The Open Network Laboratory

Charlie Wiseman, Jonathan Turner, Patrick Crowley

Department of Computer Science and Engineering
Washington University in St. Louis
{wiseman,jon.turner,pcrowley}@wustl.edu

ABSTRACT

The Open Network Laboratory is an Internet-accessible network testbed that provides access to a large set of heterogeneous networking resources including extensible multi-gigabit routers and NetFPGAs. Researchers build and configure experimental topologies with a simple Java GUI then run interactive experiments using those topologies. Real-time charts are available to monitor network statistics such as bandwidth, packet rates, and queue lengths. This paper will show how the testbed can be used to conduct sophisticated network experiments and enhance networking research through two examples.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*network communications*

General Terms

Experimentation

Keywords

Network testbeds, network experimentation

1. INTRODUCTION

The Open Network Laboratory (ONL) testbed has been expanded over the last year to include a diverse collection of resources to better support networking research and education. This paper will show how ONL can be used to conduct sophisticated network experiments and as a tool for laboratory experiences in advanced technical courses. A brief description of ONL is given here, and more information is available at the testbed website [4].

ONL is an Internet-accessible network testbed that provides access to a large set of heterogeneous networking resources including extensible multi-gigabit routers. It is an emulation testbed similar to Emulab [5]. Researchers build and configure experimental topologies with a simple Java GUI then run interactive experiments using those topologies. Users are granted sole control of all physical resources that make up their topology for the duration of their experiment. Real-time charts are available to monitor network statistics such as bandwidth, packet rates, and queue lengths. The ONL software infrastructure automatically manages all of the details of setting up the experiment, including mapping the topology to actual hardware resources and ensuring that no active experiments can interfere with one another. Testbed resources are shared via a reservation mechanism that allows users to reserve the necessary resources for their experiments in advance.

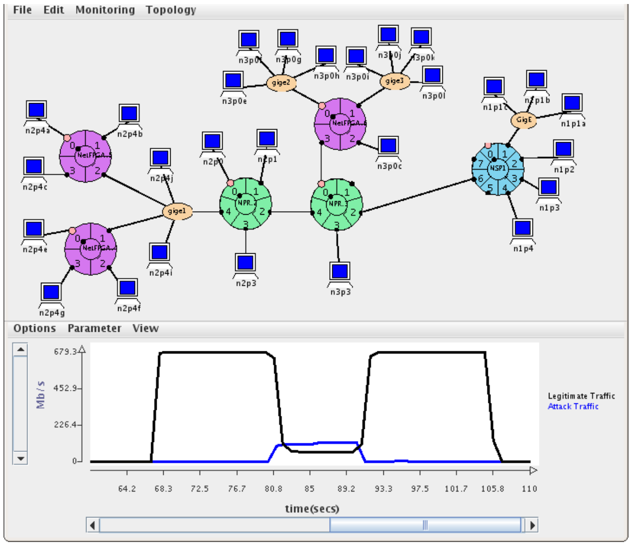

Figure 1: Example ONL session for a DoS attack prevention system.

Researchers use ONL to run experiments over a broad range of network configurations which contain diverse networking resources. This allows new applications and protocols to be rigorously tested in a variety of realistic configurations under controlled and reproducible conditions. Moreover, the various types of networking technology used in ONL provide an opportunity for researchers to gain experience with platforms that they might not otherwise have access to use. The newest addition to ONL is the NetFPGA [3]. Other resources include routers built to resemble scalable high-end routing platforms, routers built using network processors [6], Ethernet switches, and Linux PCs.

2. EXAMPLES

Two example ONL experiments are described below to show how the testbed can be used to conduct sophisticated research.

2.1 DoS Attack

The first experiment explores techniques for Denial of Service attack prevention. A standard client-server configura-

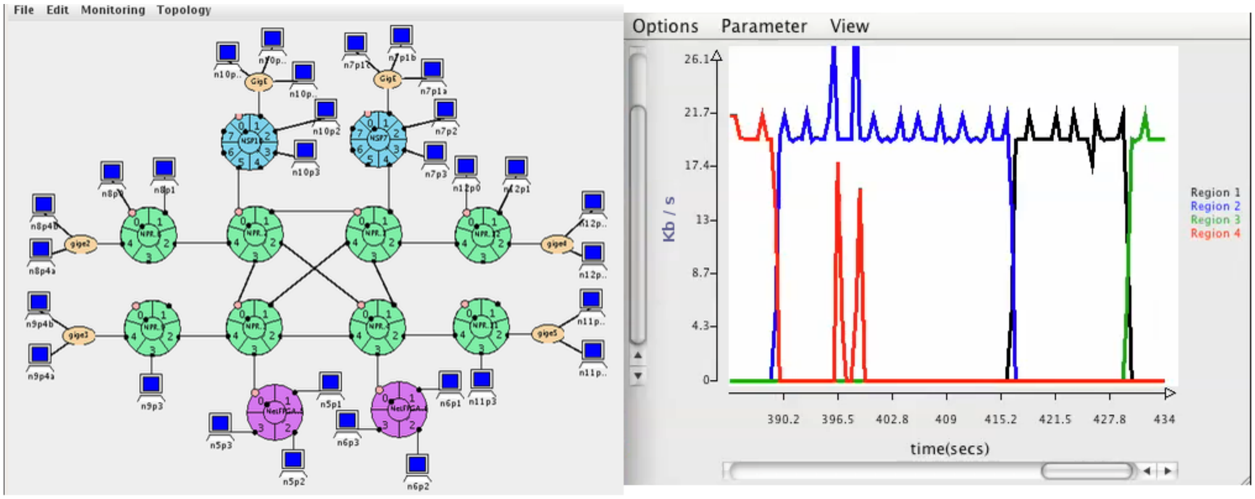

Figure 2: Example ONL session for a distributed FPS game.

tion is used in a medium-sized topology. Many clients make legitimate periodic connections to a server. A separate set of hosts is used to launch a TCP SYN flood attack on the server which causes the legitimate clients to see degraded service. To counter the attack, the router nearest the server activates a router *plugin* which can detect the sources of the SYN flood and block the attack at the router. Plugins are user-written code modules that are used to extend and enhance a router's functionality. In this configuration, both the legitimate clients and the attacking machines are located in OpenFlow [2] networks. NetFPGAs are used as OpenFlow switches, and the Nox OpenFlow controller is used. Once the sources of the attack are identified, the Nox controller is notified and the compromised machines are restricted from sending any further traffic. Figure 1 is a screenshot of an ONL session for this experiment. The top of the figure is the network topology and the bottom is a real-time chart showing legitimate and attack traffic. As the figure shows, legitimate traffic is substantially degraded when the attack starts, but returns to normal once the attack is detected and blocked.

2.2 FPS Game

The second experiment uses a larger network topology to emulate a Wide Area Network. This configuration is used to explore how router plugins can be used to provide a lightweight multicast mechanism that enables highly scalable first person shooter game sessions. In this experiment, the game world is divided into a set of regions and each region is associated with a particular multicast group. Each game server subscribes to the multicast groups for regions that it is interested in, determined by the game world location of players hosted on that server. Plugins are installed in some of the routers to support the multicast protocol. This involves handling the multicast subscription requests and packet replication at the router. The multicast protocol is designed and tailored specifically for applications such as first person shooter games which have stringent delay requirements. Traffic is generated from Linux PCs running the distributed game server, based in part on Colyseus [1]. The experiment includes a combination of player-controlled avatars and bots. Other PCs use standard traffic generators to produce cross traffic. Figure 2 is a screenshot of an ONL session running the experiment. The left side of the figure shows the network topology. The right side is a real-time chart monitoring bandwidth on some of the multicast groups which shows how traffic changes as entities move around in the game world.

3. REFERENCES

[1] A. Bharambe, J. Pang, and S. Seshan. Colyseus: A distributed architecture for online multiplayer games. In *NSDI '06: Proceedings of the 3rd Symposium on Networked Systems Design and Implementation*, 2006.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. Openflow: enabling innovation in campus networks. volume 38, pages 69–74, New York, NY, USA, 2008. ACM.

[3] J. Naous, G. Gibb, S. Bolouki, and N. McKeown. Netfpga: reusable router architecture for experimental research. In *PRESTO '08: Proceedings of the ACM workshop on Programmable routers for extensible services of tomorrow*, pages 1–7, New York, NY, USA, 2008. ACM.

[4] ONL. Open network laboratory website. http://onl.wustl.edu.

[5] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar. An integrated experimental environment for distributed systems and networks. In *OSDI '02: Proceedings of the 5th Symposium on Operating Systems Design and Implementation*, pages 255–270, New York, NY, USA, 2002. ACM.

[6] C. Wiseman, J. Turner, M. Becchi, P. Crowley, J. DeHart, M. Haitjema, S. James, F. Kuhns, J. Lu, J. Parwatikar, R. Patney, M. Wilson, K. Wong, and D. Zar. A remotely accessible network processor-based router for network experimentation. In *ANCS '08: Proceedings of the 4th ACM/IEEE Symposium on Architectures for Networking and Communications Systems*, pages 20–29, New York, NY, USA, 2008. ACM.