David Lapsley: A Practical Approach to e-VLBI over Long Fast Networks, IVS 2004 General Meeting Proceedings, p.215–219
http://ivscc.gsfc.nasa.gov/publications/gm2004/lapsley

A Practical Approach to e-VLBI over Long Fast Networks

David Lapsley

MIT, Haystack Observatory
e-mail: dlapsley@haystack.mit.edu

Abstract

Modern VLBI experiments demand transfers of up to several terabytes per day, and though modern research and education networks have backbone bandwidths of up to 10 Gbps and should be easily able to handle this load, at this speed, traditional transport options (TCP/UDP) do not perform/behave well. Many enhancements have been proposed to TCP, in particular, to improve its performance in high speed networks, however, they are not yet widely deployed. In the interim, advanced application layer transport protocols offer an alternative, easily deployable and effective solution.

In this paper, we discuss some of these protocols and look at the results of some tests using these protocols. Some of these protocols are still experimental, so they should be used/deployed only after extensive reliability and fairness testing. However, they do provide a relatively easy and efficient way in which to improve data transport rates across high speed, long round trip time networks.

1. Introduction

In order to realize the benefits of e-VLBI, data must be transported in an efficient manner at high bandwidths. However, the rate of evolution of the core backbone speeds has far exceeded the rate of evolution of the traditional network transport protocols used to support applications such as the File Transfer Protocol (ftp), scp, world wide web etc. As such, many scientists that are trying to transport large amounts of data over a wide area face a number of issues in order to realize their goal. In this paper, we describe some of the e-VLBI experiments that have been performed to date, the issues that are currently being faced by e-VLBI practitioners and some of the alternative solutions that are available for alleviating these issues. Finally, we conclude with a comment on the suitability of these alternatives for e-VLBI data transport.

The two most common transmission protocols in use today are: (1) the Transmission Control Protocol (TCP): which provides congestion avoidance and control, guaranteed, in-order data delivery and is typically used for data applications that can tolerate delay but not loss (e.g. ftp) and (2) the User Datagram Protocol (UDP): which is a light-weight datagram protocol that does not provide any form of congestion avoidance or control, does not guarantee data delivery and is typically used for real-time applications that can tolerate data loss, but not delay (e.g. audio conferencing/internet telephony).

Both protocols have their advantages and limitations. The limitations are somewhat exacerbated on networks that have high bandwidths and long round trip times (which is typical for e-VLBI). In particular, TCP throughput can suffer significantly under these circumstances. UDP on the other hand is not really useful for sustained high bandwidth data flows across shared, wide area networks as it does not provide any sort of reliability mechanism.

A number of enhancements to this layer have been proposed. These include application layer transport protocols such as the Simple Available Bandwidth Utilization Library (SABUL – which is also known as the User Datagram Transport (UDT) protocol) [3], Tsunami [1] and Reliable

Blast UDP (RBUDP) [2]. These all transport data using UDP channels that have an extra layer of reliability and congestion avoidance and control added. Many enhancements to the traditional TCP stack have been proposed: Low et al.'s Fast AQM TCP [4], Floyd's High Speed TCP (HSTCP) [6], Kelly's Scalable TCP (STCP) and Kitabi's XCP [7]. These are all implemented at the kernel layer inside the operating system.

2. e-VLBI Experiments to Date

The material in this paper is based on practical experience obtained from the co-ordination and running of many e-VLBI experiments. In this section, we provide a brief summary of the experiments in which MIT Haystack Observatory has participated.

In October, 2002 a sustained disc-to-disc transfer of 788 Mbps was achieved between Mark 5s at the NASA Goddard Geophysical and Astronomical Observatory in Maryland and the correlator at Haystack Observatory in Westford, Massachusetts. Data was simultaneously transferred from the Westford telescope to the correlator [8]. This is the highest rate e-VLBI experiment to date.

Since that time, there have been many experiments between the US and Japan in support of geodetic experiments. In June of 2003 an entire Intensive experiment was transferred from the Communications Research Laboratory (CRL) antenna in Kashima, Japan to Haystack. The data was recorded using the K5 format and then converted to the Mark 5 format for correlation on the Mark IV correlator at Haystack. The entire 40 GB dataset was transferred in 51 minutes, with the first correlation results available in under an hour. The average transfer rate was 107 Mbps. At the same time, data from the Westford antenna was transferred to Japan, where it was converted to K5 format and correlated using the CRL software correlator. An estimate of the UT1-UTC offset was obtained in under 24 hours. This was the first time this had been achieved.

Since that time, several e-VLBI transfers have been performed between Japan and the US as part of the International VLBI Service's (IVS)"T-" series and "CRF-" series experiments. In particular, e-VLBI has been used to transport some of the data for the following experiments: CRF22, CRF23, T2023, T2024 and T2026. This series of experiments has been quite important in paving the way for the "operationalization" of e-VLBI data transfers. Special software and procedures have been developed to support these types of experiments.

In addition to e-VLBI experiments, network protocol testing has been performed to determine the efficacy of various protocols for the transfer of e-VLBI data. At an International e-VLBI demonstration at the Internet2 Fall Member's Meeting in October 2003 data rates of 644 Mbps and 400 Mbps between the US and Japan were obtained using FAST TCP [4] and HSTCP [6] respectively. Since that time, rates in excess of 900 Mbps have been achieved on slightly lossy links using high speed protocols such as Tsunami [1] and UDT [3].

These experiments have provided invaluable experience in how to efficiently transport e-VLBI data across national and international wide area high speed research networks.

3. e-VLBI Data Transport Issues

TCP is used for the majority of Internet data transfers. It includes a congestion control mechanism that ensures that users competing for network resources share those resources in a manner that is in some sense fair, without driving the network into congestion collapse. TCP is used by applications such as ftp, scp and the world wide web to create data connections and reliably transport data across the network. The fundamental algorithms underlying TCP were designed roughly 20 years ago by Van Jacobson. At this time, the bandwidth and size of networks was much less than today. In current high speed networks, bandwidths of 10 Gbps are not uncommon in the network backbone, with round trip times ranging up to 100s of milliseconds. In this environment, TCP throughput can be severely impacted by packet loss. The famous Mathis equation [5] relates a TCP connection's bandwidth (BW) to its mean segment size (MSS ≡ packet size), round trip time (RTT) and the packet loss probability (p):

$$BW = \frac{MSS}{RTT}\sqrt{\frac{C}{p}} \tag{1}$$

Where C is a constant that is commonly chosen as 1.5 and BW is in bytes/second.

To gain insight into how sensitive TCP throughput is to packet loss at high speed, consider a concrete example (often used by Sally Floyd): a TCP connection with MSS=1500 bytes, RTT=100 ms and a desired BW of 10 Gbps. This requires a packet loss rate of at most one drop in every 5,000,000,000 packets. This is at most one drop every 1.67 hours and is simply not realistic in present day shared networks.

UDP is a light-weight packet protocol that does not have any congestion control implemented. It was originally intended to support real-time protocols that were loss-insensitive, but delay sensitive. UDP does not react to packet loss, but will allow an application to transmit data as fast as it wants, without regard for the amount of available bandwidth in the network, nor the UDP packet loss rate. UDP can provide high throughput, but is unreliable. Sustained high rate UDP flows can also impact other network users, and/or look like Denial of Service attacks to network administrators. This can result in a UDP flow being shut down as a results of a user complaint or network administrator concern.

The two traditional transport protocols, TCP and UDP, both have limitations when it comes to the transport of sustained, high bandwidth data flows across shared, high speed networks. In the next section, we will look at some alternatives that can be used in place of these protocols to achieve enhanced data throughput.

4. e-VLBI Data Transport Alternatives

There are basically two different approaches to enhancing the throughput of data transmission across high speed wide area networks:

- **Application Layer Protocols**: such as Tsunami and SABUL/UDT. These are implemented as user space applications that implement rate-based flow control and reliability on top of a UDP layer. These are easy to deploy and use but may be overly aggressive.
- **Enhancements to TCP**: such as FAST and HSTCP. These are implemented in kernel space and are high performance enhancements to TCP. Typically, they require a special kernel or kernel patch to be applied. They are slightly more complicated to install and use, but are a much better alternative in the long term. The major advantage they have is that once the kernel is in place, no adjustments are required to the application software.

In this paper, we investigate the performance of Tsunami and SABUL/UDT.

Tsunami is a rate-based ftp-like application. It uses two connections: a TCP connection for the transfer of control information between the client and server and a UDP connection for the transfer of data. The rate at which data is transmitted into the network is controlled by varying the inter-packet delay. This value is varied according to an algorithm that monitors the packet loss rate and adjusts the transmission rate based on this.

UDT (formerly SABUL) also uses rate-based flow control. However, UDT is a C++ library rather than an application. UDT uses just one UDP connection (as opposed to Tsunami's two connections). Both control and data are transported over the UDT connection. In a similar fashion to Tsunami, the rate at which data is transmitted into the network is controlled by varying the inter-packet delay according to the packet loss rate.

Several tests were done using Tsunami and UDT between a performance node in Tokyo[1] (that was connected at 1 Gbps to the Asia Pacific Advanced Network (APAN) backbone) and a performance server at MIT Haystack Observatory. The bottleneck link capacity was 1 Gbps (the access rate at each end), with a round trip time of $\sim$ 170 ms. At the time these results were taken, TCP performance between these two nodes was quite poor, averaging much less than 100 Mbps. This was due to intermittent packet loss on the end to end path. The results are presented in the following section.

5. Test Results

Figure 1(a) shows the throughput versus offered load for the Tsunami protocol. The dotted line represents the memory to memory throughput from Japan to the USA. Throughput increases linearly with offered load until $\sim$ 800 Mbps at which time the overall system throughput falls off. This could be due to either network or host processing/throughput limitations. The dashed line represents the disc-to-disc performance with a high speed server at the receiver end (dual Intel Xeon 2.4 Ghz with fast SCSI discs and RAID5). In this case the throughput plateaus at $\sim$ 370 Mbps. This is equal to the disc read throughput limitation on the sender side. The solid line represents the disc-to-disc throughput, where the receiving node is a slower machine (1 GHz Pentium III, with SCSI discs, but without RAID). In this case, the overall system throughput peaks at $\sim$ 280 Mbps and then plateaus at a much lower value of $\sim$ 180 Mbps. This is due to losses at the receiver (for these systems the disc read rate at the sender was higher than the disc write rate at the receiver. Thus the receiver was not able to keep up with the traffic stream from the receiver above 280 Mbps, it's internal buffers filled up and then discarded traffic resulting in an overall decrease in system throughput.

Figure 1(b) shows the throughput for a number of separate UDT trials. The solid line represents memory to memory tests from Japan to the US. Average throughput across the 5 trials was 545 Mbps, with a maximum throughput of $\sim$ 650 Mbps and a minimum of $\sim$ 300 Mbps. The dashed line represents disc-to-disc throughput from Japan to the US. The average throughput across these trials was 356 Mbps. In this case, the disc I/O throughput is the system bottleneck.

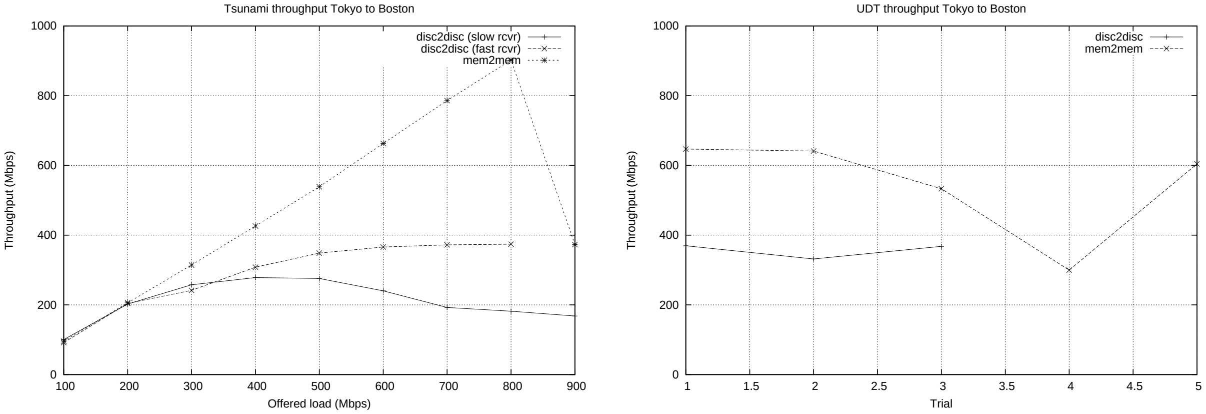
6. Conclusions

Modern research and education networks have backbone bandwidths of up to 10 Gbps. At these speeds, traditional transport options (TCP/UDP) do not perform/behave well. Many enhancements have been proposed to TCP, in particular, to improve its performance in high speed

[1] This server was kindly made available by Communications Research Laboratory

networks, however, they are not yet widely deployed. In the interim, advanced application layer transport protocols offer an alternative, easily deployable and effective solution.

In this paper, we have discussed some of these protocols and looked at the results of some tests using these protocols. Some of these protocols are still experimental, so they should be used/deployed only after extensive reliability and fairness testing. However, they do provide a relatively easy and efficient way in which to improve data transport rates across high speed, long round trip time networks.



(a) Tsunami Throughput

(b) UDT Throughput

Figure 1. Test results

References

[1] Meiss, M., S. Wallace, “Tsunami”, Available from http://www.indiana.edu/∼anml/anmlresearch.html.

[2] He, E., J. Leigh, O. Yu, T. A. DeFanti, “Reliable Blast UDP : Predictable High Performance Bulk Data Transfer”, IEEE Cluster Computing 2002, Chicago, Illinois, Sept, 2002.

[3] “DataSpace”, http://sourceforge.net/projects/dataspace/

[4] Jin, C., D. Wei, S. H. Low, G. Buhrmaster, J. Bunn, D. H. Choe, R. L. A. Cottrell, J. C. Doyle, W. Feng, O. Martin, H. Newman, F. Paganini, S. Ravot, S. Singh, “FAST TCP: From Theory to Experiments”. submitted for publication, April 1, 2003

[5] Mathis, M., J. Semke, J. Mahdavi, T. Ott, “The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm”, Computer Communication Review, volume 27, number3, July 1997.

[6] Floyd, S., “HighSpeed TCP for Large Congestion Windows”, RFC3649, Experimental, December 2003.

[7] Kelly, T., “Scalable TCP: Improving Performance in Highspeed Wide Area Networks”, Submitted for publication, December 2002.

[8] Whitney, A., et. al, “The Gbps e-VLBI Demonstration Project”, February 2003. Available at ftp://web.haystack.edu/pub/e−vlbi/demo_report.pdf