Empirical Study of Tolerating Denial-of-Service Attacks with a Proxy Network

Ju Wang, Xin Liu and Andrew A. Chien
Department of Computer Science and Engineering and
Center for Networked Systems
University of California, San Diego
jwang@cs.ucsd.edu, xinliu@cs.ucsd.edu, achien@ucsd.edu

Abstract

Proxy networks have been proposed to protect applications from Denial-of-Service (DoS) attacks. However, since large-scale study in real networks is infeasible and most previous simulations have failed to capture detailed network behavior, the DoS resilience and performance implications of such use are not well understood in large networks. While post-mortems of actual large-scale attacks are useful, only limited dynamic behavior can be understood from these single instances. Our work provides the first detailed and broad study of this problem in large-scale realistic networks. The key is that we use an online network simulator to simulate a realistic large-scale network (comparable to several large ISPs). We use a generic proxy network, and deploy it in a large simulated network using typical real applications and DoS tools directly. We study detailed system dynamics under various attack scenarios and proxy network configurations. Specific results are as follows. First, rather than incurring a performance penalty, proxy networks can improve users' experienced performance. Second, proxy networks can effectively mitigate the impact of both spread and concentrated large-scale DoS attacks in large networks. Third, proxy networks provide scalable DoS-resilience – resilience can be scaled up to meet the size of the attack, enabling application performance to be protected. Resilience increases almost linearly with the size of a proxy network; that is, the attack traffic that a given proxy network can resist, while preserving a particular level of application performance, grows almost linearly with proxy network size. These results provide empirical evidence that proxy networks can be used to tolerate DoS attacks and quantitative guidelines for designing a proxy network to meet a resilience goal.

1 Introduction

Denial-of-Service (DoS) attacks continue to be key threat to Internet applications. In such attacks, especially distributed DoS attacks, a set of attackers generates a huge amount of traffic, saturating the victim's network, and causing significant damage. Overlay networks [1] have been proposed to protect applications against such DoS attacks [1-7] . These overlay networks are also known as proxy networks [6, 8]. The key idea is to hide the application behind a proxy network, using the proxy network to mediate all communication between users and the application, thereby preventing direct attacks on the application.

Realistic study of these approaches should involve large networks, real applications, and real attacks. To date, however, studies of these approaches have been limited to theoretical analysis and small-scale experiments [1-7], which cannot capture the complex system dynamics, including packet drops, router queues, temporal and feedback behavior of network and application protocols during DoS attacks. These factors are critical to the application and proxy network performance in the face of DoS attacks. Thus, we still do not have answers to many key questions about the viability and properties of these proxy approaches. Specifically, with real complex network structures and protocol behavior, can proxy networks tolerate DoS attacks? If so, what are the key parameters to achieve effective and efficient resilience? If we use proxy networks, what are the performance implications for applications?

Our approach exploits the recent availability of a detailed large-scale online network simulator – MicroGrid [9, 10] – to study proxy networks with real applications and real DoS attacks. MicroGrid supports detailed packet-level simulation of large networks and use of unmodified applications. With MicroGrid, we are able to make detailed performance studies in large networks environment with complex, typical application packages and real attack software. Our studies include networks with up to 10,000 routers and 40 Autonomous Systems (ASes) with a physical extent

comparable to the North American continent. We believe this is the first empirical study of proxy networks for DoS resilience at large-scale, using real attacks, and in a realistic environment.

Our experiments explore a range of network sizes, proxy network configurations, attack parameters, and application characteristics. The key results are summarized below:

- Rather than incurring a performance penalty, proxy networks can improve users' experienced performance, reducing latency and increasing delivered bandwidth. The intuition that indirection reduces performance turns out to be incorrect, as the improved TCP performance more than compensates.

- Proxy networks can effectively mitigate the impact of both spread and concentrated large-scale DoS attacks in large network environment. Our experiments have shown that a 192-node proxy network with 64 edge proxies (each connected by a 100Mbps uplink), can successfully resist a range of large-scale distributed DoS attacks with up to 6.0Gbps aggregated traffic and different attack load distribution; most users (>90%) do not experience significant performance degradation under these attack scenarios.

- Proxy networks provide scalable DoS-resilience – resilience can be scaled up to meet the size of the attack, enabling application performance to be protected. Resilience increases almost linearly with the size of a proxy network; that is, the attack traffic that a given proxy network can resist, while preserving a particular level of application performance, grows almost linearly with proxy network size.

These results provide empirical evidence that proxy networks can be used to tolerate DoS attacks and quantitative guidelines for designing a proxy network to meet a resilience goal.

Our main contributions are the following. First, we provide the first large-scale empirical study on the DoS resilience capability of proxy networks using real applications and real attacks; this is a qualitative advance over previous studies based on theoretical models and small scale experiments. Second, we provide the first set of empirical evidence on large-scale network environment to prove that proxy networks have effective and scalable resilience against DoS attacks. Third, we provide a detailed performance analysis of proxy networks in large-scale network environment, and show that, in contrast to intuition, proxy networks can improve user-experienced performance.

The remainder of the paper is organized as follows. Section 2 provides background on the DoS problem and the proxy network approach. Section 3 defines the problem, and describes our approach. Section 4 briefly describes the MicroGrid simulation environment which provides new capabilities, enabling this research. Section 5 presents results and analysis. Section 6 discusses the implications of our studies, and relates our work to previous work. Section 7 summarizes the results and discusses directions for future work.

2 Background

We briefly describe the applications of concern and the denial-of-service attacks that we study in this paper. Then, we describe proxy network-based DoS defense scheme.

2.1 Internet Applications & Denial-of-Service Attacks

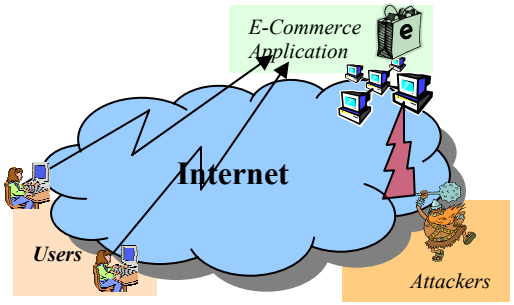

Figure 1 Internet Application and DoS Attacks

Figure 1 illustrates a typical Internet application deployment, such as an e-Commerce application. The application service runs on a cluster of servers. Users are distributed across the Internet, and access the application service via the Internet. As shown in Figure 2, in this application model, the Internet is a communication layer used to convey a well-defined application-level protocol between the applications and their users. Examples of such applications include search engines, e-Commerce, online banking, and online trading applications.

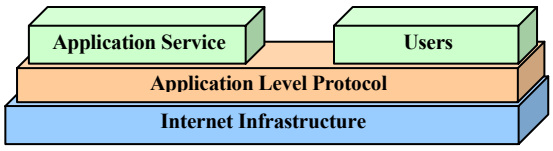

Figure 2 Application Model

DoS attacks are a major security threat to Internet applications. In a DoS attack, attackers consume resource, on which either the applications or accesses to

the applications depend, making the applications unavailable to their users.

There are two classes of DoS attacks: *infrastructure-level* and *application-level* attacks. Infrastructure-level attacks directly attack the resources of the service infrastructure, such as the networks and hosts of the application services; for example, attackers send floods of network traffic to saturate the target network. In contrast, application-level attacks are through the application interface; for example, attackers overload an application by sending it abusive workload, or malicious requests which crash the application.

Infrastructure-level DoS attacks only require the knowledge of applications' network address, i.e. IP address. Meanwhile, application-level DoS attacks are application-specific, and do not require the target application's IP address.

Distributed Denial-of-Service (DDoS) attacks are large-scale DoS attacks which employ a large number of attackers distributed across the network. There are two stages in such attacks. First, attackers build large zombie networks by compromising many Internet hosts, and installing a zombie program on each. Second, attackers activate this large zombie network, directing them to "DoS" a target. Both infrastructure and application-level DoS attacks can be used in stage two. Automated DDoS toolkits, such as Trinoo, TFN2k and mstream [11-13], and worms, such as CodeRed [14, 15], provide automation, enabling large scale attacks to be easily constructed.

This paper focuses on infrastructure-level distributed DoS attacks. In the rest of the paper, DoS attacks refer to infrastructure-level distributed DoS attacks unless indicated otherwise.

2.2 Proxy Network Approach

Proxy networks have been proposed as a means to protect applications from DoS attacks [1-4, 7]. Figure 3 illustrates a generic proxy network encompassing most of the proposed approaches [1-4, 7]. As shown in Figure 3, an overlay network, known as proxy network, is used to mediate all communication between users and the application. As long as the mediation can be enforced, the proxy network is the only public interface for the application, and the application cannot be directly attacked. Meanwhile a large set of proxies, known as edge proxies, publish their IP addresses, providing application access. The number of edge proxies can be flexibly increased. This allows scalable resilience against DoS attacks on edge proxies, and thereby allows a proxy network to shield the application from DoS attacks. Using this generic proxy network model, we study the fundamental capabilities and limitations of a wide range of proxy networks.

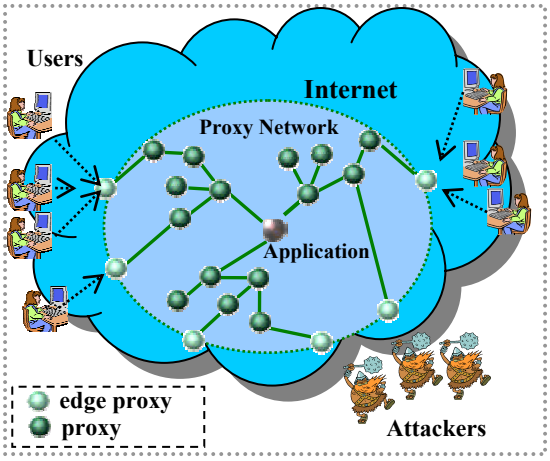

Figure 3 DoS-Tolerant Proxy Network

As discussed above, a proxy network must have two key capabilities to successfully protect applications from DoS attacks. First, a proxy network must enforce mediation so that the application can only be reached via the proxy network, thereby preventing direct DoS attacks on the application. Second, a proxy network must provide DoS-resilience mediation so that it can support continued user access to the application under DoS attacks.

Mechanisms to enforce proxy network mediation have been proposed and studied. As shown in [6, 8], it is feasible to hide an application' IP address using a proxy network, thereby enforcing proxy network mediation. Additionally, some proxy network schemes [1, 3, 5, 7] also studied slightly different mechanisms for enforcing mediation. For example, SOS [1, 7] uses filters combined with secret servlets to enforce all application access being mediated through the SOS network.

In this paper, we assume mediation can be enforced, and direct DoS attacks on the application are impossible. We focus on the DoS-resilience capability of proxy networks, and study how well a proxy network can protect user-experienced application performance under DoS attacks on edge proxies.

3 Problem Definition and Approach

3.1 Problem Definition

Little is understood about the performance or effectiveness of proxy network-based DoS defense in large-scale realistic networks. To date, studies of these problems have been limited to theoretical analysis and small-scale experiments. They do not capture real complex network structures, real temporal and feedback behavior of network and application protocols, and detailed network dynamics, such as router queues and individual packet drops. All these have important impact on application performance. Therefore, we still do not have answers to many key questions about the viability and properties of these proxy approaches.

- With real complex network structures and protocol behavior, can proxy networks tolerate DoS attacks? In particular, in large realistic networks, under various attack scenarios, how much can proxy networks mitigate the impact of DoS attacks on users' experienced performance? What are the key parameters to achieve effective and efficient resilience? How does this capability scale up when proxy networks grow in size?
- What are the basic performance implications of proxy networks? How do they affect users' experienced performance for real applications in large-scale realistic networks?

3.2 Approach

Our approach is to use newly available simulation tools for new studies that are significantly more realistic in several key dimensions, including:

- Detailed network dynamics, such as router queuing and individual packet drops.
- Real temporal and feedback behavior of network and application protocols and their interaction with other network traffic.
- Emergent properties of large-scale realistic networks, such as topology, latency and bandwidth distribution.

Since DoS attacks exercise extreme points of network behavior, correct modeling of such detail is important for realistic studies. In this context, we study the performance and DoS resilience of the generic proxy network approach. Details of our approach include:

- use of a large-scale, high-fidelity packet-level online network simulator – MicroGrid (see section 4.2) – to simulate large-scale realistic network environment, which include up to 10,000 routers and 40 ASes, comparable to the size of large ISPs.
- a real proxy network implementation and real applications deployed in the MicroGrid virtual environment.
- a large zombie network comparable to one with 10,000 zombies with DSL/cable modem connection, and a real DoS toolkit to generate attack traffic. This setting supports controlled experiments with various attack scenarios.
- a tree proxy network topology, rooted at the application with edge proxies at the leaves providing user access. The number of edge proxies is the width of the tree, and the number of hops from root to leaves is the height. For a localized application implementation, the tree corresponds to subset of links that would be exercised in all proxy networks.
- systematic study of a range of attacks, proxy network configurations, application, and resilience strategies.

We study users' experienced performance using a range of proxy network topologies to understand the basic performance impacts of proxy networks; then we generate a range of attack scenarios with different attack magnitude and distribution, and study their impact on users' experienced performance with proxy networks of different sizes to understand proxy networks' DoS-resilience capabilities and scalability.

4 Experimental Environment

We describe the key software components used in the empirical study, MicroGrid simulation environment, and the resources used in the experiments.

4.1 Software Environment

The experiments use four key components: a generic proxy network implementation, apache web server [16] as the application, a web testing tool "siege"[17] to simulate user access, and a DDoS attack tool "Trinoo"[11].

4.1.1 Proxy Network Implementation

The generic proxy network is composed of proxy nodes. Proxies are software programs that forward application messages. As shown in Figure 4, each pair

of neighboring proxies maintains a TCP connection, which is established upon proxy instantiation, according to the given topology and the bootstrap location information. The TCP connections among proxies are persistent and shared among users. Messages can be routed inside the proxy network using any given routing algorithm. The generic proxy network can be configured to capture a range of proxy networks with different topologies and routing algorithms.

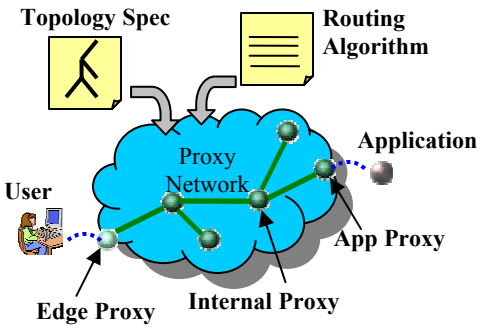

Figure 4 Proxy Network Prototype

The proxy network supports TCP applications transparently. We apply the DNS scheme used by content delivery networks [18] to direct user access to proxies. As shown in Figure 4, edge proxies listen to user connection requests, and encode application traffic into messages which are routed via the proxy network to the application. At the exit of the proxy network, application proxies (proxies that directly connect to the application) decode the messages, establish new connections to the application if necessary, and send the data to the application. Similarly, the response from the application can be delivered back to the user through the proxy network.

4.1.2 Application Service

We use Apache web server as a representative application front-end. Since we focus on the network impact of DoS attacks, specific details of the application logic at the back-end are not critical. We use Apache web server to serve files of different sizes as a representative scenario.

4.1.3 User Simulator

We use siege – a web test toolkit – to generate user requests. Siege generates web requests based on a list of URLs, and measures the response time for each of the requests. This allows us to simulate user access and collect statistics which characterize user experienced performance.

4.1.4 DDoS Attack Toolkit

Trinoo [11] is a DDoS attack toolkit generally available on the Internet. It includes a daemon and a master program. A typical trinoo network consists of a collection of compromised Internet hosts running the trinoo daemon program. The master program is used to control this trinoo network to make DDoS attacks. Given a list of IP addresses, trinoo daemons send UDP packets to the targets at the given start time. In its original form, the trinoo daemon repeatedly sends UDP packets at full speed. To support controlled experiments, we changed trinoo daemon, allowing its sending rate to be adjusted.

4.2 MicroGrid Simulation Toolkit

MicroGrid [9, 10] is an integrated online packet-level simulator that provides modeling of virtual network environments. MicroGrid allows users to configure an arbitrary virtual network, deploy it to a cluster, and then execute their unmodified applications directly in that virtual network. Three key capabilities of MicroGrid are crucial to our study.

- Ability to simulate large networks at high fidelity even at high levels of traffic. MicroGrid has demonstrated good scalability in realistic large-scale simulations of networks with 20,000 routers (comparable to a large Tier-1 ISP network like AT&T) [19].

- Support for realistic topology, routing and a full network protocol stack. MicroGrid is integrated with a topology generator maBrite[20], which can create realistic Internet-like network topologies, and set up BGP routing policies automatically based on realistic Internet AS relationships. It supports Internet routing protocols such as BGP [21] and OSPF [22]. It also supports networking protocols, such as IP, UDP, TCP [23] and ICMP [24].

- Support for direct execution of unmodified applications.

These capabilities of MicroGrid allow us to study the properties of the proxy network and detailed behavior of the system in a large-scale network environment with realistic settings, running real applications and real attacks. These capabilities are markedly greater than testbeds such as PlanetLab [25] or small scale simulators such as NS2 [26], where the scale, intensity

and range of attack scenarios that can be studied are limited.

4.3 Simulation Setup

4.3.1 Simulated Network

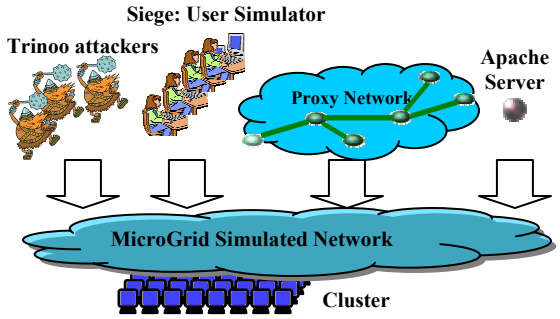

Figure 5 Experiment Setup

As shown in Figure 5, the proxy network, apache server, siege programs and trinoo attackers are deployed in the MicroGrid simulated network environment. The maBrite topology generator is used to create Internet-like Power-Law network topologies [20, 27]. We use two virtual networks in our experiments. One (named R1K) includes 1000 routers and 20 ASes, and the other (named R10K) includes 10,000 routers and 40 ASes, which is comparable to the size of a large ISP network. Both networks span a geographic area of 5000 miles by 5000 miles, which is roughly the size of the North American continent. This physical extent determines link latencies. OSPF routing is used inside ASes; BGP4 is used for inter-AS routing.

4.3.2 Physical Resources

Our experiments use two clusters. The MicroGrid simulator runs on a 16-node dual 2.4GHz Xeon Linux cluster with 1GB main memory on each machine, connected by a 1Gbps Ethernet switch. Other software components run on a 24-node dual 450MHz PII Linux cluster with 1GB main memory on each machine, connected by a 100Mbps Ethernet switch. These two clusters are connected with a 1Gbps link.

5 Experiments and Results

We study the performance implications and DoS resilience of proxy networks, and address the problems stated in Section 3.

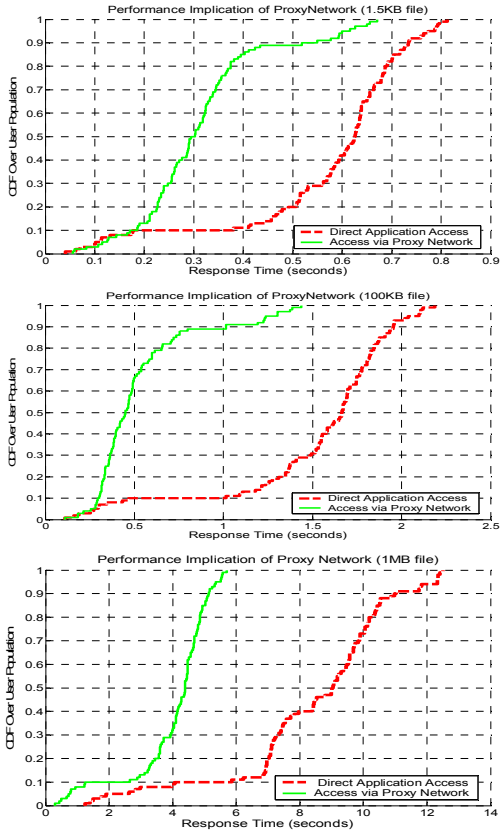

Figure 6 Proxy Network Performance

5.1 Proxy Network Performance

To understand the basic performance implications of the proxy network approach, we compare the user-observed service performance for direct application access and proxy network mediation. Users choose edge proxies based on proximity, and no user authentication is used.

The proxy network is deployed in a resource pool of 1000 hosts randomly sampled from the network. The following heuristic is used to deploy proxies in this resource pool. Edge proxies are distributed uniformly across the resource pool. Application proxies (see Section 4.1) are placed on those hosts in the resource pool which are relatively close to the application. The remaining proxies are distributed evenly between edge proxies and application proxies. This heuristic maps a proxy network to a given resource pool of Internet hosts, trying to align the proxy network structure with underlying network to avoid long detours in overlay routes.

Figure 6 shows the results in the R1K simulated network (described in Section 4.3) for a tree-topology

192-node proxy network, with 64 edge proxies. The X-axis is the response time for a user to download files of a given size (1.5KB, 100KB or 1MB). We plot the measured performance for direct access and proxy network mediation. The Y-axis is Cumulative Density Function (CDF) of user-observed response time over the user population. Hence a curve closer to the Y-axis implies that more users have good response time.

While one might expect proxies to degrade performance, the proxy network improves performance. For small requests (e.g. 1.5KB), the 50-percentile response time is reduced by half; for requests of modest sizes (e.g. 100KB), the improvement is even more significant, and so is the case of large files (e.g. 1MB). There are three main reasons for these phenomena:

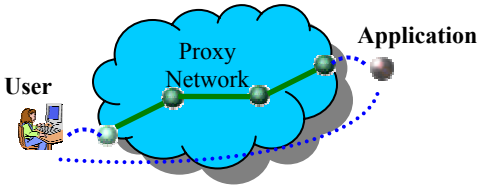

Figure 7 Direct Access vs. Proxy Network

1. Proxy network improves connection set up time. As described in Section 4.1 (see Figure 7), there are established TCP connections among proxies. For each virtual connection between a user and the application, instead of establishing a long TCP connection from the user to the application, two shorter TCP connections are established: one from the user to the edge proxy, and the other from the corresponding application proxy to the application. Both of them have small RTT (round trip time), since application proxies are close to the application, and users choose edge proxies based on proximity. Since the TCP handshake [23] takes 1.5 RTT, the connection setup cost can be reduced by one RTT between the user and the application[2]. This effect is prominent for small requests (e.g. 1.5KB) as shown in Figure 6.
2. The TCP connections among proxies are persistent, and in most cases the TCP congestion windows for those connections have already been fully opened by previous data transfers and other users' traffic. Thus, they no longer suffer a slow start phase [23] to grow the congestion window. For requests of modest sizes (e.g. 100KB shown in Figure 6), this effect is most prominent.
3. A series of shorter TCP connections can also improve throughput and robustness as studied in Logistic Networking [28]. Here we give a brief explanation, and details can be found in [28]. The throughput can be improved because the TCP throughput is roughly TCP send buffer size divided by RTT, and the connections among proxies have shorter RTTs comparing to the RTT between the user and the application. The throughput effect can be seen for large requests (e.g. 1MB shown in Figure 6).

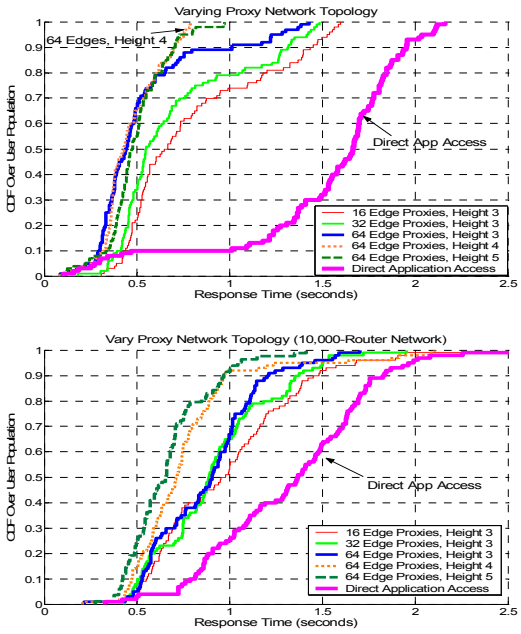

Figure 8 Performance in two simulated networks (Top: R1K network Bottom: R10K network)

To validate the generality of our analysis, we repeat the experiments (download 100KB files) for a range of tree topologies with different heights and widths in the two simulated networks described in Section 4.3 (see Figure 8), and see similar phenomena. Thus, the factors discussed above are generally applicable to proxy networks in large realistic networks, and proxy networks in general can in fact improve user-experienced performance. This is a moderately surprising result, which is not so obvious without our large scale simulation study.

These results are different to previous findings such as in [7], which evaluated the performance of WebSOS on the PlanetLab [25] testbed, and reported 2 to 10 times performance degradation. We believe that two main factors contribute to this dramatic difference. First, WebSOS uses Chord routing which does not provide shortest path routing, and the deployment of overlay nodes is not optimized either. These factors may contribute to large overhead on the overlay route. Second, the connections among WebSOS nodes are not persistent and not necessarily short. Therefore, the

WebSOS implementation cannot benefit from the TCP behaviors discussed before which greatly improve our proxy network performance. Besides these factors, user authentication on edge proxies [3] and less efficient implementation [4] may also contribute to performance overhead for WebSOS. Our results indicate that the performance of WebSOS can be significantly improved via appropriate construction, implementation, and deployment of proxy networks.

5.2 DoS-Resilience of Proxy Networks

To explore the DoS-resilience capability of proxy networks, we study user-experienced performance under a range of attack scenarios, with or without proxy networks. We use the same proxy network, which contains 192 proxies (64 edge proxies), in the simulated networks (R1K and R10K). In addition, we constructed a DDoS network, which contains 100 Trinoo daemons randomly distributed in the network. Each Trinoo daemon has a 100Mbps link. This Trinoo network is comparable to one with 10,000 zombies using DSL/Cable modem links.

Our first experiment explores whether a proxy network can really protect an application from DoS attacks. Our second experiment studies the DoS-resilience capability of the proxy network under two large-scale attack scenarios: spread DoS attacks, where attack load is distributed evenly on all the edge proxies, and concentrated DoS attacks, where attack load is concentrated on a subset of edge proxies to saturate their incoming links. We consider two user access schemes in these attack situations: static and dynamic edge proxy selection. In the static scheme, a user chooses an edge proxy based on proximity, and continue to use it even if the proxy is under attack. In the dynamic scheme, a user can switch to other proxies if the closest edge proxy is under attack. Our final experiment studies the scalability of proxy networks with respect to DoS-resilience, by varying the size and width of proxy networks.

5.2.1 Can a proxy network protect applications?

We compare the impact of a DoS attack on the application and the proxy network. In our experimental setting, the application service is connected by a 250Mbps link, and each edge proxy is connected by a 100 Mbps link. Figure 9 shows the CDF for user-observed response time of 100KB requests with or without a proxy network in the R1K network. The results show that a 250Mbps attack on the application significantly increases service response time (about 10x), and the application becomes unusable. However, when a proxy network is used, the attack has no observable impact on the user experienced performance. The reason is straightforward. By having a collection of edge proxies to dilute the impact of attack, a proxy network has a greater capacity than the application, thereby not as easily being saturated.

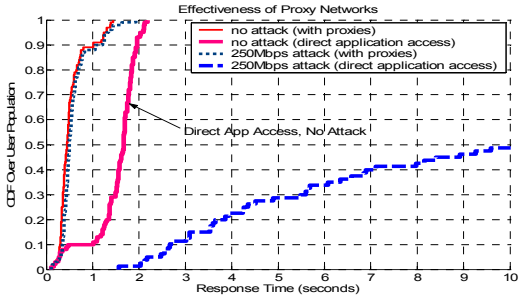

Figure 9 DoS Resilience of Proxy Network

5.2.2 Resisting large-scale DoS attacks

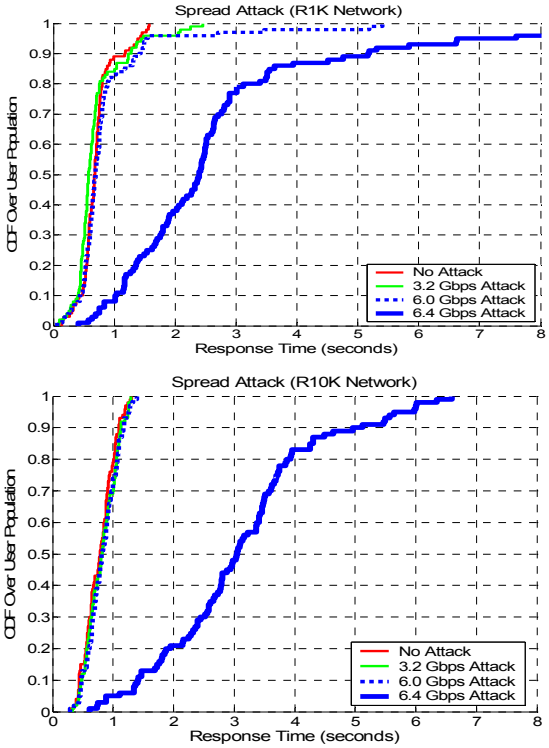

Figure 10 Performance under Spread Attacks

To investigate how well a proxy network can tolerate DoS attacks, we launch both spread and concentrate DoS attacks on the proxy network described in Section 5.1, which has 64 edge proxies and 192 proxies in total.

Each of the edge proxy has a 100Mbps uplink. In both cases, we vary the aggregated attack magnitude from 3.2Gbps to 6.4Gbps.

5.2.2.1 Resisting spreading attacks

Figure 10 shows the users' experienced performance under spread attacks. It shows that when attack magnitude is no more than 6.0Gbps (recall that the aggregated uplink capacity for all the edge proxies is 6.4Gbps), more than 95% users observe no significant performance degradation – the spread attacks have been successfully tolerated. The reason is that the edge proxies successfully dilute attack traffic; even under heavy attack loads, most of the edge proxies still have sufficient capacity left to serve user requests. Figure 10 also shows that when attack load reaches 6.4Gbps, all the edge proxies are saturated, significant performance degradation occurs for all users.

Interestingly, we can see large performance degradation for a small fraction of users (<5%) in the R1K network, when the attack magnitude is 6.0Gbps. It is due to the correlation among proxies and users (see Figure 11).

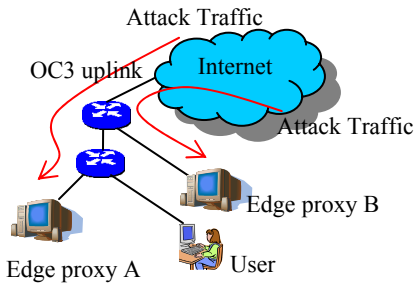

Figure 11 Correlation among Proxies and Users

Two edge proxies A and B share an uplink of OC3 (155Mbps). Before attack traffic saturates both proxies' local links (100Mbps), the shared OC3 link gets congested first. Therefore, users on these two proxies and users in the same network as these proxies will be affected. This effect limits the effectiveness of proxy networks.

5.2.2.2 Resisting concentrated attacks

Figure 12 shows of the users' experienced performance using static edge proxy selection scheme in concentrated attacks. The attack load is concentrated on a random subset of edge proxies. In this case, attack traffic saturates part of the proxy network, and a significant percentage of users are affected due to congestion and packet loss. This effect is more prominent when attack load is higher than the proxies' capacity (e.g. 4.0Gbps attack on 32 proxies).

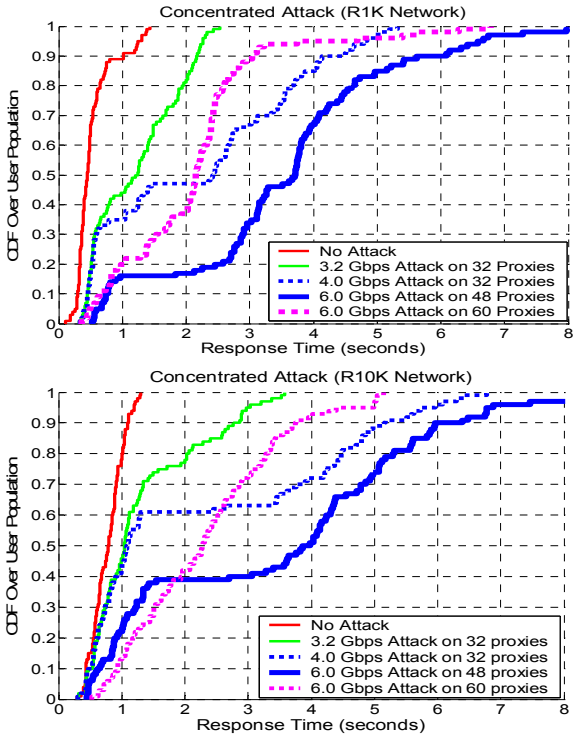

Figure 12 Performance under Concentrated Attacks (static edge selection)

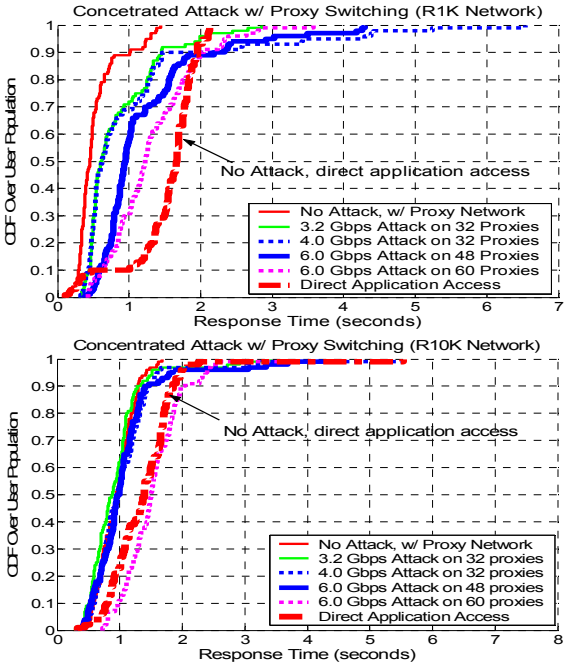

Figure 13 Performance under Concentrated Attacks (dynamic edge selection)

We repeat the experiments for concentrated attacks, and let users switch to the closest proxy not being saturated (dynamic edge proxy selection). Figure 13 shows the CDF of user-observed performance. Compared with Figure 12, the performance has been significantly improved. For comparison, Figure 13 also plots the baseline case where users directly access the application without attack traffic. It shows that, for most users, the proxy network can maintain a slightly better performance than the baseline case, even under a high attack load (e.g. 6.0Gbps). Therefore, proxy networks can resist concentrated attacks effectively.

To understand the performance gap between the attack cases and the non-attack case, we measure the users' experienced performance without attack (for the R1K network), while using the set of edge proxies they switch to during attacks (shown in Figure 14). For most users, this curve follows the attack cases closely, showing that the performance gap is due to switching edge proxies rather than congestion caused by attack traffic. Additionally, a small number of users are greatly affected by the attack, due to the limitation of the underlying network discussed in Figure 11.

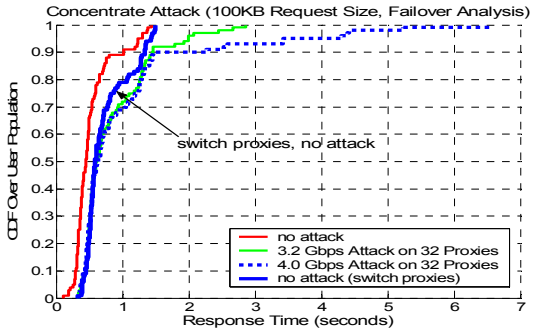

Figure 14 Analysis of Dynamic Edge Selection

5.2.3 Scalability of resilience

We explore how varying the size (width) of a proxy network affects its resilience to attacks. This is an important scaling property, showing how effective we can resist a larger scale of attacks by building larger proxy networks.

The goal of our experiment is to evaluate the amount of attack load proxy networks can withstand for a range of proxy network widths. It is difficult to directly measure the maximum attack load that a proxy network can tolerate. Instead, we set the attack magnitude to be 95% of the proxy network's capacity, and measure the user-observed performance. We define the capacity of a proxy network to be the sum of the link capacity of its edge proxies. For example, if the proxy network has 16 edge proxies and each edge proxy has a 100 Mbps uplink, then its capacity is 1.6Gbps and the aggregated attack magnitude is 1.52Gbps. Note that the capacity defined here describes the maximum attack load a proxy network can possibly resist (the load to saturate all the edges), rather than the maximum throughput of application traffic a proxy network can deliver.

Proxy network scaling results are shown in Figure 15. The X-axis is the number of edge proxies in the proxy network (they all have height 3); the Y-axis is the user-experienced response time for a certain percentile of users. We can see that for up to 95 percent users, the curves stay horizontal and less than 2 seconds (recall from Figure 6 that the 95 percentile performance for direct application access without attacks is 2 seconds). If we define 95% users not being affected by DoS attacks as successful DoS resilience, then the amount of attack traffic can be tolerated grows linearly with the size of the proxy network.

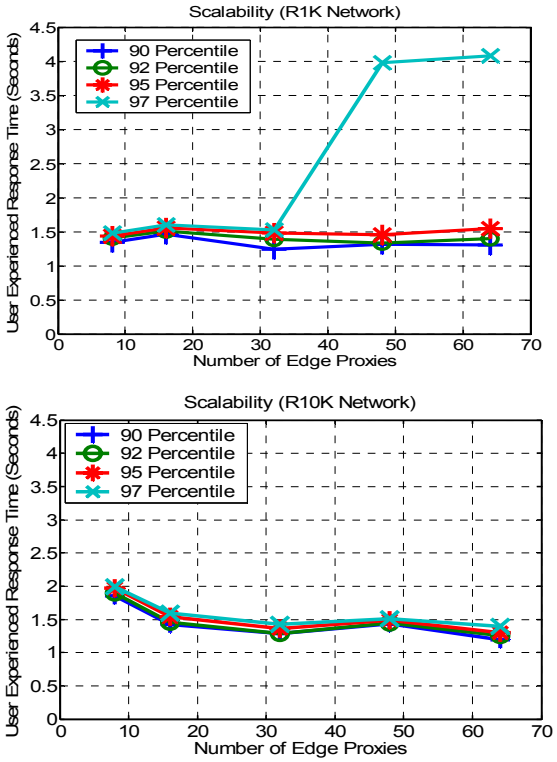

Figure 15 Resilience and Proxy Network Size

Our analysis on the spread attacks and concentrated attacks explains why there is a near-linear scaling property. For both attack scenarios, as long as there is available capacity in the edge proxies, they can keep the application accessible to users if users can switch to the

edge proxies that are not saturated. Therefore, having more edge proxies allows larger attacks being tolerated. However, we can also observe from Figure 15 that the scaling is not perfectly linear (results for the R1K network); that is due to the limitation of the underlying network discussed in Figure 11. Such phenomenon is less likely when the proxies are scattered in larger underlying networks, since in that case it is less likely to have two edge proxies in the same sub-network. We can see in Figure 15 that the proxy network achieves a much better scaling in the R10K network, which is larger than the R1K network. These results indicate that, when proxies are widely dispersed in a large network, the proxy network has the potential to achieve DoS resilience with excellent scaling properties.

In summary, we first explore the user-experienced performance using a range of proxy networks in two simulated large realistic network environments. We find that proxy networks can in fact improve the performance for TCP-based applications. Then, we conduct a series of experiments investigating the user-experienced performance under different attack scenarios using a range of proxy networks. We find that proxy networks provide effective resilience to spread and concentrated attacks – most users (>90%) can retain good performance. In exploring the properties of large proxy networks, we find that by growing the proxy network size (width), the magnitude of DoS attacks that it can tolerate grows almost linearly. Therefore, in realistic large network environments, proxy networks can have great performance potential and scalable DoS-resilience.

6 Related Work

Our focus is the capabilities of proxy networks used for DoS defense. The most related studies are those exploring the use of overlay networks to resist DoS attacks. Secure Overlay Services (SOS) [1] protects applications against flooding DoS attacks by installing filters around applications and only allowing traffic from secret "servlets". SOS uses Chord [29] to mediate communication between users and the secret servlets, without revealing the IP addresses of the servlets. WebSOS [7] is an implementation of SOS. Mayday [4] generalizes the SOS architecture, and analyzes the implications of choosing different filtering techniques and overlay routing mechanisms. Internet Indirection Infrastructure (i3) [3, 5] also uses Chord overlay to protect applications from direct DoS attacks. SOS, WebSOS, Mayday, and i3 can all be viewed as specific instances of our generic proxy network. Each of these efforts has involved some evaluation via theoretical analysis or small-scale experiments.

The primary distinctions of our work are:

First, our work differs in scale, fidelity, and realism. These other efforts except WebSOS [7] are limited to theoretical analysis and small-scale experiments, which cannot capture detailed network and application dynamics, such as router queues, packet drops, real temporal and feedback protocol behavior. All these factors are critical to application and proxy network performance. WebSOS [7] conducted larger scale experiments and performance evaluation on PlanetLab testbed. But due to the malicious nature of DoS attacks, they could not study the DoS resilience problems on PlanetLab. Therefore none of the other efforts can capture detailed application performance dynamics under DoS attacks; the validity of their results depends on the accuracy of their models, which has not been empirically validated. In contrast, our studies are based on detailed network behavior, and explore large scale network structures as well as proxy networks, with real application, attacks, and protocol software. Our work provides a qualitative advance in the scale, fidelity, and realism over the other efforts. Furthermore, our results provide the first quantitative understanding of the DoS-resilience capability of proxy networks in large-scale network environments. The primary reason we are able to undertake these studies is the novel capabilities of MicroGrid.

Second, our work differs in focus. Each of these other efforts focuses on their specific proposed solution, exploiting its structure and characteristics for analysis. Therefore the evaluation of one often applies only to that particular instance of proxy networks. In contrast, our work focuses on the fundamental capabilities and limitations of proxy networks in general. Our results apply to a wide range of proxy networks, including the other proposed solutions.

Another class of related research is on the performance and static resilience of overlay networks in general. [30] studied these issues from a graph theoretic perspective, and [31] takes an empirical approach to study the overlay network performance. There are three key differences between our work and their studies. First, our work studies the impact of DoS attacks, which affects network dynamics and the performance of real applications, which is not their focus. Second, our work studies performance of real applications, taking into account dynamic behavior of network protocols such as TCP, while their work only

considers RTT. Third, our work focuses on the performance between users and an application, while they study performance between any pair of overlay nodes.

A third class of related work is the simulation studies on Internet worms and their impact on BGP [32, 33]. They focus on worm propagation and its impact on the network, particularly on the behavior and vulnerabilities of BGP, which are not our focus. These studies are complementary to ours.

Our work focuses on infrastructure-level DoS attacks and their countermeasures. Meanwhile there are studies such as Mutable Services [34] and Roaming Honeypots [35] which explore solutions to protect Internet applications from application-level DoS attacks. These efforts focus on a different class of DoS attacks, and are complementary to our work.

7 Conclusions and Future Work

To understand the performance implications and DoS-resilience capability of proxy networks in large realistic networks, we use a detailed large-scale online network simulator – MicroGrid [9, 10] – to study proxy networks with real applications and real DoS attacks. Using MicroGrid, we are able to conduct detailed performance studies in large networks environment with complex, typical application packages, and real attack programs. Our studies include networks with up to 10,000 routers and 40 (ASes), with a physical extent comparable to the North American continent. We believe this is the first empirical study of proxy networks for DoS resilience at large-scale, using real attacks, and in a realistic environment.

Our experiments explore a range of network sizes, proxy network configurations, attack parameters, and application characteristics. The key results are summarized below:

- Rather than incurring a performance penalty, proxy networks can improve users' experienced performance, reducing latency and increasing delivered bandwidth. The intuition that indirection reduces performance turns out to be incorrect, as the improved TCP performance more than compensates.

- Proxy networks can effectively mitigate the impact of both spread and concentrated large-scale DoS attacks in large network environment. Our experiments have shown that a 192-node proxy network with 64 edge proxies (each connected by a 100Mbps uplink), can successfully resist a range of large-scale distributed DoS attacks with up to 6.0Gbps aggregated traffic and different attack load distribution; most users (>90%) do not experience significant performance degradation under these attack scenarios.

- Proxy networks provide scalable DoS-resilience – resilience can be scaled up to meet the size of the attack, enabling application performance to be protected. Resilience increases almost linearly with the size of a proxy network; that is the attack traffic a given proxy network can resist while preserving a particular level of application performance grows almost linearly with proxy network size.

These results provide empirical evidence that proxy networks can be used to tolerate DoS attacks and quantitative guidelines for designing a proxy network to meet a resilience goal.

Our main contributions are the following. First, we provide the first large-scale empirical study on the DoS resilience capability of proxy networks using real applications and real attacks. This provides a qualitative advance over previous studies based on theoretical models and small scale experiments. Second, we provide the first set of empirical evidence on large-scale network environment to prove that proxy networks have effective and scalable resilience against infrastructure-level DoS attacks. Third, we provide a detailed performance analysis of proxy networks in large-scale network environment, and show that in contrast to intuition proxy networks can improve user-experienced performance.

There are several directions for future work. First, we can study proxy networks with topologies which have multiple paths from each edge proxy to the application, in order to understand the benefit of multi-path on performance and DoS-tolerance. Second, multiple applications can share the same proxy network. We can study the correlated impact of DoS attacks on multiple applications. Third, further study is necessary to understand the impact of proxy deployment and proxy network topology on user-experienced service performance. Fourth, this paper studied the impact of proxy network depth on user-experienced service performance, but we did not study its impact on the DoS-resilience capability of proxy networks. It needs to be addressed in our future work. Last, this work focuses on congestion-based DoS attacks. It can be extended to study other forms of DoS attacks, such as SYN floods.

References

1. Keromytis, A.D., V. Misra, and D. Rubenstein. *SOS: Secure Overlay Services*. in *ACM SIGCOMM'02*. 2002. Pittsburgh, PA: ACM.

2. Stoica, I., et al. *Internet Indirection Infrastructure*. in *SIGCOMM*. 2002. Pittsburge, Pennsylvania USA.

3. Adkins, D., et al., *Towards a More Functional and Secure Network Infrastructure*. 2003, Computer Science Division, UC Berkeley: Berkeley.

4. Andersen, D.G. *Mayday: Distributed Filtering for Internet Services*. in *4th Usenix Symposium on Internet Technologies and Systems*. 2003. Seattle, Washington.

5. Adkins, D., et al. *Taming IP Packet Flooding Attacks*. in *HotNets-II*. 2003.

6. Wang, J., L. Lu, and A.A. Chien. *Tolerating Denial-of-Service Attacks Using Overlay Networks – Impact of Topology*. in *2003 ACM Workshop on Survivable and Self-Regenerative Systems*. 2003. Washington DC: ACM.

7. Stavrou, A., et al., *WebSOS: An Overlay-based System For Protecting Web Servers From Denial of Service Attacks*. Elsevier Journal of Computer Networks, special issue on Web and Network Security, 2005.

8. Wang, J. and A.A. Chien, *Understanding When Location-Hiding Using Overlay Networks is Feasible.* Elsevier Journal of Computer Networks, special issue on Overlay Distribution Structures and Their Applications, 2005.

9. Liu, X. and A. Chien. *Traffic-based Load Balance for Scalable Network Emulation*. in *SuperComputing 2003*. Noverber 2003. Phoenix, Arizona: the Proceedings of the ACM Conference on High Performance Computing and Networking.

10. Liu, X., H. Xia, and A.A. Chien, *Validating and Scaling the MicroGrid: A Scientific Instrument for Grid Dynamics*. Journal of Grid Computing, 2003.

11. Dittrich, D., *The DoS Project's "trinoo" distributed denial of service attack tool*. 1999, University of Washington.

12. Dittrich, D., *The "Tribe Flood Network" distributed denial of service attack tool*. 1999, University of Washington.

13. Dittrich, D., et al., *The "mstream" distributed denial of service attack tool*. 2000.

14. CERT, *"Code Red II:" Another Worm Exploiting Buffer Overflow In IIS Indexing Service DLL*. 2001.

15. CERT, *"Code Red" Worm Exploiting Buffer Overflow In IIS Indexing Service DLL*. 2001.

16. apache, *Apache HTTP Server Version 2.0 Documentation*.

17. JoeDog.org, *http://www.joedog.org/siege/index.php*.

18. Akamai, *Akamai Technology Overview*.

19. Liu, X. and A.A. Chien. *Realistic Large-Scale Online Network Simulation*. in *SuperComputing'04*. 2004. Pittsburgh, PA.

20. Medina, A., et al. *BRITE: An Approach to Universal Topology Generation*. in *the International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems- MASCOTS '01*. 2001. Cincinnati, Ohio.

21. Lougheed, K. and Y. Rekhter, *RFC 1106: Border Gateway Protocol (BGP)*. 1990.

22. Moy, J., *RFC 2178: OSPF Version 2*. 1998.

23. Socolofsky, T. and C. Kale, *RFC 1180 - TCP/IP tutorial*. 1991.

24. Postel, J., *RFC 792 - Internet Control Message Protocol*. 1981.

25. Chun, B., et al., *PlanetLab: An Overlay Testbed for Broad-Coverage Services.* ACM Computer Communications Review, a special issue on tools and technologies for networking research and education, 2003. **33**(3).

26. *The ns Manual (formerly ns Notes and Documentation)*, K. Fall and K. Varadhan, Editors, UC Berkeley, LBL, USC/ISI, and Xerox PARC.

27. Faloutsos, M., P. Faloutsos, and C. Faloutsos. *On Power-Law Relationships of the Internet Topology*. in *SIGCOMM'99*. 1999.

28. Swany, D.M. and R. Wolski. *Data Logistics in Network Computing: The Logistical Session Layer*. in *IEEE Network Computing and Applications (NCA'01)*. 2001.

29. Stoica, I., et al. *Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications*. in *ACM SIGCOMM'01*. 2001.

30. Loguinov, D., et al. *Graph-Theoretic Analysis of Structured Peer-to-Peer Systems: Routing Distances and Fault Resilience*. in *SIGCOMM*. 2003. Karlsruhe, Germany: ACM.

31. Jain, S., R. Mahajan, and D. Wetherall. *A Study of the Performance Potential of DHT-based Overlays*. in *the 4th Usenix Symposium on Internet Technologies and Systems (USITS)*. 2003. Seattle, WA.

32. Lad, M., et al. *An Analysis of BGP Update Burst during Slammer Attack*. in *Proceedings of the 5th International Workshop on Distributed Computing (IWDC)*. 2003.

33. Liljenstam, M., et al. *Simulating realistic network worm traffic for worm warning system design and testing*. in *Proceedings of the 2003 ACM workshop on Rapid Malcode*. 2003.

34. *Mutable Services*, New York University.

35. Khattab, S.M., et al. *Roaming Honeypots for Mitigating Service-level Denial-of-Service Attacks*. in *Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)*. 2004.

---

[1] The term "overlay network" refers to both structured Distributed Hash Tables (DHT) and unstructured overlays.

[2] Only a one-way trip is needed from the edge proxy to the application proxy, instead of a full hand-shake. In fact, once the user gets connected to the edge proxy, it can start sending data. This can be overlapped with the connection setup at the application proxy side.

[3] Use of user authentication may not be a reason for the performance degradation, because it also introduces great overhead for the baseline case of direct access.

[4] WebSOS is implemented in Java. Our implementation is in C++, and optimized to achieve comparable throughput as Apache server on Linux.