FTP 100: An Ultra-High Speed Data Transfer Service over Next Generation 100 Gigabit Per Second Network

Award Number: 53662 - US Department of Energy

Award Title: ARRA: TAS::89 0227:: TAS Recovery Act

100G FTP: An Ultra-High Speed Data Transfer Service Over Next Generation 100 Gigabit Per Second Network

1. Overview

Data-intensive applications, including high energy and nuclear physics, astrophysics, climate modeling, nano-scale materials science, genomics, and financing, are expected to generate exabytes of data over the coming years, which must be transferred, visualized, and analyzed by geographically distributed teams of users. High-performance network capabilities must be available to these users at the application level in a transparent, virtualized manner. Moreover, the application users must have the capability to move large datasets from local and remote locations across network environments to their home institutions.

To support these data-intensive distributed applications, a significant amount of work has been done to accelerate data transfer over high-speed networks. There are two main approaches. The first is protocol offload and hardware acceleration. They are among the most desired techniques to achieve high data transfer rate with marginal host resource consumption. The TCP/IP Offload Engine (TOE) is one of the early examples of protocol offload to meet above requirements. The idea of TOE is to use a dedicated hardware module on a network adapter card to execute the TCP/IP internal operations, such as segmenting, framing, packet reassembling, timing, and congestion and flow control. Research and implementation works have shown that TOE is a cost-effective technique to free the host processors from onerous TCP/IP protocol processing, and therefore to enhance the concurrency between communication and computation. Thereafter, Remote Direct Memory Access (RDMA) was proposed as another hardware-based Protocol Offload Engine (POE) realization to enable high-speed and low-latency data transfer with much less CPU resource involvement, and its use has recently become popular when converged Ethernet and data center bridge technologies were proposed and implemented. RDMA has the capability to move bulk data from the source host memory directly to the remote host memory with kernel-bypass and zero-copy operations. Three different RDMA implementations, InfiniBand (IB), Internet Wide Area Protocol (iWARP), and RDMA over Converged Ethernet (RoCE), are available today to offload RDMA protocol stack to different network architectures. Each of them has different trade-offs between system performance, cost, compatibility, and implementation complexity.

The second approach is software optimization and kernel pass techniques. This approach is based on the observation that data-intensive applications impose a formidable requirement on the CPU usage of hosts. In particular, frequent data copy within the host memory space is

expensive, but often redundant and inefficient. For example, in a typical file transfer application, file data is first read from disk drivers to the kernel memory, and then copied to the user space. From then on, the data is sent via the socket interface, and thus it is copied from the user space back to the kernel space, i.e., the socket buffer before actual delivery to network drivers. Various optimization techniques have been proposed in modern computer systems to minimize data copy. In fact, the same idea has been adopted by Web server performance optimization. In the current Linux systems, on the other hand, we have seen more standard kernel primitives to facilitate such performance optimization, for example, via the sendfile and splice primitives.

The main goal of our project is, via the aforementioned approaches, to design and evaluate high-performance data transfer software to support various data-intensive applications. First, we have designed a middleware software that provides access to RDMA functionalities. This middleware integrates network access, memory management and multitasking in its core design. We address a number of issues related to its efficient implementation, for instance, explicit buffer management and memory registration, and parallelization of RDMA operations, which are vital to delivering the benefit of RDMA to the applications. Built on top of this middleware, an implementation and experimental evaluation of the RDMA-based FTP software, RFTP, is described and evaluated. This application has been implemented by our team to exploit the full capabilities of advanced RDMA mechanisms for ultra-high speed bulk data transfer applications on Energy Sciences Network (ESnet). Second, we designed our data transfer software to optimize TCP/IP based data transfer performance such that RFTP can be fully compatible with today's Internet. Our kernel optimization techniques with Linux system calls sendfile and splice, can reduce data copy cost.

In this report, we summarize the technical challenges of our project, the primary software design methods, the major project milestones achieved, as well as the testbed evaluation work and demonstrations. The software produced by this project was made available to DOE's data intensive science applications.

2. Technical Challenges and System Design

2.1 Challenges

There are two main challenges while designing our software system.

- The software must achieve line-speed data transfer. To do so, we should avoid many bottleneck and inefficient software design. Notably, the network and host capacities are so high, that we have to maximally parallelize operations. The parallelism requires we should have multiple parallel network connections, and we should also utilize the multi-core architecture of end-hosts. In addition, we need to support pipelining various operations, including disk operations, memory access, CPU processing, and network transmission. The design of our software thus should achieve the parallelism.
- Another challenge is to integrate both hardware acceleration and software optimization techniques to support data transfer applications. For example, it is important to manage the heterogeneity of underlying RDMA architectures as described above for user applications. File Transfer Protocol (FTP) is one of the most widely-used services to transfer bulk data. Existing data transfer applications built on top of the native TCP/IP may not be able to fully utilize the available bare-metal bandwidth of the next-generation high-speed networks, because of the considerable

load on host CPU caused by kernel-based TCP/IP realization. Various RDMA implementations, as introduced above, offer opportunities to enhance the performance of data transfer service. However, despite of the emergence of industry standards such as OpenFabrics Enterprise Distribution (OFED), it could be a distraction if the user applications have to directly manage underlying RDMA devices.

2.2 Design

We designed our RFTP, the main software system of FTP 100 project, based on a general-purpose middleware layer. The middleware layer provides the necessary data communication and access functions, and manages the system resources such as buffers and network interfaces. A more detailed description of this middleware can also be found in the previous quarterly reports we submitted during the phases of the project.

We then build RFTP by extending the standard FTP protocol and software package. It has a layered architecture, as shown in Figure 1. We allow two modes: one transfer mode is solely based on the conventional TCP/IP stack, and the "R" mode, i.e., our extension, is based on our RDMA middleware layer. The middleware layer itself consists of several modules, such as buffer management and connection management. This RDMA middleware is in turn supported by low-layer protocols. It uses the common RDMA verbs on top of various hardware resource, Infiniband, RoCE and iWARP. These verbs are provided by Infiniband Verbs library (libibverbs) and RDMA ommunication manager (librdmacm). Thus, the middleware layer hides all the specifics related to hardware, while the applications, for example, FTP applications, do not have to be aware of those hardware specifics. Therefore, the middleware provides the desirable transparency. Note: Figure 1 also shows we have designed an additional disk I/O module that is based on direct I/O operations. This module encapsulates an I/O scheduler which is responsible for scheduling and supporting disk readers/writers.

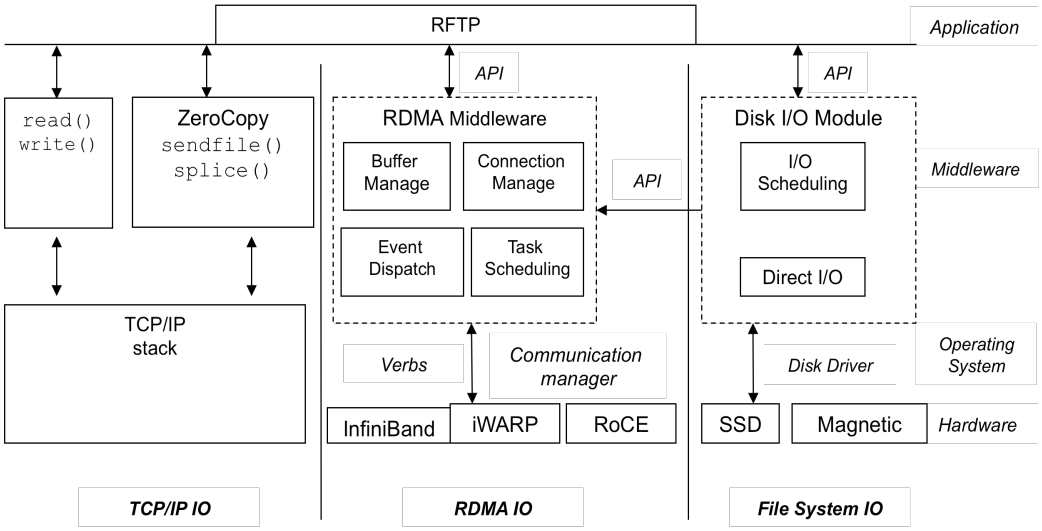


Figure 1. The RFTP application architecture

On top of this middleware, we develop the RFTP software. We extended the standard FTP's commands to allow data transfer operations over the RDMA network. The table below shows a list of extended commands:

Command	Function and Procedure
RADR	RDMA address information exchange
RSTR	1) Establish RDMA data transfer channel; 2) Send data from client to server using RDMA operations, and store the file at server side; 3) Tear down RDMA data transfer channel.
RRTR	1) Establish RDMA data transfer channel; 2) Get data from server to client using RDMA operations, and store the file at client side; 3) Tear down the RDMA data transfer channel

Figure 2. RFTP's extensions to FTP commands

Our design of RFTP has the following advantages:

- Performance advantages: one-side RDMA semantics, minimum CPU consumption, efficient disk I/O operations, and minimum data-copy.
- Flexibility advantages: RFTP allows a convenient mode switch between two implementation: (1) OFED, RDMA operation, and (2) TCP/IP, with kernel zero-copy optimization (sendfile/splice)
- Development/deployment advantages: Light-weighted, portability and compatibility (with OFED and minimum upgrade of Linux kernel).

3. Major Project Milestones

Our first major milestone is the development of our initial prototype FTP 100 data transfer software, or RFTP Version 1.0. This milestone was achieved by early 2011. Our development work involved careful software coding, debugging, and intensive laboratory test at both Brookhaven National Lab and Stony Brook University. During our development and test, we mainly utilized our local area networks (LAN) and hosts.

Our second major milestone is the deployment and performance test of our software on several types of testbed environment, and this milestone was achieved in the second half of 2011. We released the software to our collaborators of the project, for example, the climate data team at LBNL too. Our testbed environments included the follows.

- We used one high-bandwidth medium-latency MAN in the New York metropolitan area, and one high-bandwidth high-latency WAN between University of Michigan and Brookhaven National Lab, which traverses through the ESnet UltraLight network.
- We deployed our RFTP software and continued to evaluate it in the ANI testbed, specifically at the three sites, NERSC, ANL, and ORNL. Currently, we utilize 10Gbps/40Gbps RoCE links between NERSC and ANL, and we have completed some tests and continue to fine-tune our software.

Our third major milestone, as the direct results of our efforts during the previous phase, is our

live demo system displayed at the SuperComputing'2011-2013 conferences. In this demo, we connected multiple DOE labs, including NERSC, ANL, and ORNL, and conducted intensive data transfer among the sites in a WAN configuration. The demo was well received at the conference.

Our fourth major milestone is the development of a true end-to-end data transfer system. In the previous milestones, utilized memory-based data transfer to achieve ultra-high-speed data transfer, but it is more challenging for any software to overcome the other bottlenecks (host processing and storage systems). For this purpose, we moved forward our project, and developed a frontend/backend system using the iSCSI/iSER protocol for storage area networks. The design of the entire system (with a RDMA network connecting two frontend/backend systems) is shown in Figure 3.

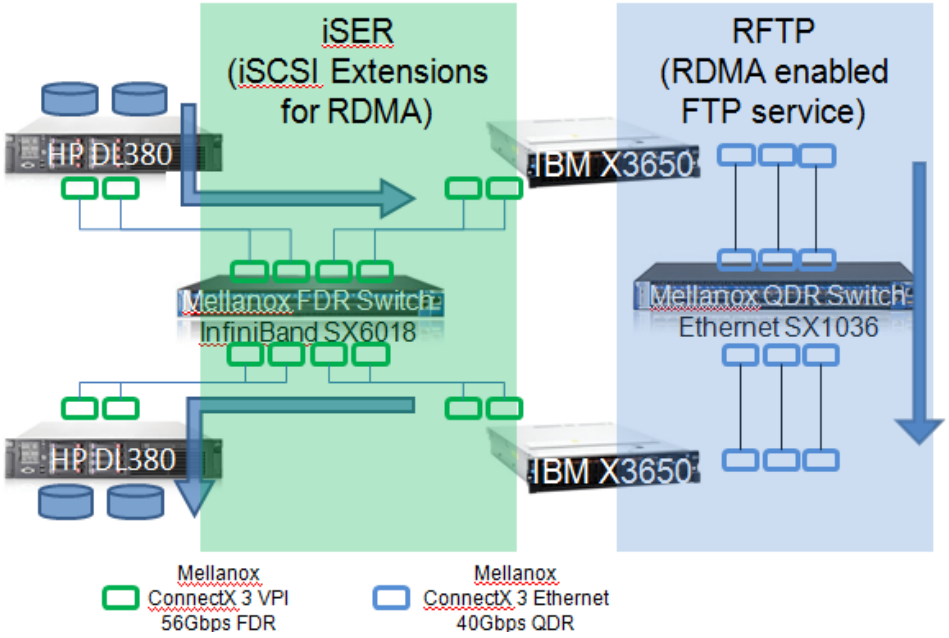


Figure 3. Full end-to-end data transfer system using iSCSI/iSER

In November 2012/2013, we achieved ***our fifth major milestone*** by developing another live demo system at the SuperComputing'2012 conference. We used both LAN and WAN configurations in our live demo. For the LAN demo, we utilized our development testbed in Figure 3. For the WAN demo, we utilized a CalTech network with RoCE connection (to be described later in this report).

4. Software Evaluation and Demos

4.1 SuperComputing'2011 Live Demo and Results

In this demo, we paired up 28 hosts, 14 each from NERSC and ANL, to run parallel RFTP data streams to conduct disk-to-memory data transfer. The maximum disk bandwidth offered via IB links is 80Gbps. Meanwhile, we paired up additional 18 hosts, 9 each from NERSC and ORNL, to run parallel RFTP data streams to conduct memory-to-memory data transfer. All these data streams shared the aggregate 100Gbps link bandwidth.

Figure 4 shows the bandwidth utilization at the time of our demo, 10:00am-10:30am on November 17, 2011, captured by the Ganglia monitoring system available at NERSC. This tool shows the double-counted traffic volume. RFTP achieves almost 80Gbps *reliable data* throughput, when the physical link bandwidth limit is 100Gbps and disk bandwidth is 80Gbps. This performance shows that our RFTP can saturate the network capacity or the internal hardware bottleneck.

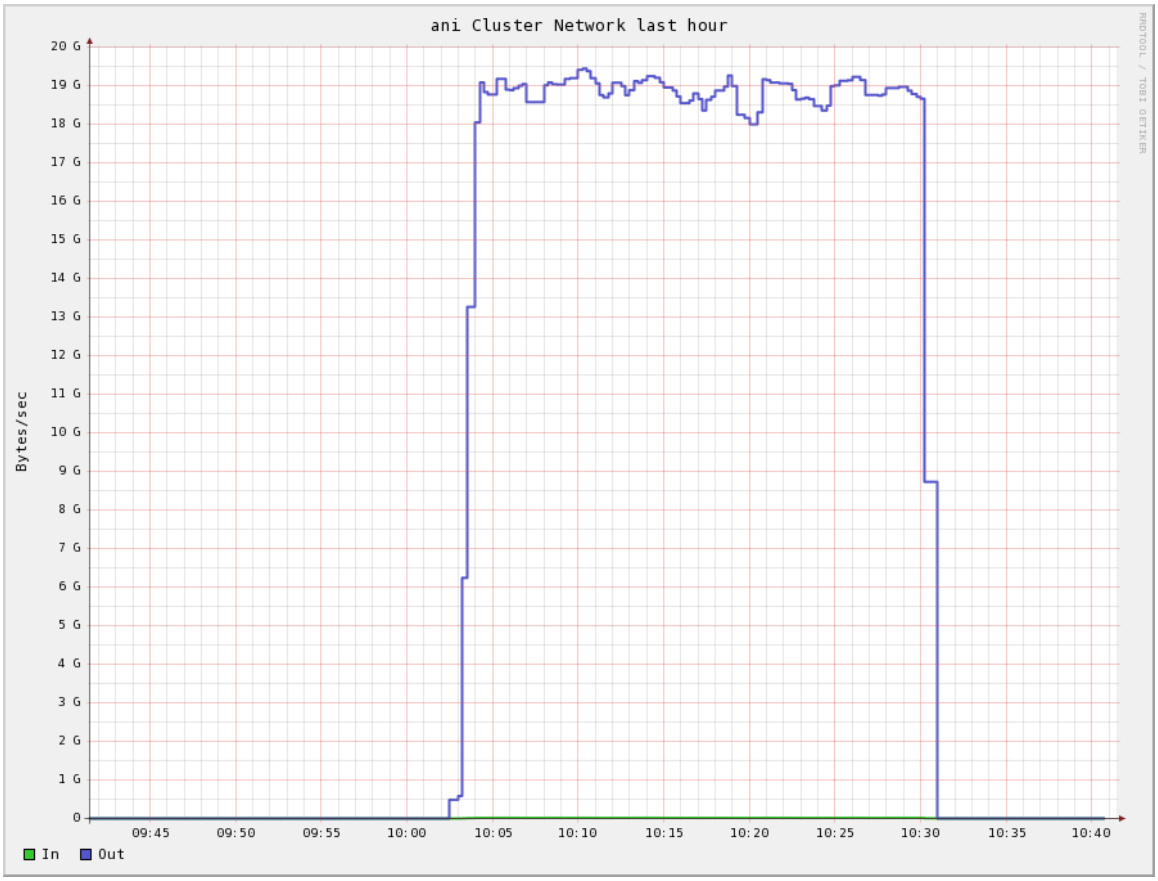


Figure 4. The bandwidth performance of RFTP at the SC'11 demo

4.2 SuperComputing'2012 Live Demo and Results

By November 2012, we have developed a full end-to-end data transfer system, and have thoroughly tested in our local testbed at Stony Brook University. To prepare for a live demo at the SupuerComputing'2012 conference, we planned two configurations: one using our local testbed, and the other using WAN RoCE connections (with helps from our collaborators from ESnet to set up the connection). However, before the conference, we found that the RoCE links had very high loss rate, and thus, the data transfer is not stable. Fortunately, at the conference, our other collaborators at CalTech had high-quality RoCE links ready, and we could utilize their networks in our demo.

The Caltech RoCE links provide 80Gbps bandwidth. Two client hosts, each with 40Gbps RoCE, loopback to a server with 2x 40Gbps RoCE interfaces. The RTT of the connections is close to 50ms. And, all hosts are equipped with Fusion IO SSDs, and provide sufficient storage capacity

and read/write bandwidth. In the demo, we showed that our RFTP software achieve the full 80Gbps end-to-end bandwidth in this configuration, as shown in Figure 5 and 6.

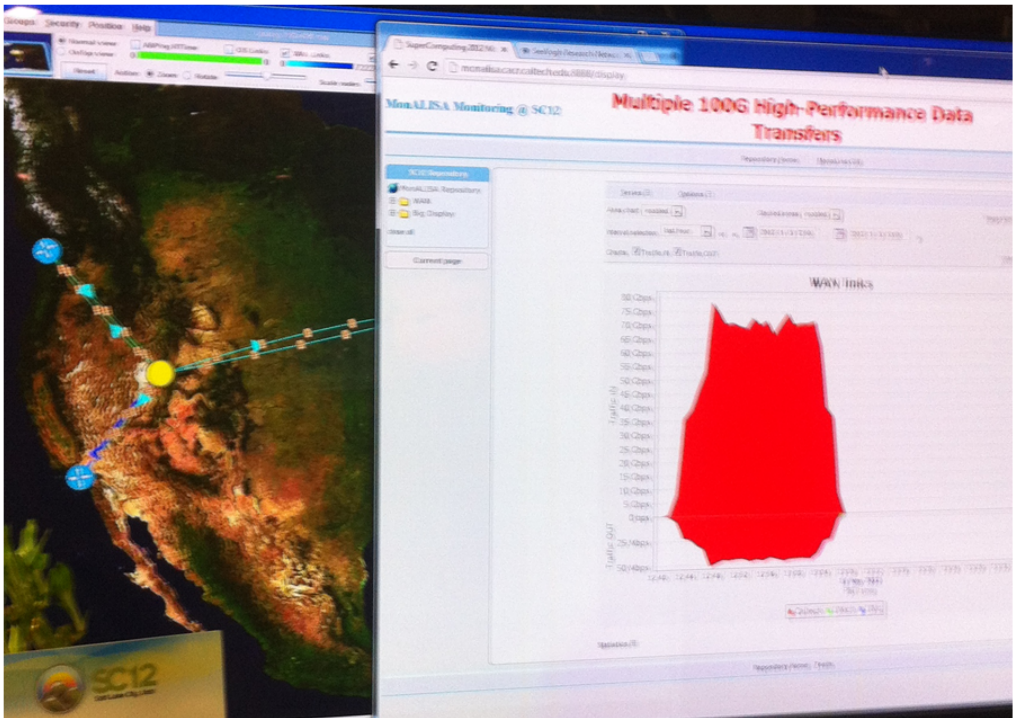


Figure 5. The demo network and our software performance at SC'12 demo

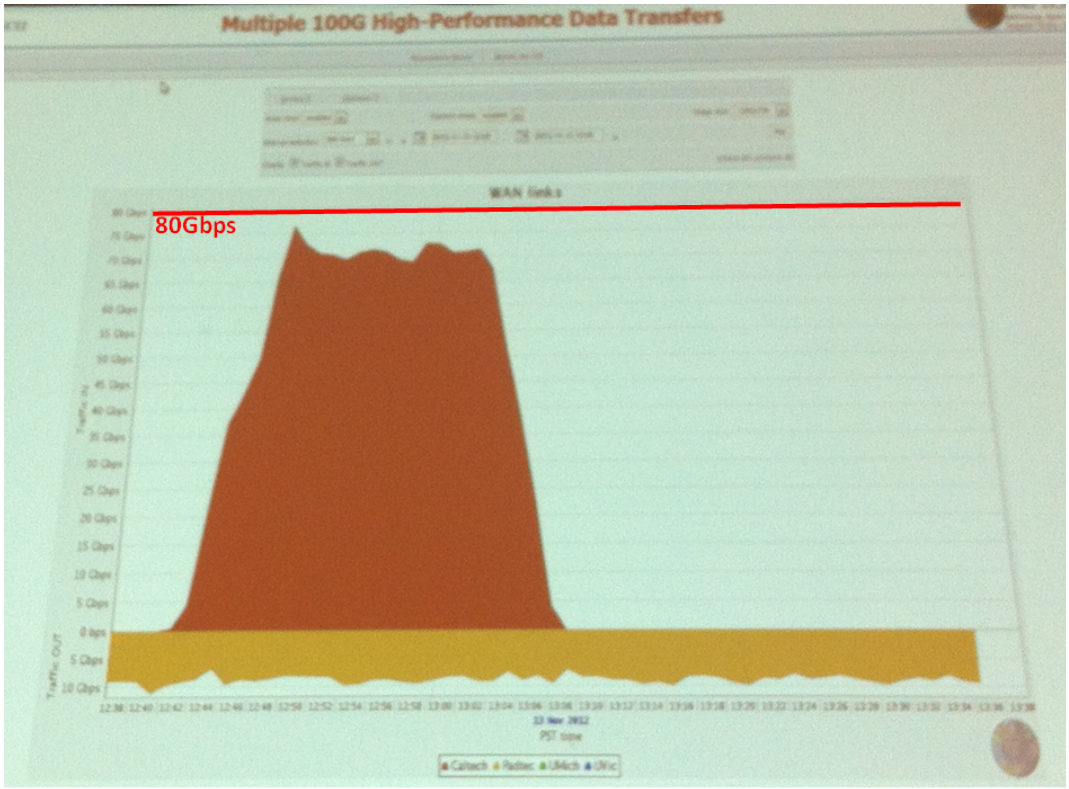
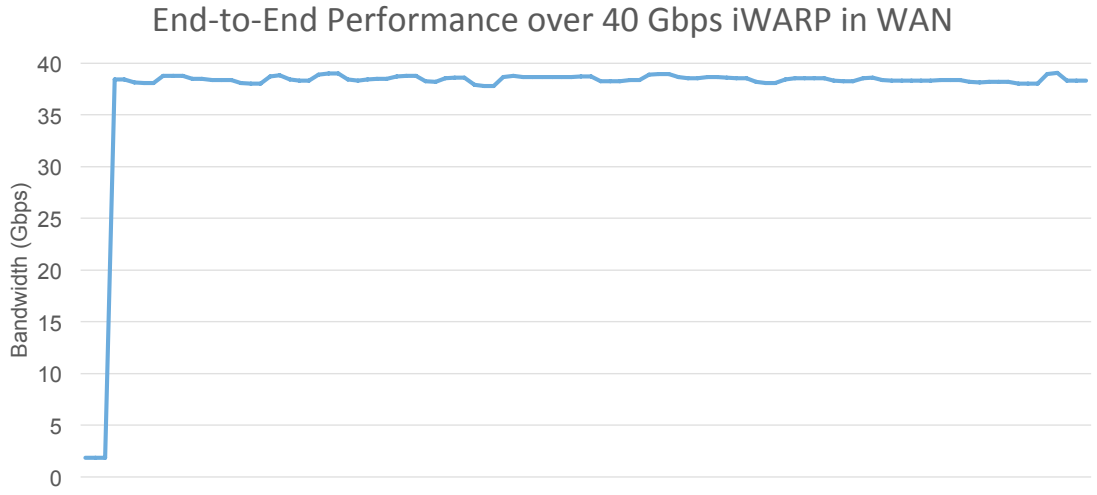


Figure 6. Our software obtained the full 80Gbps bandwidth at the SC'12 demo

4.3 SuperComputing'2013 Live Demo and Results

Our demo focuses on using 40Gbps iWARP network card to test and evaluate our RFTP software. The difference between iWARP and RoCE is that iWARP is fully compatible with TCP/IP, and can seamlessly use Internet to move data across WAN. Both RoCE and InfiniBand are not compatible with Internet and require a dedicated network and associated supporting network infrastructure. Since Chelsio releases their T5 40GE network card that can support iWARP. Our RFTP can naturally scale to Internet and move traffic. However Internet available bandwidth is undetermined and best-effort. To demonstrate its full potential in high throughput, a dedicated network link was reserved between CalTech and the Supercomputing 2013's show floor. Our RFTP can move traffic through one pair of iWARP network adaptors with a full bandwidth of 40Gbps. The following picture was taken from the sending and receiving hosts. With the new type of RDMA technology, RFTP remains to be scale to the full network bandwidth.



5. Research Publications

SELECTED JOURNAL PUBLICATIONS

1. Y. Ren, T. Li, D. Yu, S. Jin, T. Robertazzi, Design, Implementation, and Evaluation of a NUMA-Aware Cache for iSCSI Storage Servers, Accepted to IEEE Transactions on Parallel and Distributed Systems (TPDS), 2014.
2. Y. Ren, T. Li, D. Yu, S. Jin, T.G. Robertazzi, Design and Testbed Evaluation of RDMA based Middleware for High-performance Data Transfer Applications, Journal of Systems and Software, 2013.

SELECTED CONFERENCE AND WORKSHOP PUBLICATIONS

3. Y. Ren, T. Li, D. Yu, S. Jin, T.G. Robertazzi, Design and Performance Evaluation of NUMA-aware RDMA-based End-to-end Data Transfer Systems, accepted by ACM/IEEE SuperComputing 2013, November/2013, Denver, CO. (acceptance rate: 20%).
4. T. Li, Y. Ren, D. Yu, S. Jin, T.G. Robertazzi, "Characterization of Input/Output Bandwidth Performance Models in NUMA Architecture for Data Intensive Applications", accepted by International Conference on Parallel Processing (ICPP 2013), Lyon, France (acceptance rate: 30%).

5. Y. Ren, T. Li, D. Yu, S. Jin, T.G. Robertazzi, B. Tierney, E. Pouyoul: Protocols for wide-area data-intensive applications: design and performance issues. ACM/IEEE SuperComputing 2012 (acceptance rate: 21%).
6. Yufei Ren, Tan Li, Dantong Yu, Shudong Jin, and Thomas Robertazzi, "Middleware Support for RDMA-based Data Transfer in Cloud Computing", In *Proceedings of the High-Performance Grid and Cloud Computing Workshop (held in conjunction with IPDPS 2012),* Shanghai, China, May 2012.
7. Sushant Sharm, Dimitrios Katramatos, and Dantong Yu, End-to-End Network QoS via Scheduling of Flexible Resource Reservation Requests, SuperComputing 2011, Seattle, WA (acceptance rate: 21%).