 5 for detection and characterization results of the monitoring system on all 6 datasets.

In all plots, when normalized traffic volumes (i.e. $NormalizedPkts$ and $NormalizedBytes$) and moving averages of normalized traffic volumes (i.e. $NormalizedPktsMovingAvg$ and $NormalizedBytesMovingAvg$) destined for a monitored server A both exceed the threshold during a particular minute i for a certain traffic protocol p, the monitoring system signals that a volumetric DDoS attack targeting A and involving traffic of protocol p has occurred in minute i. See chapter 3 for more details on the calculations of these traffic volume metrics and thresholds, as well as the data mining algorithm that analyzes these metrics.

Note that the monitored victim servers in these datasets may not receive any traffic during certain minutes. For example, victim 35.7.72.0 of the DDOS_CHARGEN dataset does not receive any traffic between minutes 24667729 and 24667733 (11:29 AM – 11:33 AM). For visualization purposes, the plotting software linearly interpolates traffic volumes for these minutes, hence explaining the slight inconsistency between the results for DDOS_CHARGEN in Table 5.1 (see chapter 5) and Figure A.6.

A.1 DDOS_DNS_AMPL Trace Dataset Plots

The DDOS_DNS_AMPL dataset contains network traffic traces gathered at a Merit Network, Inc. border router on 7/22/2015. The traces capture a real-world reflection and amplification DDoS attack mostly based on the DNS protocol that targets victim 204.38.0.0 from about 11:00 AM to about 12:00 PM [48]. Figures A.1, A.2, and A.3 illustrate normalized network traffic volume plots (for each traffic protocol) of traffic destined for the victim around the alleged time of attack, where minute 23959380 refers to 11:00 AM on 7/22/2015. Figures A.1 and A.3 show that the monitoring system detected an attack involving ICMP traffic from 10:59 AM to 11:13 AM and 11:28 AM to 11:32 AM and UDP traffic from 11:01 AM to 11:13 AM and 11:30 AM to 11:32 AM. However, the monitoring system does not detect an attack involving TCP traffic, as illustrated by figure A.2.

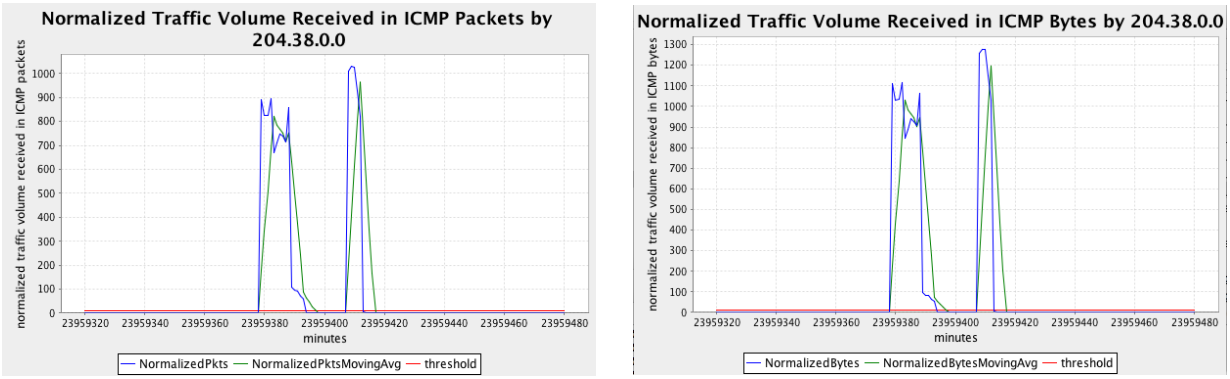

Figure A.1: ICMP Network Traffic Volume Plots of DDOS_DNS_AMPL Trace Dataset

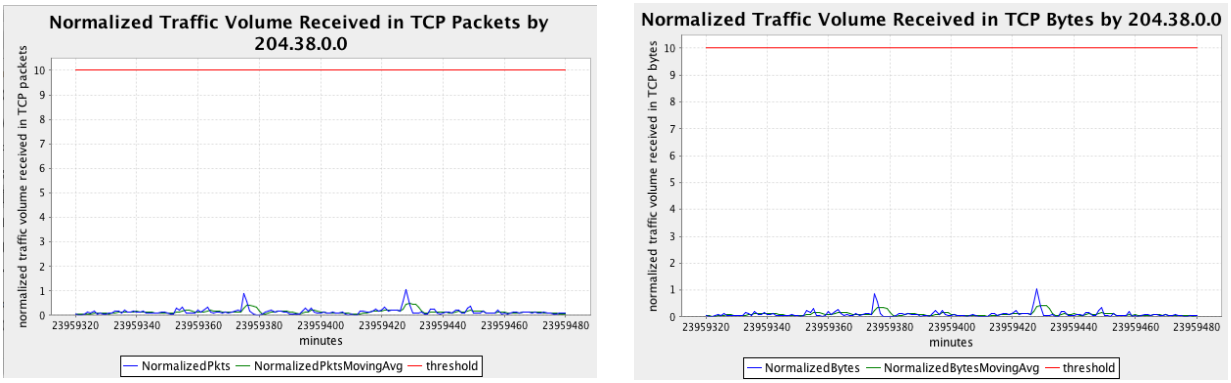

Figure A.2: TCP Network Traffic Volume Plots of DDOS_DNS_AMPL Trace Dataset

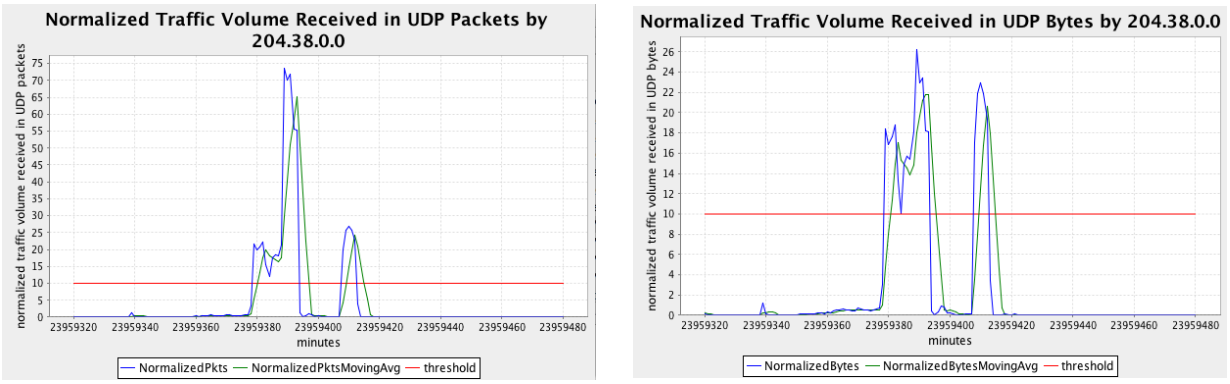

Figure A.3: UDP Network Traffic Volume Plots of DDOS_DNS_AMPL Trace Dataset

A.2 DDOS_CHARGEN Trace Dataset Plots

The DDOS_CHARGEN dataset contains network traffic traces gathered at a Merit Network, Inc. border router on 11/25/2016. The traces capture a real-world reflection and amplification DDoS attack based on the CHARGEN protocol over UDP that targets victim 35.7.72.0 starting at about 11:59 AM [49]. Figures A.4, A.5, and A.6 illustrate normalized network traffic volume plots (for each traffic protocol) of traffic destined for the victim around the alleged time of attack, where minute 24667740 refers to 12:00 PM on 11/25/2015. Figures A.4 and A.6 show that the monitoring system detected an attack involving ICMP traffic from 11:55 AM to 12:00 PM and UDP traffic from 11:54 AM to 12:00 PM. However, the monitoring system does not detect an attack involving TCP traffic, as illustrated by figure A.5. Note that figure A.4 only shows 6 minutes of normalized ICMP network traffic volume metrics because network traffic traces gathered at the border router around the alleged time of attack contain ICMP traces only during this short period.

A.3 SYN_FLOOD_ATTACK Trace Dataset Plots

The SYN_FLOOD_ATTACK dataset contains network traffic traces gathered within Merit Network, Inc. between 4:08 PM and 4:30 PM on 3/4/2011. The traces capture a real-world syn flood DDoS attack that targets a University of Michigan IRC victim server 143.213.232.0 from about 4:18 PM to about 4:28 PM [50]. Figures A.7, A.8, and A.9 illustrate normalized

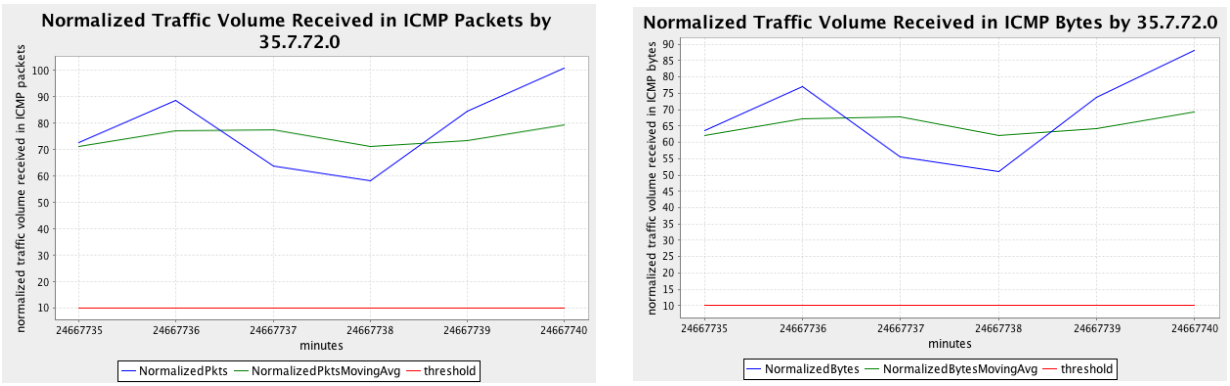

Figure A.4: ICMP Network Traffic Volume Plots of DDOS_CHARGEN Trace Dataset

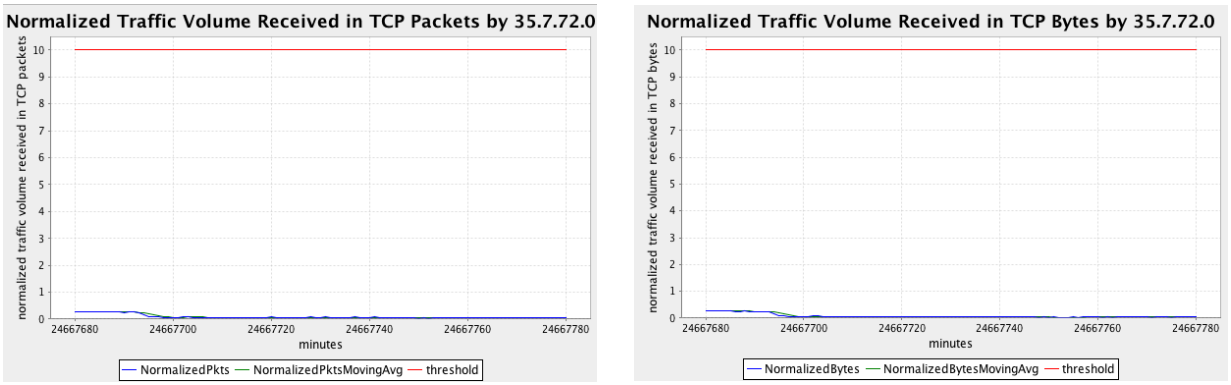

Figure A.5: TCP Network Traffic Volume Plots of DDOS_CHARGEN Trace Dataset

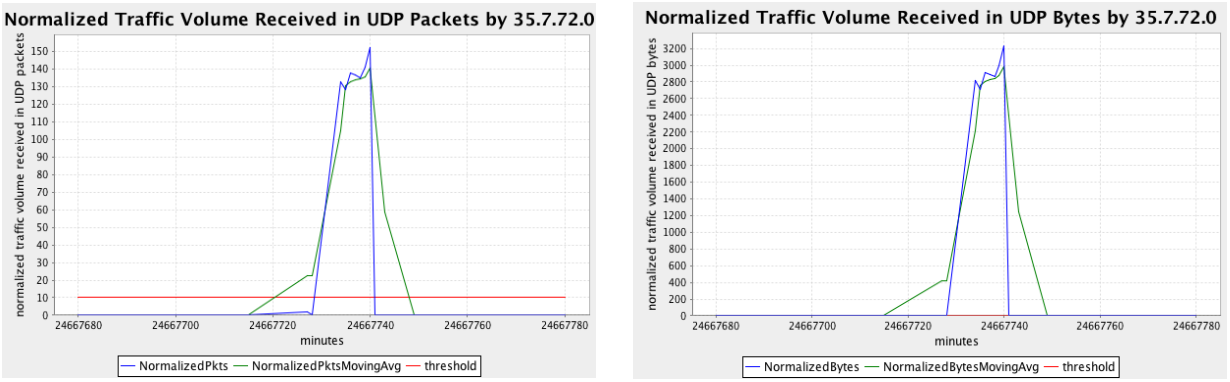

Figure A.6: UDP Network Traffic Volume Plots of DDOS_CHARGEN Trace Dataset

network traffic volume plots (for each traffic protocol) of traffic destined for the victim around the alleged time of attack, where minute 21654260 refers to 4:20 PM on 3/4/2011. Figures A.7 and A.8 show that the monitoring system detected an attack involving ICMP traffic from 4:17 PM to 4:21 PM and TCP traffic from 4:17 PM to 4:23 PM. However, the monitoring

system does not detect an attack involving UDP traffic, as illustrated by figure A.9.

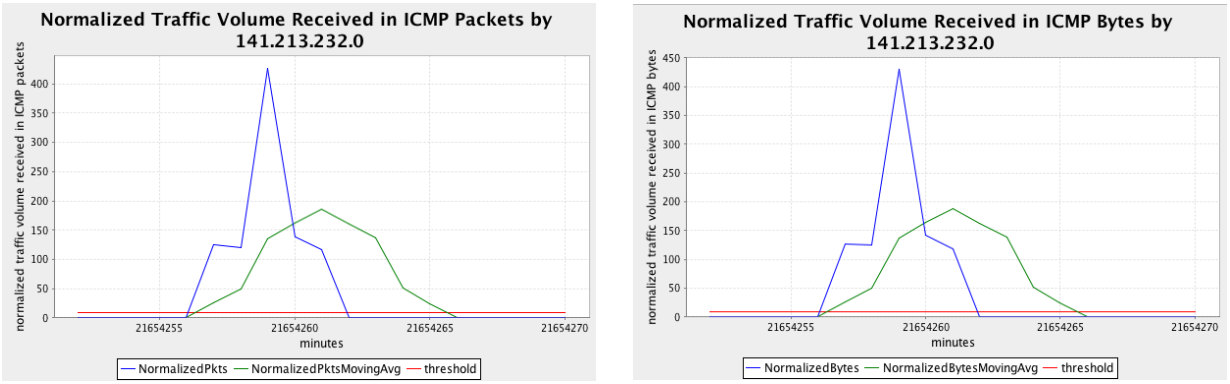

Figure A.7: ICMP Network Traffic Volume Plots of SYN_FLOOD_ATTACK Trace Dataset

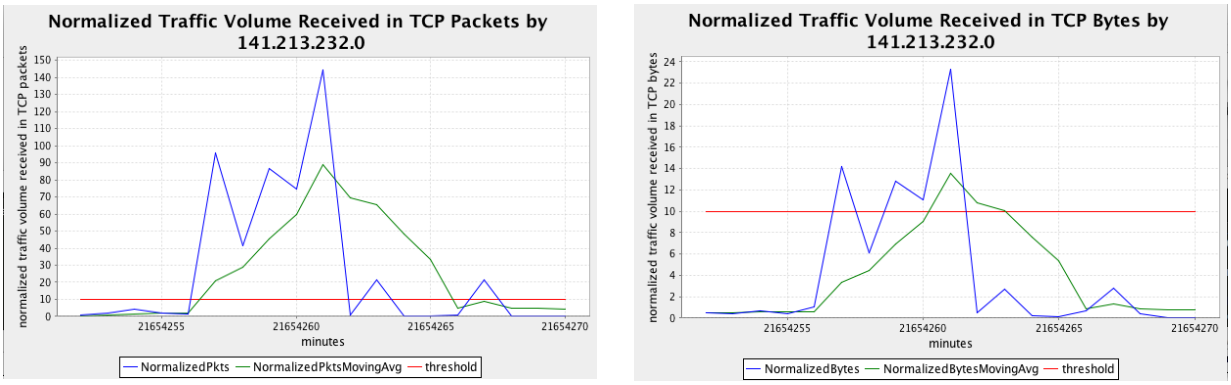

Figure A.8: TCP Network Traffic Volume Plots of SYN_FLOOD_ATTACK Trace Dataset

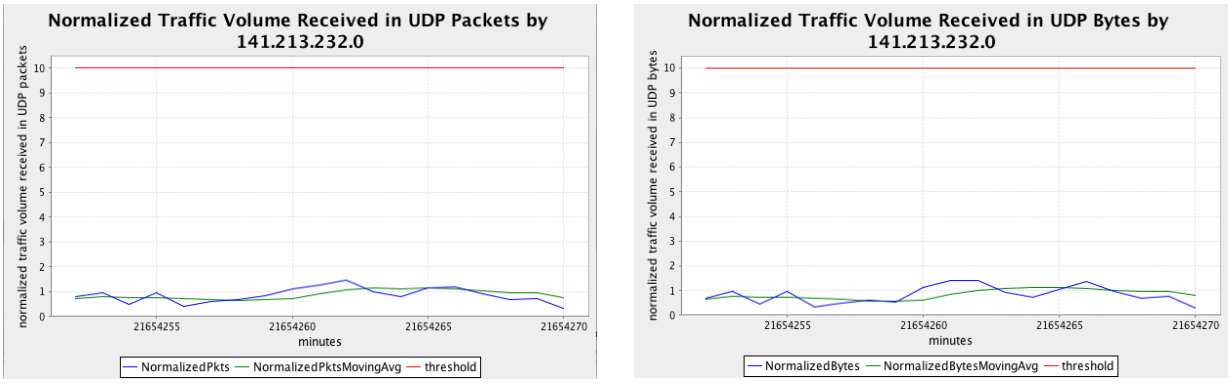

Figure A.9: UDP Network Traffic Volume Plots of SYN_FLOOD_ATTACK Trace Dataset

A.4 Deter Experiment Trace Datasets Plots

The datasets of Deter Experiment 1, Deter Experiment 2, and Deter Experiment 3 all contain 84 minutes of only TCP network traffic traces gathered at the designated router node (which has interfaces 10.1.11.2 and 10.1.2.3), which acts as a gateway node between the victim server 10.1.2.2 and the rest of the network topology (see chapter 5 for network topology). For all 3 synthesized DDoS attack traffic experiments, their traces captured a simulated TCP flood attack targeting the designated victim server 10.1.2.2 in 3 waves:

- Wave 1 occurs from minute 11 to minute 23
- Wave 2 occurs from minute 35 to minute 47
- Wave 3 occurs from minute 59 to minute 71

For Deter Experiment 1, all 6 attackers participate in each attack wave of a constant bit rate volumetric DDoS attack; however, for Deter Experiment 2 and Deter Experiment 3, only a pair of attackers participate in each attack wave, where the attack source pair changes for each wave (each pair of attackers for each wave does not overlap with other pairs of attackers for other waves). Furthermore, Deter Experiment 2 simulates a constant bit rate volumetric DDoS attack for each wave; however, Deter Experiment 3 simulates a volumetric DDoS attack with a pulsing nature for each wave in an attempt to confuse the volumetric DDoS monitoring system. Throughout the entire 84 minutes of all 3 experiments, the designated 6 normal web client nodes continue to send HTTP web traffic to the victim server at normal benign rates. Figures A.10, A.11, and A.12 illustrate normalized TCP network traffic volume plots of traffic destined for the victim around the time of attack, with each figure showing that the monitoring system detected attacks throughout the full duration of each attack wave. Note that figures A.10, A.11, and A.12 only show TCP network traffic volume plots because all 3 synthesized traffic experiments contain only TCP traffic.

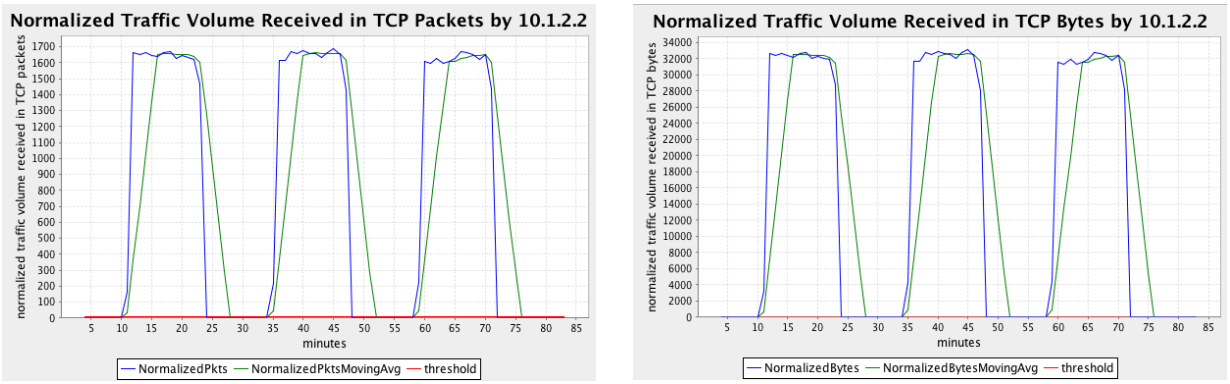

Figure A.10: TCP Network Traffic Volume Plots of Deter Experiment 1 Trace Dataset

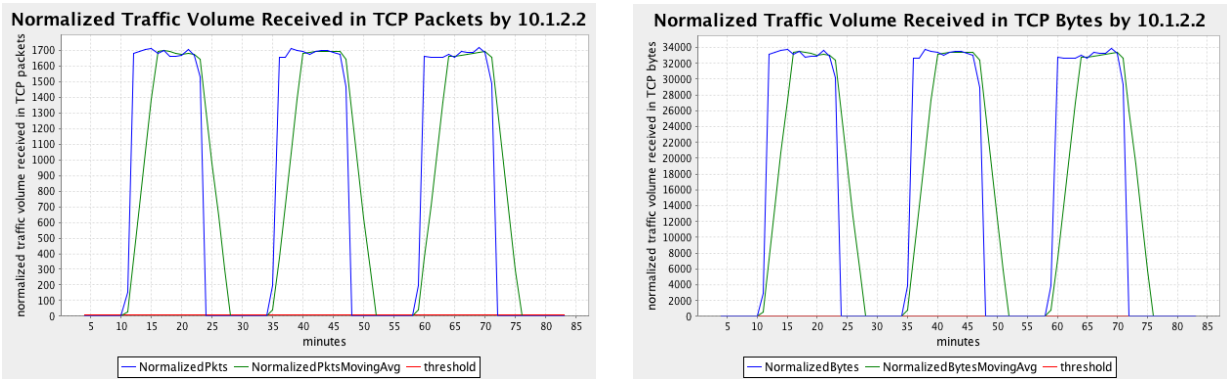

Figure A.11: TCP Network Traffic Volume Plots of Deter Experiment 2 Trace Dataset

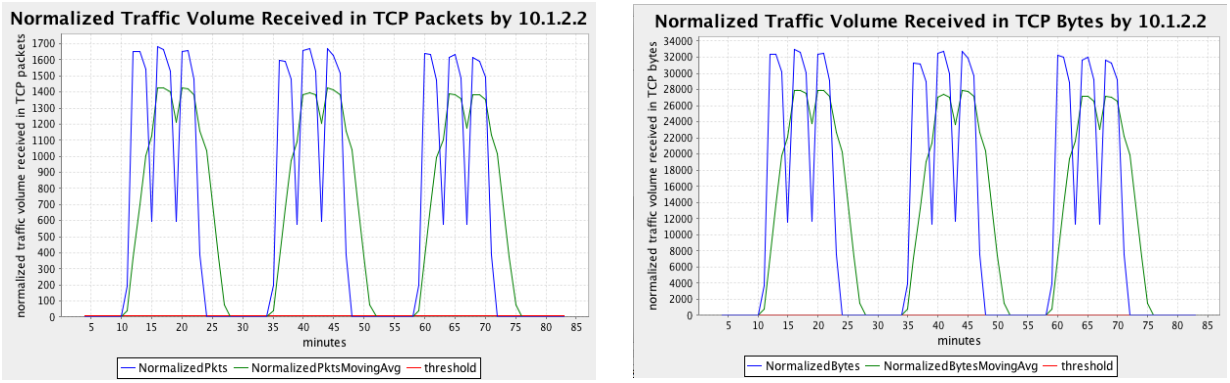

Figure A.12: TCP Network Traffic Volume Plots of Deter Experiment 3 Trace Dataset

REFERENCES

[1] L. Mathews. (2016). Someone just used the Mirai botnet to knock an entire country offline [Online]. Available: https://www.forbes.com/sites/leemathews/2016/11/03/someone-just-used-the-mirai-botnet-to-knock-an-entire-country-offline/#4bfb5f2f6c4f

[2] S. Thielman and C. Johnston. (2016). Major cyber attack disrupts internet service across Europe and US [Online]. Available: https://www.theguardian.com/technology/2016/oct/21/ddos-attack-dyn-internet-denial-service

[3] K. Arora, K. Kumar, and M. Sachdeva. Impact analysis of recent DDoS attacks. *International Journal on Computer Science and Engineering*, 3(2):877–884, 2011.

[4] S. T. Zargar, J. Joshi, and D. Tipper. A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. *IEEE Communications Surveys & Tutorials*, 15(4):2046–2069, 2013.

[5] J. Mirkovic and P. Reiher. D-WARD: A source-end defense against flooding denial-of-service attacks. *IEEE Trans. on Dependable and Secure Computing*, 2(3):216–232, 2005.

[6] M. Roesch. Snort – Lightweight intrusion detection for networks. *Proceedings of the 13th USENIX Conference on Systems Administration*, 229–238, 1999.

[7] V. Paxson. Bro: A system for detecting network intruders in real-time. *Computer Networks*, 31(23-24):2435–2463, 1999.

[8] D. G. Andersen. Mayday: Distributed filtering for Internet services. *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems*, 2003.

[9] J. Francois, I. Aib, and R. Boutaba. FireCol: A collaborative protection network for the detection of flooding DDoS attacks. *IEEE/ACM Transactions on Networking*, 20(6):1828–1841, 2012.

[10] S. K. Fayaz, Y. Tobioka, V. Sekar, and M. Bailey. Bohatei: Flexible and elastic DDoS defense. *Proceedings of the 24th USENIX Security Symposium*, 817–832, 2015.

[11] S. Lim, J. Ha, H. Kim, and S. Yang . A SDN-oriented DDoS blocking scheme for botnet-based attacks. *Sixth International Conference on Ubiquitous and Future Networks*, 63–68, 2014.

[12] J. Li, S. Berg, M. Zhang, P. Reiher, and T. Wei. Drawbridge– Software-defined DDoS-resistant traffic engineering. *ACM SIGCOMM Comp. Comm. Review*, 44:591–592, 2014.

[13] A. Khalimonenko and O. Kupreev. (2017). DDOS attacks in Q1 2017 [Online]. Available: https://securelist.com/ddos-attacks-in-q1-2017/78285/

[14] A. Khalimonenko, J. Strohschneider, and O. Kupreev. (2017). DDOS attacks in Q4 2016 [Online]. Available: https://securelist.com/ddos-attacks-in-q4-2016/77412/

[15] O. Kupreev, J. Strohschneider, and A. Khalimonenko. (2016). Kaspersky DDOS intelligence report for Q3 2016 [Online]. Available: https://securelist.com/kaspersky-ddos-intelligence-report-for-q3-2016/76464/

[16] Verisign. (2016). Q4 2016 DDoS attack trends [Online]. Available: http://www.verisign.com/assets/infographic-ddos-trends-Q42016.pdf

[17] Verisign. (2016). Verisign distributed denial of service trends report [Online]. Available: http://www.verisign.com/assets/report-ddos-trends-Q42016.pdf

[18] Verisign. (2017). Q1 2017 DDoS attack trends [Online]. Available:http://www.verisign.com/assets/infographic-ddos-trends-Q12017.pdf

[19] Imperva. (2016). Imperva Incapsula DDoS Protection [Online]. Available: https://www.incapsula.com/datasheets/ddos-protection.pdf

[20] F5. (2014). The F5 DDoS Protection Reference Architecture [Online]. Available: https://f5.com/resources/white-papers/the-f5-ddos-protection-reference-architecture

[21] F5. (2017). F5 Silverline DDoS Protection [Online]. Available: https://www.f5.com/pdf/products/silverline-ddos-datasheet.pdf

[22] F5. (2017). F5 Herculon DDoS Hybrid Defender [Online]. Available: https://www.f5.com/pdf/products/herculon-ddos-hybrid-defender-datasheet.pdf

[23] Arbor Networks, Inc. (2017). Arbor Networks SP [Online]. Available: https://www.arbornetworks.com/images/documents/Data%20Sheets/DS_SP_EN.pdf

[24] Arbor Networks, Inc. (2017). Arbor Cloud DDoS Protection [Online]. Available: https://www.arbornetworks.com/images/documents/Data%20Sheets/DS_Arbor_Cloud_Enterprise.pdf

[25] Arbor Networks, Inc. (2017). Arbor Networks TMS [Online]. Available: https://www.arbornetworks.com/images/documents/Data%20Sheets/DS_TMS_EN.pdf

[26] Verisign. (2015). Verisign DDoS Protection Services [Online]. Available: https://www.verisign.com/assets/pdf/resource-center/datasheet-ddos-overview.pdf

[27] Verisign. (2015). Verisign OpenHybrid [Online]. Available: https://www.verisign.com/assets/pdf/resource-center/datasheet-ddos-openhybrid.pdf

[28] R. Braga, E. Mota, and A. Passito. Lightweight DDoS flooding attack detection using NOX/OpenFlow. *Proceedings of the 35th Annual IEEE Conference on Local Computer Networks*, 408–415, 2010.

[29] K. Giotis, G. Androulidakis, and V. Maglaris. Leveraging SDN for efficient anomaly detection and mitigation on legacy networks. *Proceedings of the 3rd European Workshop on Software Defined Networks*, 85–90 2014.

[30] S. M. Mousavi and M. St-Hilaire. Early detection of DDoS attacks against SDN controllers. *Proceedings of the 2015 International Conference on Computing, Networking and Communications, Communications and Information Security Symposium*, 77–81, 2015.

[31] H. Liu, Y. Sun, and M. S. Kim. A scalable DDoS detection framework with victim pinpoint capability. *Journal of Communications*, 6(9):660–670, 2011.

[32] Y. Lee and Y. Lee. Detecting DDoS attacks with Hadoop. *Proceeding of the ACM CoNEXT Student Workshop*, 2011.

[33] V. Sekar, N. Duffield, O. Spatscheck, K. van der Merwe, and H. Zhang. LADS: Large-scale automated DDoS detection system. *Proceedings of the USENIX 2006 Annual Technical Conference*, 2006.

[34] S. Hameed and U. Ali. Efficacy of live DDoS detection with Hadoop. *IEEE/IFIP Network Operations and Management Symposium*, 2016.

[35] L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred. Statistical approaches to DDoS attack detection and response. *Proceedings of the DARPA Information Survivability Conference and Exposition*, 303–314, 2003.

[36] G. No and I. Ra. An efficient and reliable DDoS attack detection using a fast entropy computation method. *Proceedings of the 9th IEEE International Symposium on Communications and Information Technology*, 1223–1228, 2009.

[37] J. Zhang, Z. Qin, L. Ou, and A. X. Liu. An advanced entropy-based DDoS detection scheme. *Proceedings of the 2010 IEEE International Conference on Information, Networking and Automation*, 67–71, 2010.

[38] W. Wang and S. Gombault. Efficient detection of DDoS attacks with important attributes. *Proceedings of the 3rd International Conference on Risks and Security on Internet and Systems*, 61–67, 2008.

[39] Center for Applied Internet Data Analysis. (2017). The CAIDA "DDoS Attack 2007" Dataset [Online]. Available: `https://www.caida.org/data/passive/ddos-20070804_dataset.xml`

[40] Lincoln Laboratory, MIT. (2017). DARPA Intrusion Detection Data Sets [Online]. Available: `https://ll.mit.edu/ideval/data/index.html`

[41] S. Shahrivari. Beyond batch processing: Towards real-time and streaming big data. *Computers*, 3:117–129, 2014.

[42] J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. *Proceedings of the 6th conference on Symposium on Operating Systems Design and Implementation*, 6:137-150, 2004.

[43] M. Zaharia, T. Das, H. Li, T. Hunter, S. Shenker, and I. Stoica. Discretized streams: Fault-tolerant streaming computation at scale. *Proceedings of the 24th ACM Symposium on Operating Systems Principles (SOSP)*, Nov. 2013.

[44] M. Zaharia, M. Chowdhury, J. Ma, M. J. Franklin, S. Shenker, and I. Stoica. Spark: Cluster computing with working sets. *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing*, 2010.

[45] Apache Spark. Apache Spark – A fast and general engine for large-scale data processing [Online]. Available: `https://spark.apache.org/`

[46] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, I. Stoica. Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing. *Proceedings of the 9th USENIX NSDI Symposium on Networked Systems Design and Implementation*, 15-28, 2012.

[47] M. Armbrust, R. Xin, C. Lian, Y. Huai, D. Liu, J. K. Bradley, X. Meng, T. Kaftan, M. J. Franklin, A. Ghodsi, and M. Zaharia. Spark SQL: Relational data processing in Spark. *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, 1383-1394, 2015.

[48] Merit Network, Inc. (2015). A DNS-based amplification DDoS attack [Online]. Available: `https://www.impactcybertrust.org/dataset_view?idDataset=580`

[49] Merit Network, Inc. (2016). Internet traffic data containing a DDoS attack based on UDP Chargen protocol [Online]. Available: `https://www.impactcybertrust.org/dataset_view?idDataset=693`

[50] Merit Network, Inc. (2011). Netflow data for a SYN flood attack [Online]. Available: `https://www.impactcybertrust.org/dataset_view?idDataset=160`

[51] Merit Network, Inc. (2016). A DDoS event against the RADb service [Online]. Available: `https://www.impactcybertrust.org/dataset_view?idDataset=576`

[52] Welcome to IMPACT – Information Marketplace for Policy and Analysis of Cyber-Risk & Trust [Online]. Available: `https://www.impactcybertrust.org/home#welcome`

[53] DeterLab: Cyber-Defense Technology Experimental Research Laboratory [Online]. Available: `https://www.isi.deterlab.net`

[54] E. Begoli. A short survey on the state of the art in architectures and platforms for large scale data analysis and knowledge discovery from data. *Proceedings of the WICSA/ECSA 2012 Companion Volume*, 177-183, 2012.

[55] Greenplum database – The world's first open source massively parallel data warehouse [Online]. Available: `http://greenplum.org`

[56] Apache Impala – The open source, native analytic database for Apache Hadoop [Online]. Available: https://impala.incubator.apache.org

[57] Apache Kudu – Apache Kudu completes Hadoop's storage layer to enable fast analytics on fast data [Online]. Available: https://kudu.apache.org

[58] Apache Kudu – Using Apache Kudu with Apache Impala [Online]. Available: https://kudu.apache.org/docs/kudu_impala_integration.html

[59] Amazon Web Services. Amazon Redshift [Online]. Available: https://aws.amazon.com/redshift/

[60] Apache Kafka. Apache Kafka – A distributed streaming platform [Online]. Available: https://kafka.apache.org/