A Study of Hardware Assisted IP over InfiniBand and its Impact on Enterprise Data Center Performance

Ryan E. Grant[1], Pavan Balaji[2], Ahmad Afsahi[1]

[1]Department of Electrical and Computer Engineering
Queen's University
Kingston ON, Canada
ryan.grant@queensu.ca, ahmad.afsahi@queensu.ca

[2]Mathematics and Computer Science
Argonne National Laboratory
Argonne, IL, USA
balaji@mcs.anl.gov

Abstract— **High-performance sockets implementations such as the Sockets Direct Protocol (SDP) have traditionally showed major performance advantages compared to the TCP/IP stack over InfiniBand (IPoIB). These stacks bypass the kernel-based TCP/IP and take advantage of network hardware features, providing enhanced performance. SDP has excellent performance but limited utility as only applications relying on the TCP/IP sockets API can use it and other IP stack uses (IPSec, UDP, SCTP) or TCP layer modifications (iSCSI) cannot benefit from it. Recently, newer generations of InfiniBand adapters, such as ConnectX from Mellanox, have provided hardware support for the IP stack itself, such as Large Send Offload and Large Receive Offload. As such high performance socket networks are likely to be deployed or converged with existing Ethernet networking solutions, the performance of such technologies is important to assess. In this paper we take a first look at the performance advantages provided by these offload techniques and compare them to SDP. Our micro-benchmarks and enterprise data-center experiments show that hardware assisted IPoIB can provide competitive performance with SDP and even outperform it in some cases.**

Keywords- IPoIB; InfiniBand; Offloading; Data Center; Convergence; SDP;

I. INTRODUCTION

Current enterprise data centers typically use Ethernet networks. IP-based protocol stacks such as TCP/IP have been widely used by many applications in enterprise data centers. Such stacks have traditionally been known to incur high overheads. Consequently, high-speed networks, such as InfiniBand (IB) [16], have relied on alternative protocol stacks in order to allow applications to take advantage of the capabilities offered by the network. IB networks are a feature rich high-speed, low-latency alternative, but the costs associated with adapting current data center applications to use native IB verbs is a barrier to the adoption of IB networks in data centers.

Sockets Direct Protocol (SDP) [4] was developed as an intermediary solution to this problem. SDP was designed to bypass the TCP/IP stack and allow applications using the TCP/IP sockets API to directly utilize the features provided by InfiniBand-like networks. While promising, SDP has several limitations with respect to the breadth of its usefulness. Specifically, since SDP directly bypasses the TCP/IP sockets API, only applications relying on this API can benefit from it. Other applications that rely on additions to the IP-layer (such as IPSec [19]), those which rely on enhancements or patches to the TCP stack (such as iSCSI [22]), or those which rely on other IP-based protocols (such as UDP/IP and SCTP/IP [26]) cannot benefit from SDP.

Until recently the performance of TCP/IP on IB networks has been poor due to a lack of hardware offload support. While the methods of hardware offload for the IP stack have existed for other networks, including Ethernet networks, for many years, the introduction of such capabilities for IB networks removes some of the barriers preventing the adoption of IB in enterprise data centers. As such, it provides a compromise between offering the feature rich high-speed IB network and the required legacy support for traditional data center applications written for IP-based protocols. This is a critical step towards the goal of converging high-speed networking fabrics and existing enterprise data center networking solutions in a single system, where cost effective Ethernet technologies can be supplemented with high-speed networks in order to increase overall data center performance [7].

The desire for convergence between InfiniBand and Ethernet networks has been growing recently. As IPoIB is the interface by which socket based programs operate over IB, it is important to examine its performance in current data center environments to understand the benefits that might arise from future hybrid data center networking environments. Convergence efforts such as RDMAoE/IBoE [24], RDMAoCEE [5] and VPI [7] are already underway to combine existing high performance networks with Ethernet. A new set of standards referred to as Converged Enhanced Ethernet (CEE) [11-15,17] has opened up issues revolving around providing advanced features over Ethernet networks. Some industry vendors and researchers [5] are also proposing to include RDMA functionality over CEE [5]. Technologies such as InfiniBand over Ethernet (IBoE) [24] provide for InfiniBand packets to be encapsulated into an Ethernet frame, while still utilizing the IB network and transport layers. This allows for the features of InfiniBand to be provided over an Ethernet network, including RDMA. The development of Virtual Protocol Interconnect (VPI) [7] allows for simultaneous use of a single multi-port adapter that supports Ethernet or InfiniBand running independently on each port. This enables the use of multi-mode hot-swappable network solutions providing both native IB and Ethernet. This convergence makes the investigation of the performance of IPoIB critical as it has potential to be widely used in future hybrid data center environments.

In this paper, we take the first look at the performance advantages of such hardware assistance provided by InfiniBand to the IP stack and compare its performance to that of SDP. We perform such evaluation in the context of both micro-benchmarks as well as modern enterprise data-center environments. To the best of our knowledge this work is the first of its kind studying segmentation offload techniques for IPoIB. We find that IPoIB with offloading can provide performance equal or competitive to that of SDP in many cases, and outperform SDP in some other cases as well. This study puts TCP/IP over InfiniBand in a new light with respect to its utility and the benefits it can provide.

II. Background

InfiniBand [16] is a leading high-speed interconnect with very low latency and high throughput. Native InfiniBand verbs form the lowest software layer for the IB network, and allow direct user-level access to IB host channel adaptor (HCA) resources while bypassing the operating system. At the IB verbs layer, a queue pair model is used for communication supporting both channel-based communication semantics (send/receive) and memory-based communication semantics (Remote Direct Memory Access - RDMA). InfiniBand requires user buffers to be registered prior to being used for communication. The InfiniBand standard [16] defines four transport methods, Reliable Connection, Unreliable Connection, Unreliable Datagram and Reliable Datagram. There is also a new eXtended Reliable Connection transport available in ConnectX HCAs from Mellanox [20].

IPoIB can run over multiple InfiniBand protocols, either over a reliable connection (RC) or unreliable datagram (UD), while SDP runs over RC. A reliable connection is a negotiated connection that ensures low-level reliability of the physical network. The reliable connection mode uses an upper layer maximum transmission unit of 64 KiB, while the underlying IB network fabric uses a 2 KiB maximum transmission unit. The unreliable datagram mode uses a maximum 2 KiB packet size at both the upper layer and lower layer, and does not guarantee reliable transmission of the data, but requires less hardware support than RC.

A. Sockets Direct Protocol

Sockets Direct Protocol replaces the TCP/IP stack interface for existing TCP/IP based applications with an alternative stack. This stack handles basic flow control and provides access to advanced hardware features, including RDMA, hardware flow control [2] and a hardware offloaded transport and network stack. Due to the RDMA features used by SDP, it can only operate over a reliable network, and so it utilizes the InfiniBand reliable connection mode for transmissions. In terms of offloading, SDP offloads the majority of its workload to the NIC, enabling higher throughput and lower latencies. An overview of SDP architecture can be seen in Figure 1.

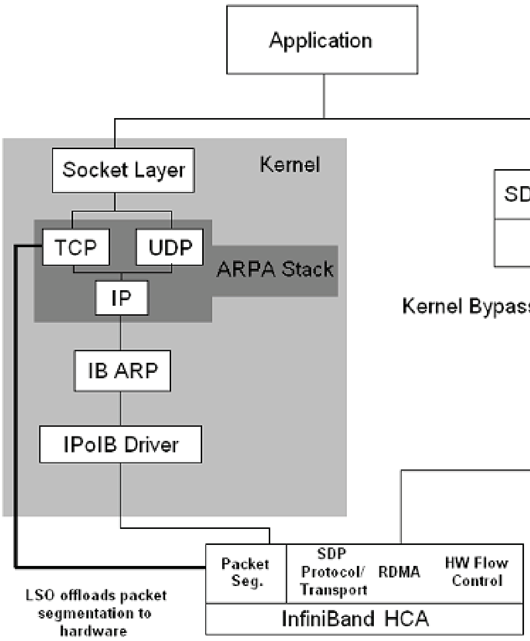


Figure 1. IPoIB and SDP Architecture

SDP can operate in two different modes, buffered-copy (BC) and zero-copy (ZC) [1, 4]. Buffered-copy uses a local buffer to hold all of the SDP incoming data, which is then copied to the application memory space when the data has been satisfactorily received. Zero-copy works by transferring data directly from application buffer to application buffer between the systems. Zero-copy works well for large message sizes, where the overhead required in setting up the application buffer transfer can be offset by a large data transfer, and the direct copy increases the available bandwidth by reducing memory accesses. Buffered-copy is more useful for small message transfers where the overhead required by zero-copy to setup the transfer has more impact than the overhead of the extra copy between the SDP memory buffer and the application buffer. The current version of OFED [21] provides an adjustable threshold switching point to allow for a transition between the two modes of operation, its default value is 64 KiB.

B. IPoIB and TCP/IP

Traditional TCP/IP implementations utilize the host for all connection management, reliability and transport functions. Through the use of the TCP/IP stack, data is split into packet-sized units, and flow control and reliability are provided through it, as shown in Figure 1. The IP stack then adds the information required for the correct routing of the packet to its destination. Therefore, the network adapter is assumed to be an unreliable datagram delivery device, with no additional capabilities.

IPoIB implements two types of offloading, a connection management offload, via IPoIB over RC and for unreliable communication, segmentation offloading via IPoIB over UD, utilizing Lareg Receive Offload (LRO) and/or Large Send Offload (LSO). For the connection management offloading, the InfiniBand network supports a native low-level reliable delivery. Therefore, TCP does not need to handle any reliability-based issues, and always receives the expected packets via the guaranteed delivery of the hardware layer. IPoIB does not currently take advantage of the available RDMA features, unlike SDP which operates exclusively over RDMA.

Segmentation offloading is available as a stateless offload when the InfiniBand adapter is using a datagram mode, as shown in Figure 1. Two such offloads exist: LRO and LSO. LSO provides a virtual MTU to the system such that large packet sizes are delivered to the adapter via the segmentation of data in the TCP/IP stack, and the data is segmented into packets at the hardware level. The primary advantage of this approach is that from the host system's perspective, the network is now capable of large MTU sizes (Jumbo-frame like behavior), which results in fewer hardware interrupts and consequently better performance. At the same time, since the packets on the network are all standard MTU-sized, there is no additional switch support required for this capability and is portable in all network environments. Most importantly, this approach benefits the IP-stack directly and hence can benefit all upper-level stacks and applications and not just those relying on TCP/IP sockets, which is the case with SDP.

LRO provides for the aggregation of incoming packets for delivery to the system. This provides the benefit of fewer (larger) packets being delivered to the stack, enhancing performance, although it can introduce some minimal additional latency to some packets as they wait to be aggregated. It should be noted that no fundamental barriers prevent IPoIB from using more advanced hardware offloading features, but IPoIB does not use them due to its current maturity.

III. Related Work

High performance sockets for high-speed networking (VIA) architectures were first proposed by Shah et al. [23]. The performance of SDP and IPoIB in a data center context for reliable connection based SDP and IPoIB has been evaluated by Balaji et al. [4], where it was observed that SDP and IPoIB can have a beneficial effect on data center performance. The performance of combined Ethernet/IPoIB/SDP traffic via VPI for ConnectX adapters has been evaluated by Grant et al. [7], in which it was shown that multi-port adapters running in IB and Ethernet mode simultaneously can be effective for increasing the performance of back-end data center architectures. All of this previous work differs from this paper in that it never considered unreliable datagrams for either IPoIB or SDP and never examined offload capabilities of InfiniBand network adapters.

Goldenberg et al. [6] first demonstrated the performance improvements that can be realized by adding zero-copy functionality to SDP. They found that the throughput of large messages could be substantially improved through the use of zero-copy. Balaji et al. [1] have developed an asynchronous zero-copy mechanism that improved upon Goldenberg et al.`s approach, through increased throughput.

The performance of socket-based protocols when using quality of service provisioning was analyzed in [8] by Gratn et al.. It was observed to have a positive effect on system performance when utilizing QoS provisioning. A greater impact was seen in the QoS provisioning of SDP over that of IPoIB.

Zhang et al. [30] have investigated the performance of InfiniBand socket-based protocols in relation to the Java communication stack. They found that the performance of the existing Java communication stack is poor and requires changes in order to provide performance inline with the capabilities of the network fabric. Other work on the Java communication stack has been done in [10], where RDMA operation functionality was added for Java through a specialized library. The work to date with Java communication stacks differs from this work in that it did not consider datagrams or offloading.

Work on offloading techniques has primarily centered on Ethernet TCP offloading, and recent 10Gb-E adapters from Mellanox have provided for TCP/UDP/IP stateless offload. A comparison between offload enabled Ethernet adapters and high-speed Myrinet and IB adapters has also been performed [3]. However, no investigation has yet been performed regarding the segmentation offloading capabilities of InfiniBand high-speed sockets based traffic.

IV. Experimental Framework

The experiments were conducted on four Dell PowerEdge R805 SMP servers. The PowerEdge R805 has two quad-core 2.0GHz AMD Opteron processors with 12 KiB shared execution trace cache, and 16 KiB L1 shared data cache on each core. A 512 KiB L2 cache per core and a shared 2 MiB L3 cache are available on each chip. There are 8 GiB of DDR-2 SDRAM on an 1800 MHz Memory Controller. Each SMP server is equipped with a ConnectX 4X DDR InfiniBand HCA from Mellanox Technologies connected through a 4X DDR InfiniBand Flextronics switch.

The operating system used is a Fedora Core 5 Linux kernel 2.6.27 implementation. The Open Fabrics distribution OFED-1.4 [21] was used as the software stack for IB. All software was compiled specifically for the machine using gcc 4.1.1.

The unidirectional bandwidth and one-way latencies are shown in Section 5. Real data center results are shown in Section 6, and they are bi-directional. Netperf [18] was used for bandwidth testing. All tests were required to meet a 99.5% confidence factor before a test was considered valid, and iperf [27] was used to confirm the results. The latency/bandwidth testing for InfiniBand verbs was performed using qperf from the OFED-1.4 software package. All bandwidth results are shown in Millions of bits per seconds (Mbps), while all latencies are expressed in microseconds (µs). All message sizes are expressed in Kibibits or Mebibits, as applicable.

Table 1 details the abbreviations used to describe the different IPoIB offloading operating modes. Results are shown according to the mode in which the IPoIB communication is taking place (RC or UD). SDP is operating over RC. For the SDP micro-benchmark results with zero-copy and buffered-copy, the switching threshold was set such that the zero-copy or buffered-copy methods were used exclusively over the entire message size range.

TABLE I. ABBREVIATIONS USED

IPoIB	
LRO	Large Receive Offload Enabled
noLRO	LRO Disabled
LSO	Large Send Offload Enabled
noLSO	LSO Disabled

In order to control the use of LRO for IPoIB datagrams, the provided module options were used to enable and disable its use. As no such control exists for LSO for our kernel, the OFED source code was modified to disable the use of LSO when conducting the corresponding LSO testing.

Table 2 summarizes the virtual MTUs used for the protocols studied in this paper. Note that offloading refers to Large Send Offload and SDP-ZC does not require segmentation.

TABLE II. VIRTUAL MTU SIZES

Protocol	VMTU Size
IPoIB-RC	64 KiB
IPoIB-UD no Offloading	2 KiB
IPoIB-UD Offloading	64 KiB
SDP-BC	64 KiB
SDP-ZC	No Segmentation

V. MICRO-BENCHMARK RESULTS

The performance benefits of IPoIB offloading techniques are presented first in Section 5.1. Section 5.2 compares the performance of IPoIB offloading with IPoIB-RC, SDP buffered-copy and zero-copy as well as IB Verbs.

A. *IPoIB Offload Comparison*

This section compares the performance of IPoIB operating over unreliable datagrams, using the offload capabilities available on current ConnectX adapters as well as IPoIB operating over a reliable connection.

Figure 2 examines the latencies of IPoIB with four possible offloading configurations, and over a reliable connection. Enabling LRO provides for lower latencies at small message sizes, where the multiple outstanding requests that are issued during the latency test can be aggregated into fewer overall interrupts, reducing the latency seen by the requests as a whole, even though individual packet latencies could be increased due to aggregation.

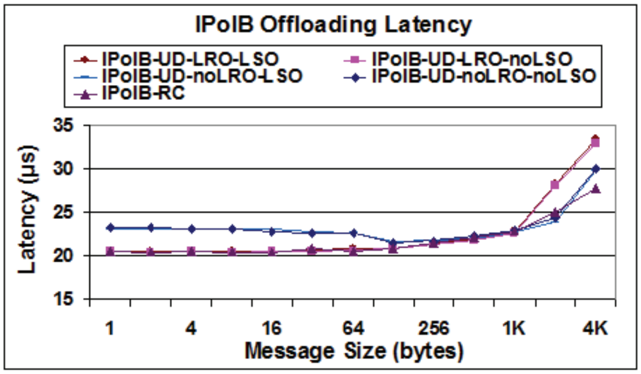


Figure 2. IPoIB offload one-way latency

At the 2 KiB message size, the latency of the LRO enabled offloads increases to above that of the LRO disabled configurations. For message sizes of 64 KiB or larger, the LRO configurations offer lower latencies, providing a 13.2% latency reduction between the best LRO enabled and LRO disabled configurations, while averaging 11.1% over all message sizes.

The maximum latency reduction occurs at 64 KiB, showing a 26.2% reduction in latency. Disabling LSO has an advantageous effect on large message latencies, providing 3.9% better latency for message sizes larger than 64 KiB when it is disabled. The best IPoIB-UD configuration, using both LSO and LRO shows a 3.2% higher latency than IPoIB-RC for message sizes up to 4KiB, but for messages less than 1KiB; there is no appreciable difference in performance between IPoIB-RC and IPoIB-LRO-LSO.

Figure 3 illustrates the bandwidth of single stream IPoIB-UD configurations with varying offload capabilities as well as IPoIB-RC. IPoIB-RC has a consistent bandwidth for larger messages of approximately 7700 Mbps while the best IPoIB-UD configuration, LRO with no LSO has a maximum bandwidth of approximately 7100 Mbps. The performance of the best IPoIB-UD configuration is understandable as a single-stream test does not fully utilize the computational resources of the system, and therefore the system can be free to quickly process the segmentation of messages, and the system CPUs operate at a higher frequency than that available on the network adapter, therefore for lower system loads, LSO's advantages cannot be fully realized for datagrams.

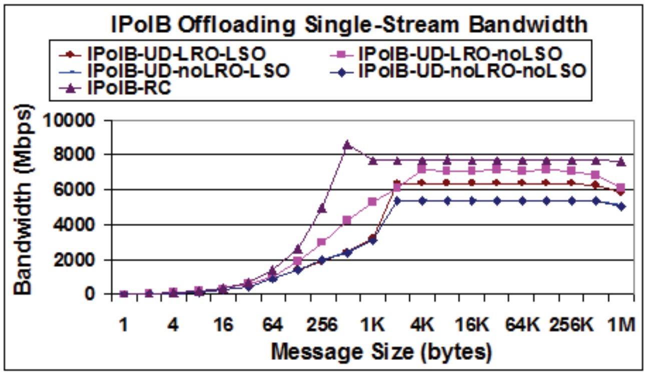


Figure 3. IPoIB offload single-stream bandwidth

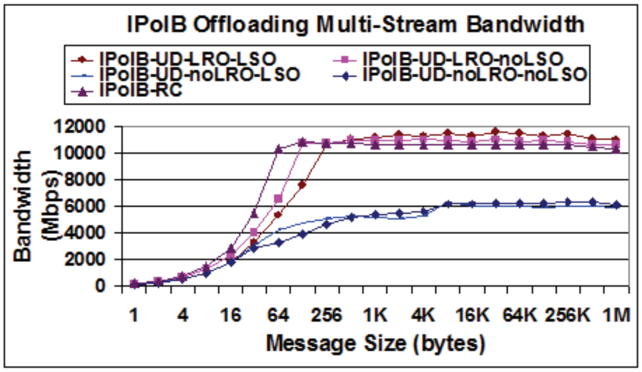


Figure 4. IPoIB offload multi-stream bandwidth

Figure 4 illustrates the bandwidth of different IPoIB offloading capabilities for the 8-stream case, where each processing core is assigned a bandwidth test thread. Contrasting these results with those in Figure 3, it is observed that the bandwidth of both LRO enabled configurations has increased significantly. The best of the configurations, with both LSO and LRO enabled, shows a 45.2% increase in maximum bandwidth over the single-stream case.

Offloading increases the overall maximum bandwidth achievable by 83.3% over the non-offloaded case, with an 85.1% average performance increase over all message sizes. In addition, we can now observe that LSO is having a beneficial effect on the achievable bandwidth. Unlike the single-stream tests, the system is more fully utilized and taking advantage of the network adapter's capability to segment outgoing packets yields an appreciable benefit. This offload can be of use in heavily loaded systems as the computational resources needed to segment outgoing packets are not needed to be expended by the CPU, and instead can be handled at the network adapter, increasing the overall system efficiency. IPoIB-RC benefits from the use of its VMTU segmentation offload, but fails to achieve the maximum bandwidth achieved by IPoIB-LRO-LSO, being out performed by 7.1%. Given that in a real data center application, multiple simultaneous data streams will be required, this shows that IPoIB-UD is the preferred mode for operation in an enterprise data center environment.

B. *Performance Comparison*

Figure 5 details the latency of the system for IB verbs, IPoIB operating in RC and UD (LRO and LSO enabled) mode and SDP over RC, operating in both buffered-copy and zero-copy modes. The baseline latency for IB verbs is only 1.22 µs for very small messages, compared to 11.4 µs for SDP buffered-copy, 11.8 µs for SDP zero-copy, 20.5 µs for IPoIB-RC and IPoIB-LRO-LSO. SDP-ZC sees 8.6% higher latency than SDP-BC. IPoIB-RC sees 75.8% higher latency than SDP-BC.

Figure 6 illustrates the baseline single-stream bandwidths of SDP and IPoIB operating over RD and IPoIB over UD, with LRO and LSO offloading. SDP is evaluated in both buffered-copy and zero-copy modes. IB verbs RDMA (RC) bandwidth is also included for reference, showing the maximum bandwidth on our system. One can observe that SDP-ZC with a one-stream load comes within 13.5% of the IB verbs bandwidth.

IPoIB-RC outperforms IPoIB-LRO-LSO by 40.2% in terms of their respective maximum achievable bandwidths, while IPoIB-RC is outperformed by SDP-BC by 37.8% in terms of their respective bandwidth peaks. Figure 7 shows the multi-stream bandwidth available to IPoIB, SDP and multi-connection IB Verbs RDMA, with 8 streams. IPoIB bandwidth with offloading enabled is only 17.5% less than that of the maximum achievable IB verbs bandwidth at their respective peaks. IPoIB-LRO-LSO sees a 81.7% peak bandwidth gain by using 8 streams over the use of a single stream. SDP-ZC maximal bandwidth is able to come within 10.3% of the IB verbs maximum bandwidth, representing excellent use of resources, and the best bandwidth available to all protocols.

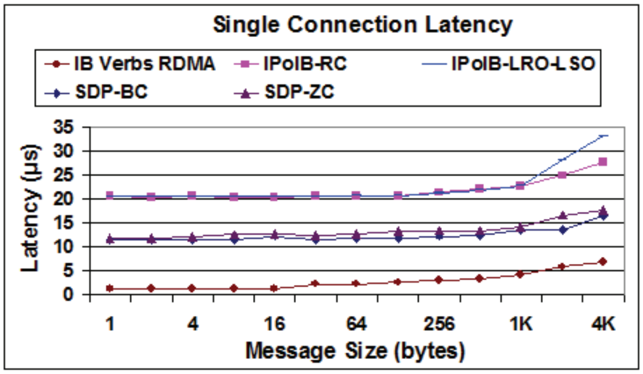


Figure 5. Baseline single-connection latency

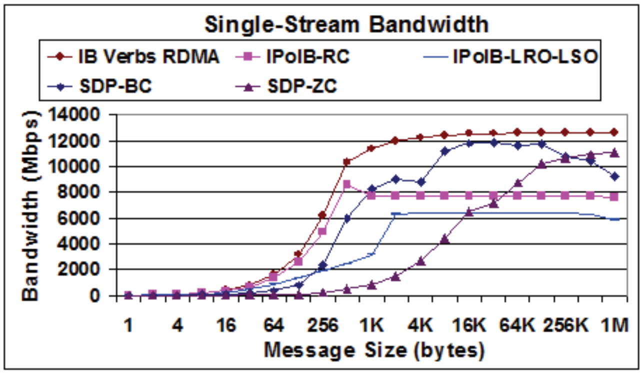


Figure 6. Single-stream bandwidth

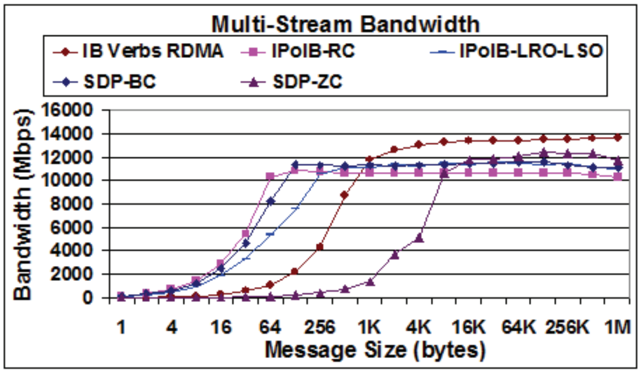


Figure 7. Baseline multi-stream bandwidth

Note that the theoretical maximum bandwidth of the ConnectX adapter is 16 Gbps, as is the speed of the PCI-Express bus (8x). However, the overhead present in the PCI-Express bus makes the maximum achievable bandwidth lower than the theoretical maximum, making it the bandwidth bottleneck.

We have found that the addition of LRO and LSO offloads for the IPoIB Datagram mode has enhanced the performance of IPoIB-UD such that it can produce maximum bandwidth 7.1% higher than the maximum bandwidth achievable by IPoIB-RC (multi-stream) and only 6.5% less than that of the best achievable multi-stream SDP bandwidth.

VI. Data Center Results

A three-tier data center architecture was setup for further testing using an Apache 2 web server for all static http and image serving, a separate JBoss 5 application server performing all server side Java processing and an additional database system running MySQL. A high-level overview of the data center implementation used for testing can be seen in Figure 8. All tiers use either IPoIB or SDP for any given test.

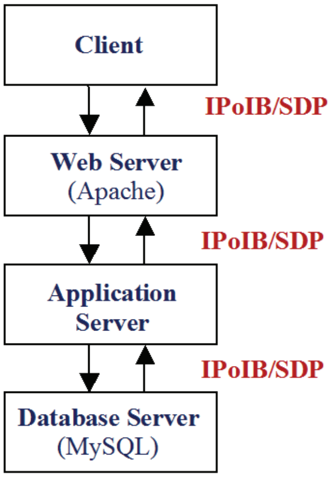


Figure 8. Test Data Center Architecture

To assess the performance of IPoIB and SDP we have used the TPC-W [29] benchmark, which utilizes a real web implementation of a bookstore, testing the latency seen by multiple emulated clients as a total request/response time seen at the client side. The average number of web interactions per second or WIPS that the system is able to sustain is also measured to determine system throughput. Strict timing requirements must be met to ensure that the system is providing reasonable service to all of its clients. The results shown in this paper are free of any errors during the entire measurement period, ensuring that the system can handle the given load consistently over longer time periods. The tests were conducted at the maximum emulated client load for each of the configurations.

The TPC-W benchmark uses specific load mixes to approximate real data center usage patterns. It utilizes static and dynamic website content, as well as databases for data storage on the server side. The benchmark can be scaled in size, and for our systems it has been implemented using 100,000 items, with the remainder of the data scaled (customers, etc.) as dictated in the specifications. The website is interacted with by a remote browser emulator, which generates multiple clients that interact with the benchmark website. These clients have specified interaction wait times and follow patterns specific to a given behaviour set to replicate real conditions.

The TPC-W implementation used was from New York University [28] and uses Enterprise Java beans. It was modified to work with our systems and updated JBoss and MySQL interfaces. The client load was created using a remote browser emulator implementation from the University of Wisconsin [9]. It has been extensively modified to work with our EJB website implementation.

Data center testing using SDP utilizes the available zero-copy threshold switching value, a message size at which SDP switches between the use of buffered-copy and zero-copy methods. This threshold value can have a significant influence on bandwidth performance, and therefore the default threshold (64 KiB) was used for testing, which has been observed to have good overall performance for both latency and bandwidth.

The results of the data center testing are shown for IPoIB-UD with offloads enabled, IPoIB-UD with offloads disabled, IPoIB-RC and SDP in Figure 9. We can observe that the highest throughput (in terms of WIPS) is achieved with the IPoIB-UD with offloads enabled mode. It achieves an average WIPS of 89.19, which is 15.4% greater than the throughput of IPoIB-UD with offloads disabled (77.30 WIPS). IPoIB-UD with offloads enabled also outperforms IPoIB-RC (84.32 WIPS), although by a smaller margin of 5.8%. It also outperforms an SDP configuration, using a 64 KiB buffered-copy/zero-copy switching threshold (to provide the best SDP mode for the data center testing), by 29.1%, with SDP having a WIPS of 69.08.

The latency results for the data center testing shown in Figure 10 are presented as a complimentary CDF, meaning that the percentage of transactions that take longer than the indicated time period is indicated by the lines on the graph. Therefore, smaller numbers are better. Examining the latency results of the different modes, we can see that both LSO and LRO have a beneficial effect on the system, reducing its latencies to below those of the non-offloaded mode and of SDP. Offloaded IPoIB-UD shows benefits for the vast majority of benchmark operations, particularly the most common requests, such as the home page and shopping cart requests, as well as search functionality. The non-offloaded IPoIB-UD shows slightly better performance for less utilized operations such as admin functions and more complex database operations, such as previous order displays. This reduced latency on more complex operations is most likely the result of non-aggregated packet reception (no-LRO) which could provide lower latencies for individual packet operations. Larger streams see a benefit of using LRO, but infrequent complex responses are delayed slightly in the LRO aggregation, as they undergo more communication between the data center tiers, requiring aggregation at all three levels of the

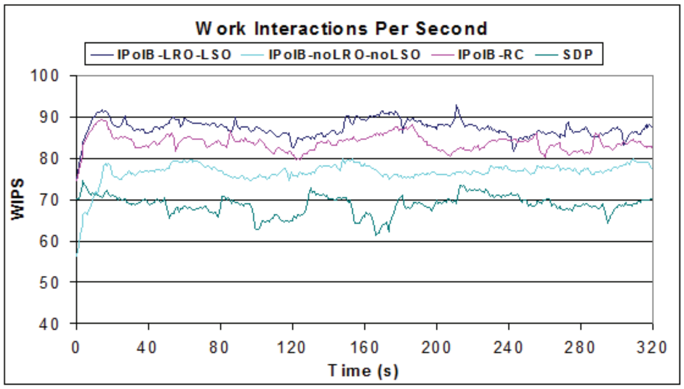
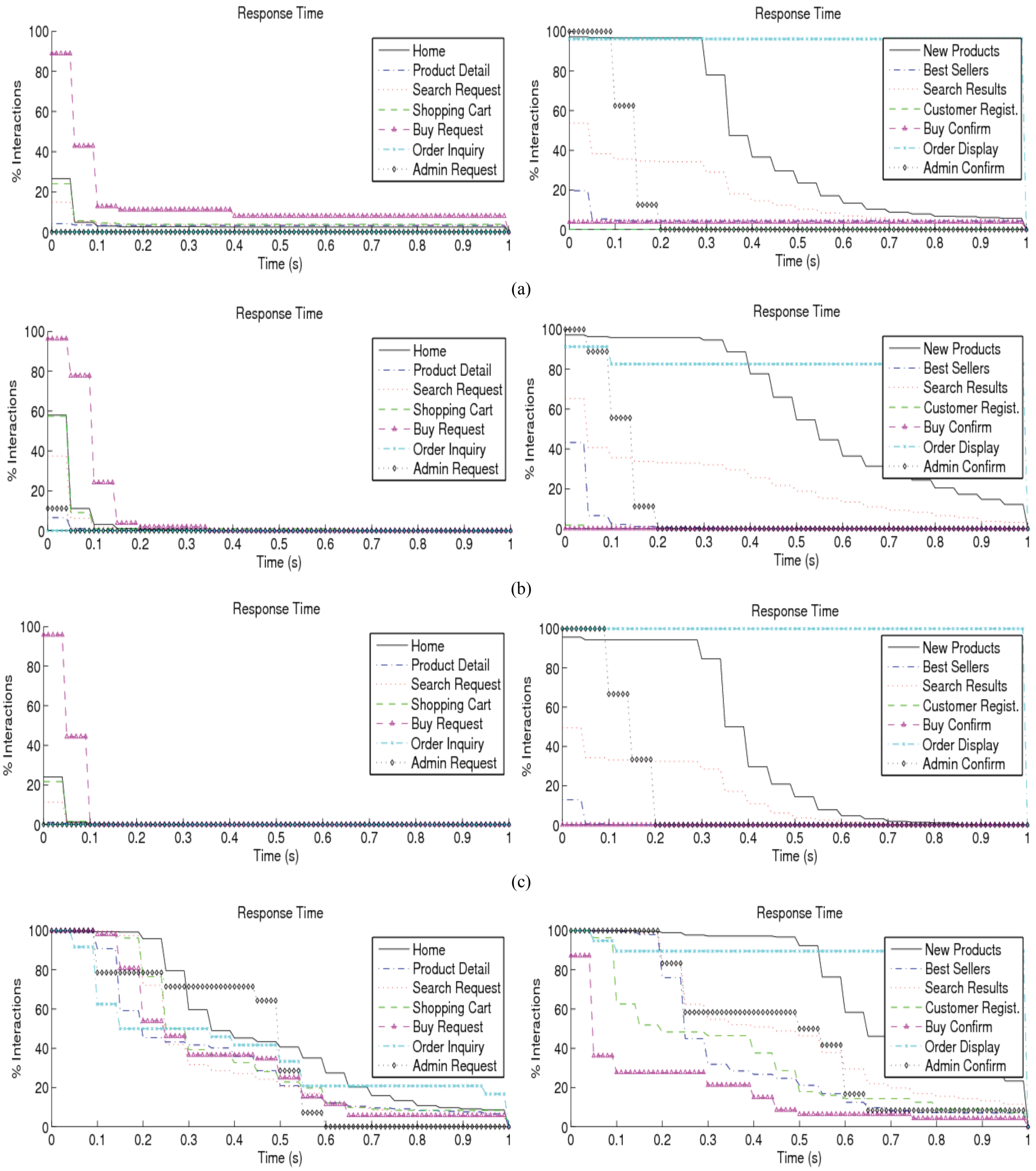


Figure 9. TCP-W Data center Throughput



(a)

(b)

(c)

(d)

Figure 10. TPC-W data center latency: (a) IPoIB-UD-LRO-LSO, (b) IPoIB-UD-noLRO-noLSO, (c) IPoIB-RC, and (d) SDP

data center. IPoIB-UD-LRO-LSO and IPoIB-RC latencies are essentially equal, with IPoIB-RC having a very small advantage. Examining the latency results for the SDP configuration leads to an unexpectedly high overall latency for all operations, higher than those observed for IPoIB configurations.

This is most likely due to the RDMA semantics underlying SDP. The data center benchmark uses a typical sockets-based configuration, which creates a large number of connections between individual tiers. For a high system load, there can be over 600 active connections between the web server and the clients, up to 200 connections between the application server and the web server and up to 50 connections between the application server and the database server. Due to the nature of the request pattern, one cannot be assured that each connection between the data center tiers has data in its send queue at all times. Whenever an inactive SDP connection must send data it requires the sending/reception of RDMA control messages, which will add one full round trip time latency to the transmission of data. Therefore, the system can experience higher latencies than would be seen in a fully saturated network situation, as the RDMA control messages cannot be hidden by sending them during existing ongoing transmissions. This has been observed to increase small message size latencies by as much as 36% for our system. The high latencies seen for the less complex interactions, such as the home page display or the shopping cart display show that high latencies are seen for operations that require only minimal participation from the lower data center tiers. Therefore, we can observe that the latency results are most probably not attributable to a problem in any single layer of the data center.

To determine if the poor results of the SDP data center was due to issues relating to the overhead associated with SDP connections and RDMA control messages, a test utilizing only 50 connections on the web server to client connection side was conducted. To create a similar load to those seen with a larger number of connections, the activity frequency of the emulated browsers was significantly raised to emulate the load of that a large number of clients would typically create. The results are shown in Figures 11 and 12. An improvement in the throughput of the system to levels higher than that of previous SDP configuration, and even that of the IPoIB configurations, and latency reductions was seen. This shows that the reduced SDP performance was due, at least in part, to the number of connections and the activity level of said connections between the data center tiers and the web server/client connection.

Examining the latency results of the reduced connection, higher client activity configuration, it is observed that the latency of the SDP system has improved significantly from the comparable SDP results in Figure 10(d). The most important of these latency changes has been the improvement in the most frequent operations, such as the home page display operation. However, when compared to the response times of IPoIB, the latencies seen in the SDP data center implementation are still higher across all operations, with the exception of the order display operation. The order display operation requires a large SQL query to the database tier.

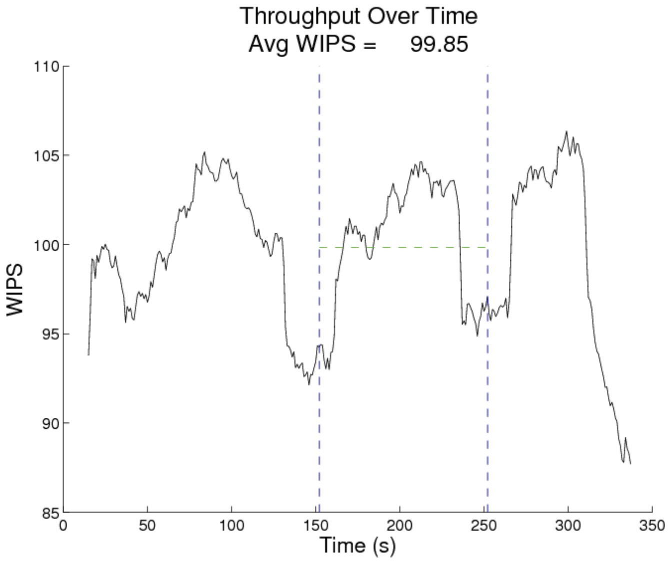


Figure 11. TPC-W data center Throughput for SDP with 50 Client Connections

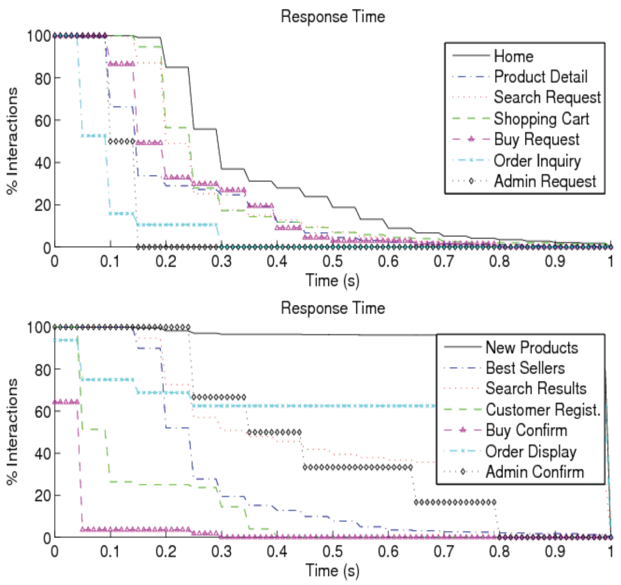


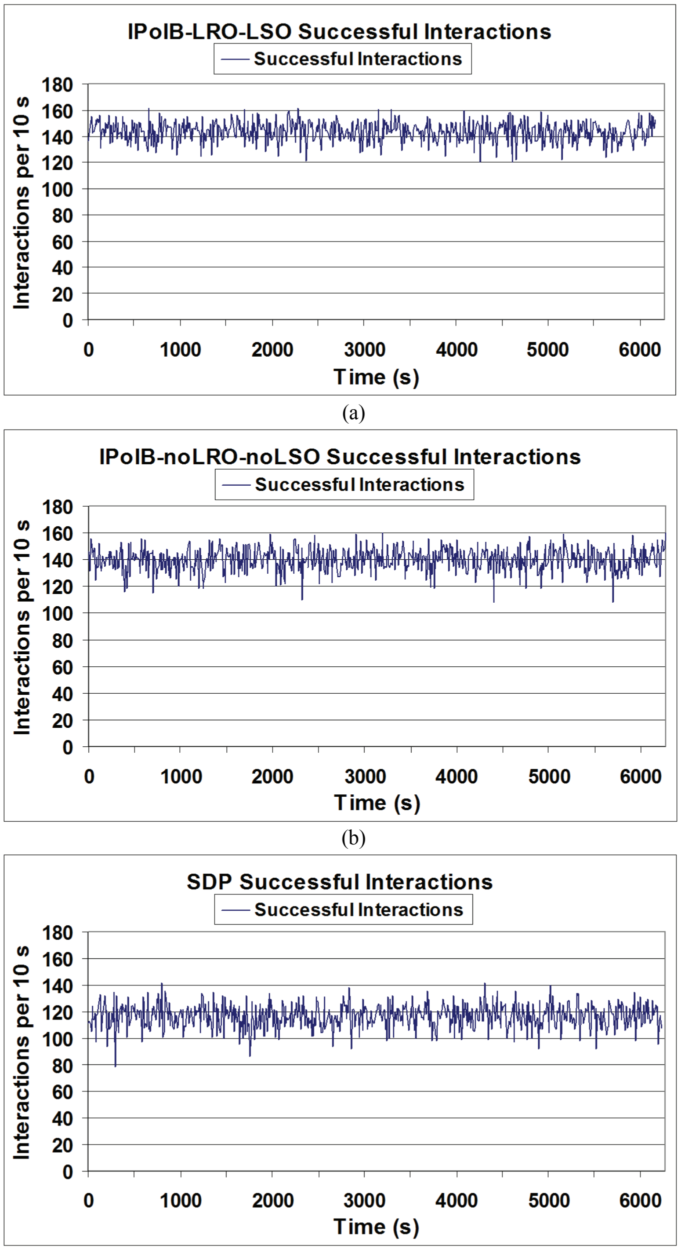
Figure 12. TPC-W data center latency for SDP with 50 client connections

Therefore, it can be concluded that for current data center implementations, and sockets-based data center software, IPoIB with offloading can provide an appreciable benefit over using SDP. It can provide higher throughput with large numbers of active connections, and provide lower latencies than can be achieved using SDP.

A. SPECWeb2009 Data Center Testing

In addition to the TPC-W benchmark, we have run the E-commerce workload for the SPECWeb2009 benchmarking suite [25]. The SPECWeb2009 E-Commerce workload simulates a commercial website complete with shopping carts, checkout and SSL encryption. It consists of a Prime client system, which connects to other client machines to provide a workload for the system under test (SUT). The SUT is made up of a webserver front end and an application server. Instead of a database back-end, a fastCGI-based backend simulator (BeSIM) is provided which provides data to queries from the system under test in a similar manner to that of a real database. The overall setup of the system is very similar to that used for TPC-W in Figure 8, with the MySQL database replaced with the BeSIM application. The client software is a Java application, the website is a jsp implementation and the backend simulator is a FastCGI application. The same Apache 2 and Jboss 5 versions were used in the testing of SPECWeb as were used for the TCP-W benchmark.

The resultant throughput of the SPECWeb2009 testing is shown in Figure 13, where the aggregate number of successful interactions over 10 second intervals is shown. It can be observed that this benchmark has much lower throughput than the TPC-W benchmark. A combination of a java driven front-end client generation system and the use of SSL for certain requests to the data center means that the network performance is hindered due to Java, as previously discussed [30], and the benchmark is much more CPU intensive due to the SSL processing, than the TPC-W benchmark, which does not operate over SSL. We can see that the general observations for



(a)

(b)

(c)

Figure 13. SPECWeb2009 throughput: Aggregate Successful Interactions per 10 Seconds (a) IPoIB-LRO-LSO (b) IPoIB-noLRO-noLSO (c) SDP

the TPC-W benchmark still hold with the SPECWeb2009 benchmark. SDP has throughput issues that are causing it to under perform its IPoIB alternatives. IPoIB-LRO-LSO is still beneficial over that of IPoIB-noLRO-noLSO, but its margin is only 2.75%. The best IPoIB scheme outperforms SDP by 22.7% in terms of throughput, which is comparable to the 29.1% difference found using TPC-W

Observing the maximum response times of the three configurations in Figure 14, we can see that the maximum response time of the IPoIB-LRO-LSO scheme outperforms the other configurations. In this case, we have a CPU intensive data center that is benefiting from the offloading of segmentation. SDP should also see this benefit, but it maintains

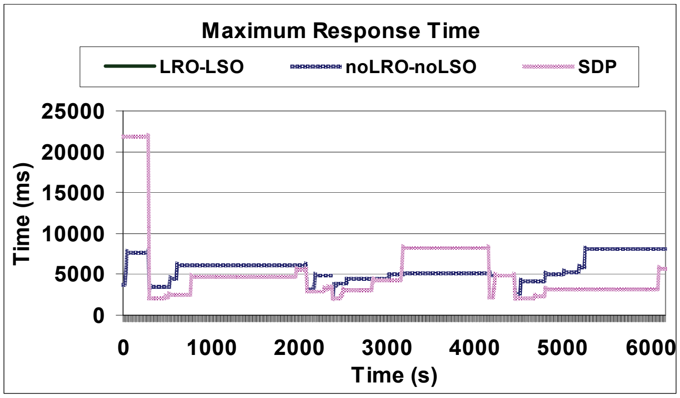


Figure 14. Maximum Response Time

a maximum response time that is the most volatile of the schemes, and on average roughly equivalent to that of non-offloaded IPoIB. It should be noted that these tests are not utilizing as many connections as the TPC-W benchmark, and therefore we expect an improvement in the performance of SDP from that of the TPC-W benchmark.

VII. Conclusions and Future Work

We have explored the performance of the IPoIB offloading capabilities of the Mellanox ConnectX InfiniBand adapters. We have found that IPoIB offloading techniques realizes performance gains in terms of bandwidth and latency over the equivalent non-offloaded traffic. Micro-benchmark results show an 83.3% improvement in maximum achievable bandwidth using segmentation offloading over non-offloaded IPoIB-UD. We have observed an 11.1% average and 26.2% maximum latency reduction using offloaded IPoIB over non-offloaded IPoIB. In addition, it was observed that the benefits of using IPoIB-RC over IPoIB-UD are minimal and therefore for future high performance networks such as CEE, the inclusion of a reliable connection mode is most likely unnecessary.

It was shown that the available IPoIB offloading capabilities can increase real data center throughput by 15.4% over that of a non-offloaded IPoIB configuration and 29.1% over that of the SDP configurations. In addition, IPoIB can provide much lower latencies than those of a data center utilizing SDP. We have analyzed issues constraining the performance of SDP and determined that a reduced number of connections than would typical be used in a traditional Ethernet data center can resolve such problems. This was further confirmed with throughput testing using the SPECWeb2009 E-commerce workload, where SDP had a lower throughput than IPoIB.

Our plans for future work in the area of high-performance sockets-based data centers include further investigation into the causes behind poor SDP data center performance and proposals to increase SDP performance in typical data center contexts. The affect on SDP performance of the zero-copy threshold value also warrants further investigation. A further verification of the results shown by exploring the behaviour of additional SPECWeb2009 workloads will also be investigated. In addition, the application of other beneficial IB features to the performance of data centers must be investigated, for example, the use of QoS mechanisms for optimizing the network traffic in modern data centers, potentially integrated with hybrid IB/Ethernet networks via VPI.

ACKNOWLEDGEMENTS

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada Grant #RGPIN/238964-2005, Canada Foundation for Innovation and Ontario Innovation Trust Grant #7154, Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract DE-AC02-06CH11357, and the National Science Foundation Grant #0702182. We would like to thank Mellanox Technologies for the resources.

REFERENCES

[1] P. Balaji, S. Bhagvat, H. Jin and D.K. Panda, “Asynchronous zero-copy communication for synchronous sockets in the Sockets Direct Protocol (SDP) over InfiniBand,” In 6th Workshop on Communication Architecture for Clusters (CAC), 2006.

[2] P. Balaji, S. Bhagvat, D. K. Panda, R. Thakur, and W. Gropp, "Advanced Flow-control Mechanisms for the Sockets Direct Protocol over InfiniBand". In the IEEE International Conference on Parallel Processing (ICPP), Sep 10-14, 2007, XiAn, China.

[3] P. Balaji, W. Feng, Q. Gao, R. Noronha, W. Yu and D.K. Panda, “Head-to-toe evaluation of high-performance sockets over protocol offload engines,” In 2005 IEEE International Cluster Computing (Cluster), 2005, pp. 1-10.

[4] P. Balaji, S. Narravula, K. Vaidyanathan, S. Krishnamoorthy, J. Wu and D.K. Panda, “Sockets Direct Protocol over InfiniBand in clusters: is it beneficial?” In IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2004, pp. 10-12.

[5] D. Cohen, T. Talpey, A. Kanevsky, et al., “Remote Direct Memory Access over the Converged Enhanced Ethernet Fabric: Evaluating the Options”, 17th IEEE Symposium on Hot Interconnect (HotI), 2009, pp. 123-130.

[6] D. Goldenberg, M. Kagan, R. Ravid and M. Tsirkin, “Transparently achieving superior socket performance using zero copy socket direct protocol over 20Gb/s InfiniBand links,” in 2005 IEEE International Conference on Cluster Computing (Cluster), 2005, pp. 1-10.

[7] R.E. Grant, A. Afsahi and P. Balaji, “An evaluation of ConnectX Virtual Protocol Interconnect for data centers,” 15th International Conference on Parallel and Distributed Systems, 2009, pp. 57-64.

[8] R.E. Grant, M.J. Rashti and A. Afsahi, “An analysis of QoS provisioning for sockets direct protocol vs. IPoIB over modern InfiniBand networks,” In International Workshop on Parallel Programming Models and Systems Software for High-End Computing (P2S2), 2008, pp. 79-86.

[9] T. Horvath. (2008, TPC-W java client implementation. [http://www.ece.wisc.edu/~pharm/tpcw.shtml], 2008.

[10] W. Huang, J. Han, J. He, L. Zhang and Y. Lin, “Enabling RDMA capability of InfiniBand network for Java applications,” In International Conference on Networking, Architecture, and Storage (NAS), 2008, pp. 187-188.

[11] IEEE. IEEE standard for local and metropolitan area networks-virtual bridged local area networks - amendment: Priority-based flow control - 802.1Qbb. [http://www.ieee802.org/1/pages/802.1bb.html].

[12] IEEE. IEEE standard for local and metropolitan area networks-virtual bridged local area networks - amendment: 10: Congestion notification - 802.1Qau. [http://www.ieee802.org/1/pages/802.1au.html].

[13] IEEE. IEEE standard for local and metropolitan area networks-virtual bridged local area networks - amendment: Enhanced transmission selection - 802.1Qaz. [http://www.ieee802.org/1/pages/802.1az.html].

[14] IEEE. IEEE standard for station and media access control connectivity - 802.1AB. [http://www.ieee802.org/1/pages/802.1ab.html].

[15] INCITS - Technical Committee T11. ANSI standard FC-BB-5 - fibre channel over ethernet (FCoE). [http://www.t11.org/ftp/t11/pub/fc/bb-5/09-056v5.pdf].

[16] InfiniBand Trade Association. InfiniBand Architecture Specification, Volume 1, October 2004.

[17] Internet Engineering Taskforce. Transparent interconnection of lots of links (TRILL). [http://www.ietf.org/dyn/wg/charter/trill-charter.html].

[18] R. Jones, "Netperf Network Performance Benchmarking Suite," 2008.

[19] S. Kent, R. Atkinson, “Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[20] Mellanox Technologies, [http://www.mellanox.com].

[21] OpenFabrics Alliance, [http://www.openfabrics.org].

[22] J. Satran, K. Meth, C. Sapuntzakis, M. Chadalapaka, E. Zeidner, “Internet small computer systems interface (iSCSI)”, RFC 3720, April 2004.

[23] H.V. Shah, C. Pu, and R.S. Madukkarumukumana, “High Performance Sockets and RPC over Virtual Interface (VI) Architecture,” In 3rd workshop on Network-Based Parallel Computing: Communication, Architecture, and Applications (CANPC), 1999, pp. 91-107.

[24] H. Subramoni, P. Lai, M. Luo, D.K. Panda, RDMA over Ethernet - A Preliminary Study, Workshop on High Performance Interconnects for Distributed Computing (HPIDC'09), September 2009.

[25] SPECWeb 2009 Benchmark suite, [http:// www.spec.org/web2009].

[26] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson, “Stream Control Transmission Protocol”, RFC 2960, Network Working Group, Oct. 2000.

[27] A. Tirumala, F. Qin, J. Dugan, J. Ferguson and K. Gibbs, “Iperf: The TCP/UDP bandwidth measurement tool,” [Http://dast.Nlanr.net/Projects/Iperf], 2004.

[28] A. Totok, TPC-W-NYU. [http://cs.nyu.edu/~totok/professional/software/tpcw/tpcw.html], 2005.

[29] TPC Council, TPC-W Benchmark, [http://www.tpc.org/tpcw/].

[30] H. Zhang, W. Huang, J. Han, J. He and L. Zhang, “A performance study of Java communication stacks over InfiniBand and giga-bit Ethernet,” In IFIP International Conference on Network and Parallel Computing Workshops (NPC), 2007, pp. 602-607.