

# High Fidelity Denial of Service (DoS) Experimentation

Roman Chertov, Sonia Fahmy, Ness B. Shroff  
Purdue University

## I. INTRODUCTION

Experimentation with security attacks introduces additional requirements compared to traditional networking and distributed system experiments. High capacity attack flows can push systems beyond their expected operational regions, and expose unexpected behaviors. Many popular simulation and emulation environments fail to account for such behaviors, and incorrect results have been reported based on experiments conducted in these environments. In addition, simulation and emulation environments sometimes introduce artifacts, altering the experimental outcome and its interpretation. Finally, identification of systems settings that significantly impact experimental results is crucial for creating repeatable experiments.

In this paper, we present the results of a careful sensitivity analysis we have conducted, which exposes difficulties in obtaining meaningful measurements from three emulation testbeds: DETER at <http://www.isi.deterlab.net/>, Emulab at <http://www.emulab.net/>, and Wisconsin Advanced Internet Laboratory (WAIL) at <http://www.schooner.wail.wisc.edu> with default system settings. We compare these results to ns-2 simulation results, and find dramatic differences between simulation and emulation results for Denial of Service (DoS) attack experiments. We select low-rate TCP-targeted DoS attacks as a case study, since these attacks have generated significant interest in the research community in the past few years. To validate our comparisons, we use a simple analytical model of TCP performance degradation, in the presence of a special case of TCP-targeted DoS attacks (those not causing timeouts), as a lower bound. Our results reveal that software routers such as Click provide a flexible experimental platform, but require understanding and manipulation of the underlying network device drivers. We also discuss our future work plans for creating higher fidelity network simulation and emulation models that are not computationally prohibitive.

The remainder of this paper is organized as follows. Section II summarizes related work. Section III describes the simple analytical model we have developed. Section IV explains the experimental setup that we use. Section V summarizes our results and the problems in achieving high fidelity DoS simulation and emulation. Finally, Section VI concludes the paper.

– This research has been sponsored in part by NSF/DHS grant 0335247 and NSF grant 0523249.

– Roman Chertov and Sonia Fahmy are with the Department of Computer Science, 250 N. University St., West Lafayette, IN 47907–2066, USA. Tel: +1-765-494-6183. Fax: +1-765-494-0739, E-mail: {rchertov,fahmy}@purdue.edu. Ness B. Shroff is with the School of Electrical and Computer Engineering, 465 Northwestern Ave., West Lafayette, IN 47907–2035, USA. E-mail: shroff@ecn.purdue.edu

## II. RELATED WORK

This section briefly describes simulation and emulation tools that are currently in use by the research community. Additionally, we give a brief background on low-rate TCP-targeted DoS attacks.

### A. Network Simulation

Probably the most well known network simulator is ns-2 [11]. This discrete event simulator has very basic link models and is mainly used for queuing and end-to-end protocol research. Routers are represented as collections of output link queues, ignoring their physical characteristics. On the other end of spectrum, the OPNET simulator [1] is a very detailed commercial simulator. It contains a diverse collection of device models that include routers, switches, servers, and mainframes. Users have the option of creating their own models for the devices. However, a plethora of complicated models is detrimental to scalability, as it becomes prohibitively expensive to run large scale topologies and there is no guarantee that the models are correct.

### B. Network Emulation Tools and Emulators

Network emulation tools range from those emulating large segments of a network to those that shape a single link. Tools like *tc*, *iproute2*, NIST-net, DummyNet, NetPath, and Click allow link shaping. Emulators like Modelnet and EMPOWER are capable of emulating large scale topologies; however, they emulate the network core leaving only the edge nodes to the user. The tools emulate the core connectivity, requested delays, loss, and link bandwidth. However, critical properties of real devices such as queuing delays, maximum packet forwarding rates, policies, and queue sizes are not accurately emulated, thus reducing the fidelity of the experiments that can be carried out.

### C. Low-Rate TCP-Targeted Denial of Service Attacks

Most well-publicized DoS attacks have utilized a large number of compromised nodes to create constant high-rate flows towards the victims. Such “flooding attacks” are effective, but have major shortcomings from the attacker’s perspective. First, the attacks are easy to detect due to the high volume of uniform traffic, e.g., UDP or ICMP. Second, the attacks can self-congest at some bottleneck and not reach the intended destination.

An attack that is less susceptible to these limitations is the low-rate TCP-targeted attack (Figure 1), introduced in [7]. This attack has generated significant interest in the research community, due to its potential to do great harm, go undetected, and the ease by which it can be generated. One variant of low-rate TCP-targeted attacks [5] is an attack that exploits the TCP Additive Increase Multiplicative Decrease (AIMD) mechanism

to cause TCP goodput degradation. The premise is that during the *congestion avoidance phase*, when packet losses occur due to attack pulses, TCP halves its congestion window, but when a successful transmission occurs, it only linearly increases its window size. Such an attack can be used to strategically target key routers or servers in the network, thus causing wide-spread degradation of TCP performance. Moreover, this attack may be difficult to detect, since it does not operate at a known frequency as in the case of the attack in [7]. Therefore, we use this attack variant as a case study in our work. This attack has not been experimentally studied in prior work, except in extremely limited settings, with no sensitivity analysis, or comparisons to analytical or simulation results.

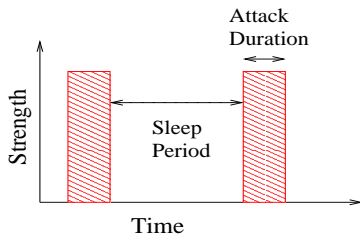


Fig. 1. Low-rate TCP-targeted DoS attack

### III. TCP THROUGHPUT DEGRADATION MODEL

In this section, we describe a simple analytical model, which is a special case of a model given in [8]. The model characterizes TCP performance degradation as a function of the TCP-targeted attack frequency. In prior work, e.g., [10], models of TCP throughput as a function of the round-trip time and loss event rate were developed. These models, however, do not consider the presence of periodic attacks. In contrast, we compute the average TCP window size as a function of the TCP-targeted attack parameters.

As discussed in Section II-C, the objective of this variant of the attack is to exploit the TCP AIMD mechanism and not to cause timeouts. Our analysis assumes that TCP Reno [2] in the congestion avoidance phase is being employed for a single flow under attack. Since Reno can typically recover from a single packet loss without a timeout, it is assumed that *every* attack pulse will induce a packet loss. A loss of a single data packet will cause a reduction of the congestion window by half in TCP Reno, after which additive increase will be employed. For simplicity of the analysis, the short fast recovery phase is ignored. The resulting TCP congestion window saw-tooth pattern is depicted in Figure 2 for a fixed attack frequency. Observe that the model also gives a close approximation of the behavior of TCP New Reno [4] or TCP SACK [9] *even* with a few packet losses with every pulse, since these TCP flavors can typically recover from multiple packet losses without timeouts.

For brevity, we omit the detailed derivation of the throughput, which can be found in [3]. We find that the average congestion window size  $W_{avg} = \frac{3t}{4rtt}$ , where  $t$  is the sleep duration and  $rtt$  is the round trip time of the TCP flow.

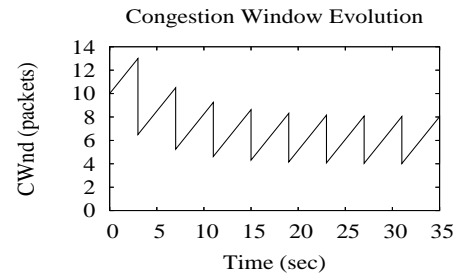


Fig. 2. Saw-tooth pattern of congestion window evolution due to periodic loss every 4 seconds.

### IV. EXPERIMENTAL SETUP

The topology we use for both simulation and emulation experiments is depicted in Figure 3. This is a simple dumb-bell topology with four end systems and two routers that connect via a link with 60 ms delay. The attacker and the attack sink are varied from one side of the topology to another.<sup>1</sup> The same basic ns-2 script is used for both simulations and DETER and Emulab testbed experiments. Due to the specific requirements of the WAIL testbed, we had to modify the *tc* script used on WAIL, so that access links have 40 ms delay and the backbone link has no delay. We have validated the results of ns-2, Emulab, and DETER for the same setup, and found them to be the same as the results with the topology in Figure 3. To create a long lived TCP flow, we used the *ttcp* tool on the testbeds to transfer a large file. Precise details about the DETER and Emulab machines can be found in [3].

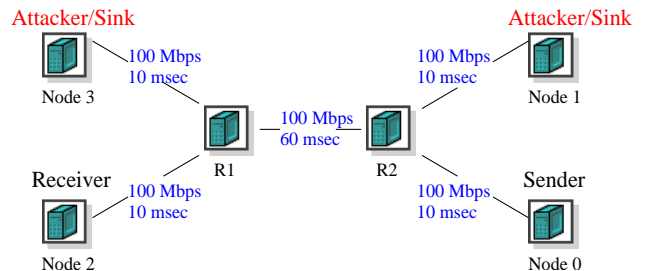


Fig. 3. Simple dumb-bell topology with 160 ms round-trip-time and 100 Mbps links.

### V. EXPERIMENTAL RESULTS

In a series of experiments on DETER, Emulab, and WAIL, we have varied the duration of the attack sleep period to change the impact of the attack on the single TCP file transfer. The length of the attack pulse was set to 160 ms which is the RTT for the TCP flow. A sample of the results is given below to demonstrate the sensitivity of the results to testbed capabilities and settings. The complete results can be found in [3].

#### A. DETER and Emulab Experiments

In this section, we compare the results from the analytical model given in Section III and the ns-2 simulator with results

<sup>1</sup>This simple topology is not representative of the Internet, but we have selected it in order to be able to analyze the results in depth. Our future work plans include experiments with multiple bottleneck configurations and other traffic patterns.

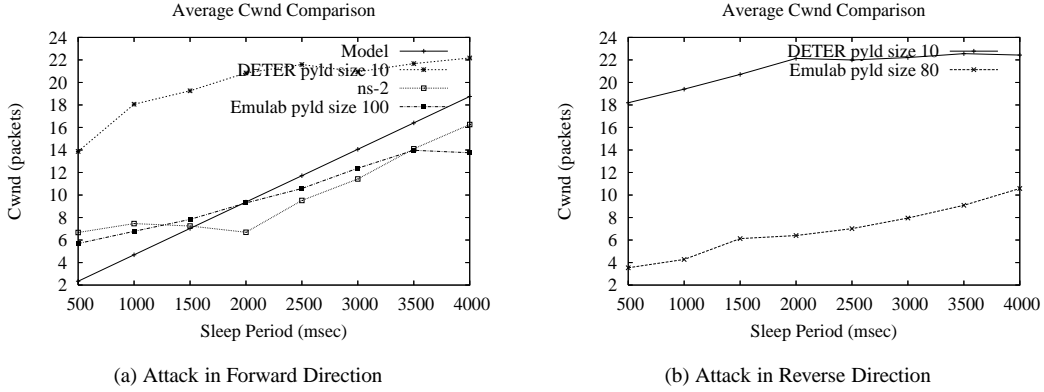


Fig. 4. Comparison of the average congestion window size from analysis, simulations, DETER and Emulab for different sleep time periods, and an attack pulse of length 160 ms. RTT is 160 ms. ns-2 results are not plotted in the reverse case because the attack has little impact.

from the DETER and Emulab testbeds. We first use the default system settings for the DETER and Emulab nodes without using Click on PC routers, since the ns-2 configuration was derived from these values. The attack tool was configured to generate packets as fast as possible during the attack pulse. In this set of experiments, the attack packet payload size is 10 bytes on DETER and ns-2, but it is set to 100 bytes on Emulab, as higher packet rates due to smaller packet sizes on Emulab cause the experiments to take several days, which is problematic in a shared and heavily used testbed like Emulab.

**DETER.** From Figure 4(a), we find that for all values of sleep time, DETER flows are *not* affected by the attack as much as ns-2 and Emulab flows. This is because the DETER PC router nodes are able to handle the attack pulse and the single TCP flow under attack. The DETER results are comparable to ns-2 results with a router buffer size of 100 packets [3]. For larger values of sleep time, the DETER curve levels off instead of increasing as with ns-2. This is because the goodput in these cases starts approaching the goodput value when no attack is present (203 KBps) for an RTT of 160 ms. This goodput value corresponds to a receiver window size of 34715 bytes (24 segments), which is the value reported by the receiver in our experiments. This receiver window size, set by the *tcp* application, limits the maximum goodput when no attack is present.

**Emulab.** Results on the Emulab testbed appear to be more similar than DETER to the analysis and ns-2 results, since the attack creates overload on the Emulab PC routers (even though attack packets are larger), causing packet loss and window cuts. We found that the attack causes a significant number of timeouts on Emulab for sleep times 500–1500 ms, while the number of timeouts is negligible for other sleep times on Emulab, and for all cases on DETER and ns-2. This can be attributed to the fact that the machines used in these Emulab experiments were older than DETER machines, and hence their buses and NICs created bottlenecks.

**ns-2.** In contrast to testbed results, packet loss in ns-2 only occurs in case of buffer overflow. The ns-2 nodes themselves have “infinite CPU and bus capacity,” and are capable of processing any flow without contention. Since the packet service times are shorter in ns-2 than on the testbeds, packet drops are

less frequent. Another difference is that, due to the bounded capacities of the physical devices on the testbed, we find that maximum testbed packet rates cannot exceed 148 Kpackets/s, while ns-2 reports up to 250 Kpackets/s. A key difference between the testbed results and ns-2 is demonstrated in Figure 4(b) when the attack is launched in reverse direction. In ns-2, there is no contention between flows traveling in opposite direction, thus rendering the attack ineffective; however, on the testbeds the attack is still potent due to physical limitations of the machines (shared buses and processors).

Another interesting observation from Figure 4(a) is the non-monotonic increase of the average congestion window for ns-2 with the increase of the sleep time. This can be explained as follows. In ns-2, the lack of overlap between sender and attacker traffic can lead to fewer Cwnd cuts than expected for certain values of sleep time, thus causing the average window to be higher. However, for other values of sleep time, synchronization of the sender and attacker or timeouts can result in a smaller average Cwnd value. Since the ns-2 simulator components in this experiment are deterministic, such synchronization effects are amplified.

**Packet sizes.** Our DETER experiments with different attack packet sizes (results not shown here for brevity) have shown that, in case of packets with 700 byte-payload, there is an even less significant goodput degradation, confirming that small packets can cause more damage on PCs and software routers due to higher packet rates, packet processing overhead, and slot based queues. Results with a payload size of 2 bytes show a slightly higher goodput degradation than with a payload of 10 bytes.

### B. Click Experiments

In the Click modular software router [6], the entire packet path is easily described, and one can easily configure a simple IP router that bypasses the OS IP stack. Simplification of the packet path yields a performance boost, making the PC router less vulnerable to overload under high packet flows. When the router is configured, each affected network device has to be included into the configuration. It is easy to change the queuing discipline and the queue depth for the queue at each output port.

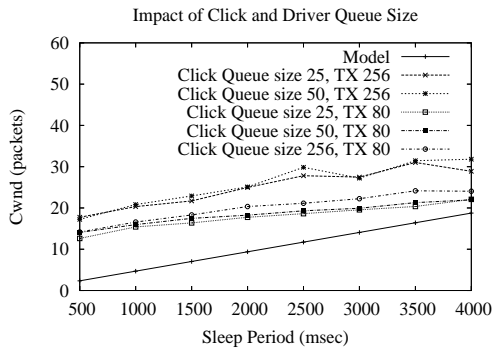


Fig. 5. Impact of varying the Click router queue and the transmit buffer ring of the device driver on DETER.

To increase the fidelity of our experiments and reduce dependence on default system settings, PC routers were configured to run an SMP-enabled Linux-2.4.26 kernel with a multi-thread-enabled *Click-1.4.3* Linux module. Nodes R1 and R2 in Figure 3 were configured to run as IP routers using Click’s programming language.

The default drop-tail queuing discipline was used. Figure 5 demonstrates that varying the TX buffer size produces *significant* variation in the results. It is also important to note that the TX buffer size has a *much more profound* impact than the Click queue size. Figure 5 clearly shows that a TX of 256 and a Click *Queue* of 50 performs much better than a TX of 80 and a Click *Queue* of 256. This implies that it is *crucial* to be aware of the driver settings.

### C. Cisco 3640 Experiments

In the previous experiments, we have used open source software routers running on PCs. The Wisconsin Advanced Internet Laboratory (WAIL) has a large variety of commercial Cisco routers which we used in the following set of experiments. Since we changed the topology on WAIL so that only the access links had propagation delays (as discussed in Section IV), we ran ns-2 experiments with the new setup and found very little difference between the new (using the modified topology) and old (using the topology in Figure 3) simulations.

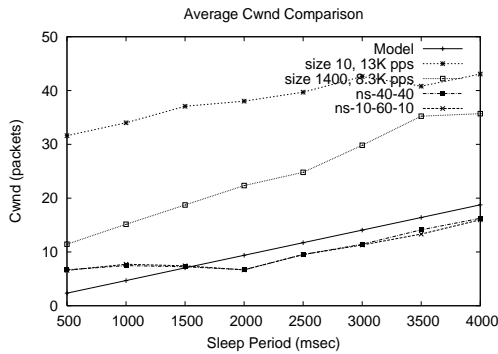


Fig. 6. Comparison of the average congestion window size and the average goodput from analysis, simulations, and WAIL for different sleep time periods, and an attack pulse of length 160 ms. RTT is 160 ms.

Figure 6 shows the results of an experiment with a low-rate TCP-targeted DoS attack as described in Section IV (dumbbell

topology, 160 ms RTT, single attacker multiplexed with the TCP flow) on the Cisco 3640 routers with Cisco IOS 12.4(5). We had to modify the attack parameters from what we used on DETER and Emulab: Two attack parameters were used: 10-byte TCP packets at 13 Kpackets/s and 1400-byte TCP packets at 8.3 Kpackets/s. We also used TCP attack packets instead of UDP packets, which resulted in RST packet traffic which imposed additional load on the routers. The attack pulse was rate limited; otherwise, most of the attack traffic was dropped on the input port queues. In this case, large packets caused more damage, which is opposite to what we have observed in the PC router experiments in Section V-A. Additionally, the fact that we had to scale down the attack significantly (13 Kpackets/s vs. 148 Kpackets/s) shows that properly configured PCs can be used to mimic lower performing commercial routers.

## VI. CONCLUSIONS

Our work has revealed key qualitative differences between the simulation and testbed results due to device specifics. These specifics have a major impact when conducting high performance experiments which stress the devices beyond the operational range envisioned when developing simulation and emulation models. It may be possible to use active probing to determine the capabilities of a network packet forwarding device and then create a general model of the device. Click can then be used to mimic a specific router, offering a higher degree of fidelity. This is the direction of our future work.

## REFERENCES

- [1] The worlds leading network modeling and simulation environment. <http://www.opnet.com/products/modeler/home.html>, 2005.
- [2] M. Allman, V. Paxson, and W. Stevens. TCP congestion control. RFC 2581, April 1999.
- [3] R. Chertov, S. Fahmy, and N. Shroff. Emulation versus simulation: A case study of tcp-targeted denial of service attacks. In *Proceedings of the 2nd International IEEE CreateNet Conference on Testbeds and Research Infrastructures TridentCom, 2006*, February 2006.
- [4] S. Floyd and T. Henderson. The NewReno modification to TCP’s fast recovery algorithm. RFC 2582, April 1999.
- [5] M. Guirguis, A. Bestavros, and I. Matta. Exploiting the transients of adaptation for RoQ attacks on internet resources. In *Proceedings of ICNP*, Oct 2004.
- [6] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The Click modular router. *ACM Transactions on Computer Systems*, 18(3):263–297, August 2000.
- [7] A. Kuzmanovic and E. W. Knightly. Low-rate tcp-targeted denial of service attacks (the shrew vs. the mice and elephants). In *Proc. of ACM SIGCOMM*, August 2003.
- [8] X. Luo and R. K.-C. Chang. On a new class of pulsing denial-of-service attacks and the defense. In *Network and Distributed System Security Symposium (NDSS)*, February 2005.
- [9] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. TCP selective acknowledgement options. RFC 2018, October 1996.
- [10] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A simple model and its empirical validation. In *Proc. of the ACM SIGCOMM*, volume 28, pages 303–314, September 1998.
- [11] UCB/LBNL/VINT groups. UCB/LBNL/VINT Network Simulator. <http://www.isi.edu/nsnam/ns/>, 2005.