Replay of Malicious Traffic in Network Testbeds

Alefiya Hussain, Yuri Pradkin and John Heidemann

University of Southern California / Information Sciences Institute

***Abstract*—In this paper we present tools and methods to integrate attack measurements from the Internet with controlled experimentation on a network testbed. We show that this approach provides greater fidelity than synthetic models. We compare the statistical properties of real-world attacks with synthetically generated constant bit rate attacks on the testbed. Our results indicate that trace replay provides fine time-scale details that may be absent in constant bit rate attacks. Additionally, we demonstrate the effectiveness of our approach to study new and emerging attacks. We replay an Internet attack captured by the LANDER system on the DETERLab testbed within two hours.**

I. Introduction

Testbeds offer a platform for testing and evaluation of networked systems in controlled and repeatable environments. They facilitate studying the impact of network conditions on the security of real network systems and applications [3]. Such study depends on approaches to providing representative network traffic.

Many approaches have been proposed to create controlled malicious and non-malicious traffic. Creating traffic in testbeds always involves some level of modeling. Some models may be completely abstract, such as constant-bitrate attacks. Others create synthetic traffic models, possibly parameterized from real network traffic (such as [10], [17]), or involve tools specific to malicious traffic generation (such as Metasploit [12]). Synthetic traffic models are attractive because they allow control and generation of any amount of traffic with few or no external constraints. Additionally, such models adapt well to testbeds and they have no privacy concerns.

The risk of synthetic models is that one gets out only what one puts in: they may miss the dynamics and subtleties of real-world traffic. Often malicious and regular network traffic reflects not only end hosts, but also aspects of the network path [6], [9]. In our previous work, we have shown attack dynamics are inherently dependent on characteristics such as number of attackers and the attacker host environment, such as the type of operating system, system load, and hardware characteristics [7]. These details are often omitted in synthetic models of malicious traffic.

Another challenge of synthetic models is that they always lag current malicious traffic. There is a delay in deploying new models, since models are designed only when malware is understood, yet attacks change and mutate rapidly. In the paper, we import denial-of-service (DoS) attacks that we have captured in LANDER, a network traffic analysis system [5], to provide malicious traffic models in the DETERLab cyber security testbed [15]. Our approach recreates source and network path conditions by *replaying* the attacks on the DETERLab testbed [15].

The contribution of this paper is to describe an approach for attack capture and immediate replay on a networking testbed. The attack replay tools provide a collective representation of one or more attackers located downstream from the capture point mapped onto a specified node in the testbed (Section II). In Section III, we show that this approach provides greater fidelity than synthetic models. We also show that our approach can quickly feed new attacks into a testbed for evaluation (Section IV), where each attack source may be isolated and modelled individually, as required by some cyber defense strategies.

We expect our tools will provide timely and realistic traffic with which to test cyber-defenses such as intrusion detection systems and firewalls. The data sets, our tools, and the DETERLab testbed are all available for use by network security researchers.

II. Approach

In this paper we discuss a methodology to detect and capture attacks using LANDER, and then replay this traffic in a scenario constructed on the DETER testbed. We discuss our approach in this section and present an analysis of the fidelity this approach may offer in the following section.

A. Capturing Network Traffic with LANDER

We capture network traffic with the LANDER system [5]. LANDER supports capture and anonymization of network traffic, using parallel processing to support high data rates and significant and varying amounts of analyst-driven-processing. It is currently in operation, with small deployments running on single (but multi-core) systems with commodity network cards, and large deployments capturing traffic at 10 Gb/s with back-end anonymization and analysis on shared 1000-node compute clusters.

For this paper, the important aspects of LANDER are that it provides the policies that enable us to get permission to collect packet, the flexibility to plug in different detection modules to detect interesting events, and the framework to extract anonymized traces in near-real time. LANDER is built around multiple queues of data, each of which triggers a callback to a system- or user-provided function. This structure supports differing policies for data anonymization. By default, data is placed into a raw queue, then anonymized, by changing IP addresses using prefix-preserving anonymization [18] and removing all application payloads. This range of policies is important to enable deployments from environments where high expectations of privacy must be enforced, to laboratories, where it can serve as a tool to audit consenting but not-fully-

trusted parties. It allows us to experiment with different analysis tools, such as our DoS-detection methods [6], as well as off-the-shelf tools like SNORT [2], Bro [13], and Suricata [1]. Once an attack is identified, we extract the packet headers for replay.

B. DETERLab: A playground for Attacks

The DETERLab provides infrastructure, tools, and methodologies for networking and cyber security experimentation. The DETERLab testbed includes 570 PCs and specialized hardware located at two sites; USC/ISI in Marina del Rey, CA and UC Berkeley, Berkeley, CA. The two sites are interconnected with a shared 10Gbps links. The DETERLab containerization tools enable researchers to allocate internet-scale topologies using virtualization of nodes. DETERLab also provides the Montage AGent Infrastructure (MAGI) tools for orchestrating a wide range of networking and cyber security experimentation scenarios [15]. The MAGI tools provide an event-based control overlay for deploying and controlling *agent* modules on the topology. Each agent enacts a particular behavior, for example, a attacker behavior, and the behavior is configured and controlled through an Agent Activation Language (AAL) [15].

C. Replaying An Attack Scenario in DETERLab

DETERLab provides a playground for attack experiments, and the MAGI tools allow researchers to systematically explore the experimentation design space. The contribution in this paper is to take traces from the real world with LANDER, and recreate the attack in DETERLab using the MAGI framework. To make this transformation, we developed an new agent for MAGI, and a scenario to replay the attack.

To replay the trace with MAGI we developed a *attack-replay* agent. It uses the Tcpreplay toolset [14] to adapt the real-world trace to the testbed. It uses tcprewrite to remap the source and destination MAC and IP addresses in the LANDER attack trace. If the original attack makes use of IP address spoofing, the source IP addresses are not modified. It then uses tcpreplay to regenerate the attack with proper timing and replay it from one or more locations on a topology in the testbed.

We generate two types of traffic inside DETER. Attack traffic and non-malicious (or background) traffic. We generate attack traffic using the attack-replay agent. We deploy additional agents to generate non-malicious traffic. We mix the two, breaking the scenario into three phases: (i) *pre-attack* phase during which we generate only non-malicious network traffic. (ii) the *attack* phase, during which we deploy the attack into the topology and observe how it interacts with continuing non-malicious traffic. (iii) The *post-attack* phase when non-malicious traffic recovers from the attack.

We generate web traffic to create non-malicious network traffic throughout the duration of the experiment. We use the MAGI webserver and webclient agents. The MAGI webserver agent deploys an Apache web server. The MAGI webclient agent uses Curl to requests objects from the web server. The webclient agent can be configured to choose a webserver in sequence or at random from a list of one or more web servers. The size of the web page can be defined as a probabilistic distribution.

We compare our replay of a real attack to an alternative attack, where we generate synthetic attack traffic with a MAGI *attack-CBR* agent. The attack-CBR agent generates a constant bit rate packet rate stream on the network that can be used to represent an attack. The agent can be parametrized with protocol, packet size distributions, port number distributions, ramp up and ramp down rates, low and high attack rates.

Lastly, we deploy a *packetCapture* agent to capture all the packets seen at the observation point in the topology. The packetCapture agent uses tcpdump to capture packets. It can be configured with a packet filter, capture packet size, and the capture file storage location.

III. Richness in Replayed Attacks

We present an analysis of the additional fidelity real-world attacks offer in this section. First, we discuss how the tools from the last section come together to allow us to systematically explore and compare the different attacks in DETERLab. We then provide a qualitative and quantitative analysis of the attack replay as compared to synthetically generated CBR attacks.

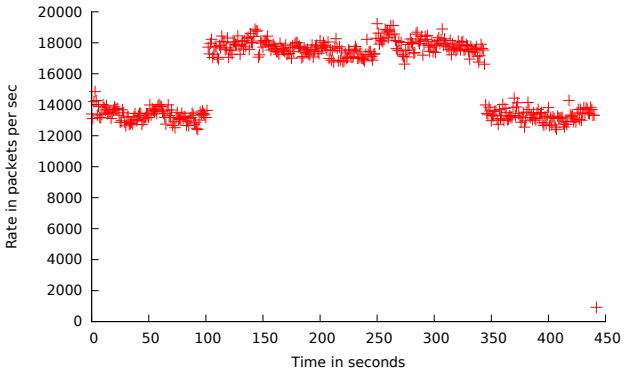
A. Traces to Experiments

We now describe how we create three types of scenarios to explore the richness of the replayed LANDER attacks and compare them to synthetically generated CBR attacks on the DETER testbed.

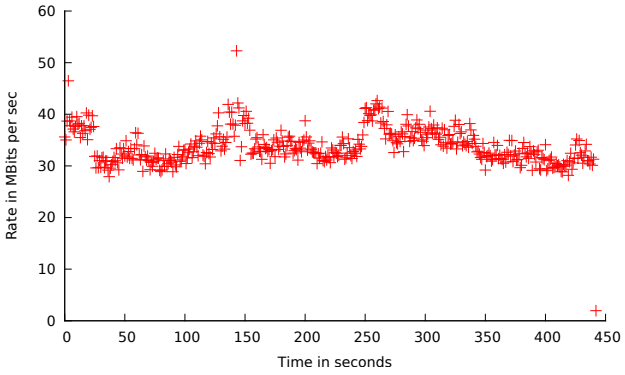
We conduct three controlled experiments on the DETER testbed: a *replay* of a real-world attack [16], a *single-attacker, constant-bit rate, synthetic* attack with a rate of 4200 packets/s, a *ten-attacker, constant-bit rate, synthetic* attack with an aggregate rate of 4200 packets/s (each attacker at 420 packets/s).

Here we use a dataset from lander with three separate denial-of-service attacks captured in 2002 and 2003 [16]. In this paper, we use the first attack, a reflector attack that send ICMP echo reply packets to a victim in Los Nettos. We refer to this attack as LANDER-2002. The mix of attack and background traffic is shown in Figure 1, both as packets per second and bits per second. The attack consists of echo reply packets from approximately 140 attackers, with each packet carrying a 28 byte payload. With small packets, we see the attack most clearly in Figure 1a; it hardly shows up when one measures bitrate (Figure 1b). We choose this attack from the dataset since it had the lowest variance in the attack rate. Even though visually this attack looks similar to a constant bit rate attack, in the next section, we systematically explore the richness in this attack as compared to synthetically generated CBR attacks.

We first process this trace to filter out attack traffic targeted to the victim. This attack was then transferred to



(a) Attack and background traffic in packets/s.



(b) Attack and background traffic in Mbits/s.

Fig. 1: Real-world Captured DoS attack LANDER-2002.

the DETER testbed and processed by the MAGI *attack-replay* agent to generate an attack on the experiment topology. Additionally, we used the MAGI *attack-CBR* agent to generate the two constant bit rate attacks as discussed above.

We generate non-malicious background traffic using four webclients agents and two webserver agents on a dumbbell topology. The web clients request a constant webpage of 10MB from a randomly chosen webserver. The attack is launched from a bank of attackers attached to one webserver. We deploy the packetCapture agent at an observation point on the bottleneck link. It captures all packets in the experiment with zero packet loss. The complete experiment description along with the traces will be made available on publication [4].

B. Attack Traffic

As discussed in the previous section, we compare two types of synthetic attacks with attack trace replay to study the dynamics.

Figure 2 show the attack rate in packets/second and the empirical probability distributions for all three types of attack. The top row shows the attack packet rates. The x-axis show time in seconds and the y-axis shows the rate in packets/second. We have parametrized the constant bit rate attacks based on the LANDER-2002 attack rate. We observe an attack rate of approximately 4200 packets/sec in all three cases. The bottom row shows both the empirical probability distribution function and the cumulative distribution function of the attack rates. The x-axis is the packet rate sampled at 1 millisecond to expose fine-timescale dynamics of the packet rate. Bars in the plot and the left y-axis show the probability distribution function as the normalized frequency of the observed packet rate. The line and the right y-axis show the packet rates plotted as a cumulative distribution function.

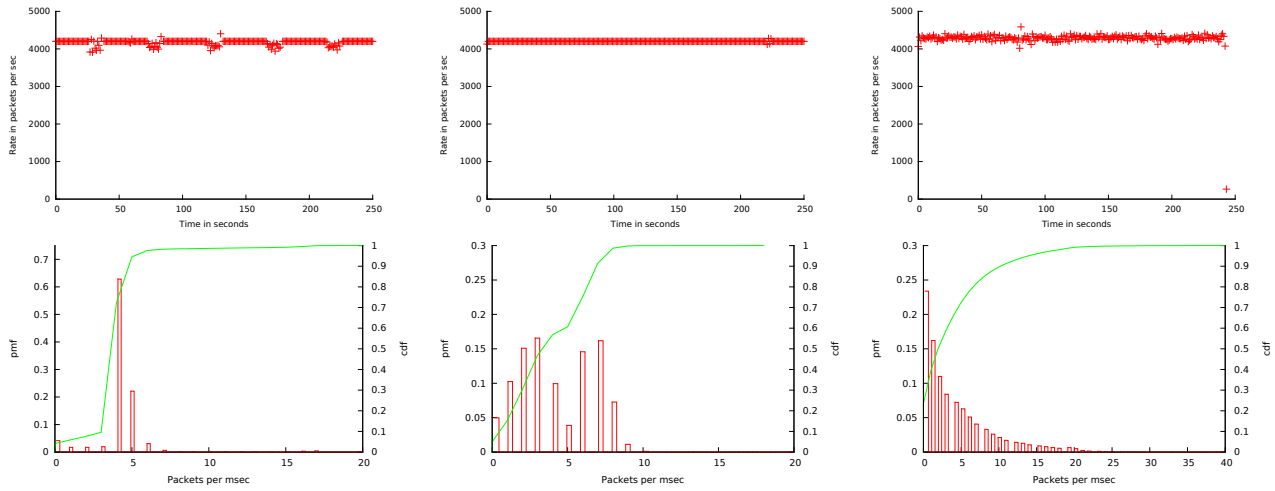
Qualitative differences in testbed attacks: We first look at overall features in the different methods of generating testbed attack traffic. We begin with the trace replay in Figure 2c. On the top right, we see the attack traffic is "noisy" with lots of fine variation. The bottom right graph shows that the exact attack rate varies quite a bit in each millisecond with a wide range of possible values, since both the PDF and the CDF show a smooth variation over a continuous range. We argue that these variations are not unusual, but are an *inherent* part in any real-world distributed denial-of-service attack. These variations arise because attacks originate on different types of hardware, sometimes with other software competing for compute cycles. They also traverse different networks en-route to their target, experiencing different kinds of cross-traffic, including delay and sometimes loss. We have seen these effects before in wide-area traces and explored them through controlled experiments [6], [7]. In this experiment we show that through replay, we can recreate these subtleties on the testbed.

By contrast, Figures 2a and Figure 2b show much less richness and variation. The synthetic, single-source CBR attacker in Figure 2a shows very regular traffic occasionally interrupted by bursts of variation (top graph, with bursts at about 45 s, 85 s, 125 s, etc.). The distribution of arrivals,however, is strongly modal, with 4 or 5 packets per millisecond 80% of the time. There is far less variability than the real-world attack. This lack of variability presents a much different attack to an intrusion detection system—an IDS that would see this periodic traffic would easily find this synthetic attack while missing the more complex, real-world one.

Figure 2b shows that more synthetic attackers provide a richer attack. Each attacker sends at a lower rate, but the aggregate rate is the same. We see more variation in the CDF (Figure 2b bottom graph), because each of the 10 attackers is slightly different. However, the aggregate traffic (top graph) is much *smoother*—the 10 attackers jitter largely cancels each other out.

We conclude that real-world traffic shows variation and jitter at timescales of milliseconds (Figure 2c). This richness is lost with synthetic attacker, even when run as teams on multiple machines. If a testbed is to reproduce the kind of real-world complexity it must employ more sophisticated tools and methodologies than just simple synthetic attacks.

Quantifying differences in testbed attacks: To quantity the difference in the spread and variance we compute the average absolute deviation (AAD). For a given sampling bin of p seconds, we define the arrival process $x(t)$ as the number of packets that arrive in the bin $[t, t+p)$.



(a) CBR: Single Attacker (b) CBR: Ten Attackers (c) Replay of LANDER-2002 trace

Fig. 2: The packet rates and empirical distributions of attack packet arrival times for the three types of attacks used in the experiments in DETERLab.

Thus, a T second long packet trace will have $N = T/p$ samples. The AAD is then defined as

$$AAD = \sum_{i=1}^{N} |(x(i) - \bar{X})|/N$$

where $\bar{X}$ is the mean of the attack rate. Since this measure does not square the distance from the mean, it is less affected by extreme observations than metrics such as variance and standard deviation. Table I summarizes the mean, median, standard deviation and the average absolute deviation for the three types of attack reply.

The constant bit rate attacks have a similar mean and median value but for the LANDER-2002 attack, the mean is almost twice the median value. The results indicate that there is a wide range of attack packet arrival rates for real-world attacks when considering the fine time scales. The arrival process is *averaged out* as the size of the sampling bin p increases. As seen in the top row on the Figure 2, all three attacks qualitatively look similar when sampled at 1 sec. However, as reported in Table I, the statistical properties of the attacks are very different when the attacks are sampled at 1 msec. We see that standard deviation is much higher as more attackers participate, and the LANDER-2002 trace has the largest standard deviation of all.

The AAD for the single source attack is the smallest indicating this experiment closely resembles a constant bit rate of 4200 packets/sec. The AAD for the other two experiments in significantly higher indicating high variability in the arrival rate of the attack packets.

These measures demonstrate the large difference between simpler synthetic attacks and trace replay. Taken with the qualitative comparison, we argue that it is essential to use trace replay if fine timescale details of attacks matter to whatever mechanism is being studied in a testbed.

metric (in pkts/ms)	**synthetic (CBR)** **single**	**multiple**	**LANDER-2002 trace**
mean	4.20	4.20	4.11
median	4.00	4.00	2.00
std. deviation	1.77	2.46	4.78
avg. abs. dev.	0.84	2.18	3.62

TABLE I: Statistical properties of two kinds of synthetic attack and captured traces.

IV. Rapid Attack Turn-around: Today's Attacks Today

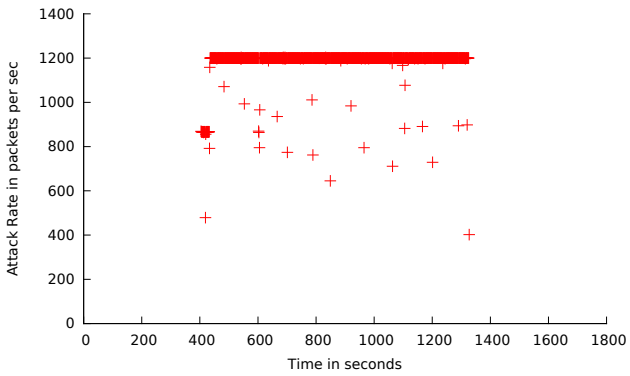
Trace replay has the promise to rapidly turn around attacks into reusable models. This speed is possible because replay of traces does not require understanding of their contents. They can therefore be used to quickly test new defenses, in parallel with analysis of the underlying attack that can lead to more parsimonious, parameterized models that will eventually explore a wider range of scenarios.

To demonstrate the ability to rapidly take an observation to a replayable tool in a testbed, we next carry out an experiment to demonstrate the process. Our target is a DNS amplification attack [8], such as those that took place in March 2013 [11]. In principle, we could watch our network for a new attack to appear. However, because of publishing deadlines, we instead decided to stage our own mini-attack.

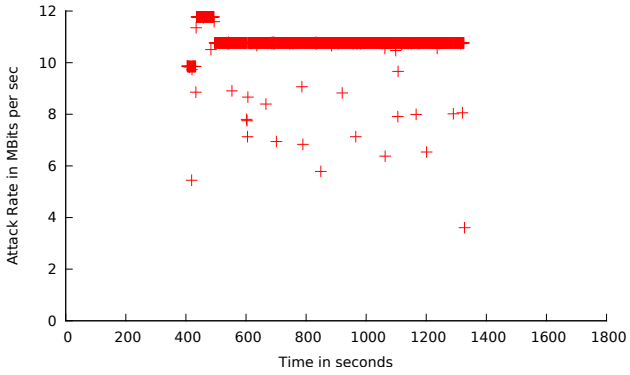
We deployed six DNS servers at ISI in Marina del Rey, California, U.S.A. An trigger computer, also at ISI, then sent a stream of 400 DNS queries per second at these servers with a spoofed source address of a machine loaned to us by Christos Papadopoulos at Colorado State University (CSU) in Ft. Collins, Colorado. We recorded traffic as it leaves ISI and as it enters CSU. We detect the attack with a custom SNORT rule. We then extract the

event	start	duration
Start of 1st segment with attack start	-0:45	1:13
Attack begins	0:00	15:23
End of 1st segment with attack start	1:13	—
SNORT detection under LANDER	6:52	19:57
Start of last segment with attack end	14:28	1:46
Attack ends	15:23	—
End of last segment with attack end	16:14	1:46
Processing trace files	22:09	6:58
Moving trace files to DETER	35:22	2:20
Swap in a topology in DETER	80:40	9:13
Process and deploy traces by MAGI	85:40	22:15

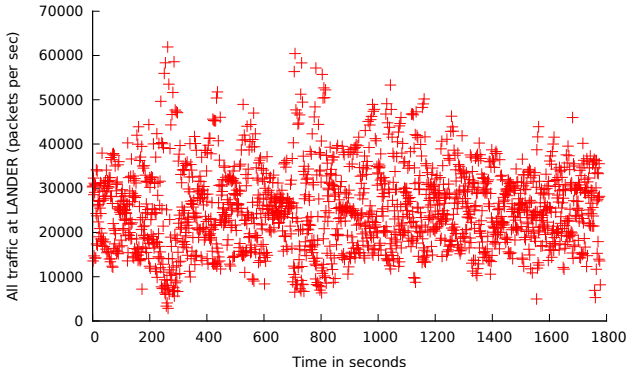
TABLE II: Timing (in minutes:seconds) of our experiment for rapid attack turn-around.



(a) DNS Reflection attack in packets/s



(b) DNS Reflection attack in Mbits/s



(c) All traffic at LANDER in packets/s

Fig. 3: A DNS amplification attack generated at ISI and captured by ISI LANDER system.

traces and hand them to MAGI for exploration in DETERLab. Figure 3 shows the attack and background traffic captured by LANDER during the attack. The attack starts at 403 seconds and lasts for about 900 seconds. Each attack packet is a 3700B DNS query response packet which is fragmented by the network into three packets. The rate changes seen at the start of the attack are due to us exploring different attack parameters for the experiment.

Table II shows the events for trace detection through replay. LANDER splits data into 512 MB segments as it arrives, and processes segments concurrently with collection. The attack lasts for only about 15 minutes and spans 10 segments. We see that although the attack lasts for more than 16 minutes, the first segment with the start of the attack completes about 1 minute into the attack. The attack continues, but this segment then becomes available for analysis. The change in traffic rates triggers SNORT about 7 minutes into the experiment. SNORT runs quickly (about 5 s per segment), but needs to examine 1 minute of traffic to establish that the attack traffic has changed the baseline from regular operation. Due to this parallelism, we detect the attack while it's still in progress.

We do not begin processing the trace files until the attack completes. When it completes, we convert the traces to pcap format, compress them, and move them from the LANDER system onto shared network. The 17 minutes of traces are 1278 MB of data compressed. This processing takes about 7 minutes. If our system was automated, processing and transfer could run concurrent with the attack, reducing this time greatly.

Once LANDER detects and provides the attack trace files, we copy the data to the DETER testbed's shared file system from where it can be accessed by the MAGI attack-replay agent during the experiment. Transferring the data to DETER takes about 2.5 minutes. We now swap-in an experiment and orchestrate it using the MAGI toolkit. Once the attack-replay agent is deployed on the attacker nodes in the topology, the agent makes a copy of the trace file on the local disk for processing. The agent then installs the tcpreplay tools, uncompresses the trace file, and changes the victim IP address and source and destination MAC addresses in the trace. Processing the trace file for replay and deploying it into the testbed takes about 22 minutes. The attack-replay agent signals the orchestrator that it is ready to start the experiment. For the scenarios explored in the paper, the attack is replayed 100 seconds after the start of the non-malicious traffic.

This exercise shows that we are able to turn around an attack into a testbed experiment in less than two hours. The main uncertainty in the process is attack detection, moving data from the locations of collection to replay, and human decision making. Our current approach is far from optimized. With better pipelining, we believe we could

replay attacks while they are still underway. Also with greater automation, some steps that currently are human-driven can be removed. We believe this simple exercise shows the potential of our approach for researchers requiring fresh data.

V. Conclusions

This paper has outlined the advantages and the potential of trace replay as a means of improving network security research. We show that replaying real–world attacks allows including fine time scale detail that are absent in synthetically generated constant bit rate attacks. Integrating LANDER data sets into the DETERLab provides higher quality data for testbed experiments. The methodology presented in this paper provides a way to rapidly incorporate *new* attacks into DETERLab experiments. Our tools and datasets are available for public use [4]. Please contact the authors if they could be of use to you.

Acknowledgments

We thank Christos Papadopoulos and Kaustubh Gadkari of Colorado State University for helping us to carry out the experiment in Section IV as the cooperative victim of the attack.

The research done by Alefiya Hussain is supported by the Department of Homeland Security and Space and Naval Warfare Systems Center, San Diego, under Contract No. N66001-10-C-2018.

Research done by Yuri Pradkin and John Heidemann in this paper is partially supported by the U.S. Department of Homeland Security Science and Technology Directorate, Cyber Security Division, via SPAWAR Systems Center Pacific under Contract No. N66001-13-C-3001. John Heidemann is also partially supported by DHS BAA 11-01-RIKA and Air Force Research Laboratory, Information Directorate under agreement number FA8750-12-2-0344.

The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of SSC-Pacific.

References

[1] Eugene Albin and Neil C. Rowe. A realistic experimental comparison of the Suricata and Snort intrusion-detection systems. In *Proceedings of the 26th International Conference on Advanced Information Networking and Applications Workshops*, pages 122–127, Fukuoka, Japan, March 2012. IEEE.

[2] Jay Beale. *Snort 2.1 Intrusion Detection*. Syngress, second edition edition, May 2004.

[3] Terry Benzel. The science of cyber security experimentation: the DETER project. In *Proceedings of the 27th Annual Computer Security Applications Conference*, ACSAC '11, pages 137–148, New York, NY, USA, 2011. ACM.

[4] Alefiya Hussain. Magi replay agent and lander-2002 attack secnario. Tools and data available at `http://montage.isi.deterlab.net/magi/hst2013tools`, July 2013.

[5] Alefiya Hussain, Genevieve Bartlett, Yuri Pryadkin, John Heidemann, Christos Papadopoulos, and Joseph Bannister. Experiences with a continuous network tracing infrastructure. In *Proceedings of the ACM SIGCOMM MineNet Workshop*, pages 185–190, Philadelphia, PA, USA, August 2005. ACM.

[6] Alefiya Hussain, John Heidemann, and Christos Papadopoulos. A framework for classifying denial of service attacks. In *Proceedings of the ACM SIGCOMM Conference*, pages 99–110, Karlsruhe, Germany, August 2003. ACM.

[7] Alefiya Hussain, John Heidemann, and Christos Papadopoulos. Identification of repeated denial of service attacks. In *Proceedings of the IEEE Infocom*, page to appear, Barcelona, Spain, April 2006. IEEE.

[8] Georgios Kambourakis, Tassos Moschos, Dimitris Geneiatakis, and Stefanos Gritzalis. A fair solution to DNS amplification attacks. In *Proceedings of the Second IEEE International Workshop on Digital Forensics and Incident Analysis (WDFIA)*, pages 38–47. IEEE, August 2007.

[9] Dina Katabi and Charles Blake. Inferring congestion sharing and path characteristics from packet interarrival times. Technical Report MIT-LCS-TR-828, Massachusetts Institute of Technology, Laboratory for Computer Science, 2001.

[10] Kun-Chan Lan and John Heidemann. A tool for RApid Model Parameterization and its applications. In *Proceedings of the Workshop on Models, Methods and Tools for Reproducible Network Research (MoMeTools)*, page to appear, Karlsruhe, Germany, August 2003. ACM.

[11] John Markoff and Nicole Perlroth. Attacks used the Internet against itself to clog traffic. *New York Times*, page B1, March 28 2013.

[12] Metasploit Framework Website. http://www.metasploit.com/.

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

[14] tcpreplay toolkit. http://tcpreplay.synfin.net.

[15] The DETER Project. http://www.deter-project.org.

[16] USC/LANDER project. Scrambled Internet trace measurement dataset, predict id `usc-lander/dostraces-20020629`. Provided by the USC/LANDER project http://www.isi.edu/ant/lander., June 2002. Traces taken 2002-06-29 to 2003-11-30.

[17] Michele C. Weigle, Prashanth Adurthi, Félix Hernández-Campos, Kevin Jeffay, and F. Donelson Smith. Tmix: a tool for generating realistic tcp application workloads in ns-2. *SIGCOMM Comput. Commun. Rev.*, 36(3):65–76, July 2006.

[18] Jun Xu, Jinliang Fan, Mostafa H. Ammar, and Sue B. Moon. Prefix-preserving IP address anonymization: Measurement-based security evaluation and a new cryptography-based scheme. In *Proceedings of the 10th IEEE International Conference on Network Protocols*, pages 280–289, Washington, DC, USA, November 2002. IEEE.