# Downscaling Network Scenarios with Denial of Service (DoS) Attacks

Wei-Min Yao, Sonia Fahmy
Department of Computer Science
Purdue University
E-mail: {wmyao,fahmy}@cs.purdue.edu

*Abstract*— **A major challenge that researchers face in studying Denial of service (DoS) attacks is the size of the network to be investigated. A typical DoS attack usually takes place over a large portion of the Internet and involves a considerable number of hosts. This can be intractable for testbed experimentation, and even simulation. Therefore, it is important to simplify a network scenario with DoS attacks before applying it to a simulation/testbed platform. Several approaches have been proposed in the literature to downscale a network scenario, while preserving certain critical properties. In this paper, we investigate via simulations the applicability of packet-level downscaling approaches to DoS scenarios. We select two representative methods: SHRiNK and TranSim. Our experiments identify the operational range of the two downscaling approaches, and propose guidelines for researches to select the most suitable downscaling approach for their own research.**

## I. Introduction

Denial of Service (DoS) attacks continue to be routinely observed in today's Internet. The goal of a DoS attack is to disrupt or disable network activities such as web browsing or online banking, without necessarily compromising or controlling the target host or network. This makes the attacks easier to conduct and harder to defend against, compared to other malicious behavior. A massive Distributed DoS (DDoS) attack on critical Internet resources or vital services can result in loss of Internet connectivity or substantial financial damages. As discussed in [1], an Internet blackout for one week can cause over 1% damage to the GDP of a developed country, i.e., up to 4.5 billion dollars. Thus, the potential threat of DoS attacks should not be ignored.

Over the past decade, several researchers have been investigating the DoS problem, yet there is still no complete solution [2]. There are several reasons why DoS attacks are hard to defend against. For one, attack packets are hard to distinguish from legitimate packets and, for another, there are new attacks emerging every day. Another major challenge in analyzing a DoS attack or developing a DoS defense mechanism is that we do not have an accurate and reliable platform on which to conduct large-scale DoS experiments. A typical DoS attack usually takes place in a large segment of Internet and involves a considerable number of hosts. Since it is undesirable to perform DoS experiments directly on the Internet, most researchers choose to use simulation or emulation. Unfortunately, it is not practical to build a very large-scale simulation or emulation experiment that can reflect all the details of the Internet. Consequently, we must first simplify an experimental scenario before we can study it using simulation or emulation. An algorithm that can downscale a DoS experimental scenario while still preserving important attack characteristics is required to make the simulation or emulation of DoS attacks practical.

Several approaches have been proposed to downscale a network scenario. We can categorize them into two groups: topology-level approaches such as [3], [4], and packet-level approaches such as [5], [6]. Due to the complexity of accurately downscaling a large network such as the Internet, each of these approaches has limitations. We observe, however, that not all network characteristics have to be preserved in a DoS experiment [7]. For example, delay jitter is not important for an FTP flow, and hence a downscaling approach can still be applicable, even when some performance properties are not preserved.

In this paper, we conduct simulation experiments to investigate the applicability of packet-level downscaling approaches to DoS scenarios. We select two representative methods that use different reduction techniques: SHRiNK [5] and TranSim [6]. Our objective is to define the operational range of these downscaling methods, especially when applied to network scenarios with DoS attacks. Our findings can provide guidelines for researches to select the most suitable downscaling approach for their own DoS research.

The remainder of this paper is organized as follows. In Section II, we discuss the related work. In Section III, we discuss our research methodology and the two downscaling approaches chosen in this paper. We present our experimental scenarios and results in Section IV. Finally, we conclude and discuss future work in Section V.

## II. Related Work

Previous techniques that aim to accelerate network simulations can be categorized into two groups: topology-level approaches and packet-level approaches. Topology-level approaches address the downscaling problem at a more abstract level. For example, they sample or partition the Internet graph, such as in [3], [4], or utilize flow modeling, such as [8], [9]. The computational savings of these approaches can be significant since either the size of network topology is reduced or the traffic is represented by flow models. However, although the graph properties of a network topology or the long term flow properties are preserved by these approaches, they are not well-suited for all network simulations since the simulation results are not sufficiently accurate for packet-level analysis [10].

Packet-level approaches such as [5], [6] aim to reduce simulation events by creating a replica of the original network scenario with fewer packets. The simulation then can be performed on the smaller replica with lower computational overhead. Unlike topology-level approaches, one can still perform packet-level analysis on the results gathered from the smaller replica. These approaches may also be applicable when performing network emulation with real hardware, e.g., using the DETER [11] or Emulab [12] testbeds.

We select two example packet-level downscaling mechanisms to investigate in this paper: SHRiNK and TranSim. Pan *et al.* [5] have proposed an approach for scalable performance prediction and efficient simulation of large networks in their Small-scale Hi-fidelity Reproduction of Network Kinetics (SHRiNK) work. Using SHRiNK, one can construct a downscaled network replica by sampling flows, reducing link speeds, and downscaling buffer sizes and Active Queue Management (AQM) parameters. The downscaled network can be used to predict performance measures in the original network. Two downscaling approaches based on flow sampling are described in the SHRiNK work. The first approach aims to reduce hardware requirements at the cost of expanding the experiment time (Section II in [5]), while the latter approach reduces the simulation time at the cost of accuracy (Section III in [5]). We study the second SHRiNK approach in this paper.

Instead of trying to reduce packet events by sampling traffic flows, Kim *et al.* have proposed a method called TranSim [6] to accelerate large-scale simulation of IP networks with TCP/UDP traffic. By applying TranSim to a simulation network, one can create an alternate network that generates a smaller number of packet events. The authors argue that by maintaining the bandwidth-delay product invariant in both networks, network dynamics (such as queue sizes) and TCP dynamics (such as congestion windows) remain unchanged in the process of network transformation.

Another noteworthy approach is DSCALE. Papadopoulos *et al.* [13] have proposed two methods to downscale a network topology that is shared by TCP flows and controlled by various AQM schemes. The first method, referred to as downscale using delays (DSCALEd), removes all uncongested links in the network topology and adds appropriate fixed delays to all packets. Since only the congested links contribute sizable queuing delays for flows, it is safe to remove all uncongested links. This method can be applied to any network without losing fidelity. However, it is not always easy to identify uncongested links in a network [14], and there is no guarantee on how many links can be pruned. The second method, called downscale using sampling (DSCALEs), further reduces the traffic intensity by sampling the traffic in the network. This method, however, is only applicable to flows that arrive according to a Poisson process. Petit *et al.* [10] also investigate methods similar to DSCALE, and point out that downscaling methods are highly sensitive to network traffic, topology size, and performance measures. This is consistent with our findings.

Other work has also considered the downscaling problem in specific contexts. For example, Weaver *et al.* [15] focus on worm attacks. Carl *et al.* [16] study how to preserve routing paths among Autonomous Systems (ASes) while reducing the number of ASes through Gaussian elimination.

## III. Methodology

The feasibility of a downscaling method depends on the traffic and performance measures of the simulation scenario [10]. Our simulation experiments are designed to examine the applicability of packet-level downscaling methods when applied to network scenarios with DoS attacks

### A. Selection of Downscaling Methods

Packet-level downscaling approaches aim to provide mechanisms for researchers to accelerate their simulation or emulation experiments. The goal of these approaches is to reduce traffic (and hence experiment time or resource requirements), while preserving queue dynamics in the network. For example, consider two traffic flows $F_1$ and $F_2$ traversing the same link. If, at any given time, 30% of the packets on the link belong to $F_1$ and 70% belong $F_2$, then, ideally, we would expect the same proportion of packets on the corresponding link in the downscaled network. Although this example goal of maintaining the same proportion of packets on any link is straightforward, it is non-trivial to achieve, especially with closed-loop TCP traffic.

Downscaling approaches can be classified into two groups according how they accomplish this. Methods in the first group assume that a session consists of many smaller flows with similar properties. As a result, a network session can be proportionally reduced by sampling the flows within the session, and reducing the resources these flows consume. In contrast, methods in the second group focus on a single network flow, and slow-down the network to reduce packet events. We select SHRiNK (Section III of [5]) to represent the former group and TranSim [6] to represent the latter.

Although SHRiNK is not intended to apply to the range of scenarios in our simulations, its basic flow sampling approach can be applied in conjunction with TranSim-style reduction of uncooperative flows (UDP traffic and DoS attack traffic). The SHRiNK approach we use (section III of [5]) also suffers from limitations with DropTail queues [5]. We find, however, that the basic SHRiNK flow sampling method is sometimes effective despite its assumptions.

It is important to note that DSCALE [13] is a more recent work from some of the authors of SHRiNK, but we did not utilize it in our experiments. This is because DSCALE (in particular, DSCALEd) requires identification of uncongested links, which can be non-trivial without performing a simulation or experiment on the original (large) network. Therefore, we elected to focus on packet-level methods that do not alter the network topology.

### B. Selection of Traffic

There are two types of traffic in our simulations: DoS attacks and legitimate (i.e., non-attack) traffic.

DoS attacks are generally classified as either flooding attacks or semantic attacks [17]. The major difference between flooding and semantic attacks is the number of attack packets. A flooding DoS attack will send a large number of packets into the network, while a semantic attack only sends a few carefully-crafted packets to the victim.

Since our goal in this paper is to evaluate network scenario downscaling mechanisms, we do not consider resources on end systems (e.g., CPU and memory) in the simulation scenario. As a result, we only focus on flooding attacks that target the bandwidth resource in the network. We select three interesting attack variants: UDP bandwidth floods, ICMP floods [17], and TCP pulsing attacks [18].

Legitimate traffic is impacted by the DoS attacks in the experiments. We select three different network applications to represent the legitimate traffic in DoS experiments. The applications are selected according to the following two requirements. First, we want applications with different QoS requirements. Second, according to [6], since the length of TCP flows has a significant impact on downscaling mechanisms, we must include applications that can generate both long and short-lived TCP flows. Third, we need both open-loop and closed-loop traffic [10]. Therefore, we select a VoIP application for media traffic, an FTP application for long-lived TCP flows, and a Web (HTTP) application for short-lived TCP flows.

### C. Performance Metrics

Ideally, we should preserve all performance metrics such as packet delay and jitter in the downscaled network scenario. However, to achieve this goal, we would have to preserve most packets, since many network protocols or applications are clocked by packets. If some packets are omitted, the protocols or applications will exhibit a different behavior, and we will unavoidably lose fidelity from the original scenario. Fortunately, not all performance metrics are required in order to judge the influence of DoS attacks on a specific network application.

According to [7] and [19], network applications can be categorized into several groups, where each group has different Quality of Service (QoS) requirements. For example, in the case of interactive applications such as Web and telnet, the primary QoS requirement is that a response is served within user-acceptable delay. In terms of network parameters, we only need to focus on the request/response delay and duration of a transaction.

There are only five traffic parameters that we need to consider to compute the QoS requirements of different applications: one way delay, request/response delay, packet loss ratio, transaction duration, and jitter. Based on the QoS requirements of a legitimate application, we use the following method to compare the performance of downscaling mechanisms applied on a specific DoS scenario.

Given a specific set of network applications that we will observe in a network scenario $S$, there are $N$ traffic parameters $x_i$ ($i$ = 1 to $N$) that can be used to define the QoS requirements of these applications. For example, if the application is a file transfer application, then $x_1$ is the request/response delay, $x_2$ is the flow duration, etc. Let $y_i$ be the corresponding traffic parameters in the downscaled scenario $S'$ (possibly after scaling by $\alpha$ or $1/\alpha$ according to the downscaling method, as discussed in Section IV). We define the normalized error for $x_i$ as $E_i = |x_i - y_i|/x_i$. This metric is similar in spirit to the *QoS_degrade* measure in [7]. We define the weighted normalized error as $\sum_{i=1}^{N} \frac{1}{N} \times E_i$. Note that we are using equal weights of $\frac{1}{N}$ for all errors in this formula (and in the paper), though different weights $w_i$ are also possible.

## IV. Simulation Results

In this section, we give the configuration of our simulation experiments and discuss our findings. All the simulations are performed on a FreeBSD machine with 2 GHz CPU and 512 MB RAM using the ns-2 (Version 2.31) [20] network simulator.

We have performed two sets of simulation experiments using the same network topology (Fig. 1), but with different queue sizes. All queues in Fig. 1 are DropTail. In the first set of simulations (labeled “Experiment I”), all queue sizes are set to 100 packets, according to the experiment setup in [6]. We select 10000 packets as the queue size in our second set of simulations (labeled “Experiment II”) according to section III of [5].

In each simulation scenario, a single legitimate traffic session travels from R1 to R4, and one DoS attack flow traverses the R2 to R3 link. We use a downscale parameter $\alpha$ ($0 < \alpha \leq 1$) where $\alpha$ indicates the degree of downscaling compared to the original network (1 means no downscaling; 0.5 means 50% downscaling; and so on).

Note that the network topology in Fig. 1 only includes the minimum links required. As discussed in [13], [10], we can eliminate uncongested links (while still preserving the Round Trip Time (RTT) for TCP flows), since they will not introduce sizable queuing delays and dependencies among flows. We find that our simple topology is a good first step for studying DoS attacks. Using such a simple topology helps in interpreting the results and identifying the causes of inaccuracies. Each of the simulation scenarios is repeated 30 times with different random seeds, and the results are averaged.

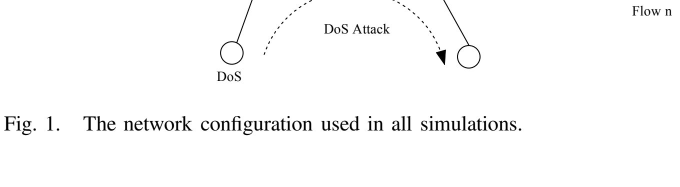

Fig. 1. The network configuration used in all simulations.

### A. Simulation Parameters

There are two types of traffic in each simulation scenario: a DoS attack flow and a legitimate traffic session. To apply the flow sampling downscaling mechanism of SHRiNK, we need to define a legitimate traffic session as a group of flows with same traffic properties. A legitimate session is composed of several flows with a specified flow arrival rate. DoS traffic in our simulations utilizes the UDP protocol with constant packet sending rate. Following the transformation for uncooperative flows described in [6], we reduce the traffic intensity of DoS flows by reducing the packet sending rate. Therefore, we only use a single DoS attack flow per session.

Recall that in Section III, we selected three different network applications, VoIP, HTTP, and FTP, as representative legitimate applications. Our setups for HTTP and FTP are inspired by [5]. In our simulations, we use the ns-2 built-in module PagePool/WebTraf to generate HTTP flows, where there is exactly one page (and one object per page) in a flow. The number of packets in each HTTP flow is Pareto-distributed with an average size of 12 packets and a shape parameter equal to 1.2. The FTP traffic is also generated with the same module and each flow contains 500 packets. We use TCP Reno since this was the TCP flavor used in [6] and [5]. The VoIP traffic uses the ns-2 constant bit rate (CBR) agent on top of the UDP protocol. Each VoIP flow is unidirectional, with a packet size of 48 bytes, and the packet inter-arrival time is 0.02 seconds. The duration of a VoIP flow is 20 seconds.

Within each legitimate session, flows arrive according to a Poisson process with a rate of $\lambda$ flows per second. We choose a value of $\lambda$ which creates sufficient load, but causes no packet drop without a DoS attack. Due to the relatively small queue size in "Experiment I," $\lambda$ is set to 50, 25, and 50 flows per second for FTP, HTTP, and VoIP flows respectively. Scenarios in "Experiment II" have flows arrive in a cluster (i.e., are almost synchronized), as discussed in Section III of [5]. To maintain some randomness in our simulation runs, $\lambda$ is set to 10,000 flows per second instead of infinity. The number of flows is set to 500.

The three DoS attacks are all implemented using the ns-2 CBR agent on top of UDP protocol. The attack rate of UDP bandwidth floods and TCP pulsing attacks is 10 Mbps, with a packet size equal to 500 bytes. The attack rate for the ICMP flooding attack is 7.68 Mbps (20K packets per second) with a packet size equal to 48 bytes.

Each legitimate session starts at time 0 and lasts for $t$ seconds. The UDP and ICMP flooding attacks start after 1/4th of $t$ and last for half of $t$. The TCP pulsing attack is triggered throughout the legitimate traffic duration. However, it is only active for 1/6th of $t$. The DoS attack is in the same direction as VoIP traffic in VoIP experiments; for HTTP and FTP experiments, the DoS attack is in the same direction as the response, not the request, since the response is more substantial.

### *B. Implementation of Downscaling Methods*

To evaluate the performance of both SHRiNK and TranSim, we follow the description of the two methods in [5] (Section III) and [6], and apply them to each network scenario. The downscaling parameter is referred to as $\alpha$. To apply SHRiNK to a network scenario, we perform the following changes in the downscaled scenario: (1) The number of flows in a legitimate session is multiplied by the factor $\alpha$. (2) The arrival rate for flows in a legitimate session is reduced by the factor $\alpha$. (3) All link capacities are reduced by the factor $\alpha$. (4) The size of all queues is reduced by the factor $\alpha$.

For TranSim experiments, the following changes are performed: (1) The propagation delay of all links is expanded by the factor $1/\alpha$. (2) All link capacities are reduced by the factor $\alpha$. (3) The number of packets in an FTP or HTTP flow is reduced by the factor $\alpha$, in order to reduce packet events using the traffic generation modules we used. (4) The packet arrival rate in a VoIP flow is reduced by the factor $\alpha$.

Finally, for both SHRiNK and TranSim, the attack rate for DoS attacks is reduced by the factor $\alpha$.

### *C. Results*

In addition to the five QoS-related traffic parameters mentioned in Section III, we consider the total number of packets in the network, since this represents the savings given by the downscaling mechanism.

Due to space limitations, Tables I, II, and III only list the relevant normalized error values for each simulation scenario in both sets of experiments. The normalized error is computed for each metric, and then as a weighted sum as described in Section III-C. There are six network parameters listed in the tables. The first (total packets) and the second (dropped packets) represent the total number of sent and dropped legitimate packets. In a downscaled network scenario (with downscale parameter $\alpha$), both of these should be reduced to $\alpha$ multiplied by their original values. The third parameter (duration) represents the average duration for a flow in the legitimate session. Ideally, this should remain the same in the downscaled network. The fourth to sixth parameters represent the one way delay, request/response delay, and delay jitter of packets in a legitimate session. For TranSim, since the propagation delays in the network are increased by the factor $1/\alpha$, the values in the downscaled network are expected to increase by the same factor. With SHRiNK, the propagation delays remain the same, so ideally the delays should be preserved. We do not compute the request/response delay for VoIP flows since the VoIP flows are unidirectional. For all the normalized error values in the three tables, the closer they are to zero, the better.

In Table I, both the legitimate traffic and DoS attack are open-loop UDP flows. Ideally, all traffic parameters should be preserved in the downscaled scenarios. However, we find that jitter is not preserved in Experiment I with both methods. TranSim performs well in Experiment I, but not in Experiment II because it does not alter the buffer size in the downscaled scenario. This can lead to errors when the queue size is large and the packet end-to-end delay is dominated by queuing delays. While TranSim proportionally adjusts the propagation delays, the average queue size (and hence queuing delay) remains unchanged in both the original scenario and the downscaled scenario. As a result, when there are large queuing delays as in Experiment II, the end-to-end delay is not adjusted to preserve the bandwidth-delay product.

Examining the number of packets in Table II, we find that the normalized errors are high with TranSim. This indicates

| VoIP | | Experiment I | | | | Experiment II | | | |
|---|---|---|---|---|---|---|---|---|---|
| DoS Attack | Downscaling Method ($\alpha$) | Dropped Packets* | One Way Delay* | Jitter* | Weighted Normalized Error | Dropped Packets* | One Way Delay* | Jitter* | Weighted Normalized Error |
| UDP Bandwidth Flood Attack | TranSim (0.5) | 0.27% | 0.01% | 4.32% | 1.53% | 1.13% | 16.16% | 62.33% | 26.54% |
| | TranSim (0.1) | 0.32% | 0.21% | 29.31% | 9.95% | 21.83% | 52.36% | 620.14% | 231.44% |
| | SHRiNK (0.5) | 3.66% | 0.08% | 31.83% | 11.86% | 1.07% | 0.23% | 14.42% | 5.24% |
| | SHRiNK (0.1) | 0.88% | 1.06% | 88.08% | 30.01% | 1.03% | 0.11% | 80.97% | 27.37% |
| ICMP Flood Attack | TranSim (0.5) | 0.05% | 0.05% | 37.64% | 12.58% | 1.13% | 21.24% | 97.85% | 40.07% |
| | TranSim (0.1) | 0.76% | 0.12% | 360.75% | 120.54% | 4.61% | 61.91% | 500.66% | 189.06% |
| | SHRiNK (0.5) | 0.53% | 0.05% | 109.77% | 36.78% | 0.27% | 0.18% | 0.98% | 0.48% |
| | SHRiNK (0.1) | 0.91% | 0.80% | 295.11% | 98.94% | 1.29% | 0.00% | 3.67% | 1.66% |
| TCP Pulsing Attack | TranSim (0.5) | 1.45% | 0.01% | 48.61% | 16.69% | 15.50% | 13.04% | 53.62% | 27.39% |
| | TranSim (0.1) | 7.15% | 0.05% | 392.13% | 133.11% | 47.42% | 9.95% | 28.67% | 28.68% |
| | SHRiNK (0.5) | 2.26% | 0.21% | 15.43% | 5.96% | 0.16% | 0.02% | 0.63% | 0.27% |
| | SHRiNK (0.1) | 2.23% | 1.92% | 38.89% | 14.35% | 0.11% | 0.06% | 2.18% | 0.78% |

TABLE I

NORMALIZED ERRORS WITH VOIP. * INDICATES TRAFFIC PARAMETERS USED TO COMPUTE WEIGHTED NORMALIZED ERROR.

| HTTP | | Experiment I | | | | | Experiment II | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DoS Attack | Downscaling Method ($\alpha$) | Total Packets | Flow Duration* | One Way Delay | Request/ Response Delay | Weighted Normalized Error | Total Packets | Flow Duration* | One Way Delay | Request/ Response Delay | Weighted Normalized Error |
| UDP Bandwidth Flood Attack | TranSim (0.5) | 30.75% | 18.54% | 3.60% | 51.91% | 35.23% | 23.16% | 53.57% | 15.70% | 15.58% | 34.58% |
| | TranSim (0.1) | 325.06% | 5.96% | 7.85% | 84.20% | 45.08% | 316.15% | 471.19% | 41.13% | 36.16% | 253.68% |
| | SHRiNK (0.5) | 14.50% | 6.39% | 3.31% | 4.33% | 5.36% | 19.52% | 1.25% | 0.26% | 0.19% | 0.72% |
| | SHRiNK (0.1) | 3.93% | 2.30% | 16.36% | 8.08% | 5.19% | 2.27% | 3.36% | 2.04% | 2.32% | 2.84% |
| ICMP Flood Attack | TranSim (0.5) | 26.37% | 52.03% | 0.07% | 9.97% | 31.00% | 21.61% | 47.12% | 0.41% | 19.35% | 33.24% |
| | TranSim (0.1) | 282.08% | 656.32% | 1.93% | 1.69% | 329.00% | 311.54% | 488.04% | 25.17% | 34.29% | 261.17% |
| | SHRiNK (0.5) | 18.88% | 5.13% | 2.02% | 5.27% | 5.20% | 19.67% | 7.22% | 5.02% | 5.11% | 6.17% |
| | SHRiNK (0.1) | 3.49% | 133.65% | 17.94% | 94.49% | 114.07% | 1.84% | 1.38% | 0.55% | 2.29% | 1.83% |
| TCP Pulsing Attack | TranSim (0.5) | 25.15% | 35.10% | 0.14% | 19.86% | 27.48% | 25.13% | 49.61% | 18.72% | 21.60% | 35.60% |
| | TranSim (0.1) | 230.82% | 348.18% | 3.63% | 35.34% | 191.76% | 328.17% | 654.10% | 23.57% | 18.44% | 336.27% |
| | SHRiNK (0.5) | 21.07% | 4.15% | 2.63% | 0.77% | 2.46% | 19.77% | 1.10% | 0.19% | 0.39% | 0.74% |
| | SHRiNK (0.1) | 1.38% | 59.34% | 23.72% | 46.24% | 52.79% | 2.98% | 3.27% | 1.38% | 0.82% | 2.05% |

TABLE II

NORMALIZED ERRORS WITH HTTP. * INDICATES TRAFFIC PARAMETERS USED TO COMPUTE WEIGHTED NORMALIZED ERROR.

| FTP | | Experiment I | | | | | Experiment II | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DoS Attack | Downscaling Method ($\alpha$) | Total Packets | Flow Duration* | One Way Delay | Request/ Response Delay* | Weighted Normalized Error | Total Packets | Flow Duration* | One Way Delay | Request/ Response Delay* | Weighted Normalized Error |
| UDP Bandwidth Flood Attack | TranSim (0.5) | 3.95% | 1.81% | 3.76% | 18.07% | 9.94% | 3.07% | 8.44% | 18.71% | 33.45% | 20.95% |
| | TranSim (0.1) | 8.16% | 27.30% | 11.94% | 69.63% | 48.47% | 22.64% | 4.20% | 37.83% | 61.95% | 33.08% |
| | SHRiNK (0.5) | 1.36% | 0.38% | 0.15% | 22.23% | 11.31% | 0.05% | 7.12% | 13.48% | 13.73% | 10.42% |
| | SHRiNK (0.1) | 2.03% | 0.81% | 3.07% | 43.66% | 22.23% | 0.09% | 0.04% | 7.33% | 6.70% | 3.37% |
| ICMP Flood Attack | TranSim (0.5) | 2.16% | 2.62% | 0.57% | 43.12% | 22.87% | 3.02% | 1.96% | 28.95% | 29.02% | 15.49% |
| | TranSim (0.1) | 11.73% | 1.46% | 2.45% | 73.01% | 37.23% | 20.03% | 8.03% | 50.88% | 58.21% | 33.12% |
| | SHRiNK (0.5) | 0.64% | 0.59% | 0.86% | 7.11% | 3.85% | 0.00% | 0.58% | 1.20% | 1.06% | 0.82% |
| | SHRiNK (0.1) | 3.15% | 4.26% | 9.03% | 41.84% | 23.05% | 0.03% | 1.50% | 3.26% | 3.42% | 2.46% |
| TCP Pulsing Attack | TranSim (0.5) | 2.40% | 3.66% | 2.96% | 48.03% | 25.85% | 0.89% | 0.16% | 6.60% | 7.25% | 3.71% |
| | TranSim (0.1) | 19.21% | 2.58% | 11.63% | 78.95% | 40.77% | 15.27% | 13.50% | 55.35% | 57.22% | 35.36% |
| | SHRiNK (0.5) | 0.25% | 0.66% | 0.09% | 1.68% | 1.17% | 0.09% | 0.20% | 0.30% | 0.53% | 0.37% |
| | SHRiNK (0.1) | 2.07% | 4.26% | 3.67% | 27.31% | 15.79% | 0.18% | 2.07% | 5.10% | 2.90% | 2.48% |

TABLE III

NORMALIZED ERRORS WITH FTP. * INDICATES TRAFFIC PARAMETERS USED TO COMPUTE WEIGHTED NORMALIZED ERROR.

that our TranSim implementation which reduces the total number of packets in a flow fails to proportionally reduce the number of packets in the downscaled scenario. This is also the cause for the high normalized error on dropped packets and flow duration. The reason for this phenomenon is that only the size of the response from server to client is reduced. The packets in the TCP three-way handshake and at least one packet of the web request (not including the TCP handshake) from client to server remained the same. These packets are necessary unless the behavior of TCP or network application is changed. This inaccuracy is significant if there are only a few packets in a flow. This is not an issue for flow sampling-based downscaling mechanisms such as SHRiNK. As a result, we suggest applying flow sampling-based mechanisms for scenarios with short-lived TCP flows.

Table III shows that SHRiNK performs better in Experiment II, and both methods work similarly in Experiment I. In Experiment II, although the bandwidth-delay product is preserved by TranSim, the packet drop changes as shown in Table IV. The intuition behind TranSim is to proportionally reduce TCP throughput by increasing the propagation delay and reducing the link bandwidth (which increase RTT). However, TCP throughput is also a function of packet drop events [21]. Packet drops are higher with TranSim than with SHRiNK with $\alpha = 0.1$ in our simulations. These results are consistent with [10] which shows that FTP is more sensitive

| Method | Experiment I | Experiment II |
|---|---|---|
| Original | 11.03% | 1.54% |
| TranSim (0.5) | 11.07% | 2.07% |
| TranSim (0.1) | 17.12% | 3.10% |
| SHRiNK (0.5) | 10.57% | 1.50% |
| SHRiNK (0.1) | 12.73% | 1.85% |

TABLE IV
PACKET DROP PERCENTAGE FOR SCENARIO WITH UDP FLOODING ATTACK AND FTP.

to downscaling than HTTP, since TCP spends more time in the congestion avoidance phase than in slow start with FTP.

From the results, SHRiNK generally outperforms our TranSim implementation, especially in Experiment II (clustered flows). This indicates that when TCP flows within a session are synchronized [22], SHRiNK sampling works well. However, SHRiNK is not always applicable since its degree of downscaling is constrained by the buffer size or number of flows in a legitimate traffic session, as discussed in [6]. Additionally, although fine-grained metrics [10] used in our experiments are not well-preserved by TranSim, coarse-grained metrics can be preserved, as shown in [6].

## V. Conclusions and Future Work

In this paper, we have conducted a simulation study to investigate the applicability of packet-level downscaling approaches to DoS scenarios. The simulations were designed to represent simple DoS scenarios with three different types of legitimate traffic and three types of DoS attacks. We select two representative packet-level downscaling approaches: SHRiNK [5] for sampling flows in a legitimate session and reducing link and buffer capacities, and TranSim [6] for reducing packet events (which we implement by reducing packets in a flow) and reducing link capacity while increasing propagation delay. We use a performance metric that captures the QoS requirement for each type of legitimate traffic. Although SHRiNK is not intended to apply to the general traffic and queues in all our simulations, its basic flow sampling approach was applicable in conjunction with TranSim-style reduction of uncooperative flows.

From our simulation results, we can draw the following conclusions: (1) The performance of downscaling mechanisms is sensitive to the type and requirements of legitimate traffic, and overall traffic load, and (2) For short-lived TCP flows and scenarios with significant queuing delay and packet drop, downscaling approaches based on sampling flows and reducing both capacity and buffer size (such as SHRiNK) appear to give a better approximation than our implementation of the TranSim approach that reduces packets in a flow.

Although packet-level downscaling approaches have the same goal – reducing the amount of traffic while preserving queue dynamics – their applicability varies based on the characteristics of the scenario to downscale. Our simulation results represent a first step towards defining the operational range of packet-level downscaling approaches for DoS scenarios. According to our findings, each method suffers from particular limitations, which indicates the possibility of designing a downscaling mechanism by leveraging different aspects of each method. This will be the subject of our future work.

## References

[1] T. Dubendorfer, A. Wagner, and B. Plattner, "An economic damage model for large-scale internet attacks," in *Enabling Technologies: Infrastructure for Collaborative Enterprises*, June 2004, pp. 223–228.

[2] J. Mirkovic, S. Dietrich, D. Dittrich, and P. Reiher, *Internet Denial of Service: Attack and Defense Mechanisms*. Prentice Hall, 2005.

[3] V. Krishnamurthy, M. Faloutsos, M. Chrobak, L. Lao, J.-H. Cui, and A. Percus, "Reducing large internet topologies for faster simulations," in *Proc. of IFIP Networking*, 2005.

[4] K. Yocum, E. Eade, J. Degesys, D., Becker, J. Chase, and A. Vahdat, "Toward scaling network emulation using topology partitioning," in *Proc. of MASCOTS (Modeling, Analysis and Simulation of Computer Telecommunications Systems)*, 2003.

[5] R. Pan, B. Prabhakar, K. Psounis, and D. Wischik, "SHRiNK: a method for enabling scaleable performance prediction and efficient network simulation," *IEEE/ACM Transactions on Networking*, vol. 13, no. 5, pp. 975–988, October 2005.

[6] H. Kim, H. Lim, and J. C. Hou, "Accelerating simulation of large-scale IP networks: A network invariant preserving approach," in *Proc. of INFOCOM*, 2006.

[7] J. Mirkovic, A. Hussain, B. Wilson, S. Fahmy, P. Reiher, R. Thomas, W.-M. Yao, and S. Schwab, "Towards user-centric metrics for denial-of-service measurement," in *Proceedings of the Workshop on Experimental Computer Science*, 2007.

[8] B. Liu, Y. Guo, J. Kurose, D. Towsley, and W. Gong, "Fluid simulation of large scale networks: Issues and tradeoffs," in *Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications. Las Vegas, Nevada*, 1999.

[9] D. M. Nicol and G. Yan, "Discrete event fluid modeling of background tcp traffic," *ACM Transactions on Modeling and Computer Simulation*, 2004.

[10] B. Petit, M. Ammar, and R. Fujimoto, "Scenario-specific topolgy reduction in network simulations," in *Proc. of International Symposium on Performance Evaluation of Computer and Telecommunicaiton Systems (SPECTS)*, 2005.

[11] "Deter." [Online]. Available: http://www.isi.edu/deter/

[12] "Emulab." [Online]. Available: http://www.emulab.net/

[13] F. Papadopoulos, K. Psounis, and R. Govindan, "Performance preserving topological downscaling of Internet-like networks," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 12, pp. 2313–2326, December 2006.

[14] F. Papadopoulos and K. Psounis, "Efficient identification of uncongested Internet links for topology downscaling," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 5, pp. 39–52, October 2007.

[15] N. Weaver, I. Hamadeh, G. Kesidis, and V. Paxson, "Preliminary results using scale-down to explore worm dynamics," in *Proc. of the 2004 ACM workshop on Rapid malcode*, 2004.

[16] G. Carl, S. Phoha, G. Kesidis, and B. B. Madan, "Path preserving scale down for validation of internet inter-domain routing protocols," in *Winter Simulation Conference*, 2006, pp. 2210–2218.

[17] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and ddos defense mechanisms," *SIGCOMM Computer Communications Review*, 2004.

[18] 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*, 2003.

[19] Nortel Networks, "QoS performance requirements for UMTS, the 3rd generation partnership project (3GPP)," http://www.3gpp.org.

[20] "Ns-2." [Online]. Available: http://www.isi.edu/nsnam/ns/

[21] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: a simple model and its empirical validation," *SIGCOMM Comput. Commun. Rev.*, vol. 28, no. 4, pp. 303–314, 1998.

[22] H. Sawashima, Y. Hori, and H. Sunahara, "Characteristics of UDP packet loss: Effect of TCP traffic," in *Proceedings of INET*, 1997.