DoS Forensic Exemplar Comparison to a Known Sample

Paul Knight *prk009@shsu.edu*
Department of Computer Science
Sam Houston State University
Huntsville, TX 77341, USA

Narasimha Shashidhar *nks001@shsu.edu*
Department of Computer Science
Sam Houston State University
Huntsville, TX 77341, USA

Abstract

The investigation of any event or incident often involves the evaluation of physical evidence. Occasionally, a comparison is conducted between an evidentiary sample of unknown origin and that of an appropriate known sample. In a Denial of Service (DoS) attack, items of evidentiary value may cross the spectrum from anecdotes to useful information in firewall logs or complete packet captures. Because of the spoofed or reflective nature of DoS attacks, relevant information leading to the direct identification of the perpetrator is rarely available. In many instances, this underscores the significance of the investigator's ability to accurately identify the tool utilized by the suspect. For a DoS attack scenario, this would likely involve a commercially available stresser or criminal bot infrastructure. In this paper, we propose the concept of a DoS exemplar and determine if the comparison of evidentiary samples to an appropriate known sample of DoS attributes could add value in the investigative process. We also provide a simple tool to compare two DoS flows.

Keywords: Denial of Service Flow Comparison, DoS Similarity Score, DoS Exemplar, Stresser.

1. INTRODUCTION

1.1 The Need

DoS attacks can occasionally result in a loss to commerce, but it almost always results in embarrassment to the victims. Investigators assigned to these cases may have a variety of artifacts to evaluate, ranging from full-packet capture to basic firewall logs. DoS attacks range from the simplest SYN flood to application specific resource depletion. Attacks can originate from a single computer, a commercial stresser, or a criminal bot. Because of the spoofed nature of DoS traffic, attribution to the source is not traditionally part of the investigative process. The purpose of our work is to establish a repeatable and forensically sound methodology for the investigator to compare DoS flows in an effort to make a determination if these streams originated from the same source. The potential attribution to the tool used to facilitate the attack could then result in legal process authorizing the seizure of the suspect server and the related databases.

1.2 Solution for Everyone

In this work, we present our method which includes a simple tool and the accompanying algorithm along with the scoring of DoS floods from five different stressers to validate the feasibility of including a "similarity score" into the investigative work flow. Reflective floods from each of the five sample stressers were captured as **.pcap* files. Floods of 30 seconds or less were used, as they provide adequate traffic for analysis, yet are small enough to be manageable with freely available tools. The weaknesses discovered through the evaluation of these samples will be presented. Finally, the concept of flow cross-section will be introduced. Additional limitations created by the possibility of shared infrastructure will be discussed as well.

2. PRIOR WORK

2.1 Complications of Attribution

There is an assortment of excellent prior research efforts on both reflective DoS attacks and attribution, but, to the best of our knowledge, it appears that most researches have stayed away from the concept of identifying a particular '*booter*' used in an attack. This is primarily due to the inherent spoofed nature of the DoS traffic and the complication of attribution when a zombie or botnet is utilized.

In the research article, *An Analysis of DDoS-as-a-Service Attacks* [1], Santanna et al. analyzed the attack characteristics of 14 distinct booters. The goal of their analysis was to add value to attack victims in the mitigation and prevention of attacks, as well as to help other researchers in understanding how stressers work, in general. They used Jaccard's Similarity Coefficient as the metric to compare flow characteristics and based on this metric concluded that, at the time, stressers only share minimal infrastructures. However, it appeared as if the future of infrastructure sharing was unpredictable.

Wheeler and Larsen, in their work titled *Techniques for Cyber Attack Attribution* [2], discussed 17 potential methods of attack attribution techniques, but conceded the difficulty and inherent limitations of such techniques, primarily because of the use of intermediaries, which both spoof traffic and use reflective protocols [2].

Hunker et al. embarked on a research effort where the focus was not on the technical aspects of attribution, but on a more general attribution framework for showing that the characteristic associated with an entity has value in the overall attribution process. This report is titled '*Report on Attribution for GENI*' [3].

Kuhrer et al.'s research work '*Exit from Hell? Reducing the Impact of Amplification DDoS Attacks',* provided insight into the root cause of DDoS amplification attacks through four methods: utilizing protocol specific fingerprinting, creating advisories, and analyzing both future attack vectors and the root cause of amplification attacks [4].

2.2 Building on Previous Work

Based on our literature survey of the vast body of work, we have identified Karami's work to be the first as it relates to validating the concept of DoS attribution to a particular stresser. He accomplished this through a collection of 30 second attacks from 23 different commercially available stressers. Karami evaluated the reflective traffic of four DoS protocols (NTP, DNS, CharGen, and SSDP) using the Jaccard's Similarity Coefficient, which provides a value of 0-1 for the similarity of IP addresses shared by two flow collections. Through the information collected in 2,667 attacks, Karami was able to propose minimum threshold scores for four of the five protocols likely to indicate a correct classification: CharGen t=.55, DNS t=.60, NTP t=.55, SSDP t=.45. DNS did not provide the same classification accuracy as the other three protocols, so Karami added additional features to the algorithm to improve performance, namely resolved domain names [5].

2.3 Prior Work Meets Current Need

In our present work, similarity scores were compared to Karami's findings for CharGen, NTP, DNS, and SSDP attacks. Similarity scores were also obtained in the SNMP and TFTP protocols. DNS findings were not as predictable, even though the samples shared characteristics like name resolution.

3. APPLICATION

3.1 Implementing a Similarity Score

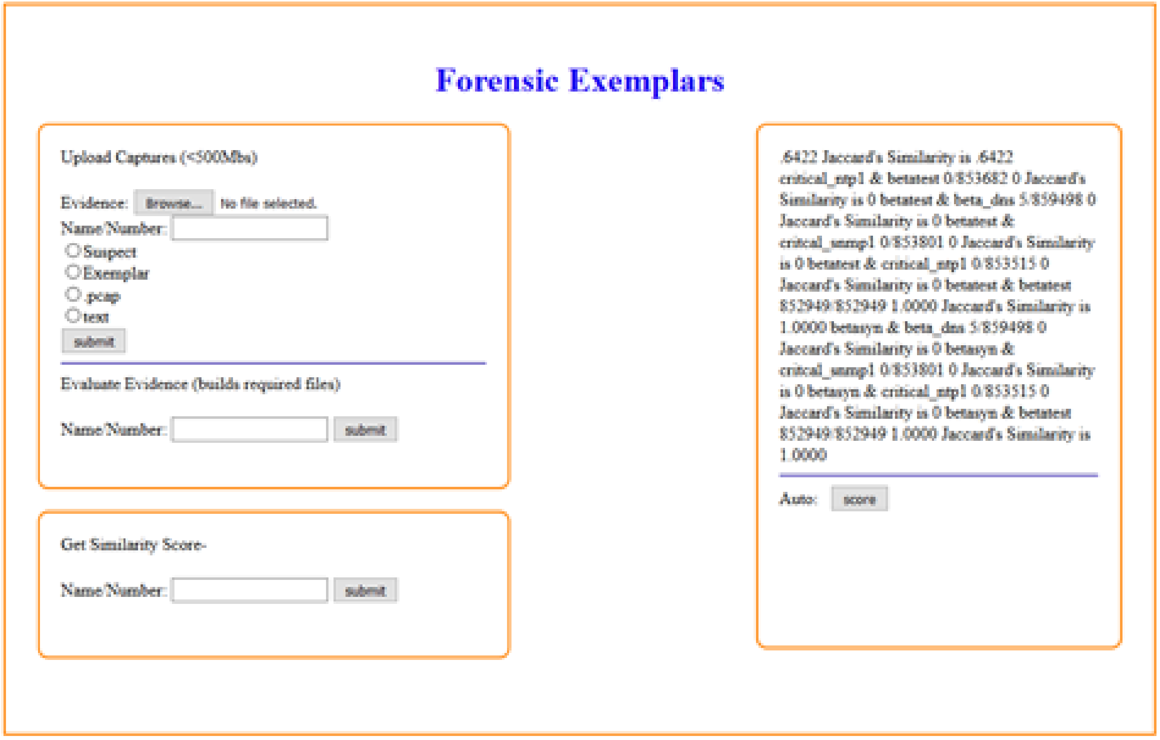
First, we present our approach to computing a similarity score. We derive a similarity score between two data flows using a three step process. First, the relevant parts of the flow are converted to text, so that they can be sorted and counted. Log files, which are already in text format, do not require this conversion step. Second, duplicate information present in a single line is removed to prevent the same value from being inappropriately included. The final step includes the sorting, counting, and comparison of the values in the two groups. Linux is the perfect tool for the final step of this data manipulation as it requires no special programming knowledge, just the basic Linux skill set and the Bash shell. For this project, a collection of known origin is referred to as an “exemplar”. A collection of unknown origin is referred to as a “suspect”.

We also created an application with a graphical user interface (GUI) to assist in the evaluation of two flow collections and provide the similarity score. Several possible use cases were taken into consideration. The first was a tool that automatically collects traffic, makes a comparison against any other flow stored in the system, and reports any score above the predetermined threshold. The second was a tool that would allow the analyst to upload evidence and make a comparison against a known exemplar for a specific stresser. The compromise was a tool that allows the analyst to upload either text or **.pcap captures* from a known exemplar or a suspect and receive the similarity score. The GUI application also compares all of the suspect files in the system to all the exemplars in the system and provides the related similarity scores. The application was developed using easily repeatable and portable Bash scripts accessed through a PHP based web interface. As such, the GUI application is best deployed in a client-server environment and made accessible to users over the Internet.

Lastly, a terminal version, which only provides a similarity score between two flow collections, was also created as a more portable alternative.

3.2 Graphical User Interface – GUI Uploads

The user interface is constructed of familiar textboxes, radio buttons, and submits buttons. Obtaining a score is a three step process; upload, evaluate, and score. The steps were separated to create maximum workflow clarity and minimize loads on the processor. Uploading large files over the Internet can create hourglass moments, as can the sorting and counting of a million IP addresses.



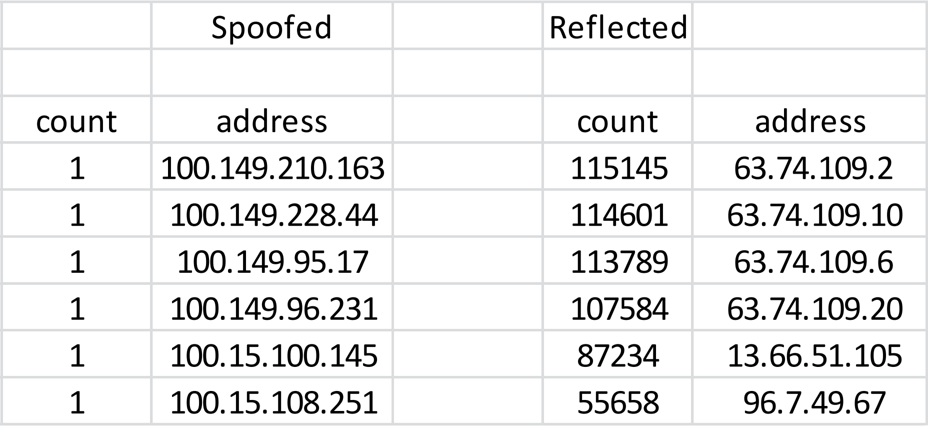
```
.6422 Jaccard's Similarity is .6422
critical_ntp1 & betatest 0/853682 0 Jaccard's
Similarity is 0 betatest & beta_dns 5/859498 0
Jaccard's Similarity is 0 betatest &
critical_snmp1 0/853801 0 Jaccard's Similarity
is 0 betatest & critical_ntp1 0/853515 0
Jaccard's Similarity is 0 betatest & betatest
852949/852949 1.0000 Jaccard's Similarity is
1.0000 betasyn & beta_dns 5/859498 0
Jaccard's Similarity is 0 betasyn &
critical_snmp1 0/853801 0 Jaccard's Similarity
is 0 betasyn & critical_ntp1 0/853515 0
Jaccard's Similarity is 0 betasyn & betatest
852949/852949 1.0000 Jaccard's Similarity is
1.0000
```

FIGURE 1: Snapshot of Tool GUI.

The first step is to upload a file into the system. The upload is the only action taken in this step. The user indicates if the file is a suspect (unknown) or exemplar (known) and if the file is **.pcap* or text. The file upload limitation was set in the *php.ini* at less than 500MB. Since traffic captures can be virtually unlimited in size, this reasonable, yet arbitrary file size, limitation was used. Some problems with file size limitations are demonstrated later in the discussion of flow cross-sections. During upload, a unique identifier, like a case number, is associated with the file.

3.3 GUI Evaluations

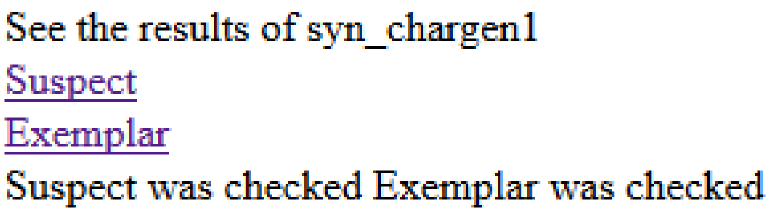
The second step is the evaluation, which is processed independent of the upload. The action is initiated with the unique identifier. The browser's autocomplete assists with accuracy. The creation of comparable IP lists consumes processor cycles, as some of the files can contain one million IP addresses. The files that are uploaded with the **.pcap* radio button selected are processed via *tcpdump* with the *–nnq* switch. The switch *–nn* eliminates the resolution of host or port names. The *–q* switch displays just enough protocol information to eliminate the duplication of IP addresses further into the packet. Text uploads are not processed by *tcpdump*. The evaluation process results in a glimpse of the IP addresses collected, organized by number of appearances. The analysts can determine for themselves if a similarity score would have relevance because the top 100 addresses are displayed. This is another arbitrary value, but shows if the traffic is reflected or spoofed.



	Spoofed		Reflected	
count	address		count	address
1	100.149.210.163		115145	63.74.109.2
1	100.149.228.44		114601	63.74.109.10
1	100.149.95.17		113789	63.74.109.6
1	100.149.96.231		107584	63.74.109.20
1	100.15.100.145		87234	13.66.51.105
1	100.15.108.251		55658	96.7.49.67

FIGURE 2: Comparison of Spoofed vs. Reflected Traffic.

The evaluate function will also report if the two files required for a similarity score are present in the system.



See the results of syn_chargen1

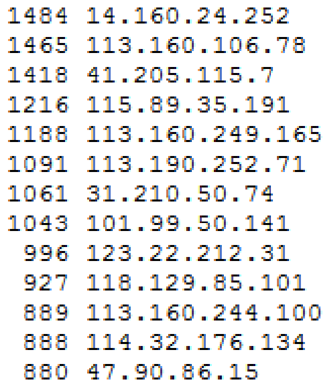
Suspect

Exemplar

Suspect was checked Exemplar was checked

FIGURE 3: Snapshot of File Evaluation.

The list created in this instance (CharGen traffic) demonstrates the view of a reflected attack.

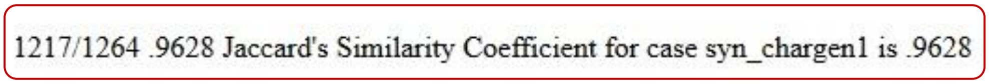


```
1484 14.160.24.252
1465 113.160.106.78
1418 41.205.115.7
1216 115.89.35.191
1188 113.160.249.165
1091 113.190.252.71
1061 31.210.50.74
1043 101.99.50.141
 996 123.22.212.31
 927 118.129.85.101
 889 113.160.244.100
 888 114.32.176.134
 880 47.90.86.15
```

FIGURE 4: Snapshot of Top CharGen Attack IP Addresses.

3.4 GUI Scores

The third step is to obtain a similarity score. This process is case specific. The unique number added during upload is entered to initiate a comparison between the suspect and exemplar files for any single case. The similarity score is simply the Jaccard's Similarity Coefficient of the two lists. In this case, the two lists are the previously processed set of unique IP addresses for both the exemplar and suspect in each case.



```
1217/1264 .9628 Jaccard's Similarity Coefficient for case syn_chargen1 is .9628
```

FIGURE 5: Snapshot of SynStresser CharGen Similarity Score.

3.5 Heavy Lifting with Bash

The Bash script performs the comparison and provides the similarity score. The *$union* is every IP address identified in either of the two lists (exemplar or suspect). The *$intersection* is the IP addresses the two lists have in common. The score is determined using the Linux *bc* command line calculator. The application calculates the intersection/union to four decimal places.

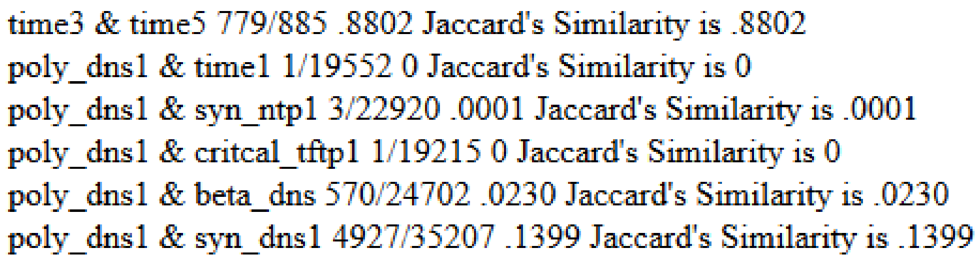
The script is displayed below:

```
##output Common to Both Columns
comm -12 <(sort $suspect) <(sort $exemplar) | uniq
##count of addresses in both columns & create the variable
intersection=$(comm -12 <(sort $suspect) <(sort $exemplar) | uniq | wc -l)
##count of unique values in both & create the variable
union=$(cat $suspect $exemplar | sort | uniq | wc -l)
##show the integers
echo "$intersection/$union"
##show the float score
bc <<< "scale=4;$intersection/$union"
##put the score into a variable
score=$(bc <<< "scale=4;$intersection/$union")
echo "Jaccard's Similarity Coefficient for case $1 is "$score
```

FIGURE 6: Bash Script for Similarity Score.

3.6 Auto-Scoring

The “Auto Score” is an additional feature that compares every exemplar to every suspect file in the system. The process uses a lot of cycles to compare all the values in the system, so it is an on-demand function. The results shown in the box on the GUI in Figure 1 are legitimate, but are present mainly for appearance.



```
time3 & time5 779/885 .8802 Jaccard's Similarity is .8802
poly_dns1 & time1 1/19552 0 Jaccard's Similarity is 0
poly_dns1 & syn_ntp1 3/22920 .0001 Jaccard's Similarity is .0001
poly_dns1 & critcal_tftp1 1/19215 0 Jaccard's Similarity is 0
poly_dns1 & beta_dns 570/24702 .0230 Jaccard's Similarity is .0230
poly_dns1 & syn_dns1 4927/35207 .1399 Jaccard's Similarity is .1399
```

FIGURE 7: Snapshot of Auto Score Results.

3.7 Terminal Version

The terminal version of the tool is capable of providing the same similarity score as the GUI, but does not auto-score against the other samples in the system. The syntax is-

```
jaccard file1 -p file2 -p
```

The `-p` switch is replaced with `-t` in the event the file is text instead of *.pcap*. A whitespace separates the variables. The source code that accompanies this paper is available in the GitHub repository [6].

4. INFORMATION COLLECTION

4.1 Stresser Selection

Reflective attack traffic was collected from Beta Booter, Critical Boot, SynStresser, and Poly Stresser. Network Stresser was never functional. Beta Booter, Poly Stresser, and SynStresser had intermittent periods of inoperability. Stressers were selected via Internet searches and based on literature survey. There were only three criteria; provide reflective attacks, accept Bitcoin, and function. The more reliable stressers were used with greater frequency.

4.2 Fire The Stressers

There were 24 *.pcap* collections conducted between April 10 and April 23, 2017. The two files collected during the proof of concept on February 20 were also included in the dataset. Pairs of DNS, TFTP, SNMP, CharGen, NTP, and SSDP attacks were collected and scored with the tool. The dataset was captured with the use of two Linux Ubuntu servers running *Wireshark* and *libpcap*. The first server was located at Microsoft Azure in Dallas, Texas. The speedtest.com python script measured the download speed at 101 Mbit/s and the upload speed at 298 Mbits/s. A second server was deployed at Linode in Dallas following the Critical Boot Stresser's refusal to attack Microsoft IP addresses. The Linode download speeds were significantly better at 343 Mbit/s. The upload speed was less at 196 Mbits/s. Because of the higher download speeds, Linode was used in the collection of the vast majority of traffic.

DoS attacks were initially set at 30 seconds. The packet capture was stopped manually instead of preselecting a value (i.e. one million packets). Packet sizes were significantly different based on the type of flood utilized. Average packet sizes ranged from 253 bytes in Critical Boot TFTP to 1177 bytes in SynStresser CharGen. Attacks were reduced to 15 seconds in many cases to avoid over collection, since the premise was to keep the files down to a manageable size less than 500MB. Even with the reduction of attack duration, many files were manually reduced in size. For consistency, any reduction contained the initial flow from the beginning to the desired length or number of packets. It is imperative to compare like sections in the flow. This will be demonstrated in detail in the subsequent sections. Flows up to 125,000 packets per second for Poly Stresser NTP Attack were observed.

5. LIMITATIONS

5.1 Flow Cross-Section

Two potential limitations to a successful comparison should be considered. The first will be referred to as the concept of flow ‘cross-section’. Because of latency, it is a foregone conclusion that attack traffic will arrive at the target at different times in any collection. Similarity scores will be affected by the point in the flow in which the IP addresses are compared. A stresser that is allowed to run through the entire list of compromised hosts would create a complete list of source IP addresses. A flow cross-section is an incomplete portion of any attack flow. Investigators hampered by the limitations of the evidence presented to them by their customers must figure out a way to compare like portions.

The graph in Figure 8 demonstrates how a 30 second attack can continue for 148 seconds with diminishing packet presentation. Though the flow diminishes, it can still skew the similarity score because addresses that appear at the beginning are not present toward the end.

The significance of comparing a similar cross-section of traffic by time can be demonstrated by the dissection of the 148 second DNS flow into three even parts consisting of 50 seconds each. Different IP addresses appear at different points in time, as seen in the following:

- In seconds 0-49, 675,056 packets were seen. The IP address 50.19.58.26 located at Amazon in Seattle, Washington was seen 38 times between 0 and .02 seconds of the flow. The address 91.109.0.201 at Mesh Digital in Sarria, Spain was seen 96 times between 0.79 and 29.38 seconds. The address 118.163.97.26 at HINET-NET in Taipei, Taiwan was seen 1435 times between 7.97 and 31.96 seconds.

- In seconds 50-99, 193 packets were seen. The address 59.104.139.111 at SEEDNET-NET in Taipei, Taiwan was seen three times between 50 and 62 seconds. In seconds 100-148, 124 packets were seen. The address 122.18.108.16 at NTT Communications in Tokyo, Japan was seen six times between 110.70 and 122.62 seconds.

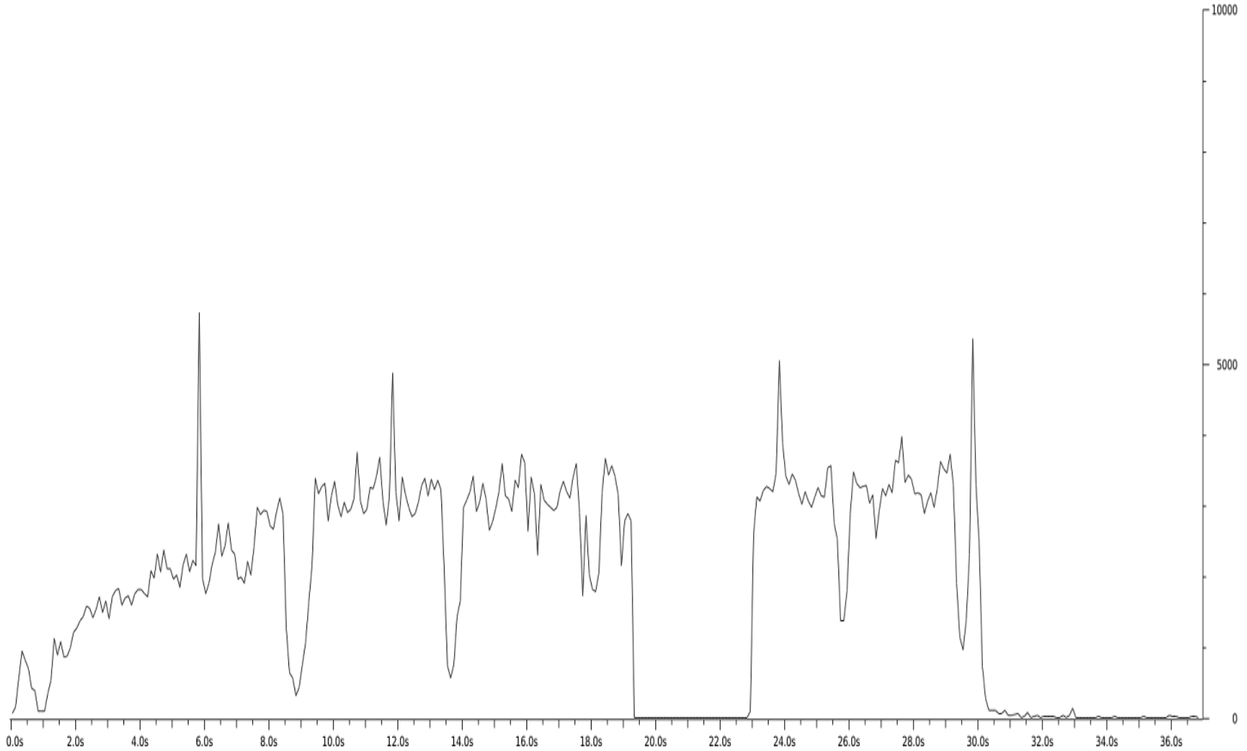


FIGURE 8: Beta Booter DNS (573 MBPS).

The graph in Figure 9 demonstrates how a 15 second attack from the Poly Stresser DNS trickled in for 110 seconds.

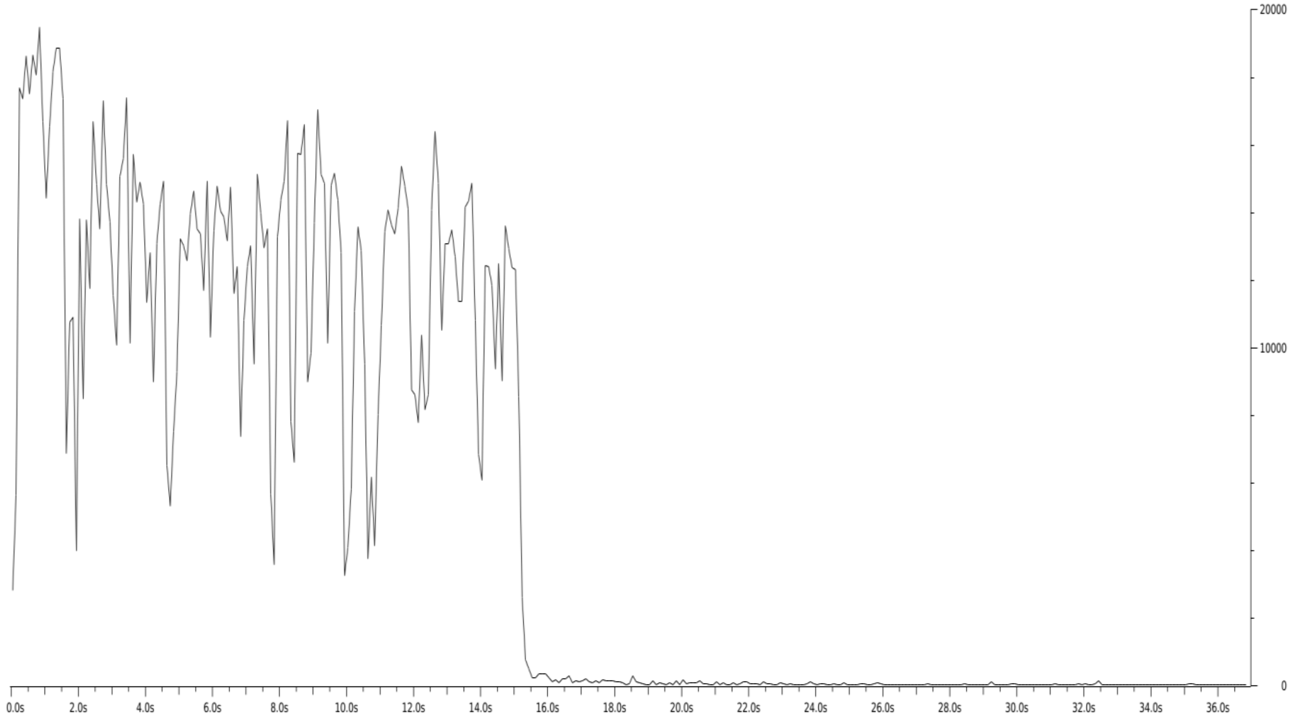


FIGURE 9: Poly Stresser DNS (1.9 GBs)

To further illustrate the cross-section concept, a 3 GB Poly Stresser NTP attack was divided into three equal parts of one million packets each, irrespective of arrival time, to demonstrate the inconsistency of scores based on the cross-section evaluated.

The similarity score of the comparison of the first million packets of the NTP suspect to its exemplar counterpart was .73 (688/942). The score of the second million packets of the NTP suspect to the original exemplar only decreased to .69 (671/965). However, the score of the third million packets of the NTP suspect to the original exemplar decreased to .06 (58/872).

In a final illustration of cross-section limitations, the same NTP flow previously divided into three equal portions of one million packets was compared to each another. Figure 10 demonstrates the inconsistency. A similarity score of 1 is expected between like sections of the NTP flow. A high similarity score of .88 is obtained between the 1st one million packets and the 2nd one million packets (1-2). However, there is only a .12 similarity between the 2nd and 3rd million packets (2-3), and a .08 similarity between the 1st and 3rd million packets (1-3). As illustrated, the similarity scores involving the third section barely resemble those of the first two.

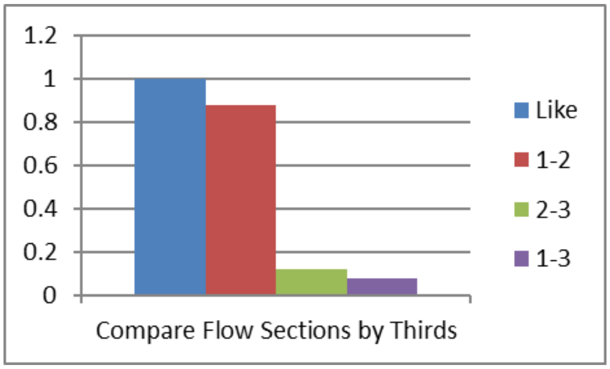


FIGURE 10: NTP Cross-Section Comparison.

Because of this cross-section issue, comparisons in our research project were made strictly to the first package of the attack to the predetermined limit; file sizes of less than 500MB or one million packets. Comparing from the beginning of the flow provided the required consistency.

5.2 Shared Infrastructure

The second limitation may be shared infrastructure. Karami [5] observed that shared infrastructure could impact the validity of similarity scores. However, comparison of like protocols from different stressers in this project matched Santanna et al.'s observations [1] - limited indication of shared infrastructure. Some overlap was expected. Because each comparison includes an exemplar and a suspect file from each stresser, the higher of the two scores is shown.

```
SynStresser NTP and Poly Stresser NTP .03
SynStresser NTP and Critical Boot NTP 0
Poly Stresser NTP and Critical Boot NTP 0
SynStresser CharGen and Poly Stresser CharGen .10
Beta Booter DNS and Poly Stresser DNS .02
SynStresser DNS and Poly Stresser DNS .19
Beta Booter DNS and SynStresser DNS .28
```

FIGURE 11: Similarity Scores for Different Stressers (like protocols).

As indicated, the DNS attacks had the highest probability of sharing compromised hosts, but there is insufficient evidence to substantiate that they share anything other than some number of compromised hosts. These findings in no way eliminate the possibility of the type of infrastructure sharing that could result in high similarity scores among different stressers. The possibility must be taken into consideration.

6. RESULTS

The results of the project's sample collection and tool comparisons are as follows:

NTP - Critical Boot, SynStresser, and Poly Stresser provided NTP attacks. Similarity scores between two flows of the same stresser were greater than those accepted by Karami's [5] minimum threshold score for the NTP protocol (.55). The similarity scores were as follows:

- Critical Boot NTP .64, SynStresser NTP .79, and Poly Stresser NTP .73.

CharGen – SynStresser and Poly Stresser both provided CharGen attacks. Similarity scores between two flows of the same stresser were greater than those accepted by Karami's [5] minimum threshold score for the CharGen protocol (.55). The similarity scores were as follows:

- SynStresser CharGen .96 and Poly Stresser CharGen .75.

SNMP – Critical Boot was the sole provider of SNMP attacks of the stressers evaluated. Karami did not evaluate the SNMP protocol [5]. The similarity score was:

- Critical Boot SNMP .98.

TFTP – Critical Boot was the sole provider of TFTP attacks for the stressers evaluated. Karami did not evaluate the TFTP protocol [5]. The similarity score was:

- Critical Boot TFTP .88.

SSDP - Critical Boot was the sole provider of SSDP attack of the stressers evaluated. Similarity scores between two flows were greater than those accepted by Karami's [5] minimum threshold score for the SSDP protocol (.45). The similarity score was:

- SynStyresser SSDP .71

DNS – DNS attack traffic was compared in Beta Booter, Poly Stresser, and SynStresser. Only one of the three similarity scores between flows of the same stresser was greater than those accepted by Karami's [5] minimum threshold score (.60). The similarity scores were as follows:
Beta Booter DNS .03, Poly Stresser DNS .51, and SynStresser DNS .77.

7. CONCLUSION

By collecting the traffic in a consistent manner and comparing like cross-sections of the attack flow, similarity scores greater than those accepted as correct classifications by Karami [5] can be achieved. In most of the tests, the flows compared were far less than 30 seconds, creating the potential for success with only a limited amount of information.

The reflective attacks NTP, TFTP, SNMP, CharGen, and SSDP provided consistent comparison results. Of course, the ideal comparison would be against every source IP address participating in every reflective attack. In the absence of that much data, consideration must be given to the cross-section of the flow; to such a degree that scores may fall below an acceptable level, even when known to originate from the same source.

In a real-life scenario, the investigator may have no control over the evidence provided by the victim. They may not know if the information was derived from a firewall capable of writing to a log at speeds sufficient for an accurate collection of the evidence or if the victim's network configuration somehow limited the volume of evidence collected. Finally, the possibility of shared infrastructure must be taken into consideration when making a final determination of the identity of a stresser, even though our research project's sample data provided little indication of the likelihood of infrastructure sharing.

Intentionally limiting by the size of the collection or using a purpose-built tool is in no way a requirement for success. Separating a list of source IP addresses from a packet capture can be accomplished with *tcpdump* from the terminal. Sorting, counting, and basic math tools like *bc* are also available in most Linux Distributions.

8. REFERENCES

[1] J. Santanna, R. van Rijswijk-Deij, R. Hofstede, A. Sperotto, M. Wierbosch, L. Z. Granville, & A. Pras. (2015) "Booters—An analysis of DDoS-as-a-service attacks". In *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on (pp. 243-251). IEEE*.

[2] D. A. Wheeler, & G. N. Larsen (2003). "Techniques for cyber-attack attribution" (No. IDA-P-3792). *Institute for Defense Analyses*, Alexandria, VA.

[3] J. Hunker, M. Bishop, & C. Gates. (2010). "Report on Attribution for GENI". *In National Science Foundation Project 1776*, 2010.

[4] M. Kührer, T. Hupperich, C. Rossow, & T. Holz. (2014, Aug). "Exit from Hell? Reducing the Impact of Amplification DDoS Attacks". In *USENIX Security Symposium (pp. 111-125)*.

[5] M. Karami, "Understanding and Undermining the Business of DDoS Booter Services," *Ph.D. dissertation*, Dept. Comp. Sci., George Mason Univ., Fairfax, VA, 2016.

[6] GitHub – prknight/Sam_Project [Online]. Available: https://github.com/prknight/Sam_Project.