Techniques for Testing Experimental Network Protocols

Brian L. Tierney
Lawrence Berkeley National Laboratory

1.0 Introduction

Designers of new network protocols or enhancements to existing protocols know that thoroughly testing a new protocol is very hard. A typical conference paper or journal article will describe a few results using the *ns* simulator, and some tests on a real network. It is rare to find a paper that actually has results for a wide range of network paths, and runs tests over an extended period of time (some exceptions include [2] and [1]). Often the reported results are in fact on very unusual testbed networks, which typically have very little congestion, and do not reflect typical internet paths.

There are a number of reasons for this. It is hard to get accounts on remote testing hosts, and even harder to get custom kernels installed. It is quite tedious to run regular, periodic tests over many paths over an extended period of time. Fortunately some progress is being made in this area. PlanetLab [10] is helping by providing a large test environment, however PlanetLab is limited to speeds of 100 Mbits/sec and only allows testing of user-space protocols such as UDT [11] and RBUDP [12]. Internet2 is working on a infrastructure to provide access to a number of testing platforms across the Internet2 network [7].

Despite these difficulties, the network research community must become more rigorous in its testing methodology; otherwise, published results will be useless for predicting the behavior of the protocol in any other network! Tests should be run periodically over an extended period of time. As part of any experiment, known sources of variability on the test systems (e.g. load on the networks and hosts by other users and applications) should be enumerated, and reasonable efforts should be made to measure those that cannot be controlled. All raw data results, including TCP header traces when possible, should be made public, for analysis by other researchers. An analysis of both the spread and center of the data (e.g. minimum, maximum, average, and standard deviation) should always be part of the published results, with more detail added if the data is not statistically "normal".

To address the issue of running extended period tests and collecting the results, we have developed the Network Tool Analysis Framework (NTAF). This talk will include a brief rant on the lack of scientific testing methodology used by some of this community, and describe how NTAF can be used to help with this aspect of the problem.

2.0 Network Tool Analysis Framework (NTAF)

We have developed a framework for running network test tools and storing the results in a relational database, which we call the Network Tool Analysis Framework (NTAF). NTAF manages and runs a set of network testing tools and sends the results to a database for later retrieval.

The goal of the NTAF is to make it easy to collect, query, and compare results from any set of network or host monitoring tools running at multiple sites. The basic function performed by NTAF is to run tools at regular intervals, plus or minus a randomization factor, and send their results to a central archive system for later analysis. For example, *iperf* can be configured to run for 20 seconds every 2 hours plus or minus 10 minutes, to a list of hosts. Some tools cannot run on the same host without perturbing each other, and NTAF is designed to accommodate this by never scheduling these tools at the same time.

The original focus on NTAF was on evaluating various bandwidth and capacity estimation tools. But we have found it to also be useful for testing new network protocols. NTAF runs on any Unix-based system, but for TCP testing it is most useful on Web100 [13] Linux systems. Web100 allows us to also monitor TCP parameters such as CWND, number of congestion events, ssthresh, and so on. If NTAF is running on a host that is using Net100 kernel [3], NTAF can alternate its tests between standard TCP Reno, HS-TCP, and scalable TCP, to allow us to compare the results of each. When combined with the Net100 WAD [3] (which can monitor and modify TCP settings for any connection) NTAF can be used to passively collect Web100 TCP data for "real" applications such as GridFTP, and to periodically modify the flavor of TCP used (e.g., Reno, Vegas, HSTCP, Scalable TCP) to collect data for comparing the throughput.

The results for all NTAF tests are converted into NetLogger events [9]. NetLogger provides us with an efficient and reliable data transport mechanism to send the results to a relational database event archive. For example, if the network connection to the archive goes down, NetLogger will transparently buffer monitoring events on local disk,

and keep trying to connect to the archive. When the archive becomes available, the events buffered on disk will be sent automatically. More details are available in [6].

Each NTAF-generated NetLogger event contains the following information: timestamp, program name, event name, source host, destination host, and value. Using a standard event format with common attributes for all monitoring events allows us to quickly and easily build SQL tables of the results. More details are in [9].

3.0 Sample Results

We have deployed NTAF servers on several hosts, including hosts at Oak Ridge National Lab (ORNL), Lawrence Berkeley National Lab (LBNL), and Pittsburgh Supercomputing Center (PSC). We are currently using NTAF to run the following tests: *ping*, *pathrate*, *pathload*, *pipechar*, *traceroute*, *iperf* (instrumented to collect Web100 information), *GridFTP*, *netest*, and a CPU and memory sensor based on procinfo. *iperf* tests are run using stock linux TCP Reno, HS-TCP, Scalable TCP, and we are hoping to soon add TCP Vegas and possibly FAST [4]. By PFLDnet we plan to report results for UDT and RBUDP as well.

Using NTAF to run the tests and send results to the SQL database, we can easily perform SQL queries to provide results such as the results shown in Table 1. The average result in this table is actually a trimmed mean , throwing out the top and bottom 5% of the results as outliers. Instead of 5-10 tests run over 2-3 hours, such as many papers on protocols report, this shows over 200 tests spread over a 1 week period, greatly increasing the credibility of the results.

Table 1 : 7 Day Summary of experimental iperf TCP Test Results

	Throughput (Mbits/second)		
Path	**Standard TCP min/max/ave/std**	**HS-TCP min/max/ave/std**	**Scalable TCP min/max/ave/std**
LBNL to ORNL	201 / 353 / 334 / 21	62 / 363 / 317 / 48	58 / 388 / 332 / 54
PSC to ORNL	44 / 277 / 134 / 48	11 / 310 / 143 / 68	23 / 331 / 211 / 71
LBNL to PSC	47 / 117 / 82 / 34	29 / 503 / 130 / 93	30 / 521 / 135 / 92
average over all paths	62 / 376 / 177	30 / 392 / 197	39 / 412 / 276

Storing the results in a relational database has proven to be very useful. For example, we can easily look for correlations between low throughput and high CPU, and possibly throw out this data on the assumption that some other process on the host was perturbing the results. We can also look for correlations between throughout and various Web100 variables, such as CWND or congestion events.

4.0 Limitations and Future Work

NTAF is still a work-in-progress, and there are several limitation to the current version, which we plan to address soon. There is no communication mechanism between NTAF servers. This means the sending side can t request changes to the receiving side to test protocols modifications that involve changes to both the sender and receiver hosts. We need to define a better mechanism for defining network experiments with co-scheduled tests, so that fairness can be evaluated. For example, how does a RBUDP test compete with a TCP Reno stream? So far NTAF has only been used on small (less than 10) of test hosts. We also want to deploy NTAF on a large number of PlanetLab hosts and explore any scalability issues that may arise.

In summary, NTAF provides a mechanism for running tests and collecting results. PlanetLab provides access to a large number of network paths for testing. The combination of these provides a large step in our ability to do better testing of new protocols. There are still a number of problems not addressed by NTAF with PlanetLab, such as how to test kernel-based protocol changes, and how to test high-end or low end-paths, as PlanetLab consists of all 100 Mbit ethernet end hosts. Network protocol researchers should try to take advantage of tools and techniques such as these whenever possible, and push for a high-speed version of PlanetLab to provide a more interesting test environment.

5.0 Acknowledgments

This work was supported by the Director, Office of Science. Office of Advanced Scientific Computing Research. Mathematical, Information, and Computational Sciences Division under U.S. Department of Energy Contract No. DE-AC03-76SF00098. See the disclaimer at http://www-library.lbl.gov/disclaimer. This is report no. LBNL-53932.

6.0 References

[1] Akella, A., S. Seshan, A. Shaikh, *An Empirical Evaluation of Wide-Area Internet Bottlenecks* , Internet Measurement Conference 2003,October 27-29, 2003, Miami, Florida, USA.

[2] Cottrell, L., Connie Logg, I-Heng Mei, *Experiences and Results from a New High Performance Network and Application Monitoring Toolkit,* Proceedings of the Passive and Active Monitoring Workshop, San Diego, April 2003.

[3] Dunigan, T., M. Mathis and B. Tierney, *A TCP Tuning Daemon*, Proceeding of IEEE Supercomputing 2002 Conference, Nov. 2002, LBNL-51022.

[4] FAST: http://netlab.caltech.edu/FAST/

[5] Floyd, S., *HighSpeed TCP for Large Congestion Windows*, Internet draft: draft-ietf-tsvwg-highspeed-01.txt, August 2003.

[6] Gunter, D., B. Tierney, K. Jackson, J. Lee, M. Stoufer, *Dynamic Monitoring of High-Performance Distributed Applications*, Proceedings of the 11th IEEE Symposium on High Performance Distributed Computing, July 2002.

[7] Internet2 End-2-end Performance Initiative; http://e2epi.internet2.edu/index.html

[8] Kelly, T., Scalable TCP, http://www-lce.eng.cam.ac.uk/~ctk21/scalable/

[9] Lee, J., D. Gunter, M. Stoufer, B. Tierney, Monitoring Data Archives for Grid Environments , Proceeding of IEEE Supercomputing 2002 Conference, Nov. 2002, Baltimore, Maryland.

[10] PlanetLab: http://www.planet-lab.org/

[11] Gu Y., R. Grossman, SABUL/UDT (Simple Available Bandwidth Utilization Library) / (UDP-based Data Transfer Protocol); http://www.evl.uic.edu/eric/atp/sabul.pdf

[12] RBUDP: http://www.evl.uic.edu/eric/atp/

[13] Web100: http://www.web100.org/