Improving High Performance Networking Technologies for Data Center Clusters

by

Ryan Eric Grant

A thesis submitted to the Department of Electrical and Computer Engineering

In conformity with the requirements for the degree of Doctor of Philosophy

Queen's University

Kingston, Ontario, Canada

September 2012

Abstract

This dissertation demonstrates new methods for increasing the performance and scalability of high performance networking technologies for use in clustered computing systems, concentrating on Ethernet/High-Speed networking convergence. The motivation behind the improvement of high performance networking technologies and their importance to the viability of modern data centers is discussed first. It then introduces the concepts of high performance networking in a commercial data center context as well as high performance computing (HPC) and describes some of the most important challenges facing such networks in the future. It reviews current relevant literature and discusses problems that are not yet solved.

Through a study of existing high performance networks, the most promising features for future networks are identified. Sockets Direct Protocol (SDP) is shown to have unexpected performance issues for commercial applications, due to inefficiencies in handling large numbers of simultaneous connections. The first SDP over eXtended Reliable Connections implementation is developed to reduce connection management overhead, demonstrating that performance issues are related to protocol overhead at the SDP level. Datagram offloading for IP over InfiniBand (IPoIB) is found to work well.

In the first work of its kind, hybrid high-speed/Ethernet networks are shown to resolve the issues of SDP underperformance and demonstrate the potential for hybrid high-speed networking local area Remote Direct Memory Access (RDMA) technologies and Ethernet wide area networking for data centers.

Given the promising results from these studies, a set of solutions to enhance performance at the local and wide area network level for Ethernet is introduced, providing a scalable, connectionless, socket-compatible, fully RDMA-capable networking technology, datagram-iWARP. A novel method of performing RDMA Write operations (called RDMA Write-Record) and RDMA Read over unreliable datagrams over Ethernet is designed, implemented and tested. It shows its applicability in scientific and commercial application spaces and is applicable to other verbs-based networking interfaces such as InfiniBand.

The newly proposed RDMA methods, both for send/recv and RDMA Write-Record, are supplemented with interfaces for both socket-based applications and Message Passing Interface (MPI) applications. An MPI implementation is adapted to support datagram-iWARP. Both scalability and performance improvements are demonstrated for HPC and commercial applications.

Statement of Collaboration

The work in Chapter 5 was performed collaboratively with Dr. Mohammad Javad Rashti. The theoretical design of send/recv datagram-iWARP was developed collaboratively. The development of the OF verbs interface and MVAPICH-Aptus implementation for send/recv datagram-iWARP were developed by Dr. Rashti, in collaboration with myself regarding the underlying native verbs stack. The development of the native iWARP stack for datagram-iWARP and all of the supporting code was done by me. The results for the MPI testing on data centers for HPC were gathered for cluster C1 collaboratively, while I performed all of the experiments on cluster C2. I designed, implemented and tested Broadcast. All of the text and other materials presented in Chapter 5 were authored/prepared by me.

Acknowledgements

I would like to first thank my supervisor Dr. Ahmad Afsahi for his valuable feedback and support in conducting the research for this thesis. Without his help, this work would not have been possible. I would also like to thank Dr. Pavan Balaji of the U.S. Argonne National Laboratory, for guidance and insightful technical advice during my doctoral studies. Thank you to the members of my thesis examining committee, Dr. Keith Underwood, Dr. Thomas Dean, Dr. Patrick Martin, and Dr. Karen Rudie for their valuable feedback concerning this research.

Many thanks to the office staff of the ECE Department here at Queen's, particularly Ms. Debra Fraser for her invaluable help in administrative matters during the tenure of my Doctoral studies.

To my co-workers in the Parallel Processing Research Laboratory, Dr. Mohammad Rashti, Dr. Ying Qian, Reza Zamani, Judicael Zounmevo, Greg Inozemtsev, Imran Faraji and Hosseim Mirsadeghi, thank you for your continued support over the years. A special thanks goes to Dr. Rashti for his very helpful collaboration on several projects during my studies and his expert advice concerning MPI.

I would also like to thank the Natural Sciences and Engineering Research Council of Canada for their generous support via an Alexander Graham Bell Canada Graduate Scholarship during my studies. I would also like to thank the Government of Ontario as well as Queen's University for further scholarship support over the last few years.

Finally, my deepest appreciation goes to my immediate and extended family for their support throughout my studies. Many thanks to my parents whose support over the last few years has been invaluable, and also to my grandparents that have provided support and a warm meal on many occasions. A special and heartfelt thanks goes to my loving and understanding wife Meagan without whom this would not be possible. Finally, I would like to thank my daughter Charlotte for being a source of inspiration and motivation.

Table of Contents

List of Figures

Glossary

AMD – American Micro Devices Inc.
ATM – Asynchronous Transfer Mode
CEE – Converged Enhanced Ethernet
CESP – Complex Event Streaming Processor
CMP – Chip Multi-Processor
CMT – Chip Multi-Threading
CPU – Central Processing Unit
CQ – Completion Queue
DDP – Direct Data Placement
DDR – Double Data Rate
DMA – Direct Memory Access
DNS – Domain Name Service
DRAM – Dynamic Random Access Memory
FDR – Fourteen Data Rate
GigE – Gigabit Ethernet (10 GigE – 10 Gigabit Ethernet)
GRH – Global Routing Header
HCA – Host Channel Adapter
HPC – High Performance Computing
HPN – High Performance Networking
HTTP – HyperText Transfer Protocol
I/OAT – Intel I/O Acceleration Technology
IB – InfiniBand
IP – Internet Protocol
IPoIB – IP over InfiniBand
IWARP – Internet Wide Area RDMA Protocol
LID – Local ID
LLP - Lower Layer Protocol
LRO – Large Receive Offload
LSO – Large Send Offload
MOM – Message Oriented Middleware
MPA – Marker PDU Aligned
MPI – Message Passing Interface
MTU – Maximum Transfer Unit
NIC – Network Interface Controller (Card)
OS – Operating System
PHP – Personal HomePage Tools (Hypertext Processor)
QDR – Quadruple Data Rate
QoS – Quality of Service
QP – Queue Pair
RC – Reliable Connection
RD – Reliable Datagram
RDMA – Remote Direct Memory Access
RDMAP – Remote Direct Memory Access Protocol
SCTP – Stream Control Transmission Protocol
SDP – Sockets Direct Protocol
SDR – Single Data Rate
SMP – Symmetric Multi-Processor
SMT – Simultaneous Multi-Threading

SRQ – Shared Receive Queue
TCA – Target Channel Adapter
TCP – Transmission Control Protocol
TOE – TCP Offload Engine
TPC-W – Transaction Processing Performance Council – Web application
UC – Unreliable Connection
UD – Unreliable Datagram
ULP - Upper Layer Protocol
VIA – Virtual Interface Architecture
VPI – Virtual Protocol Interconnect
WR – Work Request
XRC – eXtended Reliable Connection

Chapter 1 Introduction

This thesis concerns the current state of the art in high performance networking technologies for high performance computing and commercial clustered computing systems, outlining the current issues in the area and presenting novel research solutions to those issues. It will show that the enhancement of such networking technologies is both feasible and desirable, and its overall impact should be significant in both the academic and industrial communities. The ultimate goal of this thesis is to present novel networking solutions that are useful in the current environment of networking convergence, while also being scalable and useful for applications predicted to make up the majority of Internet traffic in the future. Using the described techniques, it illustrates the design of proposed next generation networks for Remote Direct Memory Access (RDMA) operation over Ethernet using unreliable datagram transports. This technique allows a source system to access remote system's memory for both write and read operations as if it were local to the source system.

Clustered computing systems are typically connected groups of individual computing systems that provide services to many clients. These services are widely varied and can range from high-performance scientific calculations to web serving data centers to credit card processing computers. The range of applications for clustered systems is very broad, and as such it is important to consider the workload of a given system when devising improvements for it. In this thesis the term "data center" is typically used to refer to the commercial use of clustered computing systems, or for describing both commercial and scientific clustered computing systems. The term "HPC data center" will be used when referring only to a clustered computing system primarily designed for use as a scientific computing system. This thesis focuses on web serving data centers, data centers processing information over the World Wide Web, as well as some discussion on high performance computing. A single one of these classifications still yields a large number of possible configurations and constraints that can be placed on the data center

infrastructure, but the primary concern of all of the systems is performance, in order to meet the real-time demands of the users while balancing this with the cost of providing said services.

It is certain that for the majority of data centers, the ability to provide fast and efficient service to clients is of the utmost importance. Increasing the performance of a data center is typically motivated by reducing the cost of providing the data center service, by doing more with less. Therefore, methods of increasing the performance of socket-based network data centers using advanced technologies for enhancements to the local system and local network performance, particularly those available to modern InfiniBand (IB) [56] networks, is an important topic for study. Such performance is crucial to data center capacity and efficiency. However, a large local network performance gain could be marginalized by poor wide area networking (WAN) performance, so it too must be addressed.

We demonstrate a method of increasing both LAN and WAN performance using remote direct memory access (RDMA) [93] for datagrams over Ethernet. Our solution utilizes a software approach to provide proof of concept for a hardware datagram-based Internet Wide Area RDMA Protocol (iWARP) [94] network adapter. This has several benefits over a traditional TCP based iWARP, where iWARP over datagrams does not have the overhead required for a TCP based system. No information needs to be stored concerning connections, eliminating a significant resource overhead. In addition, datagrams do not require as rigorous an error checking scheme as TCP packets, allowing for faster reception. The lowered complexity of the datagram approach also is advantageous for the design of hardware using an iWARP over datagram protocol as the full functionality of TCP does not need to be fabricated in silicon as a TCP offload engine, instead replacing this more complex logic with a much smaller datagram engine. This could allow for multiple parallel processing pipelines, leading to an increase in performance. Alternatively, datagram-iWARP can also be easily and inexpensively added to existing iWARP hardware implementations. The datagram-iWARP design also has the advantages observed by other researchers working on software stacks [23], namely that the server-side hardware can take

advantage of faster, less CPU intensive network operations through RDMA hardware for all client systems, instead of just those using RDMA capable hardware. Further work on this topic has yielded two interfaces capable of demonstrating datagram-iWARP's performance versus traditional iWARP for both commercial sockets-based network applications as well as scientific high performance applications compatible with the Message Passing Interface (MPI) [76][120].

1.1 Motivation

The majority of the Canadian population, whether they are aware of it or not, uses the services of computing centers daily. Some of the largest computing centers are intended only for commercial use, such as the large data centers powering Google, YouTube and Facebook. Enhancing the performance of data centers is of great concern to many data center operators. Constant improvement in speed is required in order to keep up with growing demand, and to remain competitive with competing data centers. Improving the performance of existing systems already installed in data centers is beneficial for performance improvement and lowered operating costs.

Also of great importance is the improvement in the quality of content that faster data centers can provide. As the capabilities of data centers increase, their ability to provide new content to users grows. Streaming video data centers, such as YouTube, were not conceivable in the earlier days of the Internet as the technical capabilities did not exist to either supply acceptable quality streaming video to clients, or for clients to receive the required amount of data. As Internet use continues to climb at ever-increasing rates, the performance capabilities of data centers will continue to need enhancement in order to provide next generation services to clients.

Unfortunately, very high-speed traditional networking technologies require significant CPU resources to aid in the sending and receiving of data. This is particularly troubling for dynamic request serving data centers, as they require computing power to process dynamic requests in large quantities. Techniques exist to attempt to offload some of this processing requirement, but

with traditional Ethernet networking technologies, even those with a TCP offload engine still require kernel involvement in the sending and receiving of data. Emerging technologies such as iWARP [94] and InfiniBand [56], which provide RDMA capabilities to modern networks, are desirable but these technologies still have drawbacks that must be addressed before they become adopted by mainstream commercial data centers. This thesis seeks to resolve some of the issues preventing the adoption of such technologies by major service operators.

The need for higher speed networking technologies is increasing as Internet traffic is predicted to grow at an annual rate of over 29% for the next 4 years [19]. In addition, the volume of video being transmitted over the Internet is 51% of all traffic (not including peer to peer file sharing of video), and by 2016 it is expected to reach 54% [19]. This equates to nearly 45,000 petabytes of video per month being transmitted. With this volume of video, it would take over 6,000,000 years to watch all of the video being transmitted in a single month [19]. If one includes predictions of video traffic including videos shred over peer to peer file sharing, the proportion of overall video traffic on the Internet rises to 86% by 2016 [19].

With this massive growth in the bandwidth consumed by media delivery, we face major future hurdles in effectively providing an infrastructure that is capable of offering such services reliably and in a cost effective manner. Increases to the efficiency of current networking technologies, and the introduction of current cutting edge high performance technologies can help to reduce the costs of providing such large amounts of data to a growing number of clients.

HPC clustered computing systems are also experiencing scalability issues due to their reliance on connection-based networking protocols. As individual node count grows as well as per-node processor core count the number of connections required to fully connect all processing in a running job is growing rapidly. The overhead associated with these connections continues to grow as well. Consequently, improvements to the scalability of HPC systems are important and can also lead to improvements in overall performance. This thesis explores solutions to enhance the scalability of HPC data centers as well as their performance.

1.2 Problem Statement

High performance networks are converging with Ethernet technologies, and bringing many of the benefits of their increased performance to next generation Ethernet. Several approaches have been designed and implemented for existing high-performance networks but actual deployment and usage of such networks in a commercial environment has been very limited. The current designs for RDMA-capable networks face scalability issues, particularly when considering connection-based approaches. A scalable design that can still provide RDMA functionality is needed for future generation large-scale, high-speed networks. Datagram-based UDP sockets applications cannot take advantage of the RDMA-capable network socket interfaces. A design that can provide RDMA capabilities to such applications is needed to ensure their continued high-level of performance in next generation networks.

This thesis seeks to explore the possible features of high speed networks that could be of use in next-generation Ethernet technologies, identify the most promising design choices and improve upon them. Primarily, this thesis poses the following questions:

1) How can existing high performance networking features be applied to next generation Ethernet and which features are the most effective?
2) What are the limitations to existing convergence technologies, and in what areas could their performance be improved?
3) How efficient can hybrid networking designs be for typical commercial applications? Are there protocol or networking inefficiencies that can be resolved?
4) Can advanced networking techniques like RDMA, OS bypass and zero-copy mechanisms be developed in a scalable way for future systems?
5) How can existing datagram based applications take advantage of advanced networking technology?

1.3 Contributions

This thesis makes several contributions in the area of high performance networking. The material in Chapter 3 details contributions in several areas. First, in Section 3.2, it provides what was the first look at the function and performance of InfiniBand's Quality of Service (QoS) provisioning. It demonstrates how QoS can be of use. Section 3.3 also explores high-performance sockets in a commercial data center context and IPoIB [54] offloading methods. It exposes issues with SDP [12] in such a context that were to our knowledge previously unknown. It proposes possible causes for the underperformance of SDP. In addition, it determines that the most effective offloading mechanism is Large Receive Offload (LRO), while Large Send Offload (LSO) is less important to include in next generation Ethernet solutions. Chapter 3 answers the first two questions in the problem statement, it explores which high performance features could be useful in future networks and discovers some limitations to SDP, which is a current convergence technology.

Chapter 4 presents the first implementation of SDP over IB that can take advantage of IB's eXtended Reliable Connection (XRC) [55] mode in Section 4.2. It demonstrates that the performance issues observed for SDP are not primarily caused by connection management overhead on the network interface controller (NIC). Section 4.3 continues by examining hybrid Ethernet/IB networks using Virtual Protocol Interconnect (VPI) [72], and shows that VPI can provide a solution to the performance problems seen by SDP by replacing the under performing layers with Ethernet connections. The chapter answers the third question in the problem statement by showing that hybrid networks can be of use, and illustrates that protocol limitations can be avoided by using hybrid networking solutions at different tiers of a data center architecture.

Chapter 5 builds on the previous chapter's material by providing methods of improving Ethernet performance by utilizing RDMA, but using a scalable transport solution, User Datagram Protocol (UDP). This chapter details the design of the first send/recv capable datagram-iWARP solution. Details concerning the implementation of a software version of datagram-iWARP are

discussed, and the implementation is compared to the equivalent TCP iWARP stack. Datagram-iWARP is found to have superior scalability to the TCP iWARP stack, and also benefit from improved performance for our software iWARP prototype. The chapter answers part of questions 4 and 5 from the problem statement, it demonstrates that OS bypass and RDMA send/recv techniques can be applied in a scalable way to systems. It also shows that existing HPC applications can take advantage of such networks.

Chapter 6 continues the design work in Chapter 5 by introducing the first RDMA Write method that is capable of operating over an unreliable datagram transport, called RDMA Write-Record. It is an RDMA method that is also applicable to other verb compatible networks, like InfiniBand, and allows for operation in both a message semantic mode and a true memory semantic mode. The message semantic mode makes it possible to implement an interface for use with sockets-based applications. Such an interface is described and the implementation results are shown in comparison to the equivalent Reliable Connection (RC) modes. Finally, MPI is modified to include a new transport/protocol communication path capable of handling RDMA Write-Record traffic. Chapter 6 completes the answers to questions 4 and 5 from the problem statement by demonstrating that RDMA (Write) methods can be applied favorably to future systems as well as showing that existing commercial datagram-based applications can take advantage of the RDMA methods in both Chapters 5 and 6.

1.4 Dissertation Outline

This dissertation starts by reviewing the background material and prior art for high performance networks, including 10-Gigabit Ethernet, InfiniBand [56], and iWARP [93]. All of these interconnects can be run in a socket-based mode that is typically required of Internet data centers. It also reviews messaging middleware used for HPC and other applications, as well as commercial data center architectures. Each chapter from 3 to 6 begins with a brief overview of the most pertinent prior work in the area discussed in that chapter. Chapter 3 studies techniques

that have been proposed for increasing the native performance and socket-based performance of networking technologies for data centers. Through this study it concludes which features may be of use in next generation Ethernet networks and identifies performance issues surrounding some socket interface communication methods.

Chapter 4 addresses the issues discovered in Chapter 3 and proposes two possible solutions. The first is a new adaptation to the underlying socket direct protocol for reducing connection management overhead while the second is a hybrid networking approach that takes advantage of VPI technology to offer a hybrid IB/Ethernet data center architecture.

Chapter 5 uses the information learned through the research performed in Chapters 3 and 4 by proposing a new RDMA send/recv method that is capable of operating over datagrams on an Ethernet network. The method is designed, implemented and tested in comparison to a traditional TCP-based iWARP software implementation.

Chapter 6 expands on the material in Chapter 5 by designing a novel method of performing RDMA Write operations over an unreliable datagram transport, the first of its kind for any verbs based network. The new method called RDMA Write-Record is implemented and compared to traditional RDMA Write in both commercial data center applications through a custom designed socket interface as well as high performance computing situations via an MPI interface.

Chapter 7 concludes the thesis with a final discussion of all of the material. It draws conclusions as to the best methods proposed for data centers, particularly for future data center architectures.

Chapter 2 Background

This chapter outlines the state of the art in the area of data center performance improvements, concentrating on high performance networking technologies. The chapter is divided into several sections. The first introduces the software and hardware of the data center infrastructure that is of the most interest to this thesis. The software architecture of multi-tier data centers is discussed first. Then the architectural design of the processors used in such data centers as well as I/O acceleration methods are detailed. The next section comprises the majority of background material. It explains high performance networking, reviewing Ethernet, InfiniBand and iWARP networks with a concentration on sockets-based protocols. Ethernet is reviewed as a dominant commodity network, and how it differs from the other verbs-based high performance networking technologies. Next, IB networks are discussed along with some proposed methods of increasing the performance of the socket-based IB protocols themselves. Finally, methods of increasing data center performance by leveraging RDMA capabilities and the behaviour of the quality of service mechanisms in IB networks are discussed. iWARP networks are explained in detail, reviewing each layer of the networking stack. Issues surrounding the use of TCP for a RDMA capable network are detailed as well as the networking layer introduced to resolve these issues in iWARP. The last section reviews the latest pertinent research in the area of high-performance message oriented middleware.

2.1 Overview

Data centers can be used to perform a multitude of tasks, including web serving, data storage and administration, business process management, financial management, scientific computing and many more. Each of these tasks requires different software packages and affects the overall setup of the data center. Large mainframe or system complex (sysplex) systems [47] have been used for several years with the system relying on one large central computer. These systems

function using a shared memory mechanism, where multiple processors can read and write to the same physical memory. Sysplex systems are capable of being clustered using networks, but they have proprietary coupling systems that have different needs than common rack mounted clustered data centers. In this thesis we will concentrate on data centers with multiple smaller discrete systems interconnected in a multi-tier cluster. Such cluster based web serving systems are divided into tiers, with each tier performing a portion of the overall client serving process. Data storage systems can be concentrated solely on providing fast data access from hard disks and other storage devices. Business process management and financial management can require a large number of different software support systems, individualized to the requirements of the businesses they serve. High performance computing centers for scientific applications typically require high-speed hardware that is capable of very-high speed, low latency communication, with interconnection software such as the message passing interface (MPI) [120]. Such interconnection software can also be potentially useful in a data center environment where it is required to occasionally do large computations based on incoming data streams or requests.

Ethernet networking hardware is well entrenched in current data centers and networking infrastructure. As with previous generations of proposed Ethernet replacement networks, such as Asynchronous Transfer Mode (ATM) networks [28], there is a growing trend for existing high-speed networking technologies that were previously developed for very high speed local networking to converge with Ethernet networks for deployment to commercial customers. Consequently, Ethernet networks appear to be here to stay, so future networking technologies need to co-exist with Ethernet networks by becoming inter-compatible, or adapt to easily bridge networking technologies. There is currently a push to allow existing high-speed networking stacks to work over Ethernet, such as InfiniBand's RDMAoE [71][108], which will be discussed in detail later.

As this thesis concerns network applications that are primarily commodity commercial applications, this momentum towards the future preference for Ethernet networks cannot be

ignored. Consequently, this thesis will attempt to determine what features of high performance networks (HPNs) can be applied to Ethernet or hybrid networks as well as improving Ethernet.

2.2 Data Center Architecture

This section will discuss the software and hardware elements of data centers, excluding the specifics of their interconnects, as such networks will be discussed in detail in the following section. The software architecture background will concentrate on commercial computing centers, specifically online retailing. The hardware background will focus on elements of the hardware that are not network interfaces.

2.2.1 Data Center Software Architecture

A web serving data center is made up of several tiers, each providing access to separate software systems required for the overall functioning of a data center. The server itself can have up to four tiers: a proxy tier (typically combined with the web tier, for a total of three tiers), a web serving tier (static web pages), an application tier (dynamic content) and a database tier (data storage for dynamic content).

The proxy tier is responsible for directing traffic coming into the domain to the most available systems providing the higher tier services. This allows for the data center to have many systems providing services to a single service tier, allowing for parallelization of the processing requirements of large amounts of incoming traffic. This is what allows websites such as Google, or Amazon to handle large amounts of traffic. The web serving tier provides all of the static content of a page; typically this consists of graphics, and page text that is common, such as a business name or header graphic. The application tier is responsible for dynamic content. A system such as enterprise Java or Personal HomePage Tools (PHP) would typically be the protocol for serving this tier. As requests come into the system, the application tier can provide current results out of a database (by querying the database layer) that can then be displayed on the web page. An example of this is an online retailer, they simply enter all of their stock into a

database, and then only need to create a single web page format for displaying the data. When a customer requests to look at a single item, the application layer processes the request, requesting the data from the database server, then formats it into the required output and sends it back to the client in a browser compatible format. The database tier does not provide any services directly to the client, but instead acts as a support mechanism for the application layer.

While the size of the data center determines the number of machines that are assigned to each tier's functionality, the exact allocation will depend on the nature of the web site itself, with some websites requiring more allocation to application servers, while other websites may not even require the application and database tiers. A typical multi-tier configuration is presented in Figure 2.1 where the firewall and load balancing tiers are typically implemented with routers (the proxy tier), the front end servers provide the web serving tier, and the application and database tiers provide the remaining system functionality.

High performance computing data centers are typically structured as independent nodes which communicate between nodes grouped into clusters. Communication between nodes occurs by means of software such as MPI [120], where required data is passed via network messages. MPI provides for point-to-point communication as well as broadcast and other collective communications. Such systems are typically connected together through as shallow a layer as possible of high-speed switches.

Alternatively, they can be organized into grids with other computing clusters around the world by one of many grid computing software packages like BOINC [5] and the Globus toolkit [30]. Distributed grid computing does not necessarily have to happen using data centers, but can instead be run on common desktop computers spread around the world. Cloud computing has also taken hold as an alternative data center paradigm, using clustered computing systems in a (typically) virtualized environment. Cloud computing allows multiple end users to run applications on the cloud with the cloud providing the services and storage. The first public open-source software for deploying different types of clouds debuted in 2008 [97].

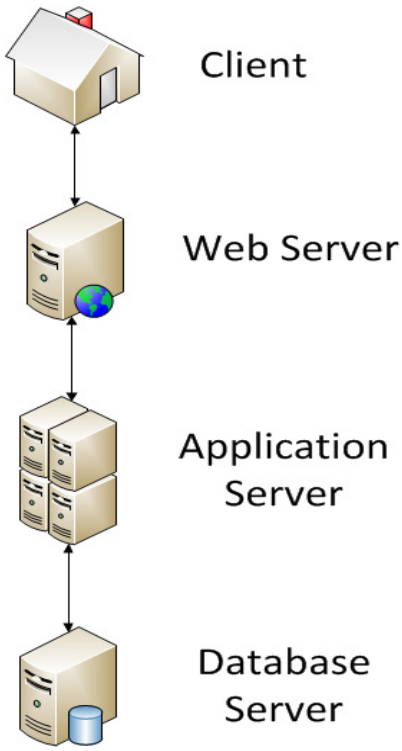


Figure 2.1: Multi-tier web serving data center

This thesis concentrates on the commercial applications of data centers, with additional discussion of high performance computing centers. While grid and cloud computing may be able to benefit from some of the performance improvements proposed in this thesis, they are not the intended target.

2.2.2 Data Center Server Architecture

Data center computing systems are typically comprised of rack mounted server systems. Current server systems typically employ more than a single CPU chip, with each CPU chip having more than a single CPU core. Many of such systems are then connected together in a rack system, with a high performance network. Most web servers utilize an Ethernet connection (gigabit or 10-gigabit Ethernet). It is also feasible, although more expensive, for them to function over higher performance networks such as InfiniBand or iWARP within their intranet. Current pressure from the high performance networking community is attempting to converge such high performance networks with Ethernet networks, making it very likely that they will appear in the

near future in commercial data centers. Efforts to extend Ethernet to add advanced capabilities are also underway, such as the proposal of Converged Enhanced Ethernet (CEE) [20].

The individual system architecture for each node in the interconnected cluster is also of interest. As each node is required to handle as many clients as possible, this requires a method of running several copies of the software responsible for serving clients. This can be accomplished in multiple ways, either by using independent processes that are running over different sockets, or by running several threads of the same application.

Multi-threading allows for multiple instances of code to be processed on a system at the same time, given adequate computational resources. Modern systems typically have more than a single processing core, and as such can take advantage of multi-threading, where multiple execution threads are run at the same time. This can be done using a single enhanced processor using simultaneous multithreading (SMT) [117], multiple processing cores on a single die via chip multi-processors (CMP) [40], or multiple enhanced processors on a single die using chip multi-threading (CMT) [107]. This provides the advantage of allowing more clients to be served from the system than if they all waited for a single processor.

Multi-threading can allow for more threads to be created on a system than the system could simultaneously execute. Web serving can be a relatively light computational process, and as such computer architectures that support many simultaneous threads have been developed. The simultaneous-multithreading approach from Intel, Hyperthreading [116], was introduced in its Pentium 4/Xeon [57] era processors, but then phased out for their new Core architecture, only being present in the core i7 [58] processors. This was done in response to the development of other CPUs designed for massive simultaneous multithreading products such as Sun Microsystems' T1 [65], T2 [101], T3 [87] and Oracle's T4 [102] processors, capable of executing 32 threads for the T1, 64 threads for the T2, 128 for the T3 and 64 for the T4 simultaneously. The T4 processor operates fewer threads with higher performance. The T2, T3 and T4 processors have built in multi-threaded support for 10Gb Ethernet [101]. This demonstrates the trend

towards increasing numbers of sophisticated processor cores on individual nodes, increasing the overall load that a single node can place on a network, as well as the load incurred by the network interface controller. The systems used for testing in this thesis have processors capable of SMT/CMT and typically run more than one thread per node simultaneously.

Processors in a data center can become a source of resource contention when tasked with the job of processing the TCP/IP stack overhead for a fast network connection. Therefore, processing performance can be enhanced by the use of techniques to offload such tasks from the CPU to other devices that can more easily handle the transfer. Intel's I/O Acceleration Technology (I/OAT) [59] has been proposed to help alleviate total CPU load on such server systems, and results [118] indicate that such technology can decrease CPU load by as much as 38%, which results in an improvement of 12% to system throughput. I/OAT functions by allowing for TCP/IP packet headers to be split from the packet data, allowing for the packet header to be compiled at the same time that the packets payload is assembled. In addition it allows for a DMA transfer to be set up between the network interface and memory, allowing for an asynchronous transfer of the data, freeing the processor to perform other tasks. The use of zero-copy protocols (transferring directly to another systems memory without interrupting either CPU, or incurring extra memory copies) is essential in reducing CPU load, as the majority of the time spent by a CPU on networking process tasks is spent waiting for memory accesses [29]. This allows the system to make progress in processing packets while allowing the payload for said packets to be fetched in parallel with other packet processing tasks instead of having to wait for the memory operations to complete before proceeding. It has been shown that multi-core computers can help to increase the performance of high performance networks [112].

The development of 64-bit many core architectures, while helping expand the capacity of server systems also aggravates the client load issue, as each individual core on a system is a source of more requests, potentially becoming as large a consumer of traffic as a whole system is

today [78]. This will help to drive the future development of faster networking technologies as they are required in order to keep up with emerging processor designs.

2.3 High-Performance Networking

High performance networks are a very important part of a modern data center. The capabilities of individual systems within the cluster necessitates the fast and effective delivery of large amounts of data. In applications such as database accesses, latency is of great concern. However, in the web server layer where graphics or other embedded media may be sent to clients, bandwidth out of said systems is very important. Tasks that require any sort of co-operation between the nodes in the cluster also require low latency and high bandwidth. Co-operating nodes are prevalent in scientific high performance computing. Sometimes the node counts for scientific computing are very large and the complexity of having all of the nodes work on a single related task can require a significant amount of interconnection between the nodes.

Several different network technologies have emerged to provide very high-speed networking. Three of the most state-of-the-art and promising architectures are covered here. It should be noted that other high-speed networks such as Quadrics QsNet II [14] (Quadrics is now defunct), and Myrinet [77] also exist, but they are less popular than the technologies presented here for use in data centers. Myricom now mostly sells high-speed Ethernet adapters. Myricom's proprietary networking technology (Myrinet Express) capable NICs are still available. Myrinet networks can function in a Myrinet Express over Ethernet (MXoE) mode, as well as Ethernet emulation (IPoMX) mode.

2.3.1 Ethernet Networks

Gigabit Ethernet networks adhere to the Ethernet standard originally created in 1975 [73][105], and use a backward compatible frame format that allows them to communicate with previous generations of NICs. The fastest current implementation of Ethernet operates at a theoretical maximum of 10 Gb/s. Modern Ethernet implementations use a switched, full duplex

(can send and receive at the same time) network architecture. Ethernet networks typically use the TCP/IP networking protocol for communication. Although several proposals have been made for QoS provisioning for Ethernet networks, so far no widespread adoption of any such QoS provisions have been deployed for Ethernet TCP/IP networks.

Up until the emergence of 10 Gb/s Ethernet, the 10Mb/100Mb/1000Mb Ethernet technologies were completely backward compatible, but the 10 Gb/s Ethernet implementations required changes to the Ethernet frame and the interconnect wire type, moving from a cat-5 twisted pair to a CX-4 or fiber optic line.

The structure of key transmission protocol stacks most typically used in Ethernet networks, such as TCP/IP and UDP protocols must be examined before investigating the function of any transmission hardware. TCP and UDP are both protocols layered on top of the IP layer. UDP and TCP are protocols that are used at a middle layer, and reside between the upper layer, a socket API, or socket interface from the operating system, and the lower layer of the network device driver. TCP provides a flow controlled connection based service, while UDP provides for an unconnected service, with no flow control. An example of connected service is the telephone network, one must specifically connect to the intended receiver using a unique address, their telephone number. Once the receiver picks up the phone, a connection is established and both parties can send and receive data over it until one of the parties hangs up. This is advantageous in that the both parties are guaranteed to receive the transmitted data, barring any catastrophic failure, while the two parties are connected. A TCP connection works in much the same way, except that the amount of data that can be sent over the connection can sometimes exceed the capacity of the connecting medium, unlike the telephone, where a maximum data rate is assured as it carries only one type of load, a voice signal.

Due to TCP's ability to flood the available bandwidth, TCP provides for flow control, which is a mechanism of regulating the speed at which data is sent to ensure that the connection is not completely flooded with data. A saturated network is problematic as it leads to large delays as

individual data elements, or packets, can be dropped by the network and are dropped much more frequently in the case of network saturation. This also helps to ensure relative fairness with others over a shared connective medium, preventing any one participant from monopolizing the network resources. TCP also provides for errors and dropped packets. TCP operates over a window of packets, allowing only those packets that belong to the current window to begin transmission. This sliding window advances when a packet at the beginning of the window is received correctly by the receiver and the receiver acknowledges its arrival to the sender. With a send and acknowledge system, it can be assured that packets are received correctly and the sender can resend packets that have been dropped or arrived in error. There are a variety of flow control and sliding window schemes that have been proposed and are in use, however, we will not go into great detail on them here, as the standard schemes used for modern systems will not be modified in any way, and therefore their behaviour will remain relatively constant throughout our proposed research. TCP is described as a stream-oriented protocol as the data delivered to it can come from an application in any size, it does not need to be delivered in already divided 'chunks', therefore the behaviour of applications using TCP can be seen as utilizing a data stream from one end of the connection to the other.

UDP utilizes a different approach to data transmission, in that it does not establish a connection before sending data. It is similar to the postal system in that one can send data to recipients without notice, and as long as they are checking their mailbox, the letter will arrive most of the time. However, there is a possibility that the package will be lost and not received. UDP has a similar sending/reception policy and delivery is not guaranteed. Therefore, UDP is referred to as a byte-oriented protocol, as the applications using it must deliver small 'chunks' of data that are pre-divided from any larger message that needs to be sent by the application. UDP does not provide flow control, and so can also flood networks. However, UDP has several advantages, firstly, since a connection does not need to be maintained, the information required to be stored concerning a connection (who is connected, what the status of the connection is, etc.) is

not needed, which makes it possible to send to many recipients at the same time, without using a huge number of local resources. In terms of our analogy, sending packages to many people at the same time is easily accomplishable, where a telephone conversation with hundreds or thousands of others requires a huge number of distinct telephone lines, particularly since in the TCP implementation, conference calling is not available.

The performance of Ethernet lags that of InfiniBand and for some operations, iWARP. Some of this difference can be accounted for in the advanced RDMA capabilities of InfiniBand and iWARP networks, which will be explained in their respective sections in this chapter. However, it is worth noting that zero-copy methods of transferring data over TCP over Ethernet have been proposed [18], and Ethernet remains the de-facto networking standard to this day. The standard does not include RDMA capabilities, which can significantly reduce the computational load experienced by a processor when sending large volumes of data. Another approach to lessening the CPU load when sending large volumes of data is to offload the processing load of the TCP/IP stack to a external device, typically called a "TCP offload engine" (TOE) [27]. The capabilities of TCP offload engines can vary and is not required in the Ethernet specifications, and there is some debate as to their efficacy [95]. Proponents of TCP offload technologies point to ever increasing network bandwidths and processors that are unable to keep up [75], and insist that offloading TCP/IP stack processing to a separate logic chip is necessary. Opponents of TCP offload cite the emergence of multi-core processor systems that can handle the additional load by dedicating a single core to TCP/IP processing [95], and avoiding the software alterations necessary to support TOEs. Due to its prevalence amongst the Internet and existing network deployments, as well as its low cost, many competing network technologies must support Ethernet compatible traffic, or at least work within the sockets-based connection paradigm such that existing programs written for Ethernet networks are compatible.

A recent proposal for the development of a new set of standards referred to as Converged Enhanced Ethernet (CEE) has opened up issues revolving around providing advanced features

over Ethernet networks. CEE is a combined number of standards from the IEEE and IETF dealing with priority based flow control (IEEE 802.1-Qbb [49]), congestions notification techniques (IEEE 802.1Qau [50]), enhanced transmission selection (IEEE 802.1-Qaz [51]), discovery and shortest path routing protocols for neighbouring networks (IEEE 802.1AB [52]), Fibre channel over Ethernet (INCITS T11 FCoE [53]), and provisioning for the transparent interconnection of lots of links (IETF TRILL [60]). The CEE enhancements allow for quality of service in a local network context and provide for Ethernet to perform as a transport mechanism for storage based traffic (using FCoE). Some industry vendors are also proposing to include RDMA functionality over CEE, as the CEE enhancements add the necessary service classes and reliability to Ethernet, making it an appropriate platform for existing RDMA technologies to use, such as InfiniBand over Ethernet (IBoE) [71].

2.3.2 InfiniBand Networks

The InfiniBand architecture [54] specifies a communication infrastructure for connecting multiple processing nodes with I/O nodes and devices, which can be seen in Figure 2.2. The InfiniBand network provides a converged system communication and I/O communication network, with hardware support providing for reliability to create a fault-free network. It is a switched network architecture that provides for multiple usable network paths from point to point, allowing for flexibility in routing traffic. InfiniBand is a state of the art high-speed low latency network, with the fastest technology; a fourteen data rate, 12X InfiniBand connection operating at a signaling speed of 168.75 Gb/s, which when used with the 66/64 bit encoding scheme gives a transmission rate of 163.64 Gb/s.

InfiniBand [56] networks function by creating queue pairs between any communicating systems. Each system contains at least one Host Channel Adapter (HCA), which can communicate with any other HCA or Target Channel Adapter (TCA). TCAs are I/O devices attached directly to the communication fabric. For example, a server system uses its HCA to communicate directly with a TCA attached to a RAID.

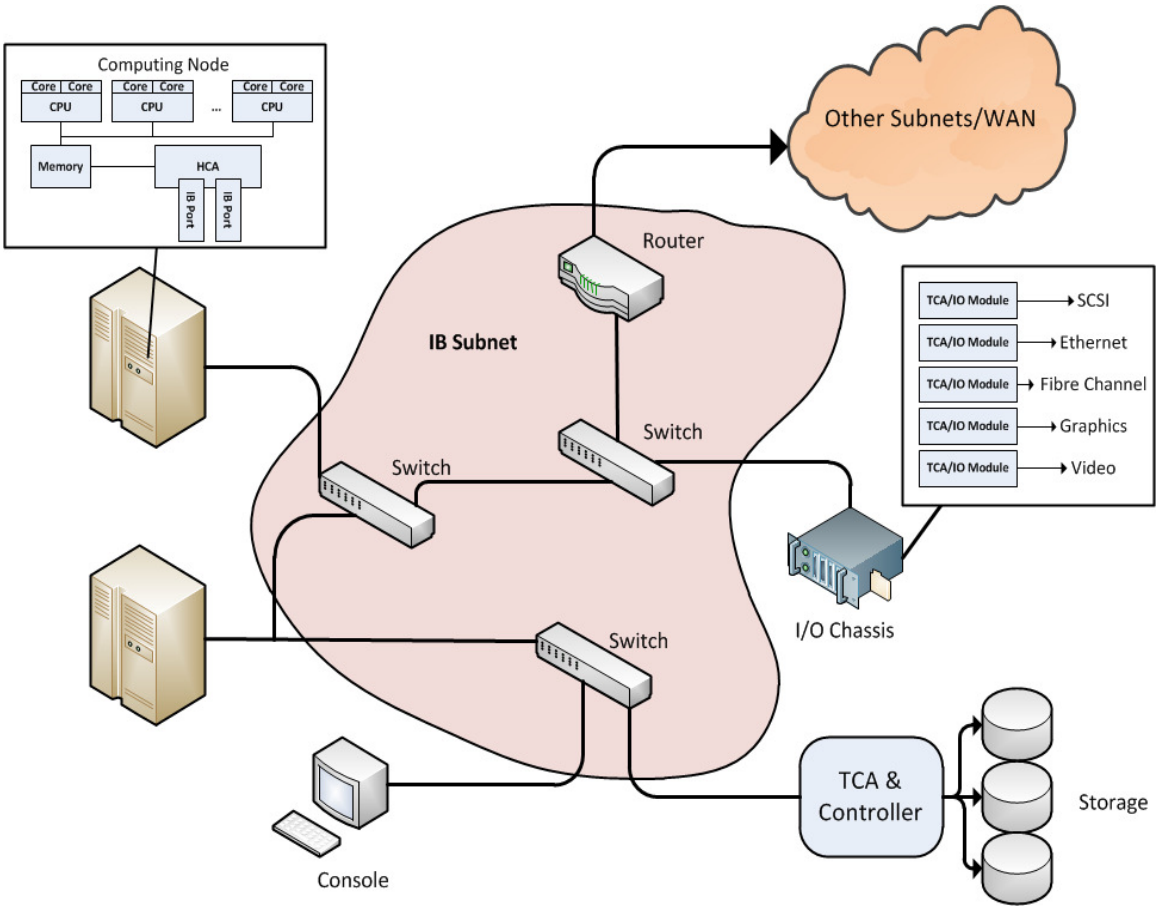


Figure 2.2: Overview of an InfiniBand system area network, adapted from [54]

InfiniBand operates over a defined packet type, and requires that software implement a defined set of functions, called verbs. When a connection is opened between two communicating processes, each creates a local work queue, a send queue, a receive queue and a completion queue. The two work queues form a queue pair. When an application wishes to send data to a receiving system, it simply posts the data it wishes to send to the local work queue, and the InfiniBand HCA takes care of the data transmission. It is possible for applications to open up multiple queue pairs for transmission, and as all of the processes on the system each can have their own work queues, they do not need to share a common transmission queue for NIC traffic. It is possible to share a receive queue (SRQ) [104] between different work queue structures, to help conserve resources when such a system is required. When operations complete, their completion is acknowledged by an entry in the completion queue. For one sided operations

(RDMA) only the initiator needs to post an event into the work queue, the sender by definition is not aware that it is going to receive a request prior to its reception. An example of the InfiniBand communication stack can be seen in Figure 2.3.

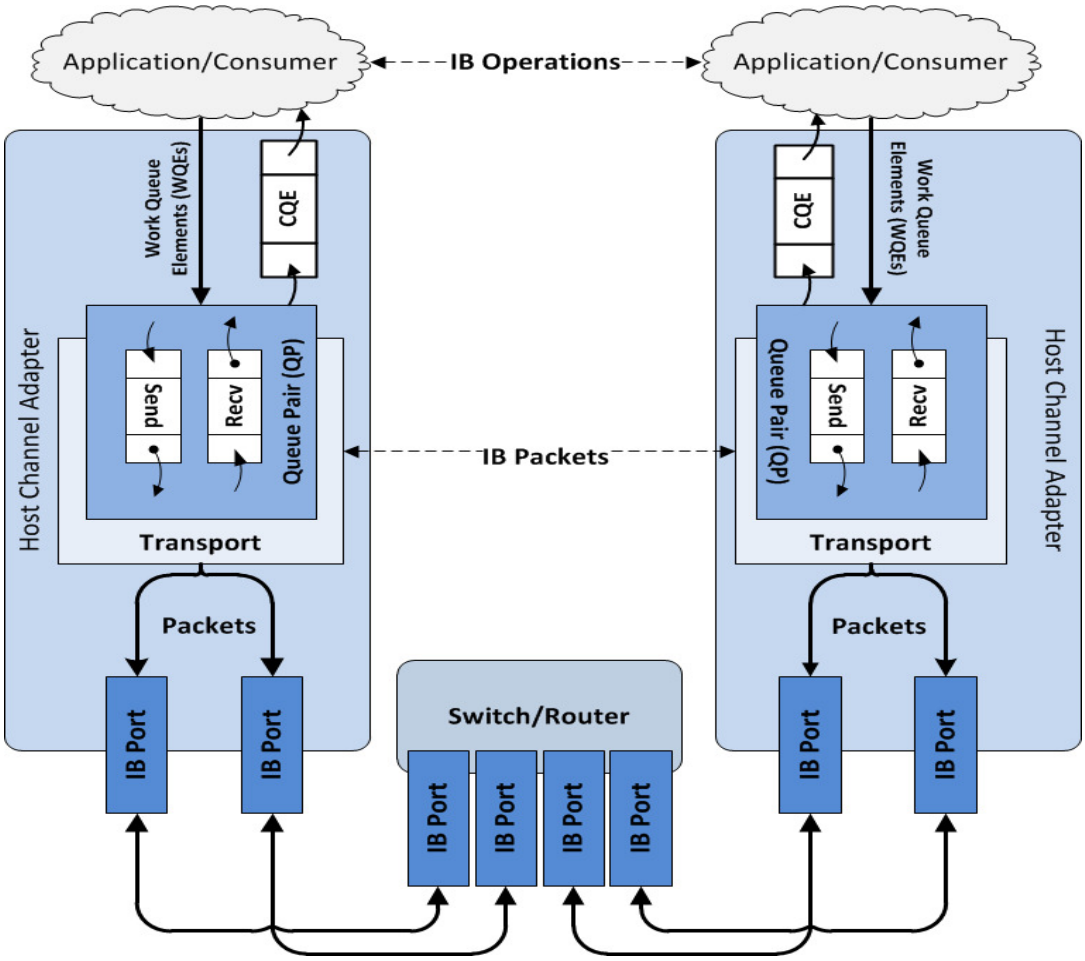


Figure 2.3: InfiniBand communication stack, adapted from [54]

InfiniBand Transport Protocols

InfiniBand provides four transport methods, Reliable Connection (RC), Unreliable Connection (UC), Reliable Datagram (RD) and Unreliable Datagram (UD). Two of the transports for Ethernet networks that were previously discussed, UDP and TCP, are representative of two of four possible transport mechanisms. UDP is an unreliable datagram (UD) protocol, while TCP is a reliable connection (RC). Although the implementation of the methods for providing such

transports varies between the two technologies, the basic concept remains identical. In addition to these two transports, we can also utilize reliable datagrams (RD), or unreliable connections (UC). However, such transports are not nearly as popular as the existing UD and RD mechanisms. Reliable datagrams can be best thought of as an acknowledged datagram protocol, which could be of great use, but very few implementations of hardware capable of providing RD support exist. Unreliable connections can best be conceived as an unacknowledged connection based protocol, which can be useful in some scenarios, but the reliability of a connection is a major attracting factor in their use.

It has been observed that the overhead of maintaining a queue pair for individual process communications can be excessive on large-scale systems [67]. Attempts to reduce the overhead incurred by having many communicating processes between nodes has been explored using shared queues over RC by Shipman et al. [104]. The eXtended Reliable Connection (XRC) transport service was introduced [55] to address this issue as well. By using XRC it is possible to maintain a connection per node to each other communicating node, and use shared queues for transmission between them. This eliminates the need for a communicating queue pair per communicating process and reduces overall overhead. XRC has also been used to enhance MPI implementations [68].

Memory and Channel Semantics

One of the most important features of InfiniBand is its support for RDMA operations. RDMA allows a system to remotely access other systems' memory for either write or read operations. RDMA permits the system to offload the processing load imposed by creating network traffic, and bypasses the typically required interaction with the kernel in handling network traffic. RDMA allows the processor to set up a data transfer and the DMA controller handles the delivery of data from memory to the HCA. This prevents a memory request from the processor to

memory and then another from the processor to the NIC as would typically be done in a non-RDMA system. This can both speed data transmission as well as reduce CPU load on the system.

RDMA requires that memory regions be registered for use with the HCA, such that they can write to the correct regions of memory, and it also requires that the network be aware of memory semantics as opposed to traditional channel semantics. Channel semantics are used in IB send/receive operations, where the sent data is received into a buffer allocated for that specific connection. When the receiving application posts a receive request and the data is received, the application can copy the data from the channel buffer to its local application buffer. This does not require the sender to be aware of the memory registration and allocation of the receiving system, as the HCA manages the local buffer storage in temporary buffers until the application is ready to receive data.

Memory semantics are used for RDMA operations. Memory semantics allow for the reference to memory regions in which incoming data is to be read from or written to. A memory region consists of a virtual memory space that has its physical addresses and access rights registered with an HCA. Memory regions must be pinned using a call to the operating system, and the memory translation must occur for the HCA to determine the correct physical addresses for the associated virtual addresses. Therefore, RDMA requires more setup time when first connecting than a channel based method, but the benefits of RDMA outweigh the setup costs shortly after registration. Access rights control whether the memory can be read from or written to, both locally and remotely. When initialized, the HCA must be informed of the virtual address of the region, its access rights, the length of the region, and have the appropriate tables for conversion of the virtual address to the physical. It is possible for some implementations to support a directly physically addressed memory location, but this requires that the physical memory region be contiguous which can lead to difficulty in finding an appropriate region for addressing, however, it eliminates the virtual to physical mapping. Keys (local or remote) are

passed along with the request that when combined with a virtual address indicate whether a request is authorized or not.

Send/Receive channel semantics are two-sided operations, meaning that both the source and the receiver need to post relevant work requests in order for the data to be transmitted successfully, while RDMA operations are one sided, where only the requestor needs to post a work request.

SDP and IPoIB Protocol Stacks

The concept of high performance sockets over the Virtual Interface Architecture (VIA) architecture [25] (predecessor to InfiniBand) was first introduced by Shah et al. [100]. The InfiniBand software fabric that implements the required verb functions for sockets is standardized and published by the OpenFabrics alliance [85]. There are also additional protocols available in the open fabrics distribution that allow applications to use standard sockets based communication calls to the InfiniBand fabric, which are translated automatically into native IB verbs. Two of these protocols are the IP over IB (IPoIB) [63] protocol and the Sockets Direct Protocol (SDP) [88]. IPoIB functions like a typical IP sockets based communication mechanism, using the TCP/IP software stack. Examining Figure 2.4, we can see that once an application makes a request to send data over the HCA, the data is passed to the kernel via the socket layer. From there it is passed to the TCP/IP stack, which uses the InfiniBand address resolution protocol (IBARP) to determine the destination. It is then passed to the IPoIB driver which enables the HCA to properly encapsulate the packet inside of an IB header/footer for proper transmission over the IB network, which then utilizes the IB HCA driver to translate the request into verbs, and then put the data out on the wire.

SDP provides socket-based functionality like IPoIB, but uses RDMA operations for transmission. It is designed such that the traffic does not need to call the TCP/IP stacks, but instead transition directly to the SDP stack and to the HCA, as shown in Figure 2.4.

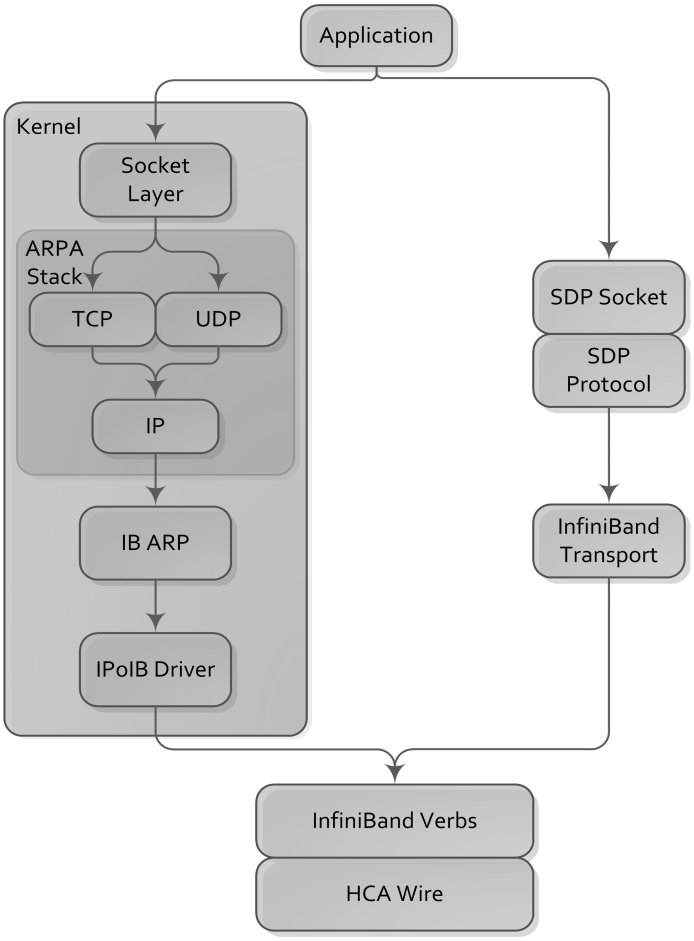


Figure 2.4: InfiniBand SDP/IPoIB software architecture

Bypassing the kernel allows for a reduction in overall system calls as well as operating system bypass functionality. However, SDP requires that RDMA capability be present and that a connected service be used. Alternatively, IPoIB can operate over both a connected or datagram service, but it does not take advantage of RDMA functionality. SDP also requires that certain control and advertisement messages be sent in addition to the data payload. SinkAvail and SrcAvail messages are required to advertise the buffers that can be sent to or read from using RDMA operations. In addition, other commands to revoke these advertisements as well as to announce the completion of RDMA write or read operations to the receiving or transferring system are required, creating a small amount of management overhead when using SDP.

SDP is capable of operating in two main modes, zero-copy and buffered copy. A zero-copy transfer bypasses the kernel and writes data directly to the application buffers, while a buffered

copy first writes the incoming data to a local buffer and then copies it over into the application buffer after reception. Due to the costs of memory registration for a zero-copy transfer, buffered copy is generally superior in performance for small messages, while zero-copy dominates for large message transfers.

InfiniBand Quality of Service

InfiniBand technology also provides the ability to offer QoS over the network. This quality of service mechanism is available when the network runs in its native IB mode, as well as IPoIB and SDP. QoS works by using multiple virtual lanes for traffic. Each virtual lane is assigned a service level through an SL2VL mapping table [96], with service levels determining the ratio of transmission credits that the virtual lane is assigned at each credit assignment interval. An example is shown in Figure 2.5. InfiniBand QoS provides two methods of QoS, a non-blocking weighted queue mechanism, and a priority queue (blocking) mechanism. In the blocking queue case, weights are assigned to service levels in a high priority queue, as well as weights to service levels in a low priority queue. When all of the packets in the high-priority queue are served, or the high priority queue reaches a given threshold of sent packets, the low priority queue is permitted to send data. In the weighted queue case, all VLs are permitted to send data in a round robin fashion, as long as they have enough credits.

Service level assignments are determined by a master table sent by the subnet manager via management datagram packets (MADs). Service levels can be assigned by transport protocol, port ranges, addresses, and even more advanced protocol, port, address combinations. Service levels can also be used to determine the specific routing path for flows. Consequently, QoS in InfiniBand networks is extremely flexible and versatile. However, due to the VL2SL tables being broadcast from a single system, the subnet manager, the frequency with which such tables are updated must be balanced with the overhead of such a broadcast operation, therefore, dynamic

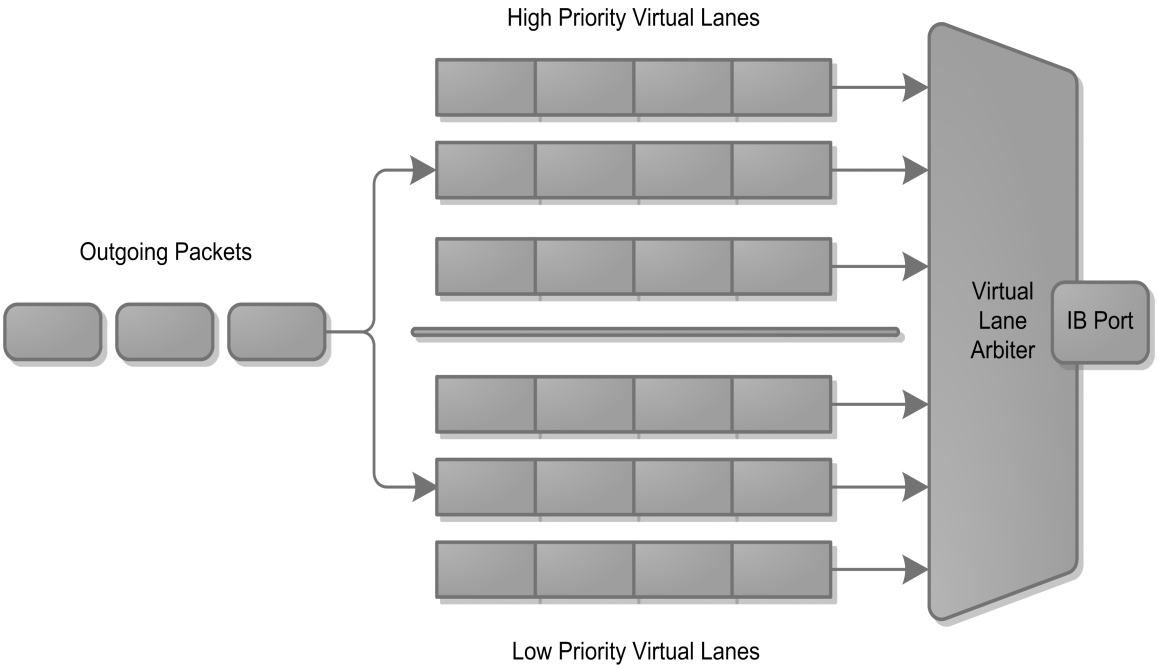


Figure 2.5: QoS arbitration

QoS assignments should be determined with care, and individual machines are not capable of having independent VL2SL mappings.

2.3.3 iWARP Networks

iWARP Ethernet [93] networks are built upon the Ethernet standard. iWARP provides for a zero-copy transfer of data via RDMA. Zero-copy allows for much higher throughputs to be achieved than a traditional buffered-copy implementation. iWARP specifications provide for a TCP offload engine which can alleviate the need for the systems CPU to execute the required TCP/IP stack instructions for creating and sending network packets. This is important in a high-speed network environment, especially when the system is performing a CPU intensive task, as the overhead introduced by TCP/IP stack processing can be significant. iWARP also utilizes OS bypass techniques in order to minimize performance loss due to context switching between the application issuing commands to the network interface card and the operating system. When commands are issued to the NIC in an Ethernet environment, the application contacts the operating system, which then switches the operating context to code designed to relay the correct commands to the NIC, then once the command is completed, the processors operation is turned

back over to the application. Using iWARP, the application can issue commands directly to the NIC without interacting with the operating system, preventing any context switch from happening. As this avoids several costly operations, such as swapping out the contents of registers to memory, OS bypass eliminates a significant amount of overhead from the command issuing process.

iWARP implements some, but not all the features provided by InfiniBand networks over the existing Ethernet network architectures. It operates over a set of defined verbs, just like InfiniBand, and is supported by the OpenFabrics Alliance [85] software distribution, allowing for translation between the verbs to occur such that verbs based programs can run over either fabric (with some small exceptions). The use of RDMA can also be extended to existing TCP sockets-based programs using SDP. iWARP utilizes an almost identical queue pair channel-based network transport as well as a memory semantic based one to that used by IB over its network architecture. Although very similar in their high-level implementation of RDMA, iWARP is only specified to work over connection-based Ethernet, unlike IB which defines datagram based transports as well as connection-based ones over its own fabric. The iWARP stack is illustrated in Figure 2.6.

The Verbs layer is the direct interface from the applications to the stack, very similar to that of IB verbs. The Remote Direct Memory Access Protocol (RDMAP) layer [94] is responsible for performing RDMA operations as well as indicating the method of the operation either as a send/receive or an RDMA based write/read. The Direct Data Placement (DDP) [99] layer is responsible for moving data from buffers to the RNIC, either from local buffers in the case of a non-RDMA send/receive or from user buffers for an RDMA operation, it is also responsible for reassembly of in-order upper layer protocol messages, and assuring reliable delivery. Security can be provided at the DDP layer by utilizing IPSec [64], and control over the registered memory regions remains at the ULP level at all times. Steering Tags (STags) point to memory regions

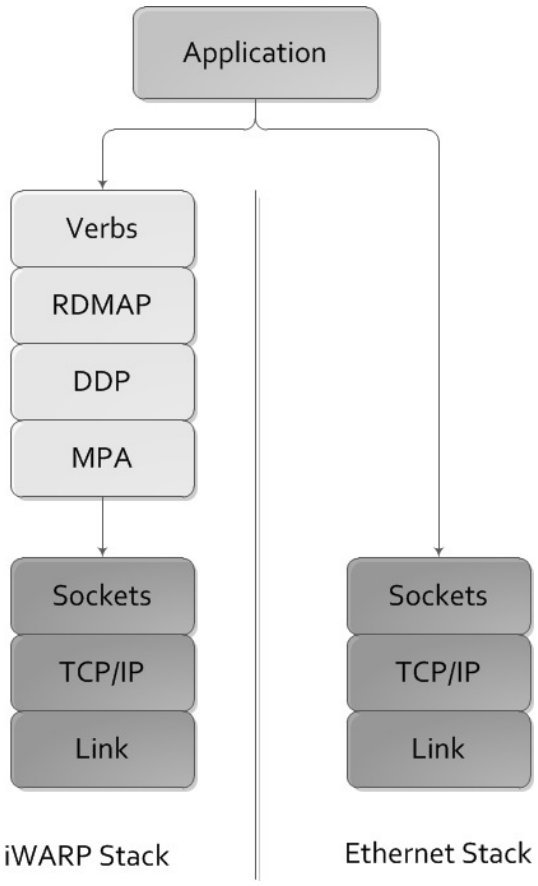


Figure 2.6: iWARP stack vs. Ethernet stack

when combined with a STag index, providing similar security to that of local and remote keys for InfiniBand as discussed in Section 2.3.2.

One of the major differences between iWARP and IB is the need for a different method of marking packets. In order to guarantee that packet boundaries are distinguishable the Marker PDU aligned (MPA) [21] layer specifically marks messages at equal intervals determined by the TCP sequence number, as shown in Figure 2.7. Each of these markers points to the correct remote DDP header portion of the message such that the remote machine can always reassemble the whole message such that the DDP layer may function appropriately. It removes the markers at the receiver's side to provide the entire original message in its entirety to the upper layer.

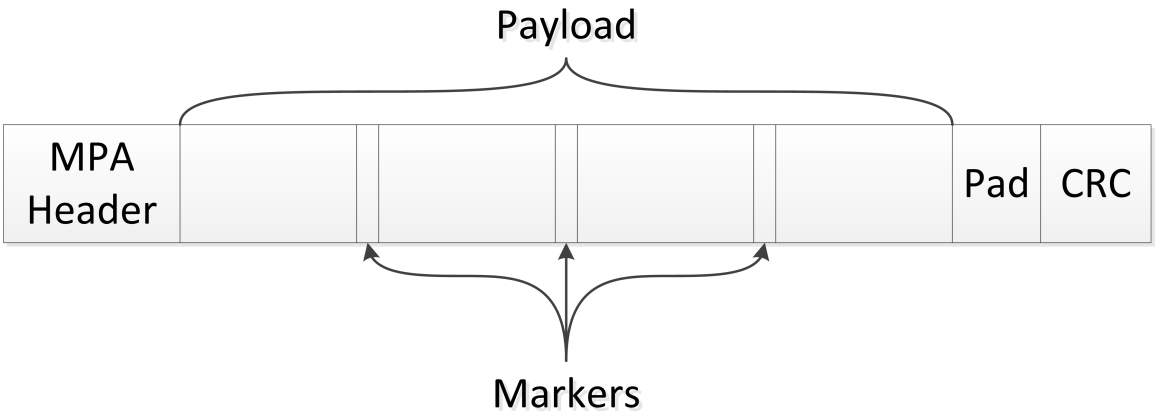


Figure 2.7: MPA marking

This mechanism is necessary to allow for middle box fragmentation, which occurs when network devices like switches split packets or combine packets and those packets then arrive out of order. Without this provisioning messages that exceed the size of a single packet in length would not be able to be placed into memory using RDMA. Therefore packet marking is required due to the use of Ethernet as the underlying network and the restrictions that it imposes due to its architecture, which was not originally designed as an RDMA capable network. iWARP does not allow direct access to the DDP layer, instead abstracting its use through RDMAP layer, which services calls made from the verbs layer. iWARP also varies from IB in that it does not support remote atomic operations.

2.4 Message Oriented Middleware

Message-oriented middleware spans a group of networking middleware approaches. Some middleware focuses on store and forward, publish/subscribe messaging for high performance networks like the Advanced Messaging Queuing Protocol [4], and have been adapted for high speed networks like InfiniBand [110].

We are primarily concerned with high performance point-to-point communications in data centers. The most common middleware to accomplish such a task is the Message Passing Interface (MPI) specification [120][76]. MPI has many available implementations for systems,

the most popular in the academic community being the MPICH-2 [6] implementation from Argonne National Labs, the MVAPICH [81] implementation from Ohio State University and OpenMPI [86].

An interesting MPI implementation is MVAPICH-Aptus [67] that utilizes a combination of reliable connections and unreliable datagram communication to provide a scalable MPI design. By adding in reliability mechanisms to ensure that all data is delivered correctly over the datagram communication channels, the functioning of MPI can be maintained when communicating with large numbers of peer processes while reducing the overhead required by not necessitating a connection for each and every communicator. Other approaches have been made to attempt to reduce the overhead of connections in MPI systems as well. The MVAPICH implementation was extended [68] to take advantage of the eXtended Reliable Connection (XRC) mode of InfiniBand, finding that memory utilization can be significantly reduced for aggressive performance MPI implementations by using XRC.

MPI has multiple communication models, but we are primarily concerned with the point-to-point communication methods. MPI implementations are typically built upon multiple different channels, which allow them to operate over different underlying networks. MPI implementations are available for most networks, and many provide support for OpenFabrics verbs-based networks, as well as Ethernet networks.

MPI operates in the same way as its namesake, by passing messages between individual communicators in a group. Each communicator is assigned a unique rank, and communication occurs between individual communicators to one or more recipients. Several control messages are also used to manage communication between processes. Communication methods also vary according to the size of the message. There are typically two different communication protocols based on message size, the Eager protocol for small messages and the Rendezvous protocol for larger messages. The exact threshold for switching from Eager to Rendezvous protocol is implementation specific. For the Eager protocol, messages are sent to the target without any

notice, like a ready to send (RTS) or clear to send (CTS) message exchange. For verbs-based queue pair networks, pre-allocated buffers are posted to a receive queue in order to receive small messages. For larger message sizes, the Rendezvous protocol is useful. The Rendezvous protocol uses an RTS/CTS behaviour to ensure that a buffer is ready at the target side before the sender initiates data transfer. This is helpful for large message sizes as it can avoid having to drop messages or create additional copies, by having memory registration performed before reception of the data. This also works well for an RDMA write operation as the location that the data is to be written to in memory can be transmitted back to the sender, and once it arrives at the target can be placed directly into the desired memory location.

Many network performance evaluations use MPI as an application middleware for performance testing with accepted benchmarking suites such as the NAS parallel benchmarks [80]. Evaluations have been performed on InfiniBand networks [66][112], and iWARP, InfiniBand and Myrinet networks [90]. Other work has focused on increasing the performance of MPI applications through improvement of the MPI middleware, like speculative Rendezvous protocols proposed in [91].

MPI also includes specifications for a remote memory access model (RMA), in which the communication model used by applications is more like a memory write/read interface than a traditional send/receive network data transfer model. Although RMA was not well received by the application developer community as it was defined in the MPI version 2.2 specifications [76], its continued development for MPI version 3 brings the possibility that such a communication model may be more accepted in the future. Consequently, RDMA capable network designs can fit well into the RMA model, and such models may become more prevalent in the future as application developers utilize additional messaging middleware communication models.

Other types of messaging middleware such as IBM's Websphere MQ [48] and JMS [111] exist as a publish-subscribe type model of communication. An open standard for such messaging exists, AMQP [4][83], and some implementations of the proposed standard are now available,

ZeroMQ [26] and from Subramoni et al. [70][110]. However, message-oriented middleware for HPC applications is the main focus of middleware in this thesis.

2.5 Summary

This Chapter has reviewed data center architectures and software as well as examining the currently available high speed networking technologies. It reviewed key issues relating to data center hardware, including CPU considerations. An overview of RDMA-capable networks was provided and both InfiniBand and iWARP networks were reviewed in detail. Methods of utilizing RDMA-based networks through a network socket interface were discussed. Finally, a review of message-oriented middleware was provided, concentrating on its application to HPC.

Chapter 3 Advanced Networking Architecture Features

As the demands for ever-increasing amounts of Internet bandwidth continue to mount, the need for server systems grows as well. Such servers are costly to purchase and maintain, and as such, increases in performance need to be accompanied by increases in overall efficiency. The efficiency of current networking approaches is not ideal and the existing inefficiencies will continue to needlessly waste increasing amounts of resources as the capacity of the system increases. One of these inefficiencies is the inability to easily handle bandwidth distribution amongst services. Existing high-speed networks have such capabilities but interfacing such technology with commodity Ethernet solutions has been difficult. Recent advances have made such an interface much easier to accomplish and the investigation of such features to determine their performance advantage on actual hardware is essential for implementing future systems that take advantage of such features for increased performance. Such approaches could significantly benefit the local network efficiency of data center systems. However, the problems relating to modern data center capacity are not limited to the local network. The wide area connection is just as important, and increasing the efficiency and performance of the wide area networking performance on server systems is critical if future server system costs are to be kept manageable.

Many new advanced network features have been recently introduced to high speed networking technologies such as InfiniBand. We will explore the Quality of Service (QoS) abilities, and datagram segmentation offloading techniques of InfiniBand networks in this section. Each of the techniques provide for the creation of network architectures that can offer enhanced functionality and performance. As the behaviour of such abilities are unknown in a real-world implementation, it must first be quantified and evaluated in order to propose environments for their use or to propose new network configurations to take advantage of them. QoS features are useful for load balancing and can also be used for multi-path communications within a local network

environment. In addition, new features such as segmentation offloading for use with IPoIB offer the opportunity to use previously under performing transmission methods (IB datagrams) to enhance overall performance.

3.1 Related Work

The performance of the ConnectX architecture, operating with MPI [76] and IB verbs has been analyzed in detail by Sur et al. [112]. The performance of SDP relative to native IB verbs and IPoIB has been documented by Balaji et al. [12]. They found that SDP was capable of outperforming IPoIB by 2.7 times on the previous generation of IB cards from Mellanox.

The benefits to utilizing a zero-copy (as opposed to buffered-copy) SDP protocol have been explored by Balaji et al. [8] as well as Goldenberg et al. [32]. Asynchronous zero-copy SDP sockets offload memory transfers from the CPU to a direct DMA-IB data transfer, allowing for much higher data transmission speeds and reduced CPU overhead. Goldenberg's implementation of zero-copy sockets [32] saw an average improvement of 29% over traditional buffered-copy methods, while Balaji's approach [8] saw improvements of 35%-200% depending on the protocol's application. Balaji et al. [13] have also explored the potential memory bottlenecks that exist in socket-based and RDMA-based communications for 10-Gigabit Ethernet networks, including a comparison to RDMA-based InfiniBand networks.

The QoS performance of native IB verb traffic over IB networks has been explored in [3] on a simulation basis. In addition, Alfaro et al. [2] have explored the impact of message sizes on the ability of IB networks to provide guaranteed QoS to applications. Reinemo et al. [96] have published a survey covering the QoS capabilities of IB networks while comparing them through simulations to the QoS provisions in many other modern network architectures. Subramoni et al. [109] have adapted an MPI implementation to take advantage of multiple virtual lanes over IB networks. They found this to be effective in terms of bandwidth and latency, as well as improving the performance of applications and collective operations over MPI.

High performance sockets for high-speed networking (VIA) architectures were first proposed by Shah et al. [100]. The performance of InfiniBand networks is well understood in most areas, particularly concerning RC modes of operation. The wide area network performance of InfiniBand networks has been examined in [16][89][121]. The performance of the 40Gbps generation of IB hardware, QDR IB technology has been studied by Koop et al. [66] using MPI. They also examined the advantages of moving to a faster PCI Express 2.0 bus, and found that DDR IB HCAs can see a 25% performance gain simply by moving to the faster bus technology. Their evaluation focused on MPI performance over the interconnect, but the results help to represent the highest performance available from the HCAs, giving an upper range to the performance possibilities of socket-based performance on the same platforms.

Zhang et al. [122] 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. They found that C based SDP and IPoIB outperformed Java by a significant margin, and that Java applications consumed far more CPU time than the equivalent C implementation when sending data over IPoIB or SDP. They attributed this behaviour to the light-weight dynamic SDP library in C, that only needs to keep a basic adherence to the standard interface of TCP/IP while taking advantage of all of the verbs layer performance advantages. In contrast, the Java implementation must actually provide a fully functional TCP/IP protocol, which adds in unnecessary complexity which causes a gap in performance between Java and C based IPoIB and SDP. The most significant finding in terms of data center concerns is that the Java IPoIB and SDP servers consume nearly 100% of the CPU resources during transmissions, while the C servers only consume approximately half of the available CPU time. We predict that this gap would have widened even further had Zhang et al. used a zero-copy based SDP.

Other work on the Java communication stack has been done in [45], where RDMA operation functionality was added for Java through a specialized library. With this new library called Jdib,

they are better able to exploit the RDMA capabilities of InfiniBand networks in Java applications. By eliminating an unnecessary copy between the Java virtual machine buffers and the operating system buffers, as well as providing a mechanism for passing data by reference between Java and C code, they have improved the performance of Java network performance for applications using SDP or IPoIB. The work to date with Java communication stacks differs from the work in this chapter 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 [10]. However, no investigation has yet been performed regarding the segmentation offloading capabilities of InfiniBand high-speed sockets based traffic.

3.2 InfiniBand Quality of Service

Quality of Service is important for modern systems as maintaining fairness in traffic flow between socket-based protocols is of concern. Providing different levels of service to different clients can be desirable. InfiniBand's QoS provisions offer a method of providing superior service to given traffic flows, or alternatively fair service to all traffic flows. QoS provisioning for IB allows for multiple virtual lanes of traffic, with each virtual lane mapped to a given service level. High priority queues provide a blocking-type arbitration for their traffic while low priority queues provide non-blocking type behaviour. QoS dictates the rate at which packets can be injected into the network as well as the average amount of data that can be sent by streams of a given service level over a period of time. This section will examine the behaviour of both low-priority and high-priority queues [36] and determine if they can be effective in providing both priority service as well as fair service to multiple data streams.

3.2.1 Experimental Results

In order to determine the effectiveness of QoS over InfiniBand networks we have conducted experiments on two Dell PowerEdge 2850 SMP servers. The PowerEdge 2850 has two dual-core 2.8GHz Intel Xeon EM64T processors with a 2MiB L2 cache for each core on a chip. There are 4GB of DDR-2 SDRAM on an 800 MHz Front Side Bus. Each SMP server is equipped with a Mellanox ConnectX 4X InfiniBand HCA, with firmware version 2.4.938, connected through a Mellanox 24-port MT47396 Infiniscale-III switch. The operating system used is a Fedora Core 5 Linux kernel 2.6.20 implementation. The Open Fabrics distribution OFED-1.3 was used as the software stack for IB. Netperf was used for measurements of SDP and IPoIB bandwidth, run with a confidence of 99.5% that the average values over a 60 second test period were within 0.5% of the reported average with 10 runs averaged.

To observe whether the QoS ability of the ConnectX InfiniBand cards can be activated when the traffic is able to saturate most of the available card capacity, we use two identical MPI both-way streams over two nodes. We prioritize one stream over the other using both weighted and blocking QoS mechanisms. Each stream was run simultaneously with at least a 30 second warm-up period, followed by an equal measurement period of which the average values were recorded, with the result reported being the average of 5 such iterations. Figure 3.1 shows the bandwidth ratio of the high-priority stream over the low-priority stream, as well as the aggregate bandwidth of both streams. The x-axis (QoS factor) in Figure 3.1 is translated as high-priority stream weight for the weighted QoS mechanism, and high-priority queue's high_limit for the blocking QoS mechanism. For instance, a QoS factor of 4 means a queue weighting of 4 to 1.

We can clearly see the effect of QoS in Figure 3.1. In both QoS mechanisms, the bandwidth ratio of the higher-priority flow to the lower priority flow closely follows the QoS weighting for small QoS factor values (up to 16). However, as the QoS factor increases, the blocking QoS mechanism yields no significant effect. On the other hand, the weighting mechanism shows higher effect on the differentiated data flows and its ratio increases up to 70 for the maximum

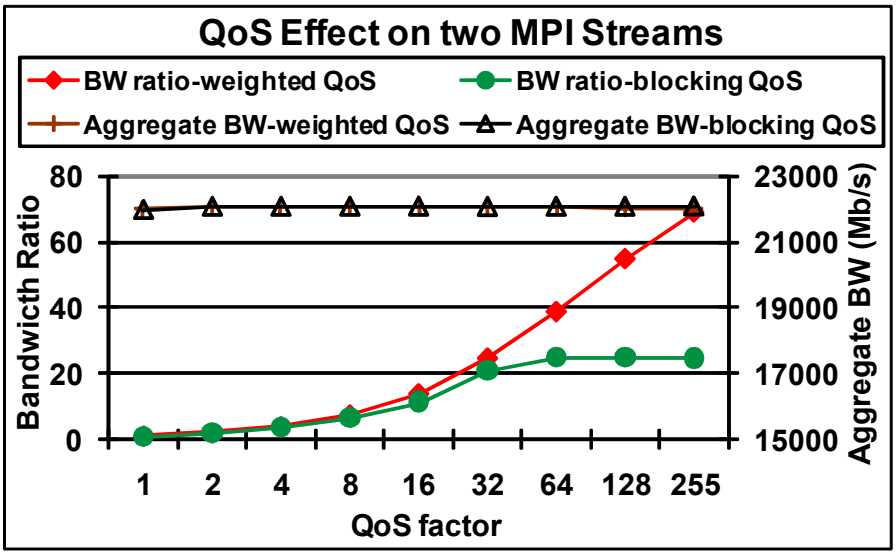


Figure 3.1: Effect of QoS on two MPI streams

weight of 255. One important observation is that the aggregate bandwidth of these two streams remains constant, and is close to 22Mb/s regardless of the QoS factor.

We will now turn our attention to the socket-based streams, which are not able to generate a high volume of traffic to put enough pressure on the cards. To do so, we employ one MPI stream as the background traffic and use three netperf TCP streams for SDP or IPoIB running over the MPI traffic. All different streams in our tests are running on different cores of the system. We examine the effect of enacting QoS on socket-based traffic and the total aggregate bandwidth using both weighted and blocking QoS mechanisms for a 512KiB message size. For weighed QoS, we show the results for up to 64 weighting of prioritized traffic over background traffic. Similarly, for blocking QoS, we go up to the high priority queue size of 64. The stream being assigned to the higher priority (blocking) queue has weightings only in that high priority queue, and is not serviced in the low priority queue.

The results for enacting a higher QoS for 3-stream SDP traffic is shown in Figure 3.2 where we can see that the SDP performance improves by 33.0% between queue weightings of 1 and 4. Assigning a priority weighting above 4 results in a diminishing return, yielding a flat performance. For blocking QoS, this is the case for high priority queue limit larger than 2.

The impact of prioritizing SDP over MPI on the aggregate traffic (SDP + MPI) is significant. We observe a 33.0% increase in SDP performance in exchange for a 15.6% decrease in the aggregate bandwidth for the 4 to 1 weighting. This drop in aggregate bandwidth is most likely due to the increased overhead of the SDP protocol over the MPI background traffic, resulting in more system related delays when SDP has higher priority. From these results we can conclude that the average queue length at the NIC is of limited size, rarely growing to have more than 4 packets for weighted QoS case, and more than 2 packets for the blocking QoS case, at any one time.

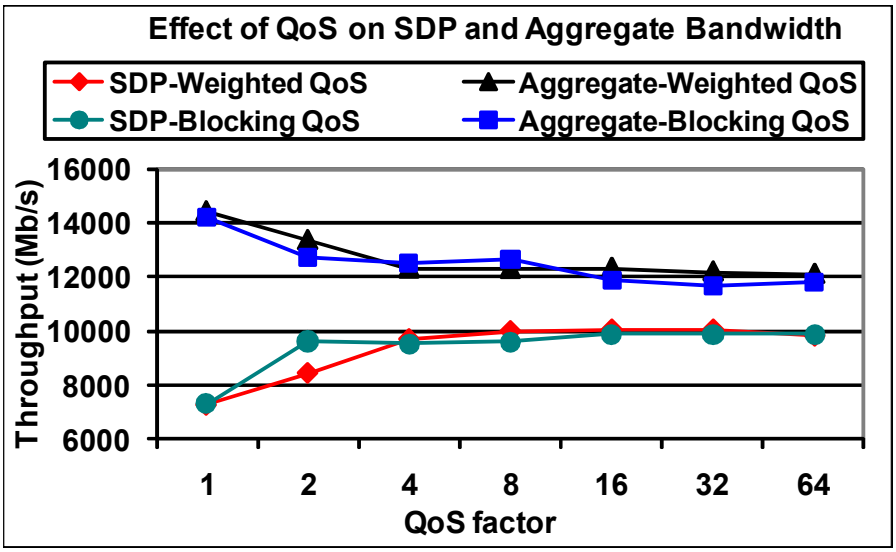


Figure 3.2: Effect of QoS on 3-stream SDP, with MPI background traffic

Next, we examine the network performance when prioritizing IPoIB traffic over MPI background traffic. As shown in Figure 3.3, there is a 72.1% increase in IPoIB performance in exchange for a 36% decrease in the aggregate bandwidth for the 4 to 1 weighting. Using a blocking QoS mechanism we see 64.2% improvement in IPoIB bandwidth accompanied by 38% drop in the aggregate bandwidth. As in the SDP case, we see diminishing returns after the weight value of 4 for weighted QoS and high_limit of 2 for blocking QoS.

It is also of interest to examine the case in which both SDP and IPoIB form all of the traffic over the network, with one protocol's traffic prioritized over the other. For the case in which SDP is the higher priority traffic, we can see the effect on the bandwidth of the SDP streams in

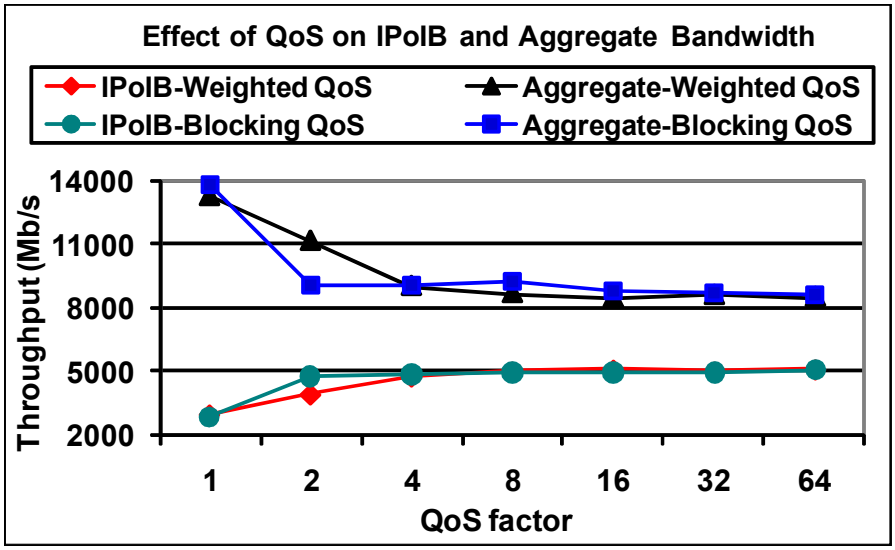


Figure 3.3: Effect of QoS on 3-stream IPoIB, with MPI background traffic

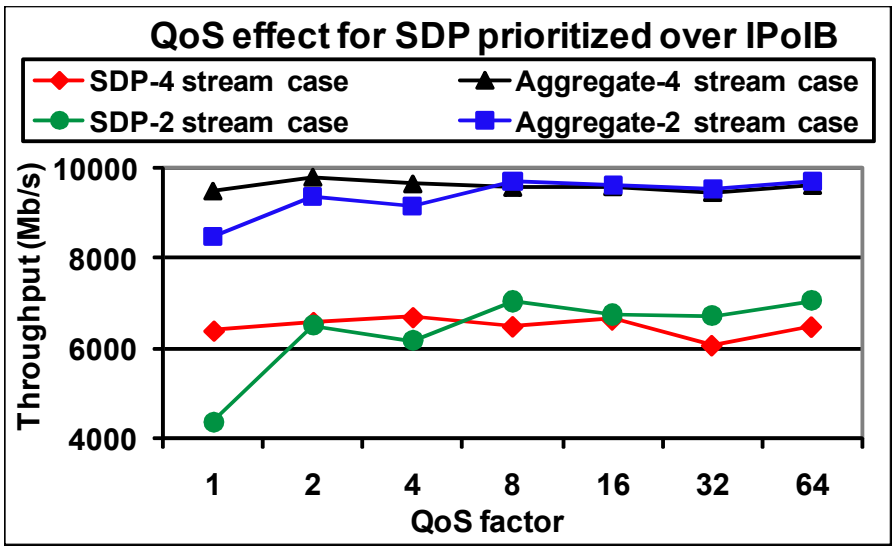


Figure 3.4: SDP performance when prioritized over IPoIB

Figure 3.4. In the 2-stream case (in which 1-stream SDP and 1-stream IPoIB are running), SDP sees a maximum bandwidth improvement of 37.7%, while for the 4-stream case, a maximum improvement of only 4.5% is seen

IPoIB exhibits interesting behaviour when it is prioritized over SDP, as shown in Figure 3.5. In the 2-stream case, IPoIB performance drops by 30.8% when it is given higher priority over SDP traffic, while the aggregate bandwidth of the combined SDP and IPoIB traffic sees a 13.2% improvement. For the 4-stream case, the effect of prioritizing IPoIB is a maximum improvement of 3.8%. Given that the QoS results for socket-based streams on our system show diminishing

returns with relatively low overall QoS factors, it prompts us to investigate the performance bottleneck that is preventing the system from reaching full utilization and limiting the size of the queues at the NIC. The performance is limited by the available bus bandwidth, as well as the buffered-copy mechanism used by SDP. In order to demonstrate that this is the case, additional tests were conducted by reducing the speed of the InfiniBand network to SDR speeds.

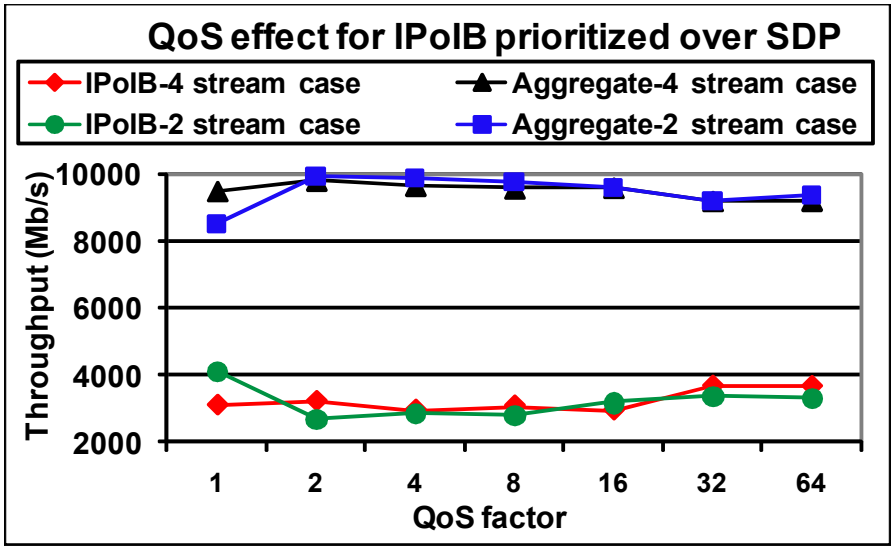


Figure 3.5: IPoIB performance when prioritized over SDP

Figure 3.6 shows the effect of weighted QoS on 3-stream SDP and 3-stream IPoIB when running over background MPI traffic in SDR mode. IPoIB and SDP are capable of occupying 36.6% and 49.7% respectively, of the maximum available bandwidth in SDR mode. In this mode, SDP and IPoIB bandwidths are increased by QoS weighting of up to 16 to 1, better than the 4 to 1 weighting in the DDR case. Similar to the DDR results, more loss in aggregate bandwidth is observed for IPoIB than for SDP. The results of blocking QoS are not shown, as they are very similar to those of weighted QoS, and thus better than the blocking QoS results in DDR. This is because in DDR configuration, queue lengths rarely build to levels that cause large amounts of blocking to occur for the MPI traffic, while in the SDR mode, blocking occurs more frequently as queue lengths are longer. As in the DDR mode, the low utilization by SDP and IPoIB relative to MPI can be partially attributed to their use of a buffered-copy mechanism, while

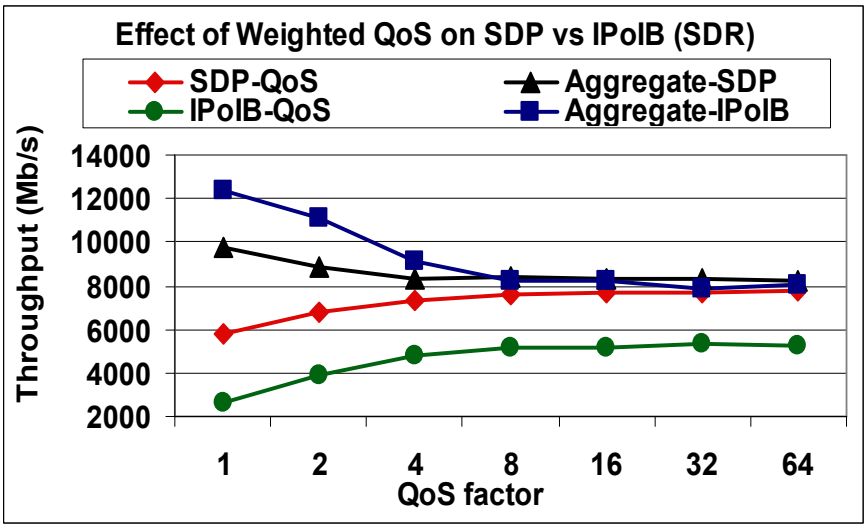


Figure 3.6: Effect of QoS on 3-stream SDP vs. 3-stream IPoIB (SDR), with MPI background traffic

MPI utilizes a zero-copy approach. We conclude that a faster node architecture coupled with a zero-copy SDP protocol will significantly enhance the QoS impact in the DDR mode.

Therefore, after exploring the functioning of the QoS provisions on real systems, it has been determined that QoS can be of use in some situations. The effect of QoS on our systems is mitigated by the system bus speed. Therefore, we anticipate that future generation systems will be better able to take advantage of the benefits of using QoS than the generation used for this testing.

3.3 IPoIB Segmentation Offload

Segmentation offloading for IPoIB datagram transmission accomplishes two key tasks, first it offloads the requirement to split large messages into packet sized data units before sending them to the HCA, thereby reducing the load on the CPU, as large messages can be sent in a single transfer. This is accomplished by the IPoIB driver providing a virtual MTU to the upper layer, this makes it appear as though the HCA can support jumbo datagrams. This virtual jumbo frame support is compatible with all existing switches and other hardware as the HCA segments the packets into their actual supported size for the hardware, this is referred to as Large Send Offload

(LSO). More importantly, at the receiver side, the reception of the packets can be offloaded using a Large Receive Offload (LRO) mechanism. LRO essentially aggregates incoming packets and forwards them to the receiving system once a time period has elapsed or a given amount of data has arrived. This is extremely useful for datagram transmission as it significantly reduces the total number of interrupts that the CPU must handle, and makes the overall DMA transmission process more efficient by combining many smaller packets into a larger DMA operation. The DMA operations of modern chipsets do not support very large transfers, typically operating at less than 1KiB per transaction, but the combination of many messages helps to eliminate the need for additional transfers of data that are not equal to the maximum DMA transmission size.

3.3.1 Micro-Benchmark Experiments

In order to determine whether offloading could be of benefit in a data center context, we conducted several tests, first to confirm that datagrams with offloading can provide performance near to that of a reliable connection and then to determine if such offloading techniques can be of use in a data center context [33][35]. For these tests we have used four Dell PowerEdge R805 SMP servers. The PowerEdge R805 has two quad-core 2.0GHz AMD Opteron processors [1] 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 [85] was used as the software stack for IB. All software was compiled specifically for the machine using gcc 4.1.1. The results for the micro-benchmark tests were obtained using Netperf [62] with a 99.5% confidence interval of +/- 0.5% over 60 seconds averaged over 10 runs and confirmed using Iperf [113]. It should be mentioned that the PCIe x8 bus's practical bandwidth (16 Gb/s theoretical) is a limiting factor on our system.

Figures 3.7 and 3.8 examine the latencies of IPoIB with four possible UD offloading configurations, as well as a reliable connection configuration. Figure 3.7 shows the results for small messages while Figure 3.8 shows the results for medium and large messages. 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. At the 2 KiB message size, the latency of the LRO enabled offloads increases to above that of the LRO disabled configurations.

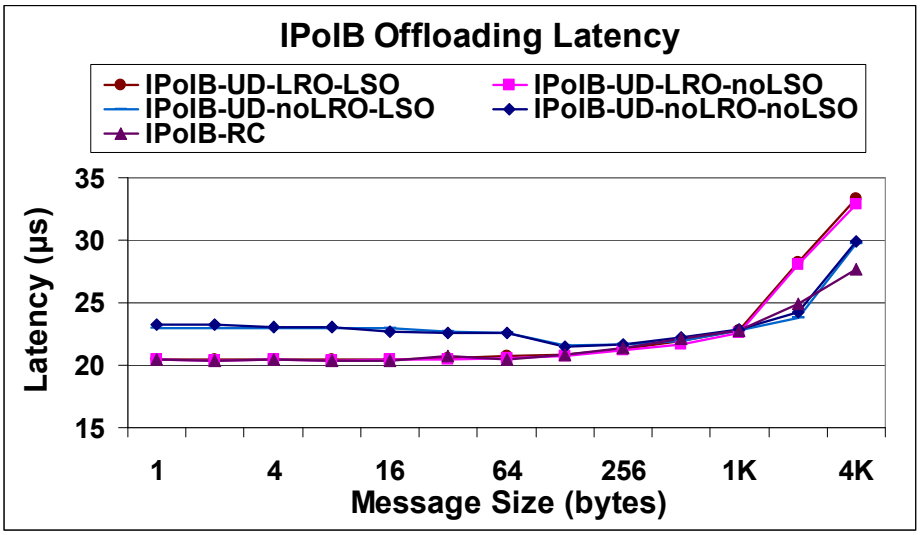


Figure 3.7: IPoIB offload one-way latency for small messages

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, IPoIB-UD-LRO-LSO 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.

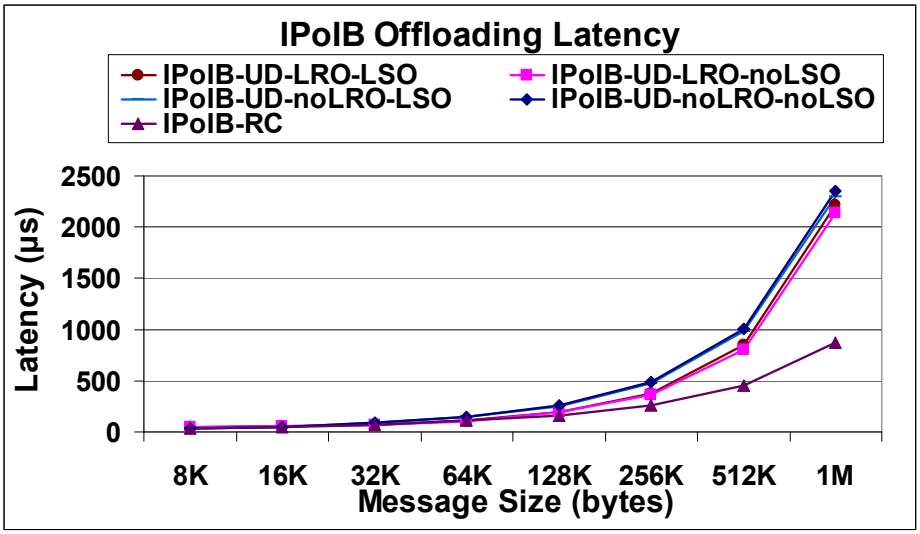


Figure 3.8: IPoIB offloading one-way latency for large messages

Figure 3.9 shows the bandwidth of single stream IPoIB-UD configurations with varying offload capabilities as well as IPoIB-RC. The highest bandwidth can be achieved using IPoIB-RC, of approximately 8600 Mb/s while the best IPoIB-UD configuration, LRO with no LSO has a maximum bandwidth of approximately 7100 Mb/s. 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. 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 3.10 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 3.9, 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

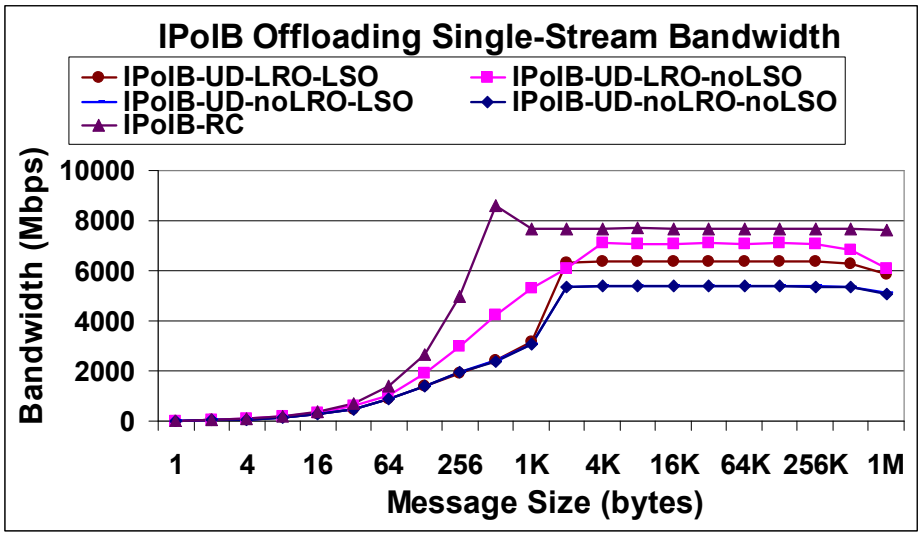


Figure 3.9: IPoIB offload single-stream bandwidth

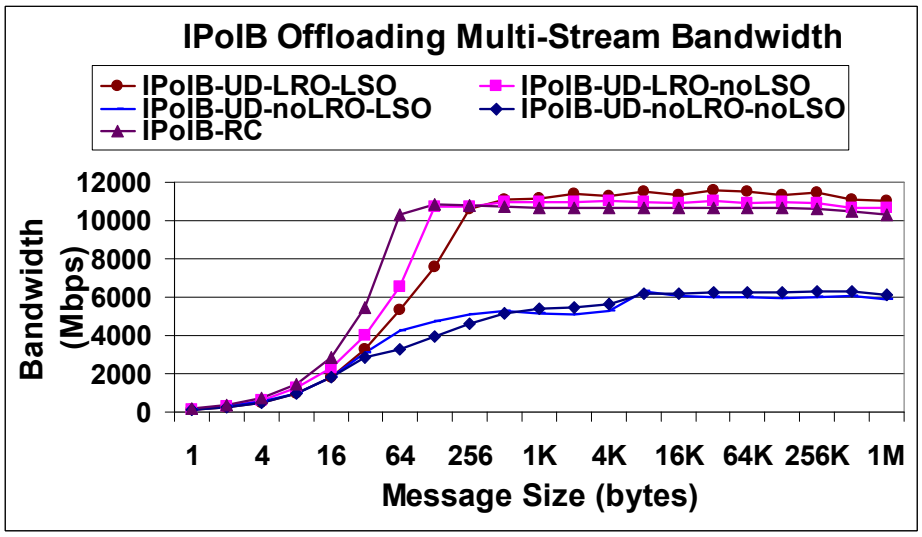


Figure 3.10: IPoIB offload multi-stream bandwidth (8 streams)

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 Virtual MTU (VMTU) segmentation offload, but fails to achieve the maximum bandwidth achieved by IPoIB-LRO-LSO, being outperformed by 7.1%.

3.3.2 Data Center Test Results

Given the results of the offloading on basic benchmarks, it is now important to discover the effect that such techniques have in a data center context. In order to do this, tests were run using a simple three-tier data center configuration. This configuration consists of an Apache 2 web server responsible for all static http and image serving, a JBoss 5 application server responsible for all server side Java processing and a database system running MySQL. A high-level overview of the data center implementation used for testing can be seen in Figure 3.11.

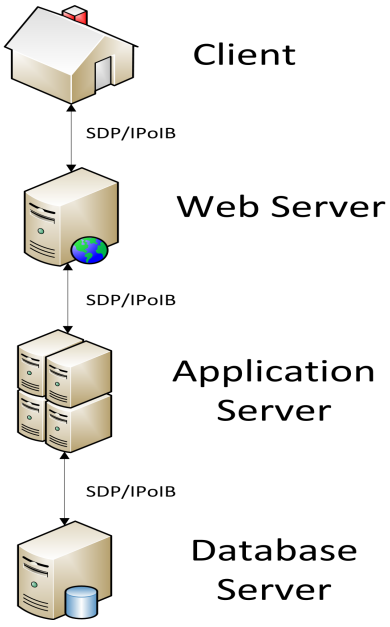


Figure 3.11: Data center layout

This overview also shows the interconnections that were available between each of the given tiers. Configurations were run using either all SDP connections or all IPoIB connections, no hybrid SDP/IPoIB configurations were considered. It is worth noting that InfiniBand is not commonly used in a commercial data center context, but may become more common in the future.

To assess the performance of offloading in the real world we used the TPC-W [115] Web Commerce benchmark, which uses a real web-based bookstore to test the latency seen by multiple clients as a total request/response time seen at the client side. It also monitors the average number of *web interactions per second* or WIPS that the system is able to sustain, which

is a measure of throughput. Strict timing requirements exist to ensure that the system is providing reasonable service to all of its clients. The results shown are free of any errors during the entire measurement period.

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 behavior set to replicate real conditions. The TPC-W implementation used was from New York University [114] 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 [44]. Is has been extensively modified to work with our EJB website implementation.

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 3.12. We can observe that the highest throughput (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). 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 3.13 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

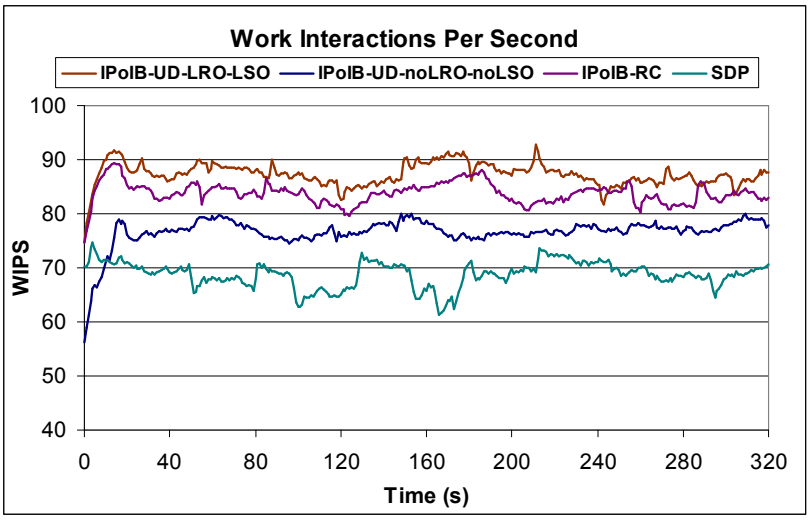
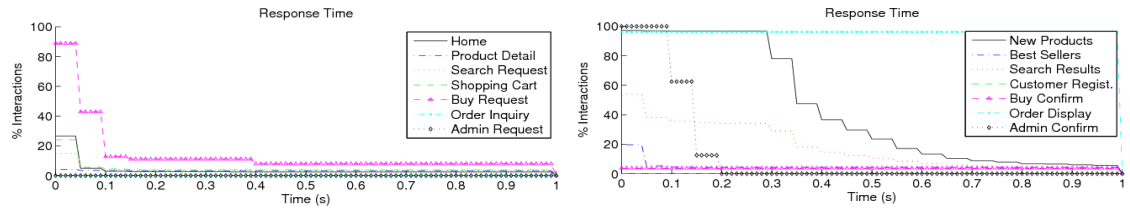
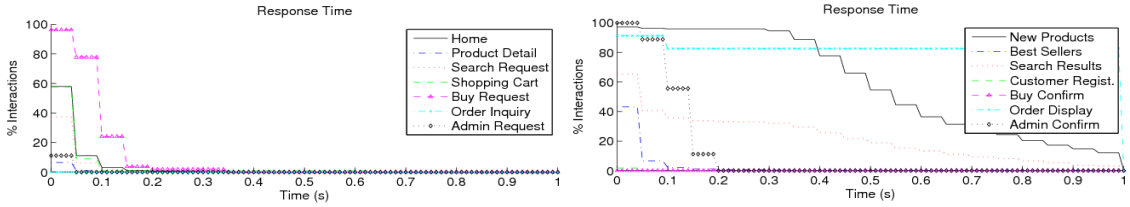


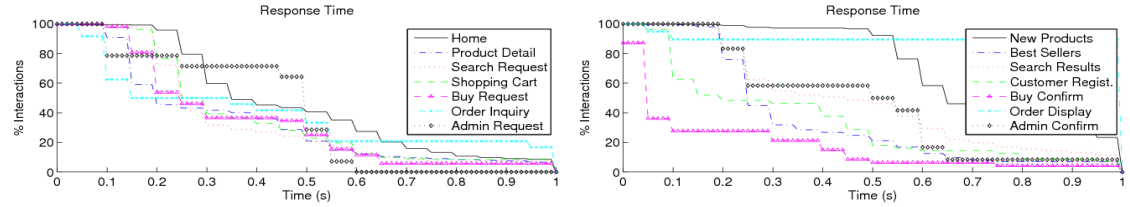
Figure 3.12 TPC-W data center throughput



(a)



(b)



(c)

Figure 3.13: TPC-W data center latency: (a) IPoIB-UD-LRO-LSO, (b) IPoIB-UD-noLRO-noLSO, and (c) SDP

have a beneficial effect on the system, reducing its latencies to below those of the non-offloaded mode and of SDP. Offloaded IPoIB 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 noLRO-noLSO IPoIB configuration 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 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 overall communication between the data center layers, requiring aggregation at all data center levels.

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 could be due to the RDMA semantics underlying SDP, or connection management overhead. 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 were 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 Figure 3.14. An improvement in the throughput of the system to levels higher than that of previous SDP configuration, and even that of the IPoIB configurations, to 99.85 WIPs was seen in addition to latency reductions. This shows that the reduced SDP performance was due, at least in part, to 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 3.13(c). 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.

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.

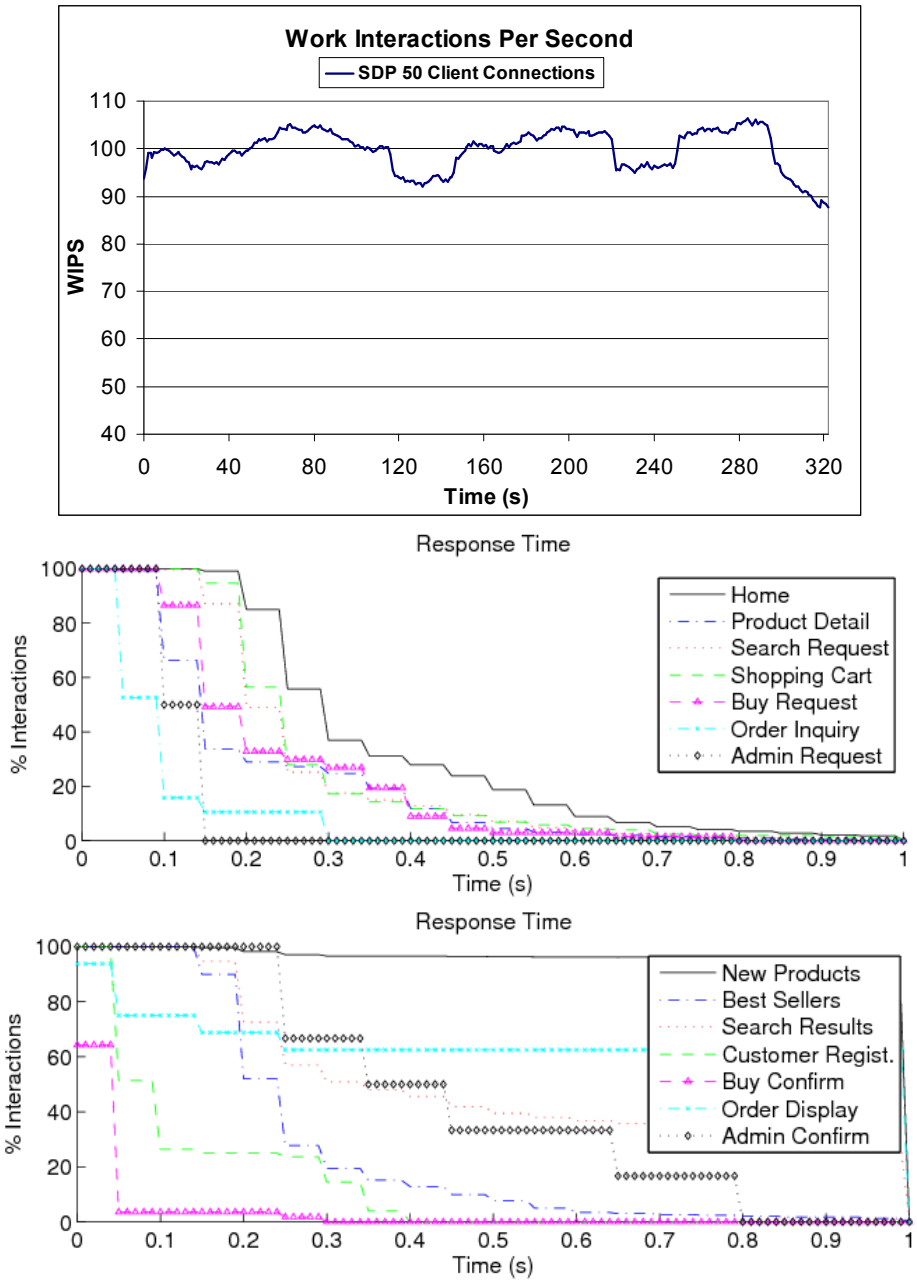


Figure 3.14: TPC-W data center throughput and latency for SDP with 50 client connections

3.4 Discussion

This chapter has explored the behaviour of QoS provisioning for an advanced network as well as offloading techniques for datagram traffic. Understanding the performance of these features is important in determining the most effective techniques to leverage in next generation Ethernet networks. Quality of Service for InfiniBand networks has demonstrated that it can be effective in

regulating the traffic of multiple streams of data on a network. The upper limits of the QoS ratios are not accessible given the level of PCI-Express bus technology that is available to us, but we expect the upper limits to be accessible with subsequent generations of PCI-Express buses, which next generation Ethernet is expected to utilize. This is especially important for very large throughput data stream applications, like high-definition video streaming, in order to allow important application control and request messages to transfer quickly over the network while it is transmitting large amounts of data. Consequently, we can conclude that such QoS provisioning of a similar type would be useful in next generation local-area Ethernet networks, and future hybrid networking architectures. Unfortunately, it can only be applied to IB network traffic and therefore, hybrid networks can only make use of QoS when the IB network links become saturated.

The evaluation of UD offloading techniques over IPoIB has shown that the inclusion of LRO offloading is very beneficial to overall networking performance, while the inclusion of LSO offloading is of less importance. It has been shown that IPoIB operating over a UD transport can be effective for web serving data center applications. It was also demonstrated that unexpected performance issues arise when using SDP in data center contexts. The supposition that this performance is caused by either low individual connection utilization or hardware-side connection management overhead was confirmed through higher intensity testing using fewer total connections. This study of offloading over socket interfaces for verbs based RDMA networks is important for understanding the usefulness of offloading techniques for future converged Ethernet networks that might use verb semantics for RDMA operations.

The following chapter will explore solutions to the observed performance problems with SDP, while examining what hybrid networking architectures are available to current generation networks.

Chapter 4 Leveraging Hybrid Networks for Data Centers

This chapter seeks to address the issues discovered in the previous chapter relating to SDP. It also aims to explore the potential of hybrid network architectures and assess their suitability compared to state of the art Ethernet networks as well as InfiniBand networks.

The previous chapter found that the performance of SDP was unexpectedly poor in data center testing versus the results from the micro-benchmark tests. With a reduced number of more active client connections, performance was more in line with that found in micro-benchmark testing. Through the implementation of a new network transport method for SDP, we can make conclusions as to the source of the previously observed network performance issues with SDP. By implementing a version of SDP that will reduce RNIC-side connection management overhead we can observe whether this was the cause of the performance issues rather than sparsely used connection transmission pipelines and protocol overhead.

The assessment of hybrid Ethernet/IB networks will enable us to determine if such network architectures are practical for next generation data centers, as well as determining the performance gap that may exist between Ethernet hardware and IB socket-based networking approaches. The study of hybrid network architectures also provides insight into possible future network convergence as the existing Ethernet-based wide area networks (the Internet), could be interfaced with data centers using localized high speed networks such as IB.

4.1 Related Work

Related work on sockets-based interfaces for IB networks, as well as the framework for data center testing used in this chapter can be found in Section 3.3. The past work done on data centers and java over high speed networks reviewed in Section 3.3 also applies to this chapter. A comparison between offload enabled Ethernet adapters and high-speed Myrinet and IB adapters has also been performed [90].

XRC has been proposed for use and implemented for Message Passing Interface (MPI) implementations. The first study of XRC for use over MPI was done by Koop et al. [68]. They determined that XRC could be of great benefit to MPI designs by reducing the number of connections required for a fully connected network by a factor equal to the number of cores in each node. This approach was extremely beneficial for MPI as some applications communicate between all processes in collective communication operations. In addition, the MPI implementation used, MVAPICH [67], pre-posts many receive buffers, so it also benefits from a reduced required number of receive buffers from the shared receive queues used with XRC.

4.2 Sockets Direct Protocol over eXtended Reliable Connections

In order to help alleviate some of the overhead incurred when opening a large number of connections between two nodes, eXtended reliable connections (XRC) were developed for InfiniBand networks.

The eXtended Reliable Connection mode is a feature currently available on InfiniBand [55] Host Channel Adapters from Mellanox [72]. InfiniBand is a queue pair based high-speed, low latency interconnect. Interaction with InfiniBand HCAs is performed through the use of verbs. These verbs provide for a full kernel bypass while allowing user level programs to directly access all of the features of the InfiniBand fabric. The verbs layer uses a queue pair model in which both channel-based communication semantics (send/receive) and memory-based communication semantics (RDMA) are supported. The InfiniBand standard [54] supports four different modes of operation, Reliable Connections (RC), Unreliable Connections (UC), Reliable Datagrams (RD) and Unreliable Datagrams (UD). Mellanox Technologies has also added support for extended Reliable Connections.

XRC is a shared connection solution where a single send queue can communicate with any of the processes on a given node, without requiring a dedicated received queue for the sender for each process. The sender creates a QP called an INI QP, which attaches to a target QP or TGT

QP on the remote node. A process needs only a single SRQ which connects to all of the TGT QPs on the node, provided they are all in the same XRC domain. Matching to SRQ from traffic on TGT QPs is done via XRC domains and assigned SRQ numbers, when both of these match, the data delivered to the corresponding SRQ. However, an INI QP is needed for each destination that a sender intends to send to. An illustration of this is provided in Figure 4.1.

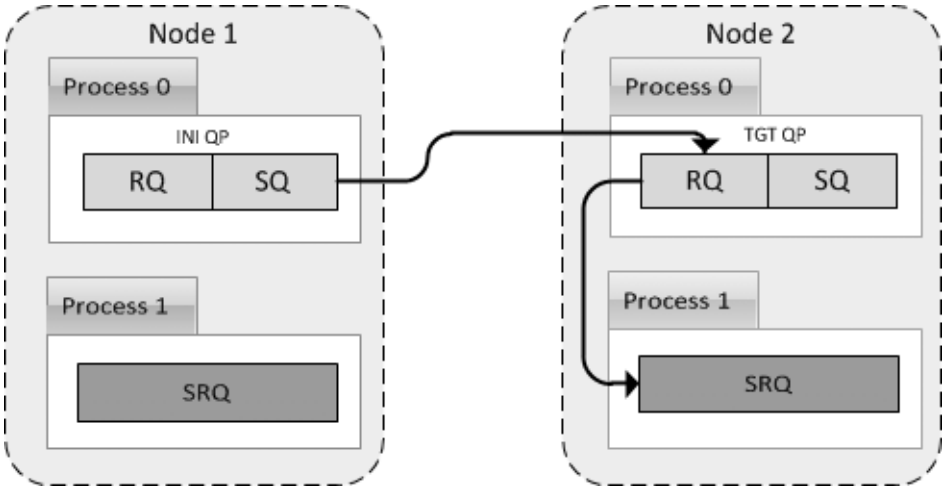


Figure 4.1: XRC communication layout example

XRC has two main benefits in terms of its communication layout. Firstly, only one INI-TGT pair is needed to provide communications from the INI queue owning process to all other processes on the target node, eliminating the need for a separate QP for each process the initiator wishes to communicate with. Secondly, the SRQ for each receiving process allows a single set of multiple receive requests to be posted to the SRQ to handle incoming data. With independent QPs, these same requests would have to be duplicated across several receive queues. This allows for memory usage savings in the case where there is interconnection between multiple processes on different nodes. For the case where only one process is running on the target node, there is no memory savings. This memory savings is also directly related to the number of pre-posted buffers that the software at the verbs level is posting to the SRQ, which in some cases such as its use with MPI implementations (MVAPICH[68]), can be significant. However, this is dependent on the individual verbs level software's implementation specifics.

In order to adapt SDP to use XRC several changes were required. The SDP code in the Open Fabrics Enterprise Distribution [85] was adapted to allow for the creation of XRC domains, as well as sending and receiving data concerning the destination SRQ. Connection setup and management needed to be changed to account for the existence of XRC domains as well as SRQs, as did connection teardown. As SDP attempts to closely mimic the behaviour of TCP, data needed to be stored according to the individual SDP sockets as to their involvement with XRC (domains, etc). The implementation supports either traditional SDP or SDP operating over XRC, it does not permit individual sockets to operate in separate modes. This means that when operating in XRC mode, a socket could belong to an XRC domain with only a single communicating partner. An attempt is made to create sockets under as few XRC domains as possible.

4.2.1 SDP vs. SDP-XRC Performance Comparison

This section examines the performance of SDP operating in its default mode (RC), versus that of SDP operating over XRC. For the development of SDP-XRC, we have used a newer version of the OFED drivers than was used for the previous chapter in order to have XRC support, version 1.5.2. As such the impact of packet segmentation has been reduced through protocol optimizations, while the other behaviour of SDP remains relatively unchanged.

The experiments in this section were conducted on the same servers used in Section 3.3.1. The OFED drivers used have been changed from version 1.4 used in Chapter 3 to 1.5.2 for XRC support. Netperf [62] was used for bandwidth testing, with the same 99.5% confidence interval used for Chapter 3. Iperf [113] was used to confirm the results. All bandwidth results are shown in Millions of bits per seconds (Mb/s), while all latencies are expressed in microseconds (μs). All message sizes are expressed in Kibibits or Mebibits, as applicable.

We first examine the latency performance of both traditional SDP as well as SDP-XRC [35]. The results are shown in Figure 4.2. We can observe that the difference in latencies between SDP and SDP-XRC is minimal, with neither version having any significant advantages. We can

observe the difference in performance over the verbs level RDMA operations, which ranges from 8 to 2 times higher than those of the base verbs, with larger messages sizes having a smaller gap in performance than those of small messages.

Examining Figure 4.3, we can observe the single stream bandwidth for both non-XRC and XRC SDP as well as compare their performance to the baseline verbs RDMA write bandwidth. XRC and non-XRC bandwidth performance is extremely close, which demonstrates that XRC's TGT QP SRQ matching is having little effect on performance. We can also observe that the large message size performance of SDP is 21-23% less than that achievable with low-level verbs.

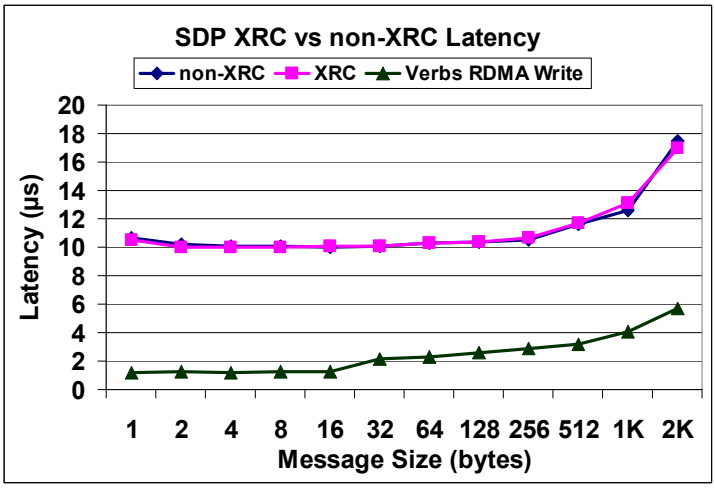


Figure 4.2: SDP vs. SDP-XRC latencies

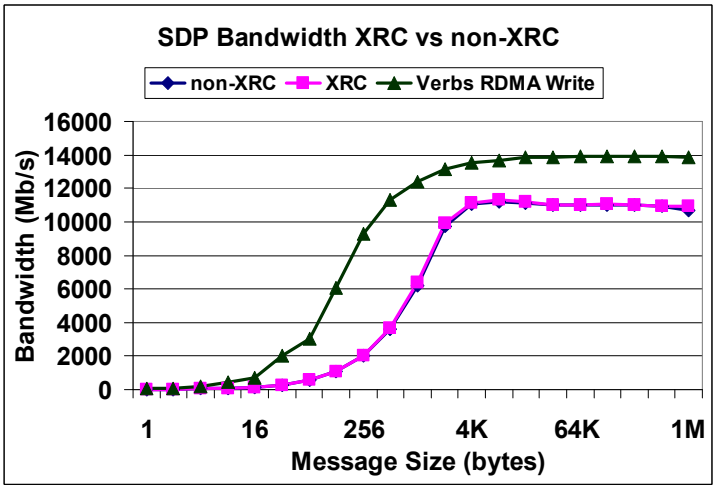


Figure 4.3: SDP single stream bandwidth: XRC vs. non-XRC

Figure 4.4 shows the bandwidth performance of SDP running in its default RC mode. It can be observed that for a single flow, the performance of SDP peaks at approximately 12000 Mb/s at a message size of 8 KiB. The performance then slowly declines to approximately 11100 Mb/s for larger message sizes. A higher number of active communicating threads shifts the message size at which maximum bandwidth is achieved to a smaller size. However, the maximum bandwidth achieved is not significantly affected by having larger numbers of threads.

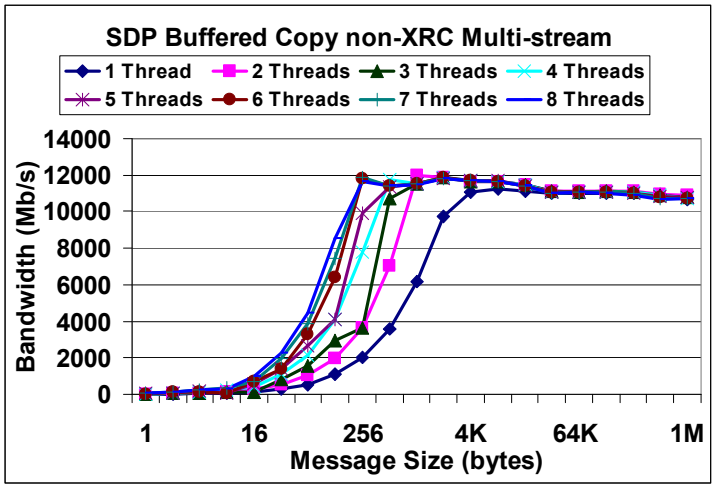


Figure 4.4: SDP non-XRC multi-stream performance

Figure 4.5 illustrates the performance of SDP running over XRC. We can observe that the performance characteristics of the XRC mode SDP implementation closely match those of the non–XRC implementation. This is to be expected as the performance of XRC versus RC should be similar, having only a very minor additional overhead in matching the incoming data to the correct SRQ as well as the correct QP participating in the use of that SRQ. Therefore we are observing that the advantages of utilizing XRC (use of SRQs and fewer full connections for communication between nodes) are available at virtually no cost in terms of network bandwidth performance.

The theoretical maximum bandwidth of the ConnectX adapter is 16 Gb/s, as is the speed of the PCI-Express bus (8x). Overhead present in the PCI-Express bus makes the maximum achievable bandwidth lower than the theoretical maximum, making it the bandwidth bottleneck.

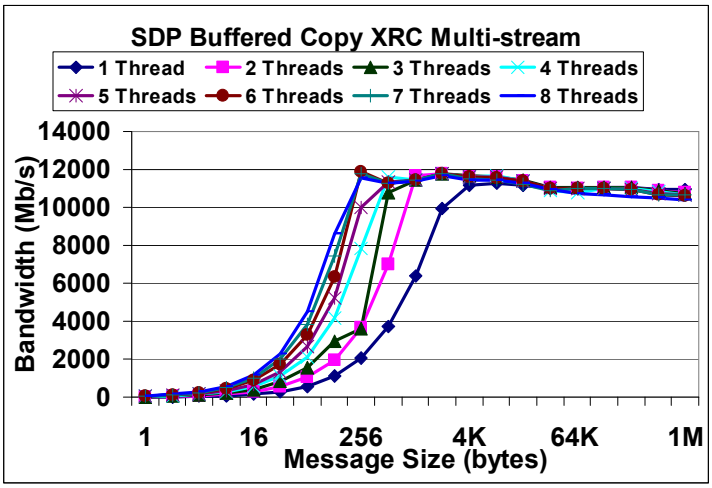


Figure 4.5: SDP over XRC multi-stream performance

4.2.2 SDP-XRC Data Center Results

The results for the SDP-XRC data center throughput testing are shown in Figure 4.6. We can observe that the throughput obtained with the SDP-XRC data center testing is 69.02 WIPS; this compares very closely with the results from the standard SDP data center of 69.08 WIPS. This also demonstrates that there are not major performance differences between OFED version 1.4 and 1.5.2 for SDP. SDP-XRC response times are shown in Figure 4.7. This demonstrates that the performance issues relating to SDP that were previously observed are not due to the device level connection handling. When combined with the observations made in Chapter 3, we can see that SDP underperforms when handling large numbers of connections. As this is not due to device level connection handling, it must be incurred due to overhead in the SDP interface. If connection handling overhead were an issue, the reduced number of full connections using XRC between the nodes should create an increase in performance. The lack of improvement suggests that the problem must be caused by another source.

It should be noted that the overall effectiveness of XRC when applied to SDP can also be limited by its connection initialization characteristics. For an ideal number of connections to be created, the process initiating the connection must connect to a target process that is not already connected to a process on the source node. In addition, to fully utilize the advantages of XRC,

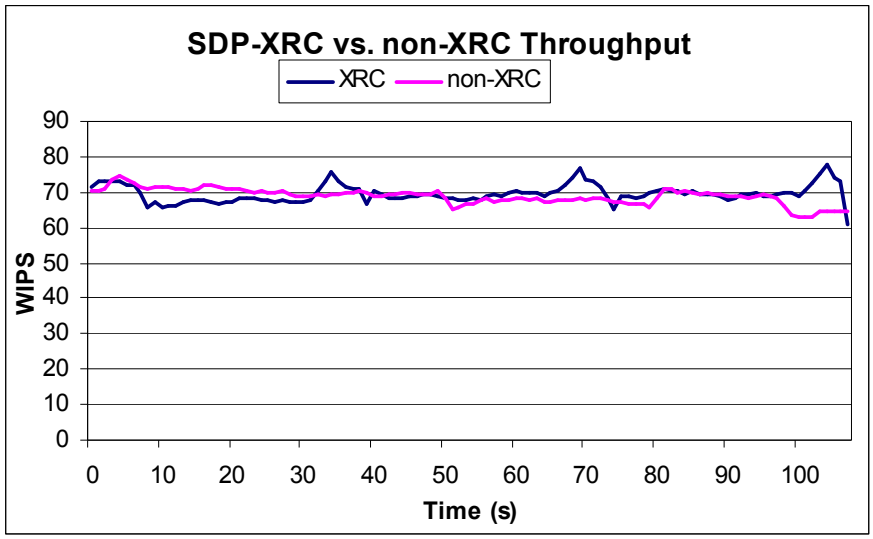


Figure 4.6: SDP-XRC and SDP non-XRC data center throughput

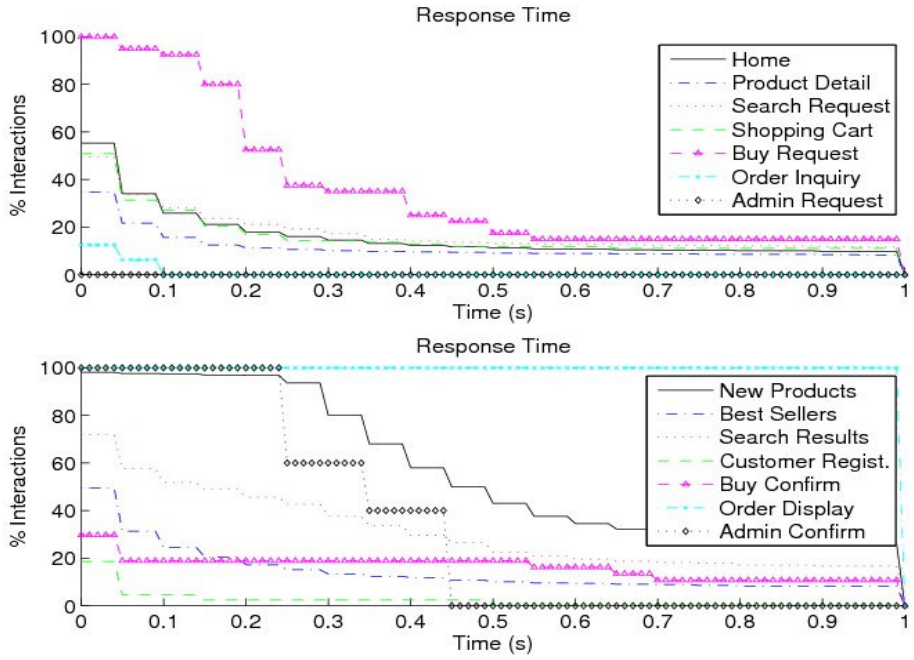


Figure 4.7: SDP-XRC data center response time

the processes on a given node must be taking advantage of the SRQ sharing abilities, meaning that a process must engage in communication with multiple other processes on the destination node in order to see a net benefit from using XRC. For bi-directional traffic in which each process on the host node communicates with only a single process on the target node, the net benefit to using XRC is reduced. Even though the benefits of XRC can potentially be somewhat marginalized by the connection characteristics of the data center software, the lack of observed

benefit from using SDP-XRC in this environment demonstrates that the poor performance of SDP relative to the other protocol alternatives is due to overhead from the SDP protocol itself and not due to the excessive usage of resources from a large number of connections. As discussed in Chapter 3, the issues surrounding the poor performance of SDP must be protocol related, as the lightly utilized individual channels (sockets) have increased overhead as control message latency cannot be hidden amongst in-progress transmissions on a given socket, which creates lower utilization than a high traffic channel, where techniques can be used to hide these protocol control messages.

Therefore, it is important to explore alternative architectures that could provide good performance for large numbers of connections by utilizing a different protocol than SDP for such tiers of the network, while utilizing SDP in areas with very active connections (which lessen issues with lightly packed communication channels) in order to leverage its excellent capabilities in such situations. IPoIB has been shown to work well for layers with many connections, and SDP can perform well for high activity layers with low numbers of connections. Therefore, a hybrid networking solution seems to be the best option for high-speed socket interface data centers.

4.3 Virtual Protocol Interconnect

VPI allows for the easy creation of hybrid networks, utilizing a single multi-port network adaptor to operate in two separate modes simultaneously. Using this method, one can utilize the Ethernet hardware and associated offloads at the same time as using the IB network hardware. This allows for a single HCA to function not only as two separate types of network adaptors but also allows the system to function as a bridge between said hardware. The benefits of such a system can be seen in its ability to use the advanced network features of IB as well as benefiting from increased signaling speed of IB in a LAN context while maintaining Ethernet compatibility for WAN transmission. Consequently, this can be used for new hybrid data center networking

configurations, with the back-end data center tiers supporting IB and the edge nodes supporting Ethernet. In addition, it can be potentially used with virtualization to support a fully re-configurable hybrid data center networking architecture, where systems can be allocated to the necessary tier without concern for the systems particular installed networking hardware. VPI should not be regarded as merely an integration of two NICs into a single card, but rather as a flexible re-configurable networking solution.

One possible concern with VPI is the performance of the system while it is providing data to both an IB port and an Ethernet port. The performance of the system as a dual network system is reliant on both ports functioning without functionality crippling degradation in either of the two ports.

4.3.1 Micro-Benchmark Experiments

In order to study the behaviour of VPI [34] we have used the same four systems used in the previous investigation in Section 4.2.1. These systems are the same ones used for data center testing in Chapter 3. Using these systems, we found a performance baseline for both latency and bandwidth to compare the capabilities of the implemented IB and 10 GigE NICs. The baseline unidirectional latencies of each of the protocols under study are presented in Figure 4.8. The minimum latency of 1.22 μs that ConnectX can achieve using InfiniBand RDMA Write is also shown in Figure 4.8 for comparison to the other methods. For the 10GigE tests, the *Adaptive Interrupt Coalescing (Receiving)* (AIC-Rx) mechanism must be considered when measuring the systems Ethernet latency. AIC-Rx allows for the alteration of the behavior of the interrupt generation pattern of a system for incoming traffic. It adapts both the number of frames that must be received before triggering an interrupt and the amount of time from the first packet that is received after triggering an interrupt.

In the case of latency tests, the adaptive algorithm is adjusting the system to create interrupts that fetch the incoming packets according to the traffic pattern, which is increasing the efficiency of interrupts, essentially this is creating a very good timed poll for data at almost exactly the

interval at which the incoming data is received, thereby reacting to them very quickly. Additionally, the responses can be aggregated and since multiple requests are on the transmission line at one time, it leads to responding to all of the requests in a much faster turn-around time than if each packet were handled independently, or a static coalescing scheme were used. This takes advantage of the nature of the request/response micro-benchmark and results in an excellent reported latency, but does not properly indicate its performance in a real world situation. IPoIB does not support adaptive interrupt coalescing and so IPoIB (with offloading) shows a higher latency for small messages than AIC-Rx 10GigE, but when AIC-Rx is disabled IPoIB shows lower latency. SDP (non-XRC) is superior to 10GigE, with a 65.4% average decrease in latency for messages up to 1 KiB. This is because it allows the system to bypass the TCP/IP stack and other software layers and translate socket-based packets directly into the verbs layer RDMA operations, while maintaining TCP stream socket semantics.

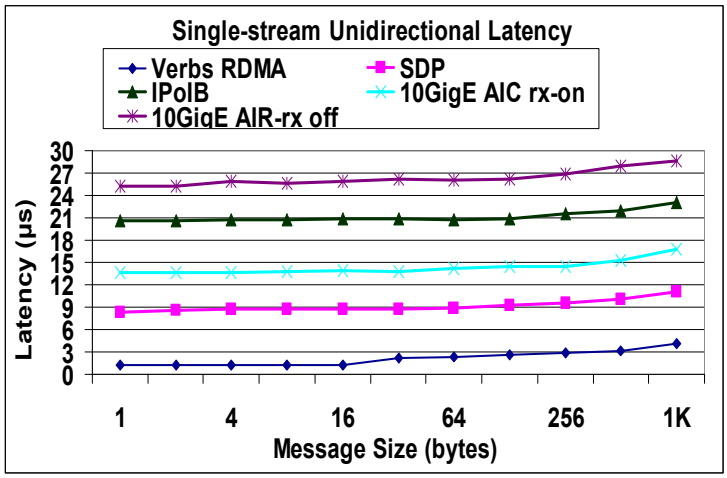


Figure 4.8: Single-stream latency

Figure 4.9 illustrates the baseline single-stream bandwidth for 10GigE (with both jumbo and normal frames), IPoIB, SDP and IB verbs. Jumbo Ethernet frames are a 9000 byte frame size while normal frames are a 1500 byte size. Jumbo frames are typically supported on 10GigE hardware, but are not necessarily backward compatible with previous generation Ethernet

networks [35]. The IB verbs bandwidth represents the maximum achievable bandwidth for our system of 12650 Mb/s for large messages.

SDP is the closest in performance to the IB verbs bandwidth, achieving a maximum of ~11880 Mb/s. It is observed that the performance of SDP takes two separate drops, at 4 KiB and at 64 KiB message sizes. The 4 KiB drop point is due to the packet segmentation of the IB fabric of 2 KiB packets, therefore a performance penalty is observed when the additional load required to segment the data stream into smaller message sizes is incurred. The other performance drop occurs at 64 KiB, which is the default threshold value for a switch between buffered-copy and zero-copy transmission methods. This drop can be remedied by adjusting the threshold higher message size.

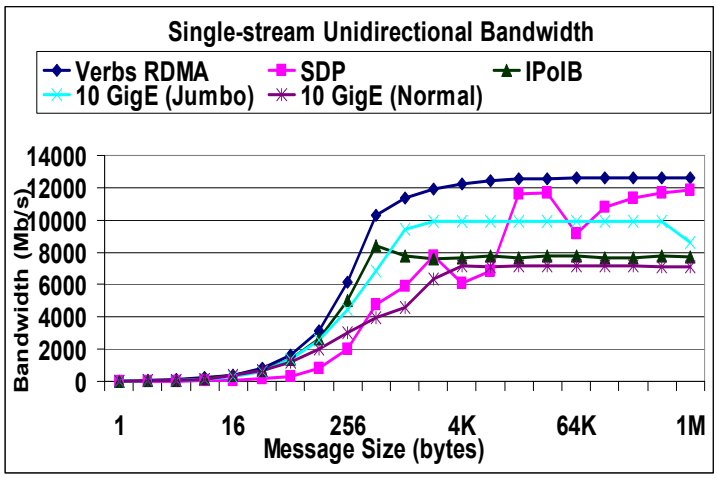


Figure 4.9: Single-stream bandwidth

The results in Figure 4.10 show the multi-stream bandwidths of 10GigE using jumbo or normal frames. The maximum bandwidth for jumbo frames is close to the single-stream results at 9900 Mb/s. The slight drop in performance noted at 1MB message sizes is eliminated by using multiple streams. For normal frames, the maximum bandwidth is 9200 Mb/s 28% higher than that for a single stream, but still below that for jumbo frames. For jumbo and normal frames we see bandwidth improvement for smaller messages sizes as the number of connections is increased, and no saturation effect occurs at the maximum load (8 streams).

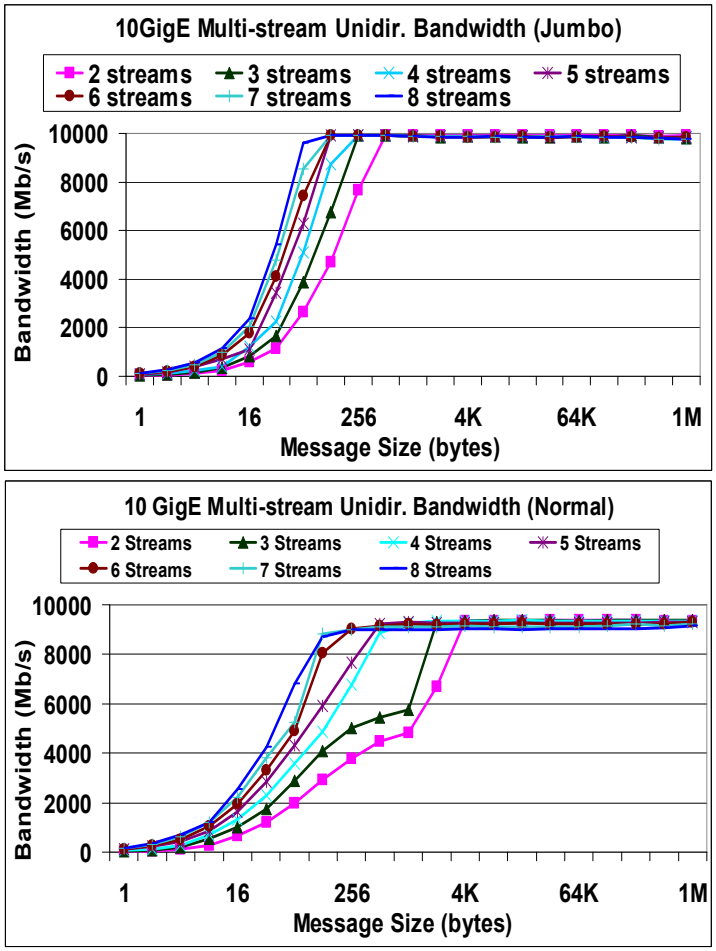


Figure 4.10: 10GigE multi-stream bandwidth (jumbo and normal frames)

The bandwidth of multi-stream IPoIB is shown in Figure 4.11. IPoIB benefits greatly from using multiple streams seeing a 30.1% improvement in bandwidth. The maximum bandwidth peaks at approximately 10800 Mb/s, or 85.4% of maximum verbs bandwidth.

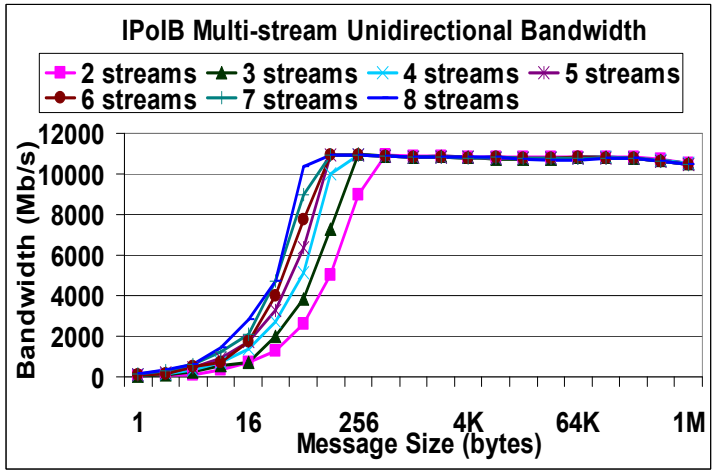


Figure 4.11: IPoIB multi-stream bandwidth

The multi-stream bandwidth for SDP is shown in Figure 4.12. The maximum bandwidth for SDP occurs with message sizes greater than 128 KiB, with a maximum bandwidth of approximately 12500 Mb/s. This shows excellent performance only 1.2% lower than that of native IB verbs. We can still observe the performance dips that occur at the message segmentation point as well as the cross over from buffered copy to zero-copy mode in the multi-stream tests. However, for four streams or more, the performance drop at those message sizes is largely mitigated, and the aggregate bandwidth is no longer significantly impacted.

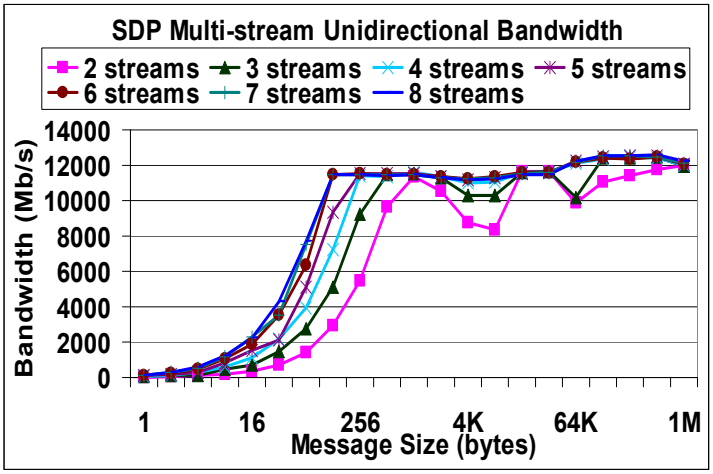


Figure 4.12: SDP multi-stream bandwidth

We have found that SDP provides the lowest latencies and the greatest bandwidth. For multi-stream bandwidth, both IPoIB and SDP have higher available bandwidths than 10GigE, resulting in maximum gains of ~9.6% and 26.2%, respectively.

Examining the results in Figure 4.13 shows that the effect of combined IPoIB and Ethernet (jumbo frame) traffic over a single HCA using different ports for each traffic type has the effect of increasing aggregate bandwidth up to ~11300 Mb/s for Ethernet/IPoIB simultaneous traffic, or 10.7% below the verbs maximum and 12.3% greater than 10GigE. There is also an uneven sharing of the bandwidth between IPoIB and 10GigE, which grows as more streams are utilized. For a single stream, the sharing is even at the highest aggregate bandwidth point. Therefore for 10GigE there is an uneven sharing of the bus occurring, representing the difference in behavior

and efficiency between IPoIB and 10GigE. This effect is more prominent with the normal frame results in Figure 4.13 with the aggregate bandwidth consequently being reduced to ~10550 Mb/s. The unevenness of the sharing is shown by IPoIB using up to 83.7% of the available bandwidth for normal frames, while for jumbo frames, IPoIB uses a maximum of 66.7% of the total bandwidth.

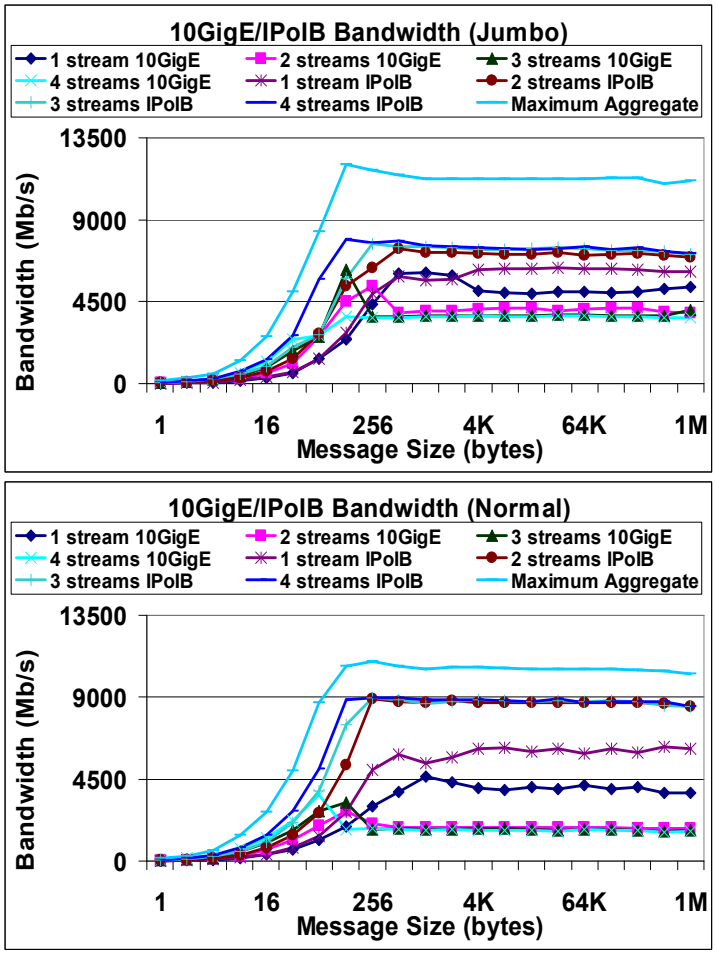


Figure 4.13: Simultaneous 10GigE/IPoIB bandwidth, jumbo and normal frames

The results of simultaneous Ethernet and SDP traffic are illustrated in Figure 4.14. This illustrates that no significant blocking is occurring and both traffic streams can exist harmoniously with an aggregate utilization within 1.2% of verbs bandwidth and 26.7% higher than with 10GigE alone for jumbo frames. The effect of bandwidth sharing can be a splitting of bandwidth within 1.2% of perfect division for larger messages with a smaller number of streams.

As the number of streams increases, the sharing declines to up to a 2 to 1 ratio of SDP over Ethernet, which will be required in future applications with PCIe Gen3. As was the case for IPoIB, the fairness of sharing the available bandwidth declines as the number of streams increases. The maximum usage of the available bandwidth occurs for large numbers of streams, where SDP can consume up to 66.8% of the available bandwidth with Ethernet using jumbo frames. Although the maximum usage is similar to the IPoIB/Ethernet case, the average utilization is only 57.9% for SDP while it is 62.3% for IPoIB, indicating better sharing for the SDP case. For normal Ethernet frames, the maximum aggregate bandwidth is similar to that for jumbo frames, being ~12400 Mb/s versus ~12450 Mb/s, but the sharing is not nearly as equal. SDP dominates the network for normal Ethernet frames, using up to 83.7% of the maximum bandwidth.

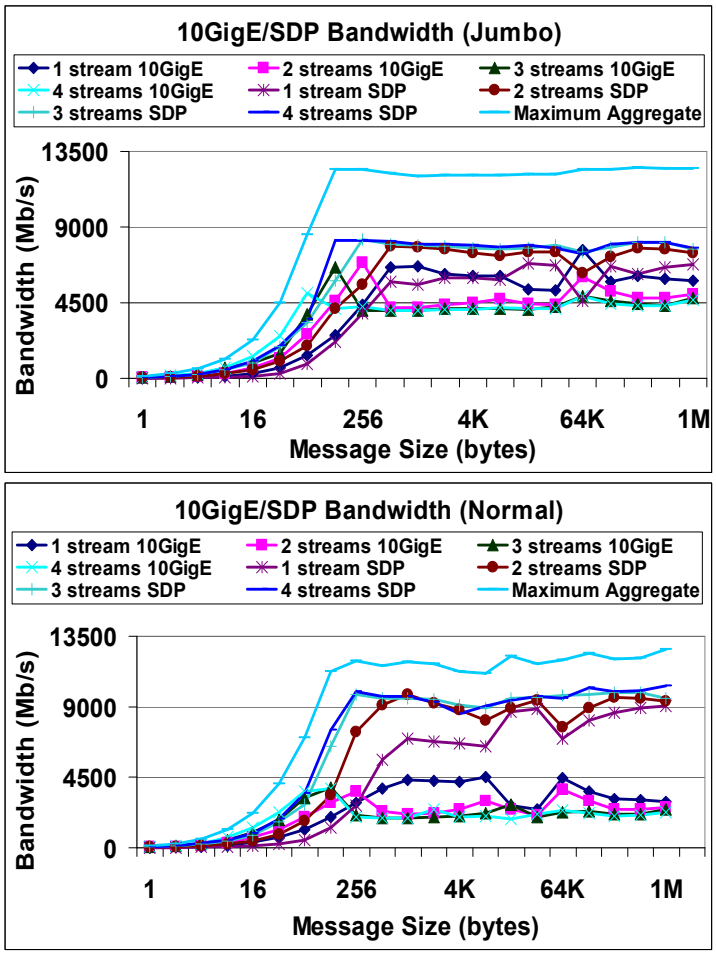


Figure 4.14: Simultaneous 10GigE/SDP bandwidth, jumbo and normal frames

Therefore, we can now conclude that SDP and Ethernet can harmoniously share a large proportion of the available bandwidth without causing any blocking to their simultaneous traffic partner protocol when Ethernet operates using jumbo frames. IPoIB and Ethernet can also share the bus effectively when Ethernet uses jumbo frames, but to a lesser degree than that of SDP/Ethernet.

4.3.2 Data Center Test Results

Now that we have established that VPI can be of use in a multi-network environment we must determine if VPI can be used effectively in a data center environment. The new setup of the data center is shown in Figure 4.15, with the possible interconnections between each tier shown. Note that while all of the nodes ConnectX cards are operating in VPI mode, only the Web server is actively sending data over Ethernet and IB at the same time. Therefore, the client to web server connection is always 10GigE, while the other connections are either all 10GigE, all IPoIB or all SDP. The underlying data center infrastructure remains the same as in Section 3.3.2.

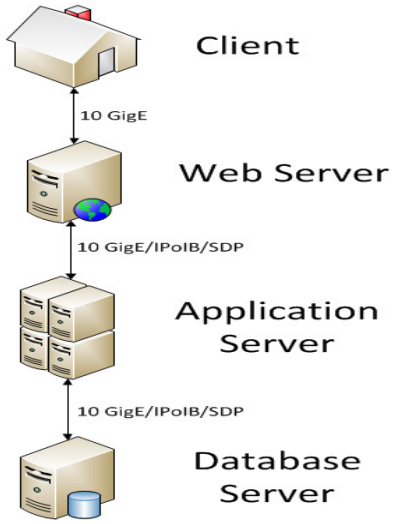


Figure 4.15: Test data center architecture

In order to evaluate the performance results from the varying data center architectures, we must first compare the overall throughput of the three types, shown in Figure 4.16. The data center setups were run with the highest workload possible that did not generate any errors over the execution period. We can observe that for the testing period, the 10GigE/IPoIB data center

architecture had the highest throughput, averaging 87.15 interactions per second. 10GigE/SDP has a throughput of 85.08 interactions per second for the given workload mixture. The all-Ethernet architecture has a slightly lower throughput of 83.06 interactions per second. The increased throughput performance of the 10GigE/IPoIB and 10GigE/SDP data center architectures is expected as they provide higher available bandwidths than Ethernet technology. This performance increase is only achievable because of improvements in data center intra-communication.

The latency results in Figure 4.17 show the percentage of the number of a certain request that take longer than the specified response time, that is, a complimentary CDF. Therefore, smaller numbers are better for any given time. Comparing Figure 4.17(a) with Figure 4.17(b) for latency results, we can see that the latency difference between an all-Ethernet and 10GigE/IPoIB VPI data center is small, with an advantage to 10GigE/IPoIB. 10GigE/IPoIB shows benefits for larger operations, such as the new products display, that requires significant interaction between the application server and the database server. There are also improvements in the buy request action, the shopping cart request and the display of the home page. Therefore, there is a net benefit to using a 10GigE/IPoIB based VPI data center as it outperforms an all-Ethernet case, and this is best seen in the most back-end intensive requests.

Comparing Figure 4.17(a) with Figure 4.17(c) for latency results, we find that the 10GigE/SDP VPI data center is in fact providing the best latencies for order display and admin functions, illustrating that the most intensive operations benefit from using an 10GigE/SDP VPI data center configuration. However, it also shows higher latencies for functions that are not very back-end server processing heavy. The given load created for the TPC-W browsing mix will not necessarily guarantee that all of the nodes in the data center are continually loaded to levels greater than the interconnect capacity. When the network is not fully loaded, SDP cannot take advantage of pipelining to reduce overall latency. Given that the data center utilizes many connections in between the given tiers (as many as 250 between the application server and web

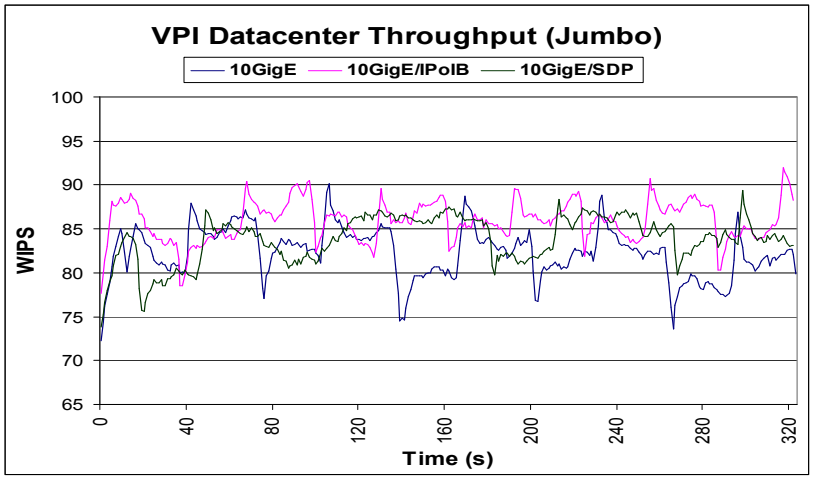


Figure 4.16 TPC-W VPI data center throughput (jumbo frames)

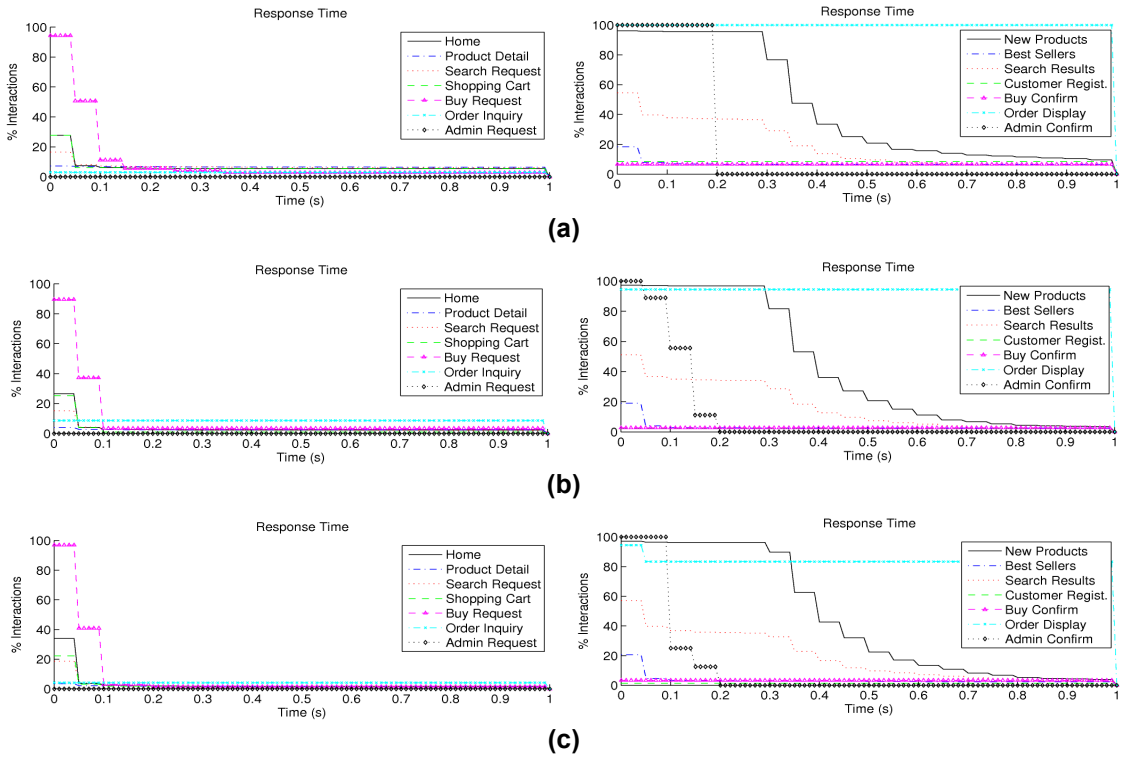


Figure 4.17: TPC-W jumbo frame data center latency: (a) all 10GigE, (b) 10GigE/IPoIB, and (c) 10GigE/SDP

server), even though the outgoing traffic may be sufficient to load a single outgoing connection, the load is distributed amongst many connections and pipelining is not as effective as it would be with fewer connections. Because SDP operates over RDMA semantics, the overhead of the required control messages can cause additional latency that would not otherwise be seen over a

heavily loaded connection, where such control messages can be hidden in the transmission latencies of incoming/outgoing traffic by pipelining the RDMA control messages.

An additional factor that might be affecting the performance of the system is the poor performance of Java's networking protocols over InfiniBand. This is reducing the potential benefit that can be seen through the application server layer as it relies on Java for dynamic processing.

Using the same framework that was used for the jumbo frame results, a data center using normal frames was tested. The results of this tests can be observed in Figures 4.18 and 4.19. The pure 10GigE data center sees the greatest drop in throughput, 75.76 WIPs for normal frames vs. 83.06 WIPs for jumbo frames. 10GigE/IPoIB and 10GigE/SDP fair better, with drops of 85.71 WIPs vs. 87.15 WIPs and 83.71 WIPs vs. 85.08 WIPs, respectively. This demonstrates that the use of normal frames is decreasing network efficiency and network processing overhead such that a small drop in throughput is seen.

The all 10GigE data center configuration suffers the most from the use of normal frames as it affects all levels of the data center. The all 10GigE data center shows a throughput of only 91.2% that of the jumbo frame configuration, or a drop in performance of 8.8%. The performance penalty for the10GigE/IPoIB and 10GigE/SDP configurations are only 1.7% and 1.6% respectively.

Examining the latency results in Figure 4.19, it can be observed that the overall latency for all of the configurations has increased. The more complex operations see greater increases in latency. This can be due to the more complex operations requiring more communication with the data center, but could also be a direct result of overhead due to extra packet processing at the Ethernet level for other bandwidth intensive tasks that are operating in parallel. This increase in latency coupled with the more erratic nature of the throughput results illustrates that the normal frame data center configuration is less able to smooth out spikes in demand than the jumbo frame data center.

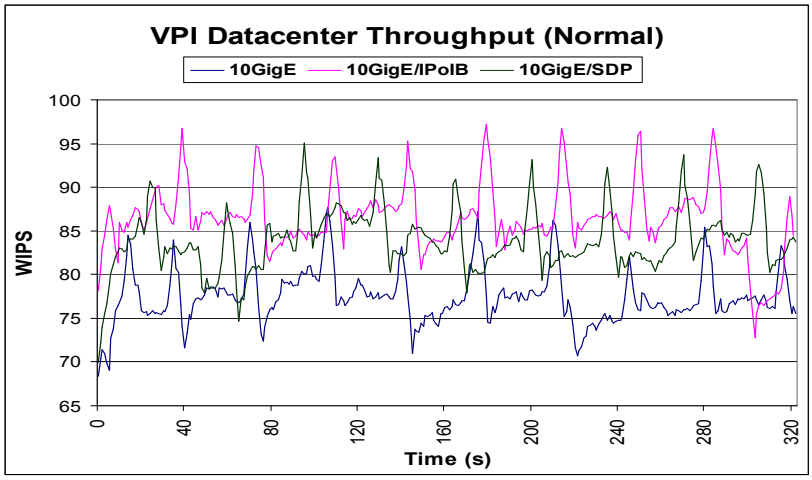
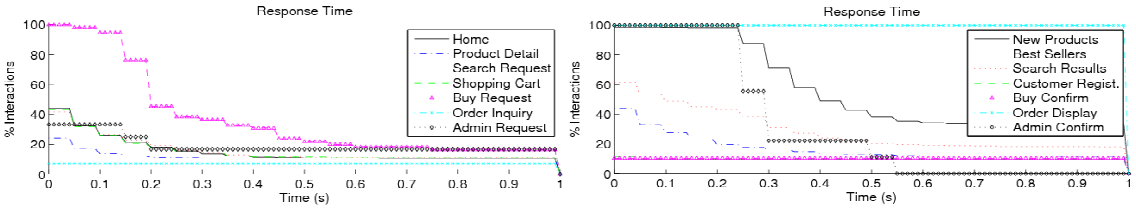
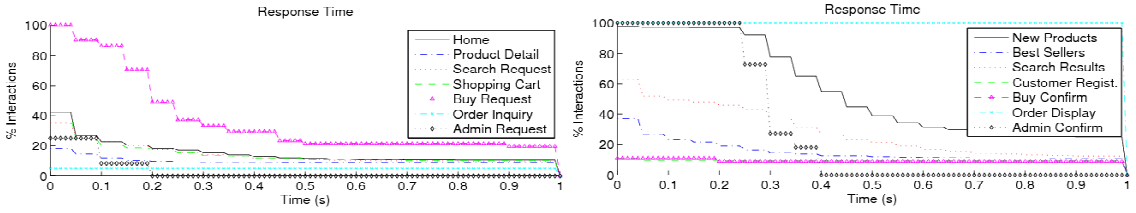


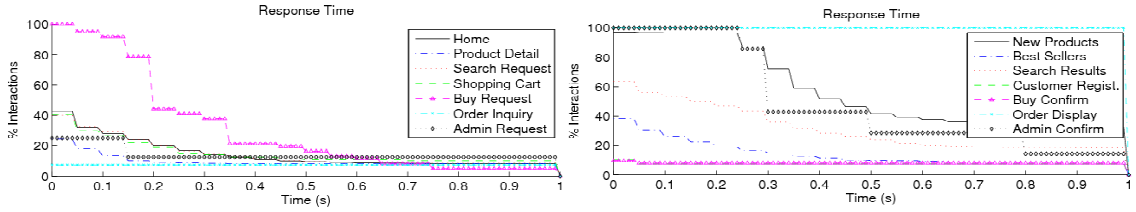
Figure 4.18: TPC-W data center throughput (normal frames)



(a)



(b)



(c)

Figure 4.19: TPC-W normal frame data center latency: (a) all 10GigE, (b) 10GigE/IPoIB, and (c) 10GigE/SDP

Examining the results of the data center tests and combining the observations with the 10GigE/IPoIB and 10GigE/SDP simultaneous bandwidth, we can conclude that hybrid data center architectures and VPI can be of benefit in several situations. We have observed that such hybrid systems can be of use in a dynamic web serving content system, where intra-data center communication latency and bandwidth are important.

4.4 Discussion

The implementation of SDP over XRC has demonstrated that the connection management overhead at the hardware level for SDP is not the main cause of its unexpectedly poor performance in a data center context. Although XRC has very little additional overhead when used in conjunction with SDP, and the benchmarks indicate that SDP's performance over XRC should be excellent, the data center performance results remain lower than expected. Given the results of the highly loaded client to web server SDP data center in Chapter 3, of almost 100 WIPs when the incoming channel is packed with greater amounts of incoming traffic it is clear that the underperformance of SDP is due to overhead associated with underutilized connections. Although the possibility of having an SDP connection between the client and the front end of the data center is unlikely to happen in the near future as client networking technologies are not that advanced, nor is SDP over IB capable of providing WAN service easily. A proxy layer could be added to aggregate traffic and feed higher traffic levels through smaller numbers of connections to a web server. but this would add layers of complexity that may not be cost-effective or desirable as technologies like VPI are applied to the data center.

VPI was shown to be effective in providing a hybrid data center that is capable of performing on the same level as that of an all InfiniBand data center. The non-hybrid all 10 Gigabit Ethernet data center architecture is also close in performance, by less than 10%, to the hybrid data center architecture when using jumbo frames. Therefore for typical web serving data center architectures, both alternatives are a good choice, with Ethernet potentially being less expensive, but the IB network having greater flexibility. Given that both the Ethernet and IPoIB networking solutions still use all or parts of the kernel networking stack, the performance of both networks can be enhanced by using OS bypass. As SDP is an OS bypass method, it was expected to be superior to IPoIB and Ethernet. However, it is demonstrated that for web serving data centers, this is not the case. For more bandwidth intensive commercial applications, SDP may have improved performance, but its TCP nature and its implementation limit it in the application space

studied. Therefore, with a UD IPoIB offloaded network architecture in Chapter 3 being the best in terms of performance, and a hybrid 10GigE/IPoIB being the best hybrid architecture in this Chapter, we can make reasonable conclusions as to what might be the most effective network design features to integrate into a single networking solution.

Given the results from this chapter and Chapter 3, a scalable Ethernet-based RDMA capable network is highly desirable for future networks. Therefore, the next two chapters detail the design, implementation and testing of an RDMA capable, unreliable datagram transport that works over native Ethernet networks. It also provides the first design for any verbs-based network to provide OS-bypass capable one-sided RDMA Write operations over an unreliable datagram transport. These designs logically follow the best features of each of the network and protocol designs studied, in the hopes of designing a superior, backward-compatible, cost effective future network design.

Chapter 5 Send/Recv iWARP Using Datagrams

Ethernet networks are some of the most ubiquitous computer networks in use today. Their wide adoption has made Ethernet networks available at lower cost than competing high-speed networks, despite lagging behind in cutting-edge network performance. Due to the large existing Ethernet infrastructure, it is desirable to upgrade existing networks piece-by-piece rather than overhauling networks via complete networking infrastructure replacement. Therefore, a high-speed Ethernet networking solution that can leverage the performance benefits of advanced networks such as operating system bypass, RDMA and offloading for both TCP and UDP traffic is very attractive. This chapter demonstrates that a method for using datagrams over iWARP is both feasible and desirable.

Some existing solutions to high-speed Ethernet have taken advantage of offloading technologies, with many offering abilities such as stateless offloading, by performing segmentation on the network adapter or calculating checksums. Solutions for stateful protocol offloading also exist, typically referred to a TCP offload engines or TOEs. Existing iWARP compatible network adapters [17] offer both stateless and stateful offloading capabilities on top of RDMA capabilities, enabling them to perform zero-copy based communications as well as bypassing the OS, providing both greater CPU availability and increased network throughput and decreased latency [41]. However, existing iWARP networking hardware provides these advanced features for TCP-based traffic. An overview of the iWARP stack versus a TCP/IP network stack is shown in Figure 5.1.

The current iWARP specification does not allow for RDMA operations to occur over a connectionless transport, and also requires that the underlying networking layers provide a reliable, in-order delivery of data. This has several limitations compared to a design that allows for unreliable data transmission and does not require the storage and manipulation of data associated with on-going network connections. For very large systems, or systems serving a very

large number of clients, the administration and storage of all of the data associated with these many individual connections can be onerous. Existing high-speed networking technologies such as InfiniBand [54] have acknowledged such limitations in connection-based networks by introducing technologies such as eXtended Reliable Connections (XRC) [54]. However, such technologies are not ideal for use in all situations as they try to reduce connection overhead by aggregating connection information between two communicating nodes for applications that open many connections between communicating nodes. Connectionless approaches eliminate this concern entirely by not incurring the overhead of connections in the first place.

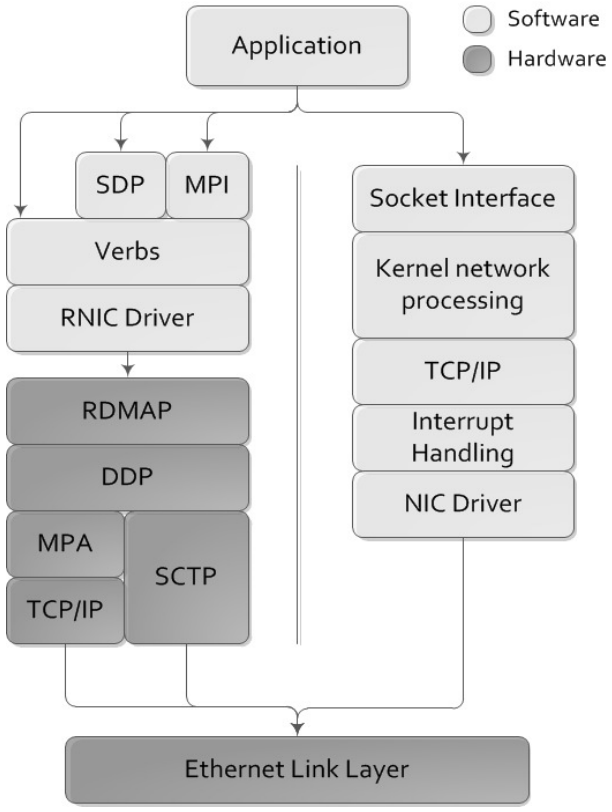


Figure 5.1: iWARP stack compared to traditional TCP/IP stack

A datagram-based solution offers flexibility to application developers in adapting the overhead of the network directly to their individual needs. For example, for a streaming application that requires time-dependent data, a reliable data stream may not be required as data may be useless if it has to be retransmitted as it is no longer current enough to be of any use. Alternatively, an

application could require reliable communication, but not need the flow control capabilities of TCP [46]. The reduced complexity of the datagram approach provides for faster communication and lower overhead than either of the current iWARP alternatives of TCP or *Stream Control Transmission Protocol* (SCTP) [82]. Datagram-based iWARP also reduces the complexity of the networking stack by not requiring the use of the Marker PDU Alignment (MPA) layer, as middlebox fragmentation is not an issue for connectionless transports. This is helpful in enhancing performance as well as reducing the complexity of datagram-iWARP hardware solutions. A high-level overview of the additions and changes to the iWARP stack that must be made to support the use of datagrams can be seen in Figure 5.2.

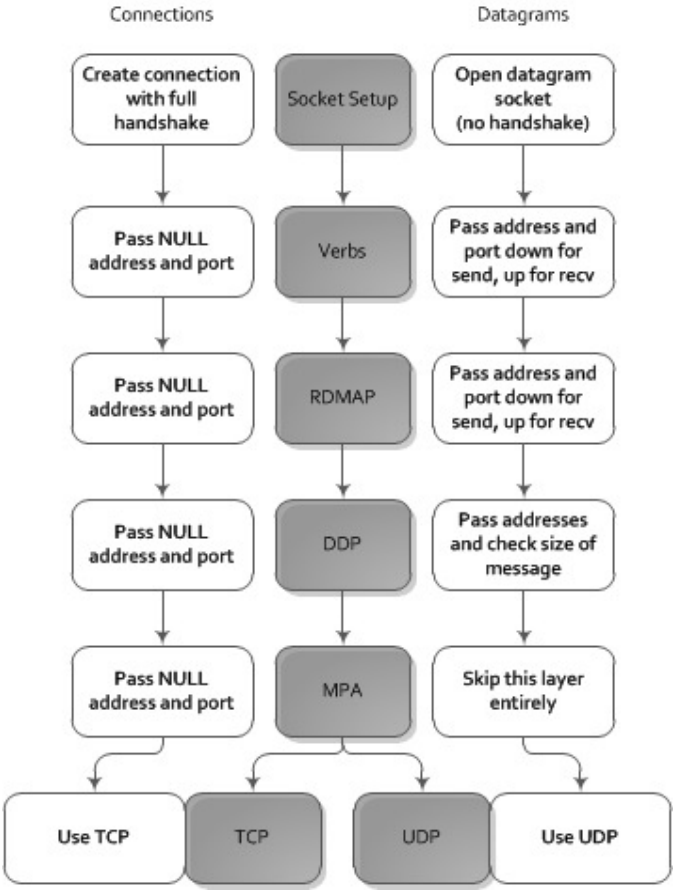


Figure 5.2: Additions to the iWARP stack data flow to support datagrams

Datagram-iWARP allows for the use of a much wider variety of applications than traditional iWARP as it allows applications utilizing datagram based semantics to use iWARP. Applications

that can traditionally make use of datagram based communication, such as media streaming or real-time financial trading applications, are part of the application set that are poised to make up ~90% of all Internet consumer traffic by 2014 [19]. VOIP and streaming media applications are typically built on top of protocols like Real-time Transport Protocol (RTP) [98], which can utilize UDP (datagrams) as a lower layer. Such applications have large throughput requirements, and therefore a hardware networking solution that is RDMA capable can reduce the burden on the CPU in transferring such large amounts of data to and from memory, thereby freeing up the CPU for other important tasks. This translates into potentially reduced CPU requirements for a system, which provides both a power savings as well as a cost savings in both initial costs as well as upkeep. Alternatively it can also result in increased system throughput, thereby increasing overall system efficiency.

5.1 Related Work

A software implementation of the iWARP stack by OSC [84] was used to develop the datagram-iWARP stack. The OSC iWARP implementation offers user-space [23] and kernel space [22] implementations of iWARP. Such software is useful in allowing for hardware iWARP capable system to enjoy the benefits of iWARP, while client systems need not have iWARP hardware to be compatible with the servers. The user-space version of the OSC iWARP stack was used for the datagram-iWARP implementation. iWARP software stacks like SoftRDMA [74] have been developed and integrated into the Open Fabrics Enterprise Distribution stack [85]. These software implementations have all been based solely on the traditional iWARP standard stack, and have not had any support for datagram modes of communication.

Performance evaluations of 10 Gigabit Ethernet NICs with offload engines have been performed [9][27]. The use of a UD transport over MPI was first demonstrated for InfiniBand networks in [69]. It uses a hybrid RC and UD channel design to reduce memory footprint while

providing high performance. MVAPICH-Aptus [67] is the latest version of this project and was used as the code-base for development of the iWARP send/recv MPI implementation using the OFED verbs interface.

The latest iWARP adapters available have been studied from a performance perspective. Ammasso iWARP adapters were studied in [24], Chelsio iWARP adapter performance was examined in [7], while NetEffect iWARP adapters were studied in [90] in comparison to IB and Myrinet networks. It was found that iWARP provided excellent latencies and throughput, while outperforming other interconnects under conditions with large numbers of simultaneous connections. In addition, iWARP had superior behaviour in the re-use of pinned memory buffers, but Myrinet was the superior technology in handling unexpected messages, as it offloads traversing the unexpected message queue to hardware, while IB and iWARP do not. Overall, iWARP had the superior bandwidth for small numbers of connections for most message sizes.

5.2 Advantages to Datagram-iWARP

The iWARP standard provides enhanced performance for reliable transports in both LAN and WAN environments. The standard is very useful for applications that require network reliability or work over existing TCP sockets semantics. The limitations of connection based transports that have been previously discussed (resource usage, complexity of TOEs), represent a problem for future systems. Local computing systems will have increased node counts and wide area systems will serve more clients per node than current hardware is capable of handling. The speed of existing iWARP RNICs is limited by the complexity of processing required for TCP streams, as the overhead imposed by the MPA layer as well as TOEs is not insignificant. The motivation behind this work is twofold. First we seek to provide RDMA functionality to a complete subset of applications (those using datagrams) that was not able to utilize such high performance networking technology. Second, we seek to "future proof" the existing iWARP standard by providing a scalable communication option that can provide good performance while allowing for

different reliability provisioning depending on individual application needs. Through the pursuit of these two goals, we also realize some other significant benefits [92]. The performance of iWARP can be enhanced through the use of less complex transport provisioning. The removal of the MPA layer is a design decision indicative of increased performance. The reduced complexity also provides the benefit of easy adoption of datagram-iWARP into existing iWARP silicon. Alternatively, datagram-iWARP provides the opportunity for a datagram offloaded iWARP solution while offering onloaded TCP processing. Such a solution would be much less expensive to produce given its lack of a TOE and MPA processing. Finally, datagram-iWARP provides the opportunity to provide broadcast [39] and multicast operations that could be useful in a variety of applications including media streaming and HPC for collective operations. This section will explore the benefits of datagram-iWARP for the untagged (send/recv) model in more detail.

Applications like online gaming or media streaming make use of unreliable datagram transports such as UDP. The overhead imposed by reliable transports is detrimental to such applications as individual datum are time dependent and limited to short time periods. For example the current heading and location of a moving object, or a single frame of a video. Receiving this information after more recent information has arrived is unnecessary. In the case of packet loss, TCP and SCTP both guarantee in-order delivery of data, therefore network jitter is experienced as there is a delay in delivering data that has already arrived but is blocked by an incomplete message at the head of the receive queue. For a datagram transport such as UDP, data is delivered as it arrives, and this helps to reduce network jitter, particularly for time-dependent data transmissions. These applications can make use of datagram-iWARP, enhancing their performance by using OS bypass, offloading and zero-copy techniques. These performance enhancing networking features are unavailable for such applications in traditional iWARP.

Scalability is another prime concern of iWARP. The communication channel isolation of the connection-based transports is excellent for flow-control and reliability of individual data streams, but it limits the resource sharing that can occur between the connections. Connections

must have a current state and communication over a connection can only have a single destination point. The resource requirements per connection are further increased through the design of some HPC middleware. Some MPI implementations pre-post receive buffers in support of Eager communication protocols on a per connection basis. This is beneficial to performance as memory is pre-registered, but uses memory for every connection. Although some schemes have been designed to reduce connection-based memory requirements in such environments, such as sharing receive queues, they are still not as efficient as a single datagram receiver of data. Datagram transports do not have to keep state information like a connection does, and for offloaded iWARP implementations, this state information must be kept in local memory on the RNIC, or else be accessed through slower system memory through requests over the system buses.

Reducing the complexity of the networking protocols through the use of UDP will help to reduce message latency and close the latency gap between iWARP and other high speed interconnects. Offloading the network processing onto the RNIC, will reduce the CPU requirements of providing high throughput traffic for datagram applications by lessening the data movement responsibilities of the CPU. This will enable businesses to concentrate their infrastructure investments directly into networking technologies and capacity, subsequently reducing the amount spent on CPU hardware while still supporting high bandwidth and low latency. This is a common benefit of WAN capable RNICs that can support both WAN and LAN-based applications. The other performance advantage of using datagram-iWARP is the existence of message boundaries. Unlike stream-based connections, having message boundaries recognized by the LLP avoids having to mark packets to later determine message boundaries. Packet marking is a high overhead activity and can be expensive to implement in hardware [21]. Therefore, avoiding it in datagram-based iWARP is a significant advantage over TCP. SCTP [82] does allow message boundaries like UDP, however it provides even more features than those in TCP and consequently is more complicated. SCTP is also not as mature as either TCP or UDP

and therefore does not have as much application support nor as long a history of performance tuning as either TCP or UDP.

The final major advantage to datagram-iWARP is its reduced implementation cost. The elements of datagram-iWARP design that improve performance also have a beneficial effect on the implementation cost. This means that adding datagram-iWARP functionality to existing iWARP RNICs would be relatively easy and inexpensive, the creation of a datagram-iWARP only RNIC would be less complex and expensive than a TCP-based iWARP RNIC. Due to this reduced silicon size, datagram-iWARP could also be used to create a highly parallelized RNIC that would be capable of handling many simultaneous requests by duplicating the communications stack processing pipelines. This would help in increasing network throughput by helping systems with many cores avoid delays due to resource availability conflicts.

5.3 Datagram-iWARP Design

Given a connection-less traffic flow, the source address and port of the incoming data must be reported to the application receiving the data. This requires that existing verbs be altered in order to pass along the required data structures for datagram-based traffic. Alternatively, separate datagram based verbs can be introduced to handle datagram traffic. Existing compatible verbs interfaces such as the OpenFabrics [85] verbs specification can allow for datagram based traffic (as send/recv datagram traffic is supported in other OF verbs compatible interconnects such as InfiniBand). Therefore, the iWARP specific verbs to handle datagram traffic can also be designed to be compatible with OF verbs through a verbs-to-verbs interface. An example of the additional verbs that support datagram traffic would be a datagram compatible post_send verb, and post_recv verb. Changes to verbs for polling the completion queue are also required, as are changes to verbs for creating and modifying queue pairs for datagram based communication. A high level overview of the changes required at each layer is shown in Figure 5.3.

As datagram QPs require different initialization information than connected QPs, the QP

creation/modification verbs need to have different inputs than existing connection-based verbs. The modification of datagram QPs also requires changes, not only to the inputs received but in the requirements for said inputs. For example, datagram QPs cannot be set up on-the-fly as data is delivered for transmission. Therefore, datagram QPs need to allow for a basic socket allocation and communication setup to occur when the QP is first created. This creates a valid socket that can then be used to transition the QP into the Ready to Send (RTS) state.

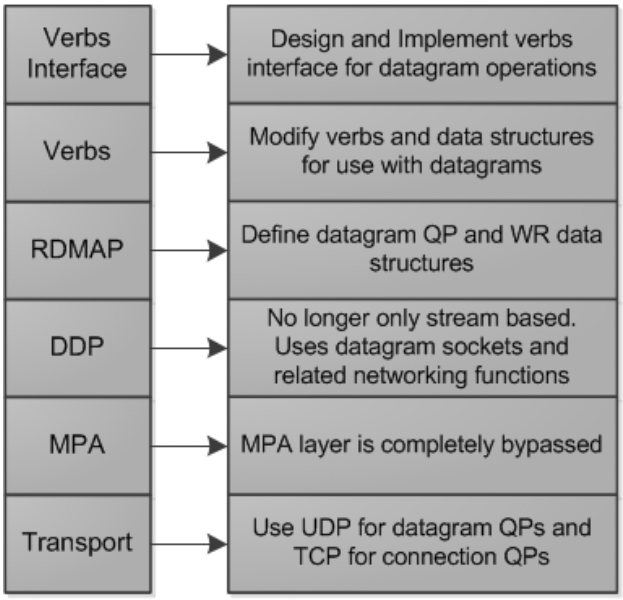


Figure 5.3 Overview of changes required for datagram-iWARP

Communication transmission verbs (send and RDMA Write) must be altered so that they receive a valid destination (IP address) and port with every operation, as they cannot rely on past behaviour to determine the destination for a given data transmission. For connection-based communication, this is not an issue as data flow over a given connection always flows to the same destination. Likewise, the verbs for receiving data need to be altered such that they deliver the address and port of the source of the transmission. For recv this is relatively straightforward, through the addition of data structures to relay this information. However, this is not done in the recv verb itself, but rather as a common change to the reception verbs, by reporting the source address and port in the completion queue element passed back to the requesting process after a completion queue poll request.

In the current iWARP standard, a work request is completed as soon as its delivery can be assured. For a reliable connection, this occurs upon passing the data to the underlying transport layer. For an unreliable transport, such an assurance can never be provided. Therefore, for datagram-iWARP, a work request completion occurs when the data is passed to the lower layer protocol (LLP) for delivery, without the expectation that it is guaranteed to be delivered.

The RDMAP specifications in the iWARP Standard [43] Section 5.1 states that LLPs must provide reliable in-order delivery of messages. The DDP standard [99], Section 5, item 3 states that the Lower Layer Protocol (LLP) must reliably deliver all packets. These specifications must be changed to allow for the operation of datagram-iWARP. It is proposed that the specifications for datagram-iWARP be changed to "may provide" and "may reliably deliver", with the requirement that for connection-based iWARP these conditions must occur.

The DDP Standard Section 5, item 8, states that errors that occur on an LLP stream must result in the stream being marked as erroneous and further traffic over said stream is not permitted. Likewise, the RDMAP specification requires an abortive teardown of an entire communication stream should an error occur on that stream. These requirements must be relaxed for datagram-iWARP, where for the case of a datagram QP, the error must be reported to the upper layer, but no teardown occurs and traffic can continue to traverse across the QP. This is necessary as the QP might be communicating with multiple targets or receiving data from multiple sources. Also an error occurring over an unreliable transport may be recoverable.

The iWARP standard has been designed for a LLP that offers guaranteed in-order delivery of data. Subsequently, for datagram-iWARP we must adapt the behaviour of the communication stack such that it can be compatible with an unreliable transport. For many cases, the applications utilizing a UD iWARP must be aware of its unreliable nature and have provisions to deal with data loss and re-ordering. This is the case for many types of applications such as streaming media applications, and even includes some implementations of HPC application middleware such as MPI. Alternatively, for the datagram-iWARP specification over a reliable datagram (RD)

transport, one that provides both delivery and order guarantees, many of these behavioural adaptations are not required. Therefore, unless expressly stated, the proposals in this section refer to those for iWARP operating over an unreliable transport.

The lack of delivery guarantees for datagram-iWARP requires that the polling for the completion of a recv operation as well as the completion of an RDMA Write-Record operation must have a timeout period associated with the polling request. Otherwise, an infinite polling loop may result, which would cause an application to fail.

Connection establishment and teardown procedures are not required for datagram-iWARP. For datagram-iWARP, a QP can be transitioned into a RTS state after a QP is created and an LLP socket is assigned to the QP. As this setup can be accomplished without any transmissions to the target of the datagram QP, the other parameters used for the QP, such as the use of CRC, must be configured by the ULP. This also eliminates the requirement that the ULP is required to configure both sides of the QP (target and source) at the same time.

As the LLP may or may not handle reliable service and order guarantees, any requirements for state information on underlying LLP pseudo-connections is the responsibility of the LLP and not that of iWARP. Therefore any associated pseudo-connection establishment/modification or termination that is required must be the responsibility of the LLP.

Error management behaviour must be changed in the iWARP stack. For datagram-iWARP, errors must be tolerated, by reporting errors as they happen to the upper layer, but not causing a complete teardown of the QP. Consequently, the application may decide on the course of action to take upon the notification that a communication error has occurred. For the case in which we are using a RD transport, the error is passed along to the application in the same manner as for a UD LLP, but the QP is placed into an error state. The error state prevents any further traffic on the QP until the error is dealt with by the application by resetting the QP state into the RTS state. This also requires that an error message detailing the error and in the case of a two-sided communication, the message sequence number to the message source. The error message is

locally placed in an Error queue as opposed to the Termination queue that would be used in traditional iWARP.

Message segmentation and the requirement for marking of messages is significantly altered for datagram-iWARP. As the proposed ULP, UDP has a maximum message size and UDP delivers said message in its entirety there is no need for message segmentation in the iWARP stack. It is the application layer's responsibility to ensure data transfers larger than 64KiB are segmented and reassembled properly. The MPA layer and its marking of packets in order to facilitate reassembly is also not needed. As datagram packets are not permitted to be further segmented by network hardware along the transmission route (unlike TCP traffic), there is no need to perform the costly activity of marking the packets so they can be re-assembled correctly at the target system.

Datagram-iWARP always requires the use of Cyclic Redundancy Check (CRC32) when sending messages. This requirement allows for the creation of datagram QPs without any communication between the source and target nodes as well as ensuring that no conflicts occur in terms of CRC use over a datagram QP, given that it can communicate with several other systems. As the proposed LLP for the UD transport, UDP, does not require the use of CRC, it is required that it be performed in the iWARP stack such that data transmission errors are identifiable.

5.4 Software Implementation

To evaluate the proposed datagram extension to the iWARP standard we have developed a software implementation of datagram-iWARP based on a TCP-based iWARP implementation from Ohio Supercomputer Center (OSC) [84]. This allowed for a direct comparison between the UD and RC modes of datagram-iWARP, using a publicly available RC implementation. An overview of this implementation including the additional upper layer interfaces that we have added, namely the iWARP socket interface, compatible MPI layers as well as modification to the existing OF verbs interface in order to make it compatible with more OF verbs middleware and

applications. The changes required to the verbs, RDMAP, and DDP layers, as described in Section 5.2, are reflected in the layers of the stack indicating both UD and RC support shown in Figure 5.4. While this implementation is fully capable of operating over a reliable UDP transport, all of the testing and results of the implementation were performed over an unreliable UDP.

Modifications to the existing incomplete OF verbs interface on top of the native software iWARP verbs was necessary due to the base RC iWARP implementation that was used for the design. OF verbs were originally designed for InfiniBand (OpenIB verbs) but are now also used to support iWARP hardware in a unified driver. As the RC implementation uses native iWARP verbs, they must be translated over to OF verbs for use at the MPI layer. An existing MPI implementation [67] was adapted to use the OF verbs interface of our iWARP stack. A socket interface is also added, to facilitate the use of socket-based applications without the need to re-write their existing networking code.

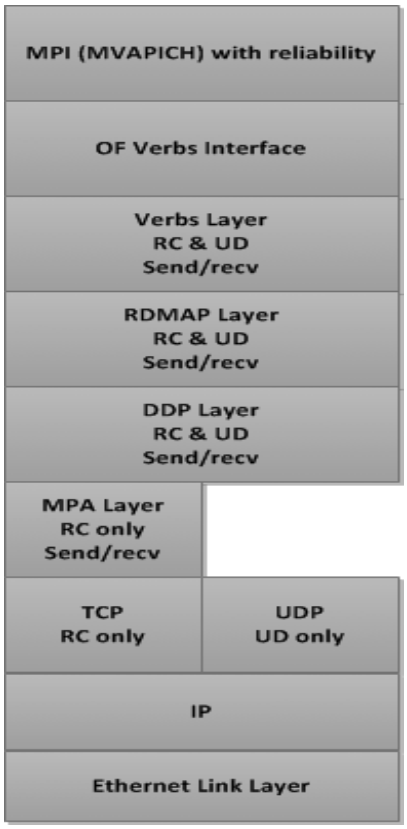


Figure 5.4 Software implementation of datagram-iWARP

Our datagram-iWARP implementation uses CRC error checking at the lower DDP layer for datagrams to ensure correct reception of individual packets. CRC checks may be performed at the UDP layer as well, and thus it would be redundant to do so at the DDP layer as well. Therefore, it is recommended that CRC checking be disabled at the UDP layer to further enhance performance. The implementation uses round-robin polling over sockets, to ensure fair service to all QPs in the software RNIC. It has been enhanced to use I/O vectors for UDP communication (similar to TCP) to avoid extra sender and receiver side copies, for improved performance. I/O vectors enable the gathering/scattering of data from non-contiguous memory locations in a single operation avoiding any intermediate memory copies when assembling a datagram with its header and associated data payload. Our implementation also avoids segmentation of DDP messages into MTU-size datagrams, using the available Ethernet hardware segmentation provisions.

5.5 Experimental Results and Analysis

Two computing clusters were used to conduct the evaluation. The first cluster, named C1 has four computing nodes, with two quad-core 2GHz AMD Opteron processors, 8GB RAM and a NetEffect 10-Gigabit Ethernet (10GE) card on each, connected with a Fujitsu 10GE switch. The OS used was Fedora 12 (kernel 2.6.31). The second cluster, C2, has 16 compute nodes, with two dual-core 2.8GHz Opteron processors, 4GB RAM and a Myricom 10GE adapter on each, connected through a Fulcrum 10GE switch. The OS used on cluster C2 is a Ubuntu Linux distribution, kernel version 2.6.27.

5.5.1 Micro-benchmark Performance Results

Micro-benchmarks were used to test the performance of datagram-iWARP compared to a traditional TCP-based iWARP implementation on cluster C1. All results are an average of a minimum of 10 test runs, using the microbenchmark suite from OSC iWARP software release, with the datagram microbenchmarks adapted directly from the TCP based code. For the verbs level, the latency results for native iWARP verbs is shown in Figure 5.5. Figure 5.5 combines the

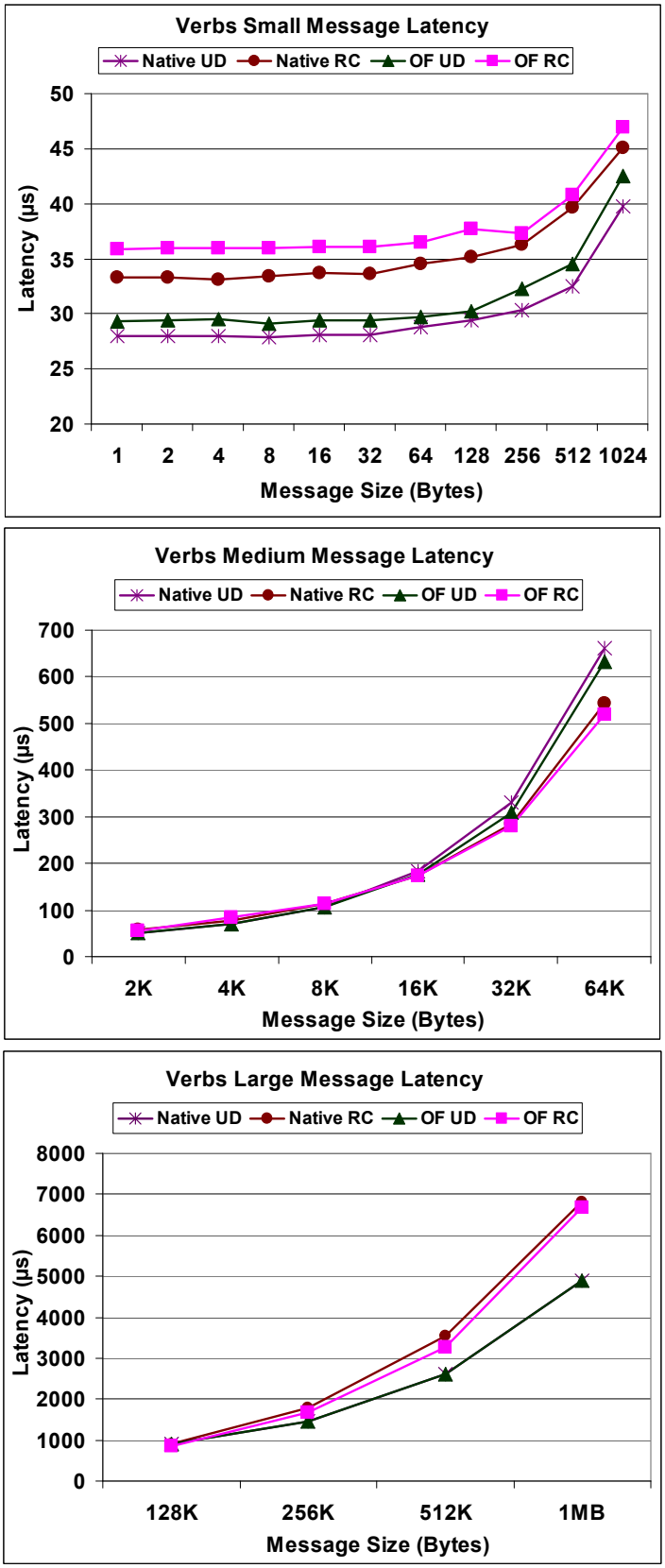


Figure 5.5: Verbs ping-pong latency

ping-pong latencies for send/recv in both RC and UD modes for both native (original iWARP) verbs and the OF verbs interface. The lowest latencies are for UD send/recv of 27-28µs for messages less than 128 bytes. The OF verbs interface introduces a small amount of latency but does not alter the overall order of the results.

Small messages, those under 2KiB in size, realize an 18.1% improvement in performance by using UD send/recv over RC send/recv. RC send/recv has a slight performance advantage for medium sized messages, those messages between 16KiB and 64KiB in size. The UD send/recv then demonstrates superior performance for large messages of 128KiB or greater. The reason for this performance advantage at medium message sizes for the RC mode is that at a 64 KiB message size the maximum size of a datagram message is reached and the message must be segmented into individual 64KiB chunks before they are passed to the UDP layer, this introduces a small amount of overhead. For larger messages this overhead can be hidden more effectively, so UD performs better than RC.

Datagram-iWARP has latencies that are 5-6 µs better than traditional RC iWARP for small messages. The reasons behind this improved performance are the use of the simpler UDP transport, and the exclusion of the MPA layer processing. The local area network environment in which these tests were performed also provides very low network transmission error and packet loss rates, which helps to increase the effectiveness of the UD -based transport approach.

For the unidirectional bandwidth results shown in Figure 5.6, it can be observed that the native verbs bandwidth of datagram-iWARP is superior to traditional iWARP. Native UD send/recv demonstrates this performance advantage, having an improvement of 33.4% over native RC send/recv at 256KB messages. The reasons behind this performance advantage are the same ones that were observed being advantageous for latencies, lowered transport protocol overhead and lack of MPA layer.

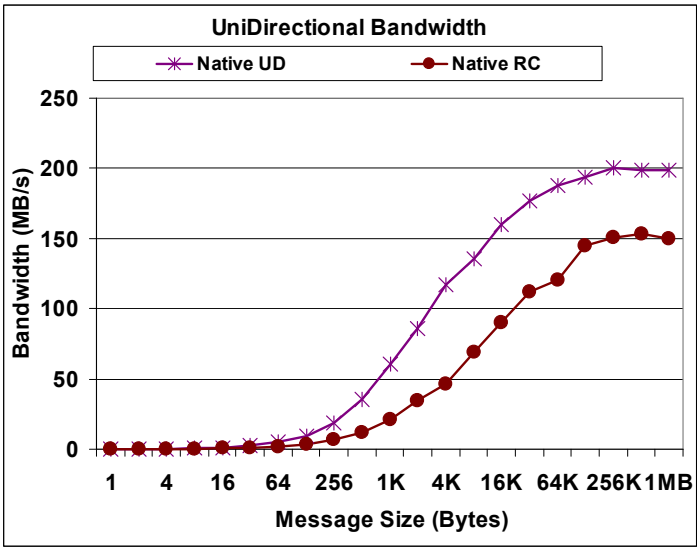


Figure 5.6: Unidirectional native verbs bandwidth

Broadcast Microbenchmark Results

To illustrate some of the benefits that datagram-iWARP has over traditional iWARP, datagram-iWARP was extended to include support for broadcast operations via the IP-based broadcast provisions. Broadcast essentially replicates packets and sends them to multiple end nodes in the network. As the interconnection between the systems-under-test are relatively simple, the functional demonstration of broadcasting for iWARP send/recv shows the advantages of packet replication at the hardware layer. Receiver managed data placement is necessary for broadcast operations as the replication of packets means that every broadcast packet is the same when it arrives at the targets. Theoretically a broadcast/multicast operation could use sender managed data placement (RDMA Write), but only if all of the clients could have an identical STag and memory region for data reception.

Examining Figure 5.7, we expect that the bandwidth will increase relatively proportionally to the number of clients. What is observed follows these expectations, but begins to show diminishing returns for large message sizes. The results for two and three receivers scale well for small and medium messages. As the message size grows, we begin to experience greater overhead associated with the copying of large messages by the broadcasting switch, leading to

non-ideal scaling (less than the single receiver bandwidth multiplied by the number of receivers). This overhead results in approximately 12-13% less bandwidth than an ideally scaling case, for messages greater than 64KiB with 3 receivers. The overhead of the broadcast operation is not excessive, and is preferable to having the software iWARP stack replicate packets and sending them to multiple targets. There is also a slight change in the slope of the bandwidth curve at the 8KiB message payload size. At this size, the local network MTU is exceeded and packets need to begin to be segmented into multiple packets at the IP level. This extra processing causes a slight drop in the efficiency of the networking operations.

The results for the functional demonstration of broadcast operations over send/recv are easily translatable to multicast operations as the broadcast is a generalized version of multicast where every end node in the local network is automatically a member of the multicast group.

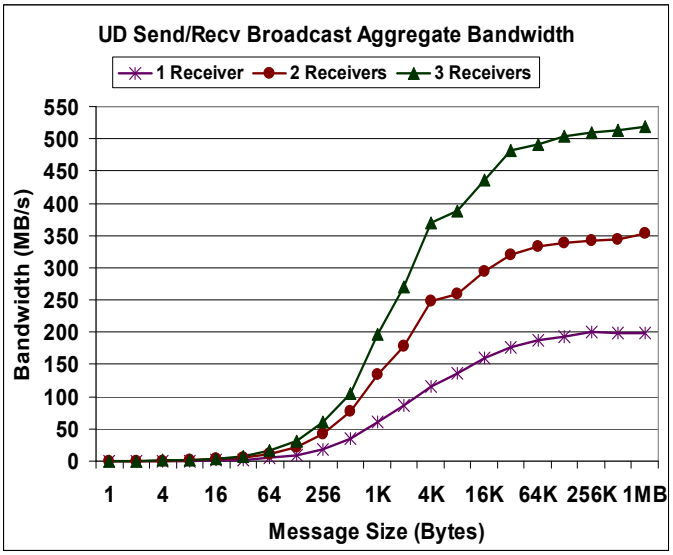


Figure 5.7: Send/Recv broadcast aggregate bandwidth results

MPI Microbenchmark Results

As the MPI layer is an important communication middleware it is also of interest to observe the micro-benchmark results for latency and bandwidth at this layer. These results are for MPI which interfaces with the iWARP software implementation through a lightweight OF verbs interface that adds very minimal overhead. The majority of the difference between the verbs

level tests and the MPI level benchmarks is caused by the overhead of the MPI middleware layer itself. It should also be noted that MPI is adding some basic lightweight reliability mechanisms which add some additional overhead as well. These methods have been seen to be effective for the LAN environments that MPI typically runs over.

For the MPI microbenchmark tests in this chapter, all results were obtained using an average of 5 runs of the MVAPICH microbenchmarks with each run comprising at least 100 iterations. The ping-pong latency over MPI is shown in Figure 5.8 for send/recv modes of iWARP operation. Results show the same trend of superior performance of the datagram-mode iWARP over traditional iWARP for MPI performance.

Figure 5.9 shows the bi-directional bandwidth for send/recv and RDMA modes. The bi-directional bandwidth test uses two pairs of processes on two nodes, communicating in opposite directions. One of the processes of each pair posts a window of non-blocking receive calls (if applicable), while the other posts a window of non-blocking send calls and they synchronize at the end of the test. MPI in either UD mode offers a higher bi-directional bandwidth for most cases than the comparable RC mode. The improvement for send/recv is on average 14.5% for large messages.

The drop in performance for the UD mode and the leveling out of bandwidth for the RC mode at the 8KiB message threshold is a direct result of the workings of MPI. The 8KiB message size is the threshold switching point for the Eager and Rendezvous protocols. When switching from Eager to Rendezvous, there is a small performance penalty. The rendezvous mode for both RC and UD modes still uses the send/recv communication mechanism, but a buffer is pre-negotiated prior to the data being passed to the transport layer. This avoids a memory copy for large messages or having to speculatively pre-post large recv requests, causing performance degradation and/or excessive memory utilization.

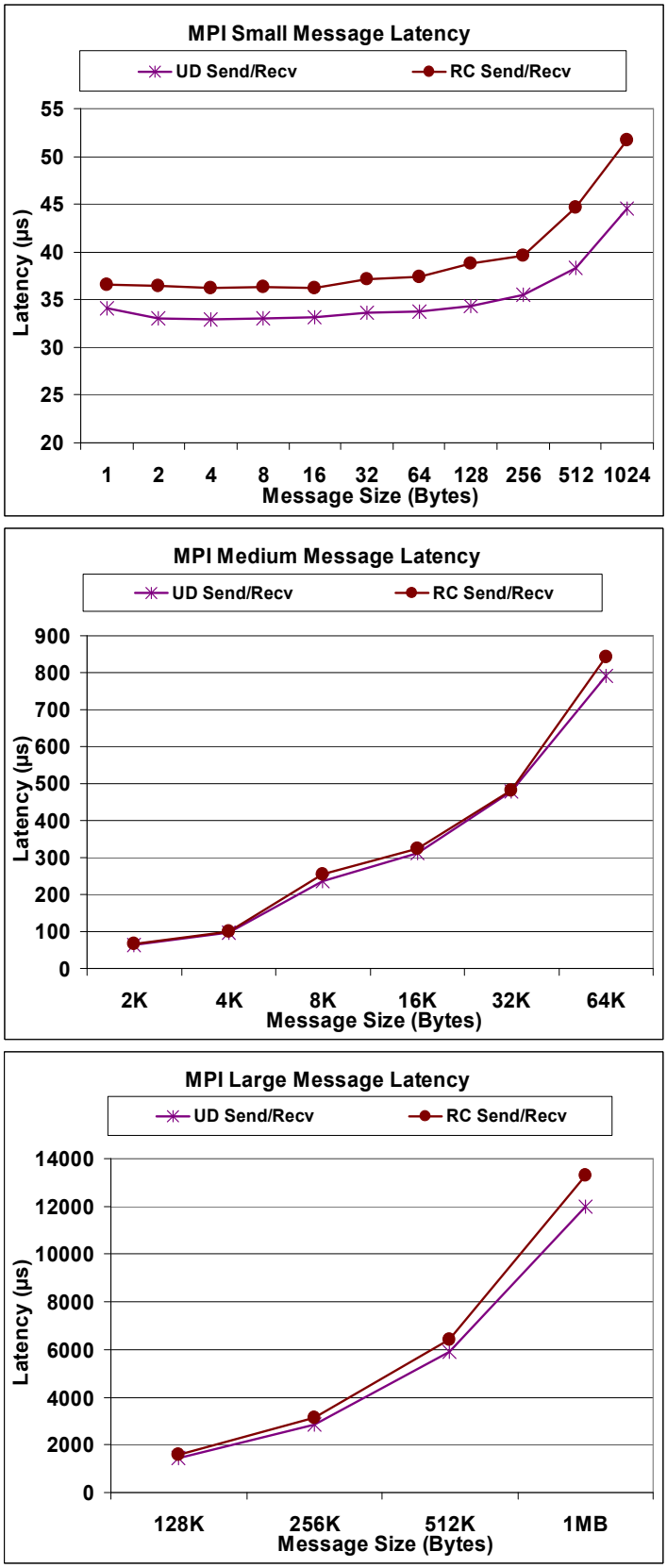


Figure 5.8: MPI ping-pong latency

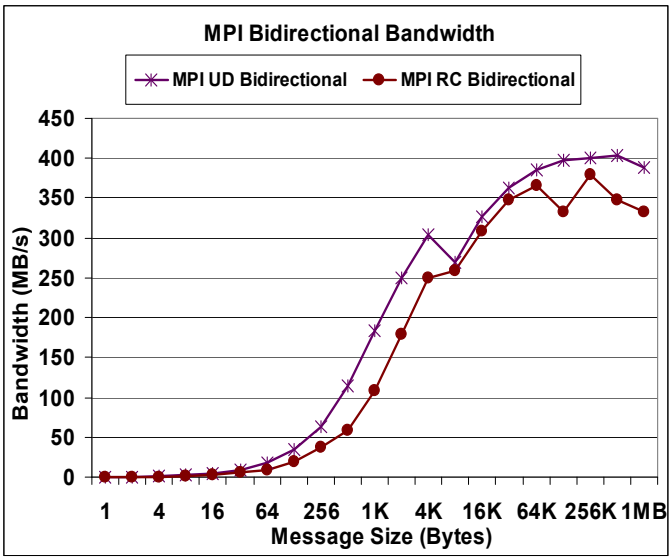


Figure 5.9: MPI bidirectional bandwidth

5.5.2 MPI Application Results

MPI application performance results are shown in Figure 5.10 for several applications from the NAS benchmark suite version 2.4 [80], CG.B, MG.B, and LU.B. Also included are results from two applications Radix [103] and a multi-grid solver SMG2000 [15]. The results show the total time the applications spend communicating as well as the total application runtime. Communication time was defined as all of the time spent during runtime executing MPI communication primitives. These tests were run on both clusters C1 and C2.

The results for the communication time for cluster C1 show that for larger numbers of processes, between 10% and up to greater than 30% improvement can be seen depending on the application. The results for the C2 cluster are even better, seeing gains greater than 40% for larger numbers of processes. For overall application runtime, the improvement is also quite good, with improvements of almost 20% for large numbers of processes on cluster C1. Cluster C2 sees even greater gains with over 45% improvement for the SMG2000 application. The reason behind the greater performance of the C2 cluster is that it has fewer cores on more nodes, thereby requiring more inter-node communication which lets the C2 cluster benefit more from improvements in the network communication efficiency.

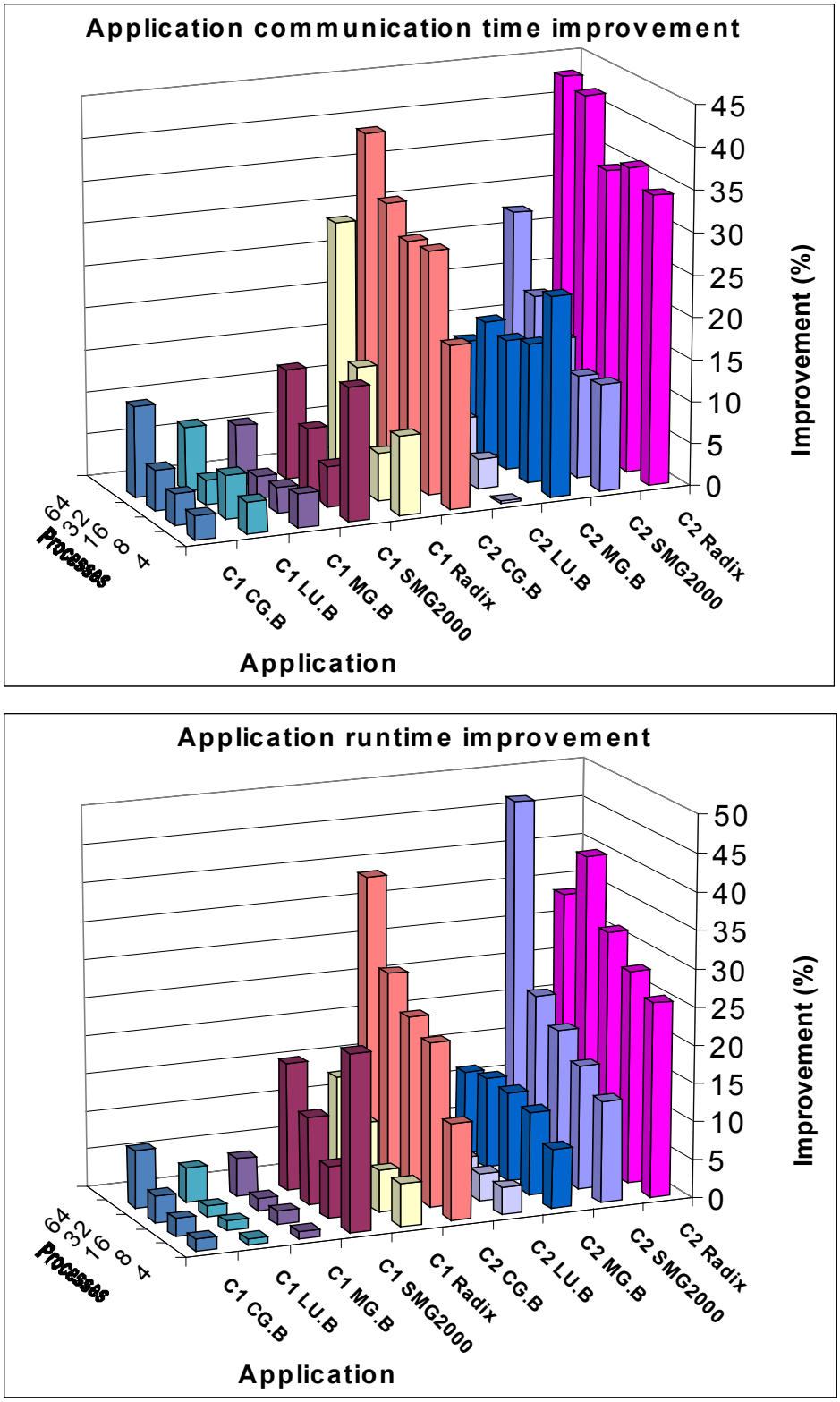


Figure 5.10: MPI application communication time and rumtime benefits for send/recv

The scalability of datagram-iWARP is also of interest in the area of high performance computing. Therefore, the memory usage of the MPI applications under study was determined for both UD and RC modes for comparison.

MVAPICH is designed to provide very high levels of performance, and subsequently allocates many resources to communication. For Eager protocols, provisions must be made to allocate buffers for incoming data of various sizes. In order to accommodate this MVAPICH pre-allocates a number of general buffer pools with different sizes for each process. When a QP is established a number of buffers are picked from these pools and pre-posted as receive buffers to the QP. A default number of 95 receive buffers are picked from the pools and posted to the QP. With an average size of 8KiB for each buffer, approximately 800KiB of memory are required per connection for each process. The reuse of a UD QP for communicating with several other nodes reduces the overall memory allocated to these buffers by not requiring a pre-posted set of receive buffers for every other node that a process must communicate with.

To measure the memory usage for the software-based iWARP, the total number of memory pages allocated to each MPI job in Linux is aggregated. Figure 5.11 shows the improvement in application memory usage of send/recv datagram-iWARP over RC iWARP. The results show that as the number of processes increases so does the percentage amount of memory saved. This demonstrates the scalability of datagram-iWARP, where the advantage of having fewer pre-posted buffers per node increases with the total number of nodes in the system.

Applications which do not perform communication with many other computational nodes have lower memory savings benefits than those that are more highly interconnected. This occurs for some many of the NAS benchmarks as the number of connections per node does not scale exponentially with the number of processes. Other applications such as Radix and SMG2000 are more interconnected and therefore show larger benefits from using datagram based communication.

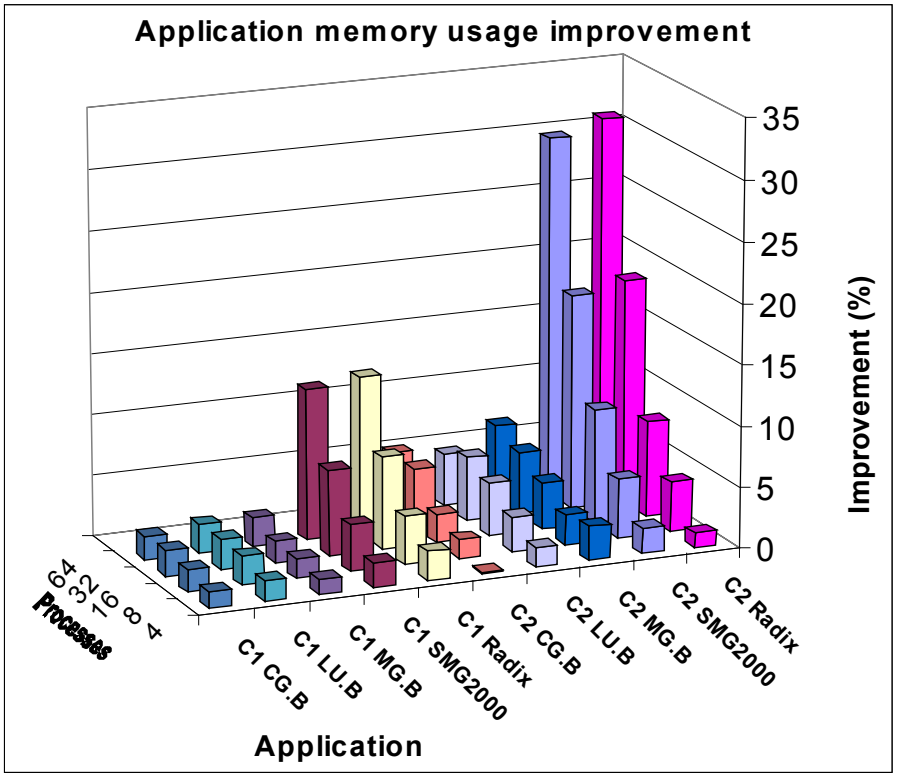


Figure 5.11: Memory usage improvement and scalability trend for send/recv

5.6 Discussion

The design of datagram-iWARP send/recv has demonstrated that the UD transport can provide better performance as well as scalability at the software level over a comparable TCP based stack. The removal of the MPA layer as well as the simplification of the underlying LLP transport in terms of reliability and TCP-based flow control has had the beneficial effect expected during the design of datagram-iWARP. It has been shown that the existing iWARP stack can be easily adapted to use a datagram transport. The adaptation of existing MPI implementations to use a datagram-iWARP send/recv communication mechanism has demonstrated increased performance at the application level as well as increased scalability. Increased scalability is especially important for send/recv mode for MPI as it is an excellent communication strategy for smaller message sizes using the MPI Eager protocol. As the Eager protocol uses many pre-posted buffers, its memory usage can be high when large numbers of connections are in use as many pre-posted buffers need to be used.

Some limitations exist on the current implementation of datagram-iWARP send/recv. First of all, the target process must always have a recv request posted before the data arrives, requiring that the target node be actively involved in any data transfer. A one-sided communication mechanism could help to alleviate this problem in the case that one does not want the target to be actively involved in data transfer. The send/recv model also requires that the target node handle all of the memory buffer matching, that is matching an incoming data payload to the appropriate memory location.

The datagram-iWARP implementation also has some limitations that are expressly required due to the send/recv model, but could be improved upon. For example, the handling of large messages inside the iWARP stack could be beneficial and it would be interesting to further examine. The performance of send/recv under packet loss is an outstanding issue that still need to be examined, as the WAN environments that iWARP may operate in are subject to packet loss.

The next chapter will address these issues through the introduction of a one-sided RDMA Write model for use over datagrams. It will examine large message performance and the performance of both send/recv and the one-sided model under packet loss conditions.

Chapter 6 One-Sided RDMA over Ethernet over Datagrams

In Chapter 5, we proposed send/recv datagram support for iWARP for use in high-performance computing environments, with corresponding reliability mechanisms. It was shown that for HPC applications, datagram-iWARP can provide significant memory savings of over 30% and runtime improvements of up to 40% on moderately sized HPC clusters, while providing higher bandwidth than connection-based iWARP. This chapter extends the work in Chapter 5 by proposing the first RDMA operation over unreliable datagrams that can significantly increase iWARP performance and scalability and expand the application space that iWARP can serve to include some very network-intensive applications.

In order to support RDMA over unreliable datagrams, we demonstrate *RDMA Write-Record*, a proposal for the design and implementation of, what is to our knowledge, the first RDMA design over an unreliable datagram transport. It is designed to be extremely lightweight and to be used in an environment in which packet loss occurs. This allows for its use in situations where data loss can be tolerated, but can be supplemented by a reliability mechanism (like reliable UDP) for those applications that cannot deal with data loss. RDMA Write-Record is particularly useful in the circumstances that a server is sending a large chunk of data to a client. In the situation where the client does not need a guarantee that all of the message will arrive entirely intact, such as streaming audio or online gaming, where missing data can be skipped over with little noticeable degradation, this can be of great use. The proposed design for RDMA Write-Record is not limited to iWARP and can be utilized in any other RDMA-enabled network such as InfiniBand [54], with minor modifications.

Applications in both HPC and data center domains can significantly benefit from the RDMA capable datagram-iWARP. VOIP and streaming media applications are typically built on top of protocols like RTP [98], which can utilize UDP as a lower layer. The large overhead that processing such huge amounts of data creates can be overcome by using a hardware RDMA

solution that offloads the intensive data shuffling activities from the CPU and performs them directly to/from memory. By using RDMA capable datagram-iWARP we can significantly reduce the CPU load of commercial systems delivering high bandwidth media, allowing for lower overall system cost due to much lower CPU requirements while providing the required performance to utilize 10-gigabit Ethernet hardware. In addition, because we do not need to keep connection data for each client, a datagram based solution is much more scalable than traditional iWARP.

A major factor blocking the widespread adoption of datagram-iWARP could be the application interface. For traditional iWARP this has been solved by the adoption of the *Sockets Direct Protocol* (SDP) [88]. iWARP utilizes a set of verbs for communication that are not immediately compatible with the traditional socket interface. The task of re-writing applications to make use of iWARP verbs is a large and intensive one. Therefore SDP was designed to translate sockets based applications to use the verbs interface. SDP does not provide any support for datagram-based applications, only those using TCP, as it closely replicates the behaviour of a stream socket, not a datagram socket. Therefore, we have designed a socket interface to allow applications to harness the speed and features of datagram-iWARP without having to re-write any software. This interface is a proof of concept to show that datagram-iWARP can be adapted to use such important interfaces. In addition to its functionality over sockets, datagram-iWARP is also applicable within an HPC context using verbs.

6.1 Related Work

The software implementation of iWARP that was used as the code on which datagram-iWARP was developed is the same as that described in Section 5.1. The data center framework and testing use the same TPC-W benchmark tools first detailed in Section 3.1.

Methods of increasing data center performance by leveraging IB RDMA capabilities for dynamic caching of request responses were addressed in [79]. They cache dynamic content at the

proxy server level, allowing for data to be deemed cacheable or non-cacheable. Using RDMA read operations, it is determined whether the data currently in the cache at the proxy server is up to date. By using RDMA read operations the data in the back end servers is retrieved while avoiding any interrupts to the current processes. If the data is identical, no transfer occurs, the proxy simply serves the locally cached data, thereby speeding the overall transmission of the data to the client. They demonstrate that in certain circumstances the performance of a data center can be increased by an order of magnitude using such caching schemes.

An attempt has been made to translate the data center based memcached object caching system, commonly used in commercial data centers over to use InfiniBand [61], by using multiple transports, including both RC and UD modes to enhance performance. It is found that by using the hybrid transport solution, better scalability can be achieved than using a single transport. Interestingly, SDP is found to perform poorly in this environment supporting the results in Chapters 3 and 4.

In [11] a method of providing support for legacy sockets while still taking advantage of iWARP verbs was proposed. This is important as one of the most attractive features of iWARP is its backward compatibility with existing Ethernet networks and socket-based applications. Other networking technologies have begun to support Ethernet as an optional transport, such as IB over Ethernet (IBoE aka RoCE) [71].

SDP has been proposed as a possible protocol for use over iWARP [7]. This is an attempt to leverage the RDMA capabilities of iWARP while keeping backward compatibility with existing sockets based applications. Balaji et al. [9] examined the impact that providing out of order packet support in an iWARP stack has on the performance of a system. They argue that a fully featured iWARP stack that handles out of order packets can be of use in specific scenarios, although the exact design of which (iWARP onloaded or offloaded) is extremely dependent on the upper layer application characteristics.

6.2 One-Sided Datagram-iWARP Advantages

Many of the advantages of one-sided, or tagged model communication, in this case known primarily as RDMA Write-Record, are the general advantages of utilizing datagrams over a connection based transport. As was discussed in Section 5.2, these advantages include reduced reliability and flow control handling requirements, increased performance via the exclusion of the MPA layer, and overall low overhead. The additional benefits of low implementation cost and increased scalability also still apply. SCTP can provide a message-based transport and bypasses the MPA layer like datagram-iWARP, but it has disadvantages in that it is even more complex, with more features than TCP. Therefore the overhead incurred by using SCTP would be higher than that of unreliable datagrams, as well as its per-stream memory consumption.

The advantages of RDMA Write-Record over send/recv for datagram-iWARP are the elimination of the need to match incoming messages with a recv request in the receive queue. The immediate nature of the tagged model, allows the source node to take responsibility for the data transfer with no interaction required on the part of the target node. The only action of the target node required is to poll the completion queue for notification of events, or request a validity map.

Verbs-based network hardware that is currently in use, such as InfiniBand adapters, show that the RDMA Write methodology can reduce latencies, with the smallest latencies obtainable on the device being those of an RDMA Write operation. These differences typically measure less than 10 μs, with RDMA Write operations taking as little as 1 μs. This may not seem significant in terms of wall clock time, but these latencies can be important in HPC for LAN environments. The performance of RDMA Write/Read versus send/recv can be very implementation dependent, but based on existing network implementations the gap between performance for RDMA Write and send/recv could be reasonably expected to translate to datagram-iWARP performance as well. Unfortunately, these performance differences will most likely not be observable in a

software implementation of datagram-iWARP, manifesting themselves only with a true hardware solution.

6.3 RDMA Write-Record Design

Datagram-iWARP represents a significant shift in the overall design of iWARP, as the current standard is based entirely upon reliable connection-based transports that provide ordered delivery. All of these requirements are not supported with UDP, and although a reliable UDP implementation [42] can provide some of the required features, it still does not match the existing standard. It is important that a design be compatible with both unreliable and reliable datagram transports, as applications that currently use UDP as a transport are capable of handling data loss, like media streaming, VOIP applications or streaming data such as financial market feeds. While data loss is not desired, it is not a fatal error in such applications, and therefore the required overhead for reliability may be avoided in some circumstances. In addition, some socket-based applications for commercial data centers must already assume an unordered transport, as they make use of UDP. However, applications that currently use TCP can also be supported via a reliable UDP implementation [42] that provides the order and reliability guarantees they require. In the case of one-sided RDMA operations like RDMA Write, the order of the data is not important unless the same memory location is being written to multiple times with no control messages, a condition that is highly not recommended in the existing standard.

A high-level overview of some of the required added support for datagram-iWARP can be seen in Figure 6.1. As mentioned in Section 5.2, datagram-iWARP does not require the MPA [21] layer. This avoids the costly exercise of inserting markers into the data payload and consequently will help to enhance performance, as fewer operations on the data are required before placing it out on the transmission line. These changes are compatible with both unreliable and reliable lower UDP layers.

Our datagram-iWARP solution is mostly compatible with the existing iWARP standard. However, several layers, both DDP and RDMAP are defined as requiring that the lower layers be reliable. Therefore specific standards requirements must be relaxed in order to facilitate datagram-iWARP, in addition to other changes required to support datagrams that are not explicitly contrary to the defined standard. Some of these changes also apply for send/recv datagram-iWARP, but are also required for RDMA Write-Record [37]. Such changes are:

1. In the DDP standard [99], Section 5, item 3, requires that the Lower Layer Protocol (LLP) must reliably deliver all packets. This requirement must be relaxed for unreliable datagram-iWARP.
2. The DDP standard [99], Section 5 item 8, states that if an error happens on a Lower Layer Protocol (LLP) stream, the stream must be marked as erroneous and no further traffic can travel over it. This requirement is also loosened in datagram-iWARP, as we don't invalidate LLP streams (or QPs) when errors occur due to data loss, they are simply reported, but the QP is not forced into the error state for unreliable service. In the case of a reliable LLP, the error should be reported, but the connection teardown is not required, as no such connection exists, the QP simply transitions into the error state.
3. The RDMAP Standard [94] Section 5.1 states that the LLPs must provide reliable in-order delivery of messages. This cannot be provided for in datagram-iWARP, but should be handled by applications as current UDP applications or Upper Layer Protocols (ULPs) currently do. For a Reliable Datagram (RD) solution, this is not required as an RD LLP should provide order and reliability guarantees.

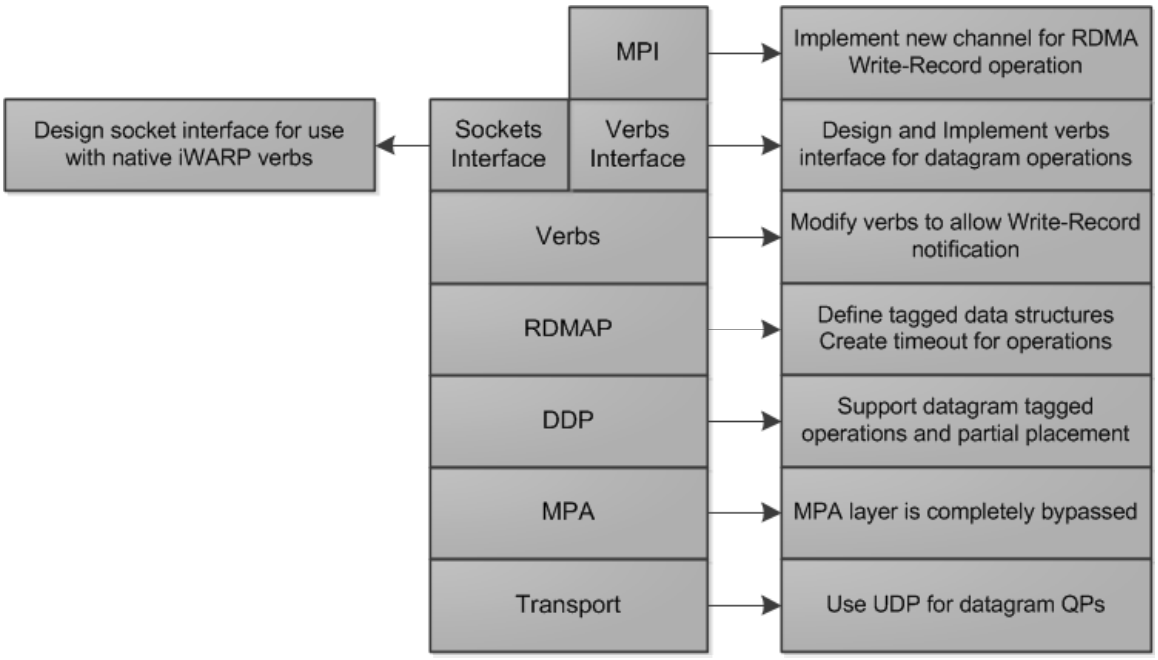


Figure 6.1: Changes for datagram-iWARP to support RDMA Write-Record

4. Datagram-iWARP requires either the establishment of new verbs, or the adaptation of existing verbs for methods of creating and manipulating datagram-iWARP sockets. We require a datagram type QP, as well as a method for initializing datagram QPs. In addition, we require verbs that allow for the inclusion of destination addresses and ports when posting a send request. We also require a datagram receive verb that allows for the sender's address and port to be reported back to the calling application. This is accomplished by expanding the work request elements to include the source address of an incoming message. In addition the completion queue elements need to be altered to include information concerning the source address and port for incoming data. These changes are required throughout the stack to support datagrams.
5. Datagram-iWARP does not require packet marking for either UD or RD modes, therefore the MPA layer [21] can be removed.
6. Operating Conditions: Datagram-iWARP always requires the use of Cyclic Redundancy Check (CRC32) when sending messages. In addition, there is no initial set up of operating conditions exchanged when the QP is created; the operation conditions are set

locally, and should be communicated through the ULPs. Given that we are using an unreliable transport, using messages spanning multiple datagrams makes the system vulnerable to packet loss. While useful in conditions of low network congestion, it can cause significant problems based on the existing standard. With datagram-iWARP it is recommended that the application layer perform segmentation and assembly for messages larger than the defined maximum UDP packet size, 64KiB. Although for UD RDMA operations it is possible to perform reassembly of larger messages in the iWARP stack.

There is currently no method proposed that allows for RDMA operations over unreliable datagrams for any RDMA capable interconnects. The one-sided RDMA write operation requires a significant change in design due to the unreliable, out-of-order nature of the transport. The defined RDMA Write operation requires the lower layer provide in-order delivery. For datagram-iWARP, using a reliability scheme at the lower layers cannot be assumed, so we can be assured of neither reception nor in-order guarantees. Therefore, we need an operation that can determine the validity of data on the target side without a supporting operation generated at the source side.

Currently, RDMA Write over reliable connections can notify the target application of when data is valid by completing a send/recv operation after the successful RDMA Write [94]. This has the purpose of informing the application that there is valid data to read. Alternatively, some implementations also use a flagged bit in memory that is polled upon, and when set, the operation is known to be complete. Over an unreliable transport this does not work for several reasons. Firstly, the RDMA write may not complete successfully, and the send/recv operation does, which causes the target to receive invalid data (as the source considers the write operation complete when all data have been passed to the LLP). Alternatively, the RDMA operation may complete but the send/recv is lost. Using high-level acknowledgments is not an ideal solution to these problems as such acknowledgements may also be lost and cause unnecessary re-transmission.

Therefore, a new method called RDMA Write-Record was developed for use over unreliable datagrams. RDMA Write-Record must log at the target side what data has been written to memory

and is valid. The target application can then request this information to determine what data is valid by reading the appropriate completion queue entries. These completion queue entries can be designed as either individual entries for each logical chunk of data in a message or can be a validity map; essentially an aggregated form of individual completion notifications. For messages of a size less than or equal to the Maximum Transfer Unit (MTU) of the LLP, this notification is very simple, comprising only a single completion entry or single entry validity map. This method differs from send/recv over UD as there is no matching receive request posted, the data is simply placed in the correct memory location. However, unlike a traditional RC RDMA Write operation, no further communication is required between the source and target and the source completes the operation at the moment that the last bit of the message is passed to transport layer. The differences between the two approaches can be seen in Figure 6.2.

RDMA Write-Record is an especially useful operation for datagrams as it has very low latency achieved by being able to directly write into allocated memory with a pre-determined data sink location. It also fits well into the semantics for sockets, in that it can easily be used within a send/recv semantic that is compatible with socket-based applications. The send side initiates a request, which completes as soon as the data are delivered to the UDP layer. In order to support a socket type interface, the receiver can then poll for the completion of an RDMA Write-Record operation. This method is also valid for a reliable transport, although in practice, the solution of polling on a signalling bit to determine completion is a lower-overhead method of performing an RDMA operation. In a reliable transport, the in-order and reliable delivery make the monitoring of what data is valid unnecessary at the iWARP level, as it is handled at the lower level.

RDMA Write-Record is somewhat similar to send with the solicited event verbs defined for iWARP. In send with solicited event, the target machine can receive a send, match it to a posted receive and signal an event, if the system supports such an operation (for example an interrupt).

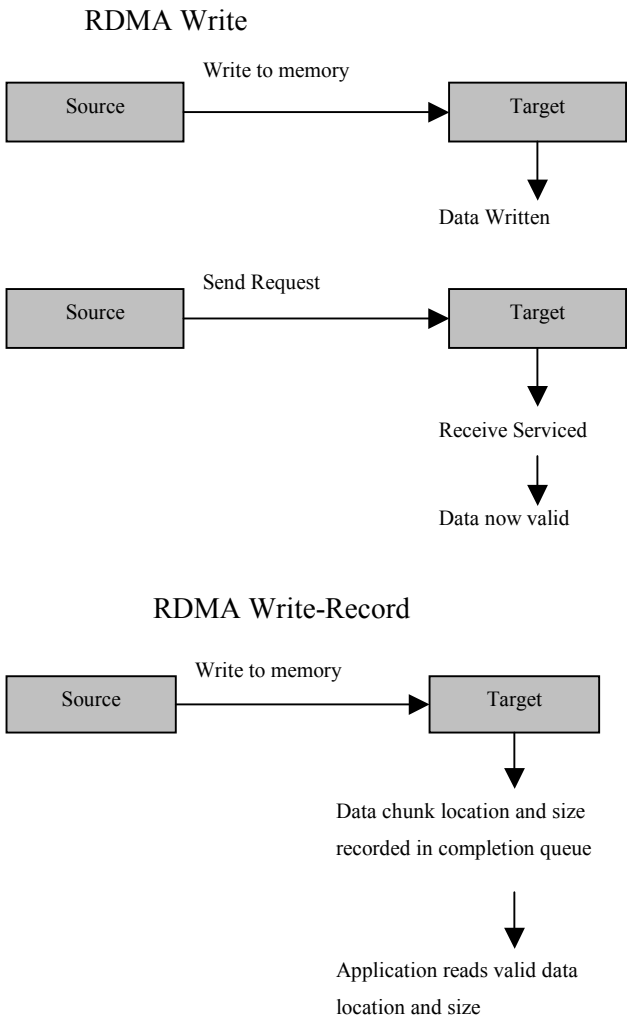


Figure 6.2: Comparison of RDMA Write over RC and RDMA Write-Record over UD

This differs from RDMA Write-Record as it is a two-sided operation, and it also creates an event at the target, where RDMA Write-Record simply creates a completion event queue element, that must be actively read in order to determine that the operation completed. RDMA Write-Record also has some similarities with the RDMA Write with immediate verb defined for InfiniBand networks. RDMA Write with immediate differs from our RDMA Write-Record as it requires that a receive be posted at the target to receive the immediate data. Our proposed solution does not require a posted receive whatsoever, making it a truly one-sided operation.

In designing a networking solution operating over an unreliable transport, consideration must be made for packet loss, particularly if message sizes greater than the network MTU are

supported. We have designed support for partial message placement through RDMA Write-Record. This allows for some data loss, for applications that can tolerate some packet loss, such as streaming audio, or online gaming. A small subset of applications also exists that is capable of determining if the incoming data feed is valid or not and compensating for invalid inputs. We provide support for informing applications of the valid memory areas that have been written to, in order to support applications that can handle invalid input streams as well as those that can tolerate some data loss. The performance benefits of this approach versus the whole message delivery provided by send/recv is illustrated in Section 6.6.1.

WANs normally run using a 1500 byte MTU, with applications using message sizes smaller than the MTU. Datagrams are technically defined up to a maximum size of 64 KiB. Therefore, it is preferable to package each message sent over RDMA Write-Record as a complete unit that spans only one datagram packet, preferably the size of the network MTU. In order to enhance relatively error-free local area network transmission performance, and make our solution compatible with varied network MTUs, we have designed in support for larger message sizes than the expected network or datagram MTU. This can lead to efficiency increases for applications that can use large messages over low-loss networks. Its use over existing WAN infrastructure, particularly congested networks, is limited as the penalty for packet loss can be high.

Typically, in the event of packet loss while sending large multi-packet datagram messages, the entire message must be discarded. This can be alleviated to some extent by the addition of a reliability mechanism for UDP, but it is preferable to have multiple independent requests in an environment with frequent packet loss. With the proposed changes to Ethernet, like CEE, that defines error free channels; systems can make use of large message sizes to increase efficiency.

6.4 RDMA Read Design

RDMA read over datagrams [39] is a challenging design due to its request response nature. In order to perform a traditional RDMA Read, the operation source node sends a request to the target node; the target node then sends the source node data from the target's memory. Much like an RDMA Write, the source must have some knowledge of the valid areas of memory on the target in order to make its request. Once a request has reached the target node, it processes the request and sends the required data back to the source. In a reliable connection environment this request-response exchange may be delayed due to network conditions, but communications will never fail to reach either the target or the source unless a catastrophic network error occurs.

In an unreliable datagram environment, such delivery guarantees cannot be made. This makes the fundamental request-response behaviour of RDMA Read somewhat problematic. It is vulnerable to data loss, and when this does occur, it can be difficult to determine at which stage the loss occurred. For example, if a source sends a request, the request is received at the target and the target responds, but the response is dropped, the source does not know if it should resend the request because the request was dropped, or if the response was the source of failure. The target is never informed that the source received the data so it can remain ignorant of any data loss.

RDMA Read is also problematic in that it requires that state information be kept about current requests. Consequently, requests cannot be allowed to fail to complete without providing some method of dealing with the failed outstanding requests. In order to accommodate the possibility of failure, it is desirable to only report the failure should its status be requested by the application. A solution to removing failed requests is to create a reasonable timeout period after which requests are automatically removed from the active request list. This time must be sufficiently large to account for network delays, but not so long as to allow a large buildup of failed requests to swamp the networking resources. Obviously, this time period will be system and network dependent, and consequently, it has been designed to be adjustable.

After a request has reached its timeout period, it is removed from the active list. This timeout period is not monitored in a real-time fashion, the timeouts of the individual requests are only checked when traversing the list to complete a RDMA Read request, or when specifically requested. This eliminates this activity as a potential source of performance degradation for other RNIC operations.

Although methods of providing some sort of reliability in the operation itself are possible, they do not fit the definition of unreliable datagram-iWARP. Reliability is the responsibility of the application or of the chosen LLP, which could be a reliable datagram implementation. For an unreliable network environment it is desirable to have communication not require multiple communication stages between systems.

6.5 RDMA Datagram-iWARP Software Implementation

The software implementation of datagram-iWARP was developed using a TCP-based iWARP implementation from [23]. A socket interface was added, which is described in more detail in Section 6.5.1.

The software implementation replicates the functioning of all of the iWARP layers as a user-level library. It provides a verbs interface that applications can use to interact with the iWARP stack. The expanded stack is shown in Figure 6.3. The changes required to the verbs, RDMAP, and DDP layers as described in Section 6.2 and 6.3 were implemented in this software stack. In addition, changes were required to the code to allow for the use of datagram transports at a lower layer that would not be required in a hardware implementation of datagram-iWARP. Our software implementation takes advantage of I/O vectors to minimize data copying and enhance performance. This stack from the high-level view of Figure 6.4 is not different from Figure 5.4, except for the addition of the optional socket interface, RDMA Write-Record and RDMA Read.

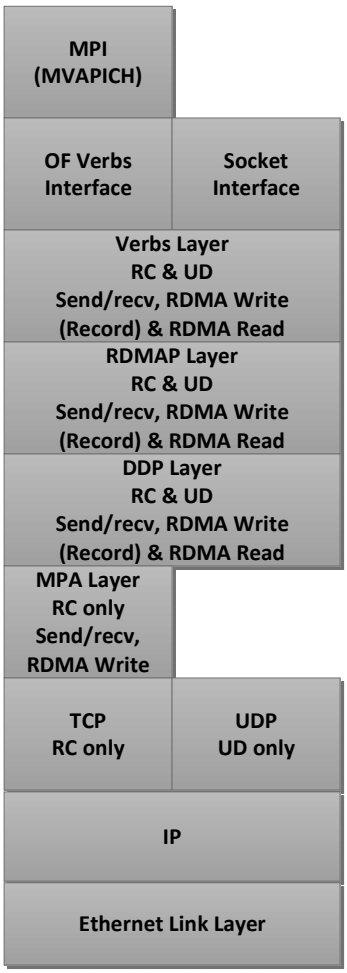


Figure 6.3: The software implementation of datagram-iWARP with RDMA Write-Record

Like the send/recv design, RDMA Write-Record requires the use of CRC at the DDP layer to ensure correct reception of individual packets. CRC checks may be performed at the UDP layer, and therefore it would be redundant to do so again at the DDP layer. Therefore, it is recommended that CRC checking be disabled at the UDP layer to further enhance performance.

6.5.1 Datagram-iWARP Socket Interface

The iWARP socket interface was designed to serve as a layer that translates the socket networking calls of applications over to use the verb semantics of iWARP. This has the significant benefit of allowing existing applications to take advantage of the performance of iWARP while not requiring that they be re-developed to use the verbs interface. This interface is a proof of concept showing that it is possible to write a user-level interface for socket applications to directly use datagram-iWARP hardware. Such functionality is provided for reliable

connection-based RDMA through the Sockets Direct Protocol [88]. No such protocol exists for unreliable transports, although the concepts used in SDP could be adapted to offer functionality for datagram-based traffic. Therefore, our socket interface should be regarded as a demonstration that such functionality could conceivably be implemented in a full SDP-like protocol specification. In order to fairly compare the UD versus RC results, support for both UD and RC operations has been included in our socket interface implementation.

Our socket interface works by dynamically preloading the interface code before running an application, overriding the operating system networking calls to sockets, re-directing them to use iWARP sockets instead. Unlike SDP, our design does not override the creation of sockets, only the data operations related to them, and does not seek to replicate datagram socket behaviour like SDP does for TCP. As such, it uses the socket initialized by the application directly over the iWARP software stack. In the software solution this makes sense as the socket must be passed to the kernel networking stack when it is used by a LLP like UDP. In a hardware solution, this design would be expanded to override all socket creation as well, so that the relevant hardware QP could be created and associated with a "dummy" file descriptor number, and a full protocol specification could be developed to enable more efficient use of RDMA Write-Record.

The iWARP socket interface operates by allowing for both TCP and UDP-based iWARP sockets to be opened, using the relevant iWARP lower layer protocol. It tracks the socket to QP matching so that each socket is only associated with a single QP. For datagram-iWARP, the work request posted to the QP is assigned a destination address at the time of a send.

When a call is intercepted, the interface determines the type of socket that is performing the request, and uses either RC or UD as appropriate for the socket type. Information about the destination address, source address or port is not stored in the interface, only the QP to file descriptor mapping and whether the file descriptor has been previously initialized as an iWARP socket. The remaining required information is stored in the socket data structure.

This solution is much more lightweight than traditional SDP, but less robust as applications can modify sockets. As such it is suitable for determining the performance of datagram-iWARP with some popular socket-based applications. However, it is not a comprehensive alteration of the existing SDP standard in order to adapt it specifically to use a datagram semantic. Altering SDP would have the positive benefits of hiding the socket creation from the applications, and is essential for a hardware implementation to separate the socket based networking semantic at the application level from the verbs based semantics of the hardware itself.

Such an addition to the SDP protocol would be lengthy and require major changes or additions to the existing standard. Our goal with the iWARP socket interface is to demonstrate that a datagram socket to verbs translation is possible, and demonstrate the potential benefits that datagram-iWARP could bring to existing datagram sockets-based applications.

6.5.2 Datagram-iWARP RDMA Write-Record MPI Implementation

Building upon the MPI implementation for datagram-iWARP send/recv mode, the MVAPICH-Aptus [67] implementation can be further extended to support RDMA Write-Record. RDMA Read could also be supported, but given its poor performance in the verbs benchmark, it is not implemented. The existing implementation is insufficient for RDMA Write-Record for two main reasons. First, the behaviour of RDMA Write-Record differs from that of a traditional RDMA Write, therefore the RDMA Write code available for use in MVAPICH-Aptus is only of use for areas of behaviour that overlap with RDMA Write-Record. Second, the MVAPICH-Aptus RDMA Write protocol is only supported for use over reliable connections such as traditional iWARP TCP. Consequently, an additional transport/protocol path had to be created which enabled RDMA Write-Record to function correctly, and the implementation had to be altered so that the new RDMA Write-Record network code could be used over a datagram transport. Many of the same requirements that were originally discovered for the datagram-iWARP send/recv code were also applicable to RDMA Write-Record, there is no need for GRH, LIDs, service levels, shared receive queues or XRC support. The connection setup requirements

were altered such that datagram-iWARP could be supported, namely transmitting extra data during the connection setup such that connections could be established. This is required as LIDs are not supported in datagram-iWARP, so data such as IP address and port are required instead of the connection LID.

Therefore, for the newly created networking protocol for RDMA Write-Record, several existing protocols could be used in creating the new RDMA Write-Record method. The existing Rendezvous protocol was expanded to also include a RDMA Write-Record mode, using elements of the existing send/recv (UD compatible) Rendezvous method, combined with elements of the RC-based RDMA Write code. Some behavioural changes could also be made to the implementation. Upon reception of the data at the target node, the RDMA Write-Record method can deliver the data without the need for the follow-up Rendezvous finish message to arrive.

Some minor changes required to support the traditional iWARP RC-based RDMA Write operation were also made (such as GRH, LID support etc.), such that it could be used as a comparison baseline against datagram-iWARP RDMA-Write for testing. These changes were much less extensive than the changes and additions needed to support datagram-iWARP.

6.6 Experimental Results and Analysis

This section details the experimental platform used for performance testing the datagram-iWARP implementation. It also reviews the performance results of verbs level micro-benchmarks for all of the discussed modes of operation, and investigates the performance of MPI applications as well as two commercial applications, a media streaming application VLC [119] and a SIP [31] application/benchmark SIPp.

The experimental results were obtained from the C1 systems, used in Chapter 5. The interested reader is referred to Section 5.5 for full details.

6.6.1 Micro-benchmark Performance Results

The micro-benchmarks results in this section are an average of at least 10 micro-benchmark runs, with each run comprising thousands of iterations. The results of the native verbs latency of datagram-iWARP are shown in Figure 6.4, for send/recv, RC RDMA write, UD RDMA Write-Record and RC and UD RDMA Read. We can observe that the lowest latencies for small messages are UD send/recv and UD RDMA Write-Record, which are in the range of 27-28µs for messages less than 128 bytes.

Therefore in terms of latency datagram-iWARP is consistently better than RC send/recv, RC RDMA Write, and RC RDMA Read, which has latency around 33µs for messages under 128 bytes long. For message sizes up to 2KiB, UD send/recv offers an 18.1% improvement in latency over RC send/recv iWARP, while UD RDMA Write-Record offers a 24.4% improvement over RC RDMA write, and UD RDMA Read offers a 20.28% improvement over RC RDMA Read. For messages between 16KiB to 64KiB we can observe that the performance of RC send/recv is slightly better than that of UD RDMA Write-Record, UD RDMA Read and UD send/recv, but the UD based iWARP solution (send/recv, Write-Record and Read) have better latencies for larger message sizes.

The latencies observed are relatively high empirical values when compared to those expected for existing networks. Verbs level RDMA write latencies can be in the 3 µs range for current iWARP networks [17]. However, it must be remembered that these latencies are observed for a software implementation of a networking stack. They function over a software iWARP stack and use the kernel's networking stack for TCP and UDP. As such there is a significantly higher latency than can be achieved with modern hardware solutions that take advantage of OS bypass. Therefore, the hardware networking solutions available can avoid latency incurred by the need to context switch over to the operating system to processing a networking task.

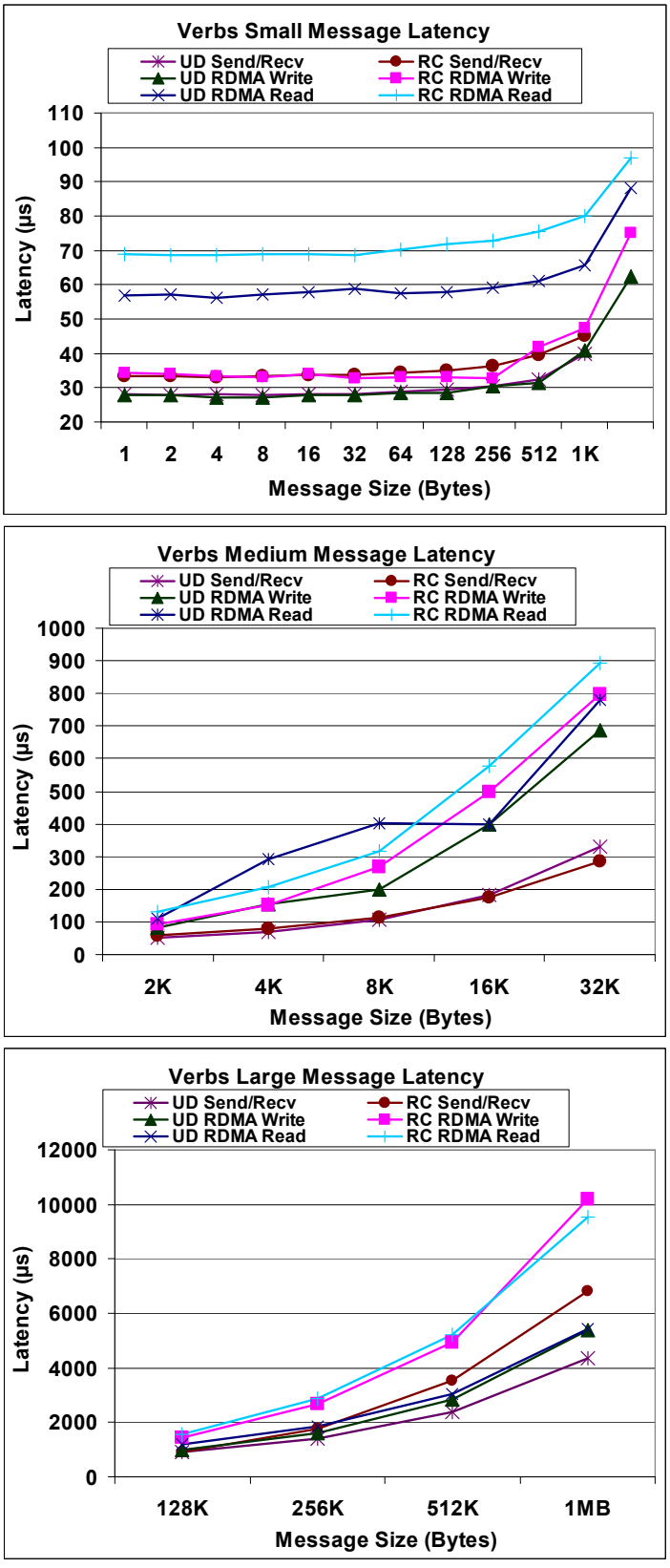


Figure 6.4: UD iWARP vs. RC iWARP verbs latency

The software approach has an advantage in that it uses a more sophisticated and higher speed processor to handle the networking stack processing. However, the overhead associated with context switching and the kernel stack processing, do not offset the advantages of avoiding the context switch.

Figure 6.5 shows the unidirectional bandwidth in which one side is sending back-to-back messages of the same size to the other side. The performance of messages between 1KiB and 1.5KiB is of great importance as such message sizes are the most likely to be used by many commercial applications, as this message size matches that of the MTU of the underlying layer. For 1KiB messages UD RDMA Write-Record has a bandwidth of 188.8% higher than RC RDMA Write and UD send/recv has a maximum bandwidth of 193% higher than that of RC send/recv. RDMA Read for both UD and RC perform poorly at this MTU compared to send/recv and RDMA Write-Record.

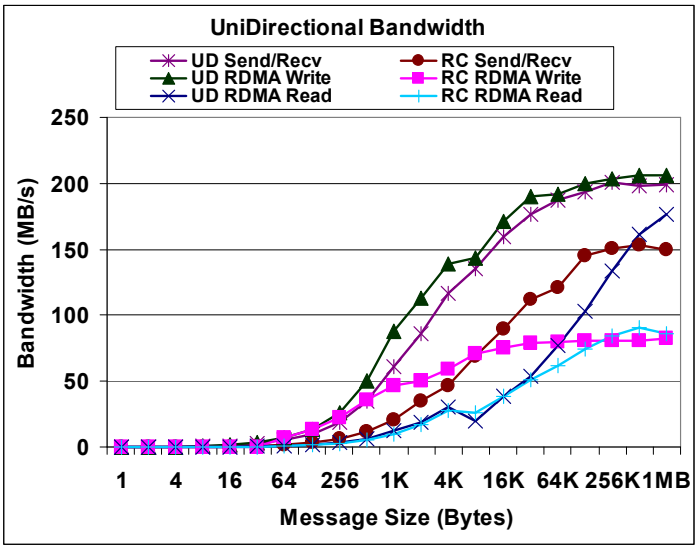


Figure 6.5: Unidirectional Verbs bandwidth

For messages larger than 1.5KiB, multiple datagrams comprise a single message, and they are recombined at the receiver to form the full message. Such a scheme is only useful in networks with low packet loss rates, but the results in such an environment are excellent. We find that for very large messages, UD RDMA Write-Record is the dominant method. Examining the bandwidth results for large messages (larger than 128KiB) in Figure 6.5, we can observe that UD

iWARP is the winner. UD send/recv offers a maximum of 33.4% improvement over RC send/recv occurring at 256KiB messages. Most importantly, RDMA Write-Record has a significant 256% advantage at message sizes of 512KiB over RC RDMA Write. UD RMDA Read similarly outperforms RC RDMA Read (maximum 76.1%), although it under performs compared to other UD methods. Given this underperformance related to other UD methods and the issues surrounding reliability detailed in Section 6.4, RDMA Read will not be analyzed further for applications or packet loss considerations.

Performance with Packet Loss

In order to study the proposed system under packet loss conditions, the Linux traffic control provisions were utilized. Using the traffic control mechanisms, a FIFO queue that normally dequeues messages as fast as they can be delivered to the underlying hardware was configured to drop packets at a defined rate. By examining the bandwidth of UD send/recv datagram-iWARP under various packet loss conditions in Figure 6.6, we can see that the theoretical evaluation done in the design stage follows the results seen under real conditions. For the UD send/recv mode, the impact of packet loss is significant for large message sizes even under very low packet loss conditions. A loss rate of 0.1% is close to the observed packet loss rates for intra-US web traffic, while a 0.5% loss rate is in line with expectations of loss between a western European-US transmission [106]. Packet loss rates of 1-5% are observable for traffic to such locations as Africa and parts of Asia. As such we can observe that the solution of partial delivery of messages like that provided for with RDMA Write-Record improves performance over the required full message delivery used in our send/recv method.

Figure 6.7 illustrates that the partial placement RDMA Write-Record method performs better in low packet loss conditions even for larger message sizes. We observe a drop at 64KiB messages as these messages exceed the maximum sized MTU of the UDP layer, requiring multiple UDP messages.

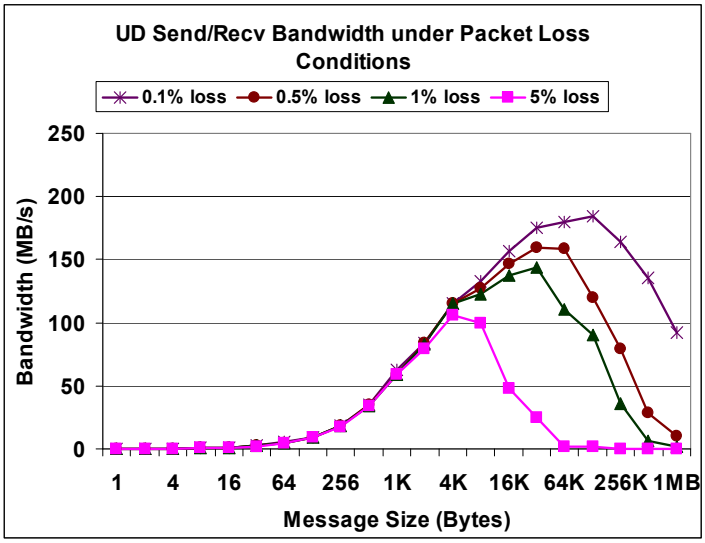


Figure 6.6: UD Send/Recv bandwidth under packet loss conditions

For messages under 64KiB and greater than the network MTU of 1500 bytes, multiple packets are segmented at the sender (IP layer) and recombined at the target machine and delivered by the UDP layer as a single large message. Any loss of the smaller packets making up this large UDP packet results in the entire (up to 64KiB) message being dropped. For messages larger than 64KiB, the partial placement feature of RDMA Write-Record makes it capable of maintaining high bandwidth under packet loss conditions. Segments (64K) are placed in memory as they arrive and their reception and location is recorded. So for messages that comprise many 64KiB UDP segments some of the overall message can be saved even if some 64KiB UDP segments are discarded. However, high packet loss rates can cause total breakdown of the bandwidth, as the final packet must arrive for the partial message to be placed into memory and those parts that are valid are declared as such. Loss of this final packet results in the loss of the entire message. Therefore, large loss rates (~5%) can lead to very low throughput for large sized messages. However, the performance for more typically used smaller messages is excellent.

Overall, we can observe that datagram-iWARP clearly has great benefits in terms of bandwidth for our software implementation. We would therefore expect that the hardware proposed by this proof of concept would have excellent throughput, although obviously limited in

its maximum bandwidth by the link speed itself. These performance results show that such a scheme can provide high bandwidth, while in hardware, requiring no CPU intervention.

The effectiveness of iWARP UD RDMA Write-Record is clear in that it has latency comparable to the best alternative method (UD send/recv), and it clearly has the best bandwidth performance of any of the methods. In Ethernet networking in a commercial environment, the bandwidth performance is much more important than the latency performance, as even moderately sized transmission ranges lessen the impact of the lowered latency at the system side.

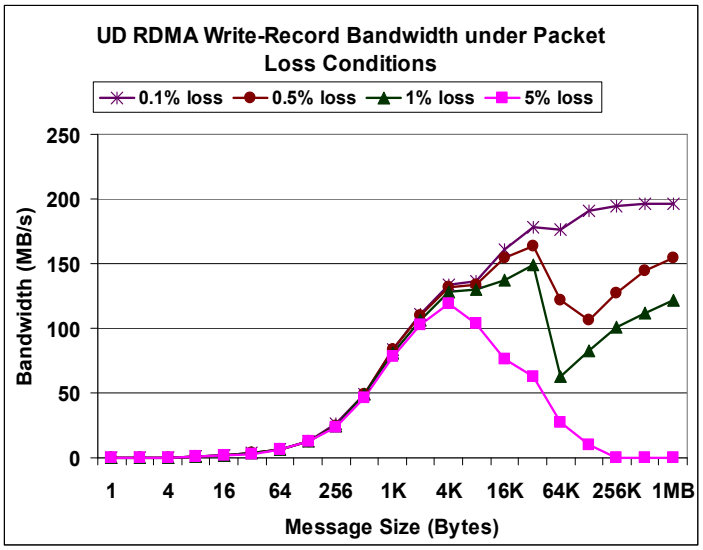


Figure 6.7: UD RDMA Write-Record bandwidth under packet loss conditions

MPI Micro-benchmark Results

The ping-pong latency over MPI is shown in Figure 6.8 for both UD and RC modes. As in Chapter 5, the results are an average of 5 runs of the MVAPICH microbenchmarks, with each run comprising 100 iterations. MPI uses two different protocols for sending messages of different size. The Eager protocol uses send/recv for small messages and the Rendezvous protocol uses RDMA Write (Record) for larger messages. MPI implementations typically support a threshold message size at which this protocol change occurs, for these tests the threshold is 64KiB, a good performance point found in verbs testing. It can be observed from Figure 6.8 that the datagram mode of iWARP has superior MPI performance over RC, consistent with the verbs level results.

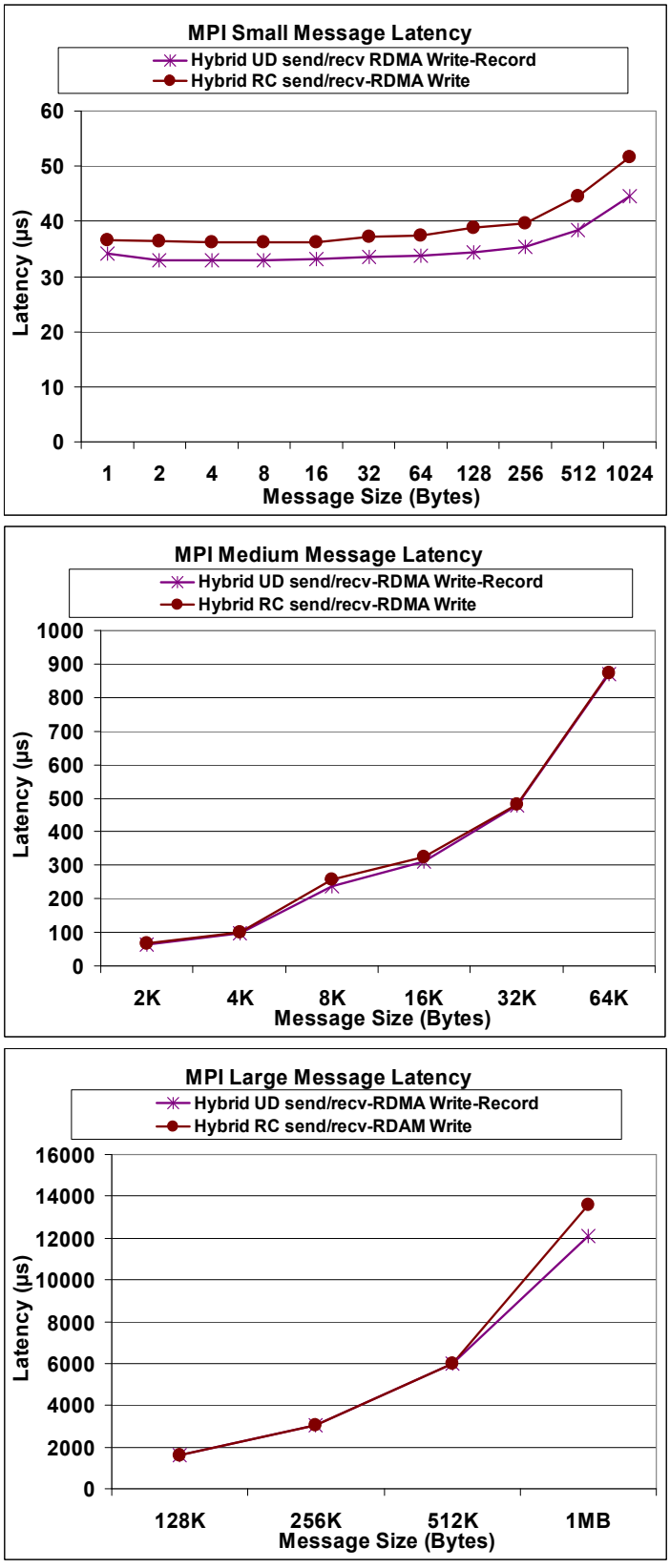


Figure 6.8: MPI ping-pong latency

A method has been developed to support RC RDMA Write operations for small messages, called fastpath RDMA, in some MPI implementations [81]. This method lowers RDMA latency times for small messages by using a pre-determined set of buffers, avoiding the need for pre-negotiation before every network data transmission. This makes using RDMA operations for small messages more practical. However, this approach has the same scalability issues with connection-associated buffers that occur for send/recv.

Figure 6.9 shows the bi-directional MPI bandwidth for UD and RC modes. The bi-directional bandwidth benchmark uses two pairs of processes on two nodes, communicating in opposite directions. One of the processes of each pair posts a window of non-blocking receive calls (for the send/recv case, no posted calls are necessary for RDMA), while the other posts a window of non-blocking send (or RDMA) calls with synchronization at the end of the test.

MPI in either UD mode offers a higher bi-directional bandwidth for most cases than the comparable RC mode. The improvement for large messages (RDMA Write-Record) is 20.7%. Less protocol processing and lower reliability requirements allow the UD-based communication methods capable of pushing more data on the wire in each direction.

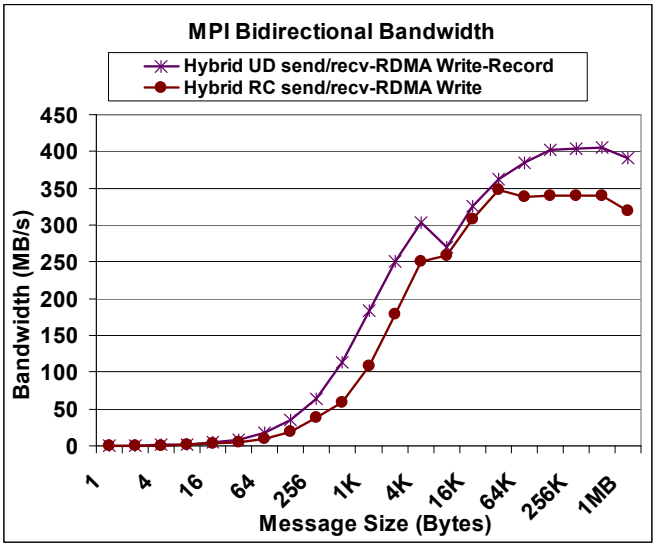


Figure 6.9: MPI bidirectional bandwidth

6.6.2 MPI Application Results

The results in this section are for the class B CG, MG and LU benchmarks from NAS Parallel Benchmark (NPB) suite version 2.4 [80], as well as the Radix [103] and SMG2000 [15] applications. All results are presented for 4, 8, 16, 32 processes. The percentage improvement in the application runtimes is shown alongside the percentage improvement in overall communication time. Communication time was measured as the time spent performing all communication primitives: MPI blocking and non-blocking send and receive and MPI wait calls.

The improvements possible using RDMA Write-Record over RC-based RDMA Write were also examined for MPI. This involved setting a threshold for a Rendezvous protocol that uses either RDMA Write or RDMA Write-Record from the iWARP stack. For this, the same set of applications used in Chapter 5 was run on cluster C1, with the results presented in Figure 6.10. The Rendezvous threshold was set for a 64KiB message size for those tests which send large messages. Some of the applications do not send messages of a size greater than 64KiB, therefore in order to observe RDMA Write (Record) in use the MPI implementation's default 8KiB message size threshold was used when running these applications.

By comparing the percentage improvements of the applications over send/recv in Figure 5.9 and the hybrid UD send/recv-RDMA Write-Record method in Figure 6.10 we can see that some of the same patterns occur. Overall the improvement of the hybrid UD send/recv-Write-Record method over that of hybrid RC send/recv-RDMA write method is better than that of send/recv UD over RC. We still see excellent percentage improvement in application runtime for four processes of SMG2000, like that seen for send/recv. The direct performance comparison of runtimes to the send/recv results in Figure 5.8 shows that LU, MG and Radix improvement of the hybrid UD send/recv-RDMA Write-Record method over the hybrid RC send/recv-RDMA Write is superior to the improvement of send/recv UD over RC, the other applications show more improvement between the send/recv methods.

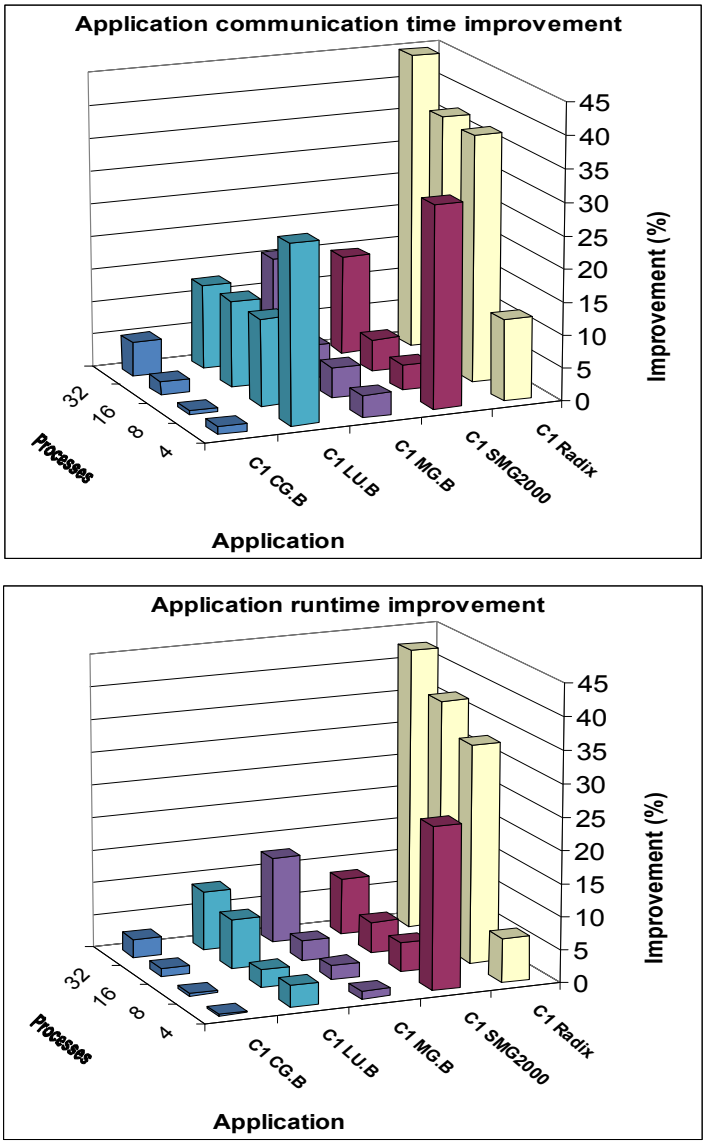


Figure 6.10: MPI application communication and runtime improvements for hybrid UD send/recv/RDMA Write-Record over RC RDMA Write

The application with the greatest improvement over its RC equivalent for the hybrid UD send/recv-RDMA Write-Record mode is Radix. It sees a 42.8% improvement in runtime and 44.9% improvement in communication time over its RC equivalent. The pure send/recv mode used in Chapter 5 (Figure 5.9) shows only a 14.3% and 29.9% improvement over RC send/recv. However, the hybrid mode is not the superior method for all applications, as CG operating in a pure send/recv mode has 7.6% and 10.6% improvements in runtime and communication over its RC equivalent versus 2.8% and 5.3% for the hybrid case. This is due to the message size

characteristics of the CG benchmark, as it does not send messages larger than 64KiB, yet does send messages larger than 8KiB. As the MPI eager/rendezvous threshold was set to 8KiB in this case, the extra round-trip latency from the rendezvous RDMA Write-Record method becomes detrimental to performance.

Application Memory Usage

For the RDMA Write-Record versus RDMA Write methods, there is a negligible memory savings from UD over RC, mainly due to the fact that the buffers are negotiated when using the rendezvous method, which does not require pre-posting many buffers as the buffer sizes and locations are allocated and assigned as needed through a rendezvous negotiation prior to data transmission. Therefore, for such a buffer management system examining its memory usage is of limited utility. However, it should be noted that the memory savings available to datagram-iWARP send/recv are still applicable and exist for the use of send/recv when under the Rendezvous threshold. They are not presented here to avoid unnecessary duplication. The interested reader is referred to Figure 5.10 in the previous chapter for an overview of the potential memory savings available to send/recv.

6.6.3 Commercial Application Results

The iWARP socket interface has been tested using a variety of custom coded socket tests, as well as testing using widely accepted socket based software. It is designed to work over any application that uses a datagram or stream socket. It has been tested with VideoLan's VLC [119] a popular media streaming application as well as SIPp [31], a testing framework for load testing SIP servers.

VideoLan's VLC media player (*VLC)* [119] was chosen for performance testing of datagram-iWARP. In order to assess datagram-iWARP's real world performance benefits for a media streaming application it was necessary to compare VLC's UDP streaming mode with an RC compatible mode (HTTP-based) for a UD vs. RC comparison. In comparing the two approaches as seen in Figure 6.11, we can observe that the UD mode of operation results in a 74.1% reduction

in media initial buffering time over the HTTP-based RC alternative. This represents a significant increase in throughput for the system of almost three times the throughput of a RC based system. There is more overhead involved in the HTTP method, and therefore the performance gap between the application modes is due only partially to the datagram-iWARP to RC-iWARP difference.

However it can be observed that the performance difference between send/recv and RDMA Write-Record is minimal. This is due to the need in the software socket interface to provide support for many buffers passed into a single socket. In order to effectively support the use of multiple buffers on a single socket, we have elected not to re-exchange (advertise) remote buffer locations for every new buffer due to the required overhead and subsequently reduced performance, but to copy the data over to the supplied buffer location instead. This makes both RDMA Write-Record and send/recv almost identical in terms of performance when using our socket interface. Therefore in the next sub-section, we will present the UD results as a single performance number instead of separating the results out for both methods. Future enhancements to performance similar to SDP's buffered copy/zero copy methods will allow for a differentiation in real-world performance, by using zero-copy for large message sizes and buffered copy for smaller messages. This also increases memory efficiency, as intermediary buffers are not required for large messages, as they are written directly into the application buffer.

We have examined the base response time for interaction with the SIPp server. For request/responses under a server under light load, we found the request response averages seen in Figure 6.12. We observe that the UD-iWARP response time is a 43.1% improvement over that of RC-iWARP. This can be attributed to the TCP overhead incurred.

Using SIPp [31] we have investigated the overall memory benefits of using datagram-iWARP over traditional RC iWARP. A SIPp server and client were generated and configured as a client and server using a basic SipStone client-server test. Figure 6.13 details the memory savings that are possible when using datagram-iWARP on a SIP server compared to RC in such a scenario.

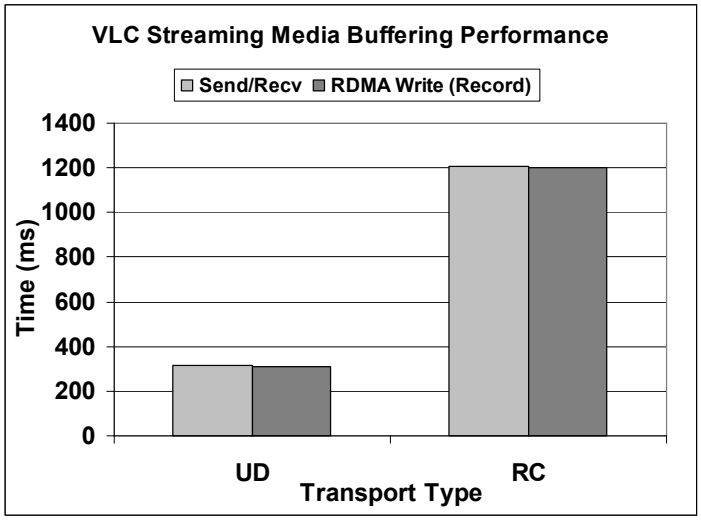


Figure 6.11: VLC UD streaming vs. RC-based HTTP streaming

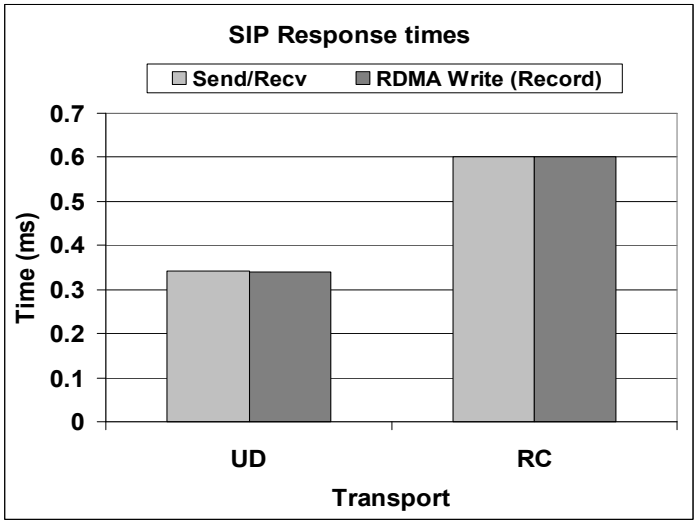


Figure 6.12: SIP response times

The memory savings were calculated using the sum of the SIPp application memory usage and the allocated slab buffer space used to create the required sockets. This means that the memory usage is a whole application space (including iWARP memory usage) memory usage comparison including kernel space memory for the sockets. SIPp was configured to generate a load emulating many clients, which creates a single UDP port for each client. We find that at 10000 clients, we have a memory savings of 24.1%. Theoretical calculations based solely on the iWARP socket size (using one socket per client, and determining the amount of memory saved by using a datagram structure over a connection data structure) predict that such a case would result in a 28.1% memory improvement over RC. The resulting 4% difference can be directly attributed

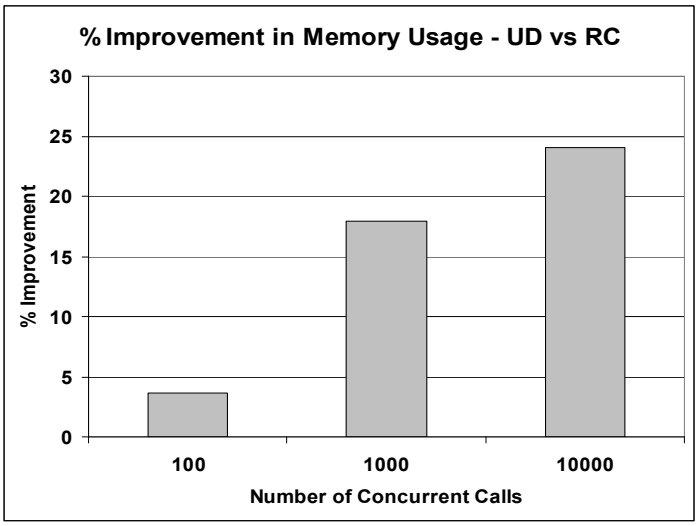


Figure 6.13: SIP improvement in memory usage using send/recv datagram-iWARP over traditional iWARP

to the application's memory usage, which would require some additional bookkeeping to keep track of the states of the calls over the UDP ports to determine when to close the ports. Although the actual amount of memory used in this test is not excessive, it can become more onerous on systems supporting hundreds of thousands to millions of clients.

We have measured the overhead caused by using the datagram-iWARP socket interface when doing the most network intensive task available during video streaming, the pre-buffering required before beginning playback. We have found a very minimal approximate 2% increase in overall pre-buffering time when using our socket interface over using native UDP. As such, the interface has very low overhead, as the overall overhead of the software iWARP solution and the socket interface is only a 2% penalty over the native UDP networking stack, which our iWARP software solution uses at the lower layers. Therefore, we can conclude that for such applications, the software iWARP solution is viable, and a hardware iWARP solution should therefore be able to significantly improve performance while having very little overhead running over a software socket interface.

6.7 Discussion

Datagram-iWARP has been shown to be effective in both HPC and commercial data center environments. The advantages of a scalable connectionless transport are numerous and the efficiency improvements that can be realized are significant. As datagram traffic is expected to dominate future WAN traffic [19], the inclusion of RDMA over datagram support for modern high performance interconnects is an important step in broadening the application space for RDMA-enabled network technologies. This represents a major milestone in the development of RDMA technology. It is the first proposal that provides RDMA over unreliable datagrams, and is consequently very scalable.

Chapters 5 and 6 have provided a design for a full featured datagram-iWARP solution. Chapter 6 proposed a new RDMA operation, RDMA Write-Record, which can be utilized on any RDMA-enabled network, including datagram-iWARP. In addition, we have implemented and tested both a fully functional software datagram-iWARP stack as well as a socket interface for providing iWARP functionality to existing socket-based applications. The performance of the datagram-iWARP send/recv and one-sided RDMA operations were explored with real data center applications, finding that one-sided RDMA Write-Record can have significant performance benefits over send/recv. It was found that the bandwidth of datagram-iWARP can exceed that of traditional iWARP by up to 256% for RDMA operations using large message, and that RDMA Write-Record can outperform RC RDMA Write with up to a 24.4% improvement in latency. It was also discovered that the overhead of the software iWARP socket interface is minimal.

We have examined the bandwidth performance of iWARP under various packet loss scenarios and determined the MTUs most appropriate for given network conditions. We also determined that the considerations that we made for packet loss have had the desired effect on the overall bandwidth of our proposal; providing increased bandwidth and partial delivery for those applications that can take advantage of such features.

An MPI implementation was adapted to allow for the use of RDMA Write-Record and a comparison of the performance of it in combination with UD send/recv was conducted versus their RC equivalents. The performance benefit of using both methods was found to work well for applications that send large messages.

We evaluated the performance of some real-world applications, which demonstrated that datagram-iWARP could be useful in such contexts. It was observed that the real-world memory scalability and performance of datagram-iWARP over SIP is excellent, providing a memory savings of 24.1% and performance improvement of 43.1%, and that performance over VLC can outperform RC by 74.1%. These benefits will scale well with large commercial clusters. We have shown that sockets-based applications can take advantage of datagram-iWARP, and that it would be beneficial to develop a protocol similar to SDP, but for datagrams, that will leverage the advantages of RDMA Write-Record for socket applications, to translate the verbs performance benefits of Write-Record over to the sockets domain.

Chapter 7 Conclusions

7.1 Advanced Networking Technologies

The study of QoS in InfiniBand networks and UD-based offloading techniques on modern InfiniBand HCAs demonstrated that such techniques are important to networking performance, and next generation Ethernet networks should support such features. The research on InfiniBand QoS was to the best knowledge of this author, the first of its kind performed on InfiniBand [36]. The study of UD offloading on IPoIB was also the first of its kind performed [33][35].

For quality of service, the importance of being able to prioritize specific streams of traffic is very useful. Next-generation networks may be able to take advantage of multiple channels of different priority, and even different service level guarantees. One could provide a high speed unreliable channel and a lower speed guaranteed delivery channel, or several channels of each reliability type could be offered with differing performance. The possibility of future hybrid networking data centers further increases the usefulness of QoS, as traffic can be prioritized based on the protocol; for example, by prioritizing SDP traffic above that of TCP or UDP traffic. As the current level of technology in this area does not allow for hybrid network based QoS, this is an area of research that will deserve further study in the future.

The findings from the study of UD offloading techniques for IB HCAs yields important data regarding the performance improvement of subsequent generations of Ethernet networks. Primarily, it is found that the most important offloading occurs at the receiving side, in the data fetching from the HCA. Although send offloading reduces the CPU's involvement in network data transmission, the advantages in terms of performance are limited. The ability to pass large data payloads for segmentation on the HCA is of limited concern, while the ability to pass large amounts of data in a single transfer from the HCA is much more important. It is the aggregation of the incoming data packets that allows for increased networking efficiency. However, the parameters for how much data must arrive or how much time must elapse before delivering data

must be intelligently determined, as excessively poor choices can have an adverse effect on performance. As CPU capacity increases at the rate of the addition of extra cores to a processing die, the need for offloading send segmentation operations will most likely decrease. Such operations can be handled by the faster CPU, and if capacity is less of an issue than it is today, it may be desirable to not offer send offloading.

Receive offloading remains important even with increasing CPU capacity. The aggregation of incoming data allows for a more efficient use of the internal system buses, by only triggering events after a sufficient amount of data has arrived. In the future, CPUs could be dedicated to fetching data from the NIC, or such tasks could be significantly simplified by having the NIC perform direct memory placement of the data through a UD RDMA operation.

7.2 Hybrid Networking

In the study of advanced networks, performance issues surrounding the high-performance sockets interface, SDP, were discovered in a pure IB networking environment. The reason for this performance issue could be potentially attributed to connection level management overhead. Therefore, an SDP implementation was extended [35] to use eXtended Reliable Connections (XRC), which should reduce connection management overhead. This implementation demonstrated that connection management overhead is manageable at the hardware and software level. Therefore, after analysis, the SDP protocol overhead itself was found to be the main contributor to reduced performance. By utilizing a reduced number of connections, the performance can be greatly increased by the resulting increase in per connection traffic. This result leads to investigation of hybrid Ethernet/IB networks, where the most connection intensive links, the incoming client/server links from the WAN communication, are replaced with Ethernet connections.

The investigation of hybrid network data centers [34] was the first work to utilize Virtual Protocol Interconnect to create a hybrid network data center. It showed that such network

architectures can be superior to a completely IB network for web serving data center applications [35]. Dual-mode HCAs, with Ethernet and IB ports operating simultaneously work well as a network interface between traditional Ethernet and IB networks. The hybrid networking solution was found to be the best in terms of performance for a web serving data center over all other alternative architectures. The testing comparing jumbo frame Ethernet with normal frame Ethernet illustrated the benefits of using jumbo frames, while quantifying the performance penalty for using normal frames using various data center network configurations.

7.3 RDMA-Enabled Ethernet

The development of RDMA capable Ethernet operating over an unreliable datagram transport [92] is an important step in providing a high performance Ethernet solution that is applicable to the vast majority (along with traditional iWARP) of socket-based networking applications. The scalability of the solution is excellent, as is the case with most datagram-based transports. In addition, the UDP transport over Ethernet is able for the first time to offer true one-sided RDMA Write capability, through our newly design RDMA Write-Record mechanism [37][38][39]. This expanded ability not only significantly increases the applications space to which RDMA operations are available, but also provides superior performance over traditional methods of providing such functionality.

The extension of Ethernet to leverage the abilities of high speed networks was begun through the development of traditional TCP-based iWARP, and the design of UDP-based iWARP offers a fully featured networking solution that can compete with existing high speed networks both on speed and features. Datagram-iWARP also offers the possibility of providing lower cost high performance networking by offering only UD based offloading, where datagram-iWARP hardware could be potentially produced for less cost than either traditional iWARP or combined traditional/datagram-iWARP.

RDMA enabled Ethernet is also capable of operating in a software/hardware mode, where the hardware at the server side could be running iWARP to efficiency reasons while the individual clients could run a software iWARP stack. This makes it immediately useful at the server side, where more expensive networking hardware can be fully utilized, justifying its cost. The individual clients may not benefit significantly in terms of performance, but the increased efficiency of the data center hardware will bring benefits in expanded capacity, offering access to a greater number of users. Reduced cost will also benefit both the data center operator and the user as less hardware is required for the same amount of traffic (including reduced CPU requirements), service providers can pass on a portion of the cost savings to their customers. Ultimately, should RDMA-enabled Ethernet become widely used in the data center, the technology can eventually be migrated to the client side as the manufacturing quantities of hardware reduce individual chip cost. At this point clients would also see an appreciable increase in performance, which will be required for very high definition streaming video or other future high bandwidth applications.

7.4 Network Convergence

The overarching theme of this thesis is the topic of network convergence. The work detailed here has provided an insight into what aspects of current high performance networks would be useful to include in a future converged networking solution, as well as those features that are not as important. The investigation into hybrid networking architectures demonstrates the abilities of the current generation of networking technologies to work in tandem to produce a well performing data center.

The momentum towards Ethernet remaining the network of choice for future generation networks for commercial applications is an indicator of the great possibility that it will continue to remain the most widely used wired networking technology. Existing efforts to enable other networking stacks, such as InfiniBand, to work over native Ethernet frames, as well as the

development of hybrid technologies such as VPI demonstrate the further push towards convergence of these networking stacks with Ethernet. Our extensions to the iWARP standard, in order to bring many of the advantageous features of networks like IB to Ethernet is a step towards convergence of the RDMA networking technologies and existing Ethernet networks.

iWARP is excellent in terms of its convergence potential, as it is already operating over Ethernet natively. It also has the advantage that it can operate over the IP layer, which allows it to easily extend to wide area networking solutions. The alternative IB-stack over Ethernet solutions (RDMAoE), do not operate over the IP layer, and as such there is a greater barrier to adoption for network solutions that require wide area networking capabilities than datagram-iWARP. Therefore, we believe that the designs presented here are a good solution to bringing superior performance to next generation Ethernet networks, while providing all of the functionality of existing Ethernet networks, and backwards compatibility. This make datagram-iWARP in combination with traditional connection-based iWARP well suited for replacing the existing Ethernet infrastructure.

7.5 Future Work

The implementation of the proof of concept software iWARP networking stack is an excellent step in the direction of proving the performance and applicability of datagram-iWARP. However, some of the best features of iWARP cannot be leveraged in a software solution. OS bypass and zero-copy capabilities can only truly be leveraged through the use of a hardware solution.

Additional features can be added to datagram-iWARP. The current implementation only supports broadcast operations, therefore, support for multicast operations could be added. As iWARP operates over Ethernet natively, existing multicast protocols like IGMP can be used. Adding support for such operations for a send/recv operation is relatively straightforward, RDMA

Write-Record support is not straightforward and could only use existing protocols over Ethernet under certain circumstances.

The design and development of a fully-featured UDP interface for datagram-iWARP or any unreliable datagram verbs-based network is a valuable future project. By opening up the application space of UDP networking based programs, as well as those that utilize both TCP and UDP, the usefulness of verbs-based networks can be enhanced. To further increase the number of applications supported for datagram verbs networks, the extension of datagram-iWARP to work over a reliable datagram transport could be implemented. The techniques developed for the unreliable datagram-iWARP design detailed in this thesis are 100% compatible with a reliable datagram transport layer, as it will simply act as a the UDP transport would, without any packet loss. The use of a reliable datagram transport would be a new development for verbs networks, as no current verbs networks utilize reliable datagrams in practice. The reliable datagram mode of IB networks is defined in the IB specifications [54], but it is not currently implemented on any available IB hardware. The operation of datagram-iWARP over reliable datagrams does not require any different specifications than that provided for the unreliable design developed here, but the method of providing reliability must be carefully chosen so as to not add excessive overhead while providing an adequate method of ensuring delivery.

The design of RDMA Write-Record is applicable to other high speed networking architectures, like InfiniBand, although some additional design would be required in order to integrate the ideas behind the RDMA Write-Record transaction into a format that would be compatible with the existing IB UD transport.

The development of Converged Enhanced Ethernet will provide opportunities to introduce RDMA Write-Record type methods to traditionally non-RDMA type networks. The use of datagrams for RDMA operations is especially attractive for CEE as it has guaranteed delivery channels that can make use of lightweight UD transports.

References

[1] Advanced Micro Devices. AMD Athlon X2 dual core. [http://www.amd.com/us/products/desktop/processors/athlon-ii-x2/Pages/amd-athlon-ii-x2-dual-core-processors-desktop.aspx], Last accessed on: Sept. 20, 2012.

[2] F.J. Alfaro, J.L. Sánchez and J. Duato. "QoS in InfiniBand Subnetworks," *IEEE Transactions on Parallel and Distributed Systems,* 15(9):810-823, Sept. 2004.

[3] F.J. Alfaro, J.L. Sanchez and J. Duato, "Studying the influence of the InfiniBand packet size to guarantee QoS," In *10th IEEE Symposium on Computers and Communications (ISCC'05),* pages 989-994, 2005.

[4] AMQP Working Group. Advanced message queuing protocol. [http://www.amqp.org/], Last accessed on: Sept. 20, 2012.

[5] D. Anderson, "BOINC: a system for public-resource computing and storage," *Grid Computing, 2004. Proceedings. Fifth IEEE/ACM International Workshop on,* pp. 4-10, 2004.

[6] Argonne National Labs, MPICH. [http://www.mcs.anl.gov/research/projects/mpich2/]. Last accessed on: Sept. 20, 2012.

[7] P. Balaji, S. Bhagvat, R. Thakur and D. K. Panda, "Sockets direct protocol for hybrid network stacks: A case study with iWARP over 10G ethernet," in *Proceedings of the 15th International Conference on High Performance Computing (HiPC),* 2008, pp. 478.

[8] P. Balaji, S. Bhagvat, H. Jin and D. Panda, "Asynchronous zero-copy communication for synchronous sockets in the sockets direct protocol (SDP) over InfiniBand," in *Proceedings of the IEEE International Parallel and Distrbuted Porcessing Symposium Held in Conjunction with the Workshop on Communication Architecture for Clusters (CAC 06),* 2006, on CD, 8 pages.

[9] P. Balaji, W. Feng, S. Bhagvat, D. Panda, R. Thakur and W. Gropp, "Analyzing the impact of supporting out-of-order communication on in-order performance with iwarp," in *Proceedings of the 2007 ACM/IEEE Conference on Supercomputing,* 2007, on CD, 12 pages.

[10] P. Balaji, W. Feng, Q. Gao, R. Noronha, W. Yu and D. Panda, "Head-to-toe evaluation of high-performance sockets over protocol offload engines," in *Proceedings of the IEEE International Cluster Computing,* 2005, on CD, 10 pages.

[11] P. Balaji, H. Jin, K. Vaidyanathan and D. Panda, "Supporting iWARP compatibility and features for regular network adapters," in *Proceedings of the IEEE International Conference on Cluster Computing,* 2005, on CD, 10 pages.

[12] 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 *Proceedings of the IEEE International Symposium on Performance Analysis of Systems and Software,* 2004, pp. 10-12.

[13] P. Balaji, H.V. Shah and D.K. Panda, "Sockets vs RDMA interface over 10-Gigabit networks: an in-depth analysis of the memory traffic bottleneck," in *Proceedings of the Workshop on Remote Direct Memory Access (RDMA): Applications, Implementations, and Technologies,* 2004, on CD, 10 pages.

[14] J. Beecroft, D. Addison, D. Hewson, M. McLaren, D. Roweth, F. Petrini and J. Nieplocha, "QsNetII: Defining High-Performance Network Design," *IEEE Micro,* pp. 34-47, 2005.

[15] P. N. Brown, R. D. Falgout, J. E. Jones, "Semicoarsening multigrid on distributed memory machines." SIAM Journal on Scientific Computing - 21 (2000), pp. 1823-1834.

[16] S. Carter, M. Minich and N. S. V. Rao, "Experimental evaluation of InfiniBand transport over local-and wide-area networks," in *Proceedings of the 2007 Spring Simulation Multiconference,* 2007, pp. 419-426.

[17] Chelsio Corp. T4 iWARP Adapter. [http://www.chelsio.com/products/t4_unified_wire_adapters/], Last accessed on: Sept. 20, 2012.

[18] H. J. Chu, "Zero-copy TCP in solaris," in *Proceedings of the Annual Technical Conference on USENIX 1996 Annual Technical Conference*, 1996, pp. 21-21.

[19] Cisco Systems Inc. (May 30, 2012). The Zettabyte Era, [http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/VNI_Hyperconnectivity_WP.pdf], Last accessed on: Sept. 20, 2012.

[20] D. Cohen, T. Talpey, A. Kanevsky, U. Cummings, M. Krause, R. Recio, D. Crupnicoff, L. Dickman, P. Grun, "Remote Direct Memory Access over the Converged Enhanced Ethernet Fabric: Evaluating the Options", in *Proceedings of the 17th IEEE symposium on High Performance Interconnects*, 2009, pp. 123-130.

[21] P. Culley, U. Elzur, R. Recio, S. Baily, et. al. "Marker PDU Aligned Framing for TCP Specification (Version 1.0)", RDMA Consortium, October 2002.

[22] D. Dalessandro, A. Devulapalli, P. Wyckoff, O. S. Center and O. Springfield, "iWarp protocol kernel space software implementation," in *Proceedings of the 20th International Parallel and Distributed Processing Symposium,* 2006, on CD, 8 pages.

[23] D. Dalessandro, A. Devulapalli and P. Wyckoff, "Design and implementation of the iWarp protocol in software," in *Proceedings of the 17th IASTED International Conference on Parallel and Distributed Computing and Systems,* 2005, pp. 471-476.

[24] D. Dalessandro, P. Wyckoff, O. S. Center and O. Springfield, "A performance analysis of the ammasso RDMA enabled ethernet adapter and its iWARP API," in *Proceedings of the 2005 IEEE International Conference on Cluster Computing,* 2005, on CD, 7 pages.

[25] D. Dunning, G. Regnier, G. McAlpine, D. Cameron, B. Shubert, F. Berry, A. Merritt, E. Gronke and C. Dodd, "The Virtual Interface Architecture," *Micro, IEEE,* vol. 18, pp. 66-76, 1998.

[26] FastMQ Inc. ZeroMQ. [http://www.amqp.org/], Last accessed on: Sept. 20, 2012.

[27] W. Feng, P. Balaji, C. Baron, L. Bhuyan and D. Panda, "Performance characterization of a 10-gigabit ethernet TOE," in *Proceedings of the 13th Symposium on High Performance Interconnects,* 2005, pp. 58-63.

[28] W. Fischer, E. Wallmeier, T. Worster, S. P. Davis, A. Hayter, "Data communications using ATM: architectures, protocols, and resource management," *Communications Magazine, IEEE* , vol.32, no.8, pp.24-33, Aug. 1994

[29] A. Foong, T. Huff, H. Hum, J. Patwardhan and G. Regnier, "TCP performance re-visited," in *Proceedings of the 2003 IEEE International Symposium on Performance Analysis of Systems and Software,* 2003, pp. 70-79.

[30] I. Foster, "Globus Toolkit Version 4: Software for Service-Oriented Systems," in *Proceedings of the IFIP International Conference on Network and Parallel Computing*, 2005, pp 2-13.

[31] R. Gayraud et al., "SIPp Traffic Generator", May 2012; http://sipp.sourceforge.net/.

[32] D. Goldenberg, M. Kagan, R. Ravid, M. Tsirkin, M. T. Inc and S. Clara, "Transparently achieving superior socket performance using zero copy socket direct protocol over 20Gb/s InfiniBand links," in *Proceedings of the 2005 IEEE International Conference on Cluster Computing,* 2005, on CD, 10 pages.

[33] R. E. Grant, P. Balaji, and A. Afsahi, "A Study of Hardware Assisted IP over InfiniBand and its Impact on Data Center Performance", in *Proceedings of the 2010 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS 2010)*, pp 144-153.

[34] R. E. Grant, P. Balaji and A. Afsahi, "An evaluation of ConnectX virtual protocol interconnect for data centers," *in Proceedings of the 15th International Conference on Parallel and Distributed Systems (ICPADS 2009)*, 2009, pp. 57-64.

[35] R. E. Grant, P. Balaji, and A. Afsahi, "High Speed Sockets for Data center Applications," under review at the Journal of Concurrency and Computation: Practice and Experience (CCPE).

[36] 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 *Proceedings of the the 37Th International Conference on Parallel Processing (ICPP 2008)*, pp. 79-86.

[37] R. E. Grant, M. J. Rashti, P. Balaji, and A. Afsahi, "RDMA Capable iWARP over Datagrams", in *Proceedings of the 25th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2011),* 2011, pp. 628-639.

[38] R. E. Grant, M. J. Rashti, P. Balaji, A. Afsahi, "Remote Direct Memory Access over Datagrams," U.S. Patent Pending, Dec. 2011.

[39] R. E. Grant, M. J. Rashti, P. Balaji, A. Afsahi, "Scalable Connectionless RDMA over Ethernet," under review at the International Journal of Parallel and Distributed Computing (JPDC).

[40] L. Hammond, B. A. Nayfeh and K. Olukotun, "A Single-Chip Multiprocessor," *IEEE Computer,* vol. 30, pp. 79-85, 1997.

[41] B. Hauser, "iWARP Ethernet: Eliminating Overhead in Data Center Designs", NetEffect Inc., 2006.

[42] E. He, J. Leigh, O. Yu and T. DeFanti, "Reliable blast UDP: Predictable high performance bulk data transfer," in *Proceedings of the IEEE International Conference on Cluster Computing,* 2002, pp. 317-324.

[43] J. Hilland, P. Culley, J. Pinkerton, R. Recio. "RDMA Protocol Verbs Specification (version 1.0)", RDMA Consortium, October 2002.

[44] T. Horvath. TPC-W java client implementation. [http://www.cs.virginia.edu/~th8k/downloads/], Last accessed on: Sept. 20, 2012.

[45] W. Huang, J. Han, J. He, L. Zhang and Y. Lin, "Enabling RDMA capability of InfiniBand network for java applications," in *Proceedings of the International Conference on Networking, Architecture, and Storage, NAS'08.* 2008, pp. 187-188.

[46] G. Huston, "TCP Performance," The Internet Protocol Journal - Volume 3, No. 2, Cisco Systems, June 2000.

[47] IBM System Z - Parallel Sysplex [http://www-03.ibm.com/systems/z/advantages/pso/], Last accessed on: Sept. 20, 2012.

[48] IBM. Websphere MQ. [http://www.ibm.com/software/integration/wmq], Last accessed on: Sept. 20, 2012.

[49] 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], Last accessed on: Sept. 20, 2012.

[50] 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], Last accessed on: Sept. 20, 2012.

[51] 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], Last accessed on: Sept. 20, 2012.

[52] IEEE. IEEE standard for station and media access control connectivity - 802.1AB. [http://www.ieee802.org/1/pages/802.1ab.html], Last accessed on: Sept. 20, 2012.

[53] 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], Last accessed on: Sept. 20, 2012.

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

[55] InfiniBand Trade Association, “InfiniBand Architecture Specification, Volume 1, Annex A14: Extended Reliable Connected (XRC) Transport Service”, March, 2009.

[56] InfiniBand Trade Association. [www.infinibandta.com], Last accessed on: Sept. 20, 2012.

[57] Intel Corp. Intel Xeon processor. [http://www.intel.com/content/www/us/en/processors/xeon/xeon-processor-e7-family.html], Last accessed on: Sept. 20, 2012.

[58] Intel Corp. Intel Core i7 processors [www.intel.com/content/www/us/en/processors/core/core-i7-processor.html], Last accessed on: Sept. 20, 2012.

[59] Intel Corp., (2006) "Accelerating High-Speed Networking with Intel I/O Acceleration Technology," [http://www.intel.com/content/dam/doc/white-paper/i-o-acceleration-technology-paper.pdf], Last accessed on: Sept. 20, 2012.

[60] Internet Engineering Taskforce. Transparent interconnection of lots of links (TRILL). [http://www.ietf.org/dyn/wg/charter/trill-charter.html], Last accessed on: Sept. 20, 2012.

[61] J. Jithin, H. Subramoni, K. Kandella, W. Rahman, H. Wang, S. Reddy, D. Panda, "Scalable Memcached design for InfiniBand Clusters using Hybrid Transports," in *Proceedings of the 2012 IEEE Symposium on cluster, Grid and Cloud Computing*, 2012, pp. 236-243.

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

[63] V. Kashyap, "IP over InfiniBand (IPoIB) architecture," RFC 4392, April 2006.

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

[65] P. Kongetira, K. Aingaran and K. Olukotun, "Niagara: a 32-way multithreaded Sparc processor," *Micro, IEEE,* vol. 25, pp. 21-29, 2005.

[66] M. J. Koop, W. Huang, K. Gopalakrishnan and D. K. Panda, "Performance analysis and evaluation of PCIe 2.0 and quad-data rate InfiniBand," in *Proceedings of the 16th IEEE Symposium on High Performance Interconnects,* 2008, pp. 85-92.

[67] M. J. Koop, T. Jones and D. K. Panda, "MVAPICH-aptus: Scalable high-performance multi-transport MPI over InfiniBand," in *Proceedings of the IEEE International Symposium on Parallel and Distributed Processing,* 2008, on CD, 12 pages.

[68] M. J. Koop, J. K. Sridhar and D. K. Panda, "Scalable MPI design over InfiniBand using eXtended reliable connection," in *Proceedings of the 2008 IEEE International Conference on Cluster Computing,* 2008, pp. 203-212.

[69] M. Koop, S. Sur, Q. Gao, D. K. Panda, "High Performance MPI Design using Unreliable Datagram for Ultra-Scale InfiniBand Clusters," in *Proceedings of the 21st ACM International Conference on Supercomputing (ICS07)*, 2007, pp. 180-189.

[70] G. Marsh, A. P. Sampat, S. Potluri and D. K. Panda, "Scaling advanced message queuing protocol (AMQP) architecture with broker federation and InfiniBand," 2009. [ftp://ftp.cse.ohio-state.edu/pub/tech-report/2009/TR17.pdf], Last accessed on: Sept. 20, 2012.

[71] Mellanox Technologies. (2008). "The case for InfiniBand over ethernet." [http://www.mellanox.com/pdf/whitepapers/WP_The_Case_for_InfiniBand_over_Ethernet.pdf], Last accessed on: Sept. 20, 2012.

[72] Mellanox Technologies. [www.mellanox.com], Last accessed on: Sept. 20, 2012.

[73] R. M. Metcalfe and D. R. Boggs, "Ethernet: Distributed packet switching for local computer networks," *Communications of the ACM,* vol. 19, pp. 395-404, 1976.

[74] B. Metzler, P. Frey, A. Trivedi, “A Software iWARP Driver for OpenFabrics” IBM Zurich Research Lab, Presented in OpenFabrics Alliance 2010 Sonoma Workshop, March 2010.

[75] J. C. Mogul, "TCP offload is a dumb idea whose time has come," in *Proceedings of the 9th Conference on Hot Topics in Operating Systems,* 2003, on CD, 5 pages.

[76] MPI Forum, ‘MPI: A Message Passing Interface Standard,” v2.2, September 2009.

[77] Myricom Inc., Myri-10G overview. [http://www.myricom.com/products/network-adapters.html], Last accessed on: Sept. 20, 2012.

[78] G. Narayanaswamy, P. Balaji and W. Feng, "Impact of network sharing in multi-core architectures," in *Proceedings of 17th International Conference on Computer Communications and Networks, 2008. ICCCN'08.* 2008, on CD, 6 pages.

[79] S. Narravula, P. Balaji, K. Vaidyanathan, H. W. Jin and D. Panda, "Architecture for caching responses with multiple dynamic dependencies in multi-tier data-centers over InfiniBand," in *Proceedings of the Fifth IEEE International Symposium on Cluster Computing and the Grid (CCGrid'05)-Volume 1,* 2005, pp. 374-381.

[80] NAS Parallel Benchmarks, version 2.4: [http://www.nas.nasa.gov/Resources/Software/npb.html], Last accessed on: Sept. 20, 2012.

[81] Network-Based Computing Laboratory, “MVAPICH: MPI over InfiniBand, iWARP and RDMAoE”, Ohio State University: [http://mvapich.cse.ohio-state.edu/], Last accessed on: Sept. 20, 2012..

[82] Network Working Group, “Stream Control Transmission Protocol (SCTP)”, Editor: R. Stewart, IETF RFC4960, September 2007.

[83] J. O'Hara, "Toward a commodity enterprise middleware," *Queue,* vol. 5, pp. 48-55, 2007.

[84] Ohio Supercomputer Center, “Software Implementation and Testing of iWARP Protocol”: [http://www.osc.edu/research/network_file/projects/iwarp/iwarp_main.shtml], Last accessed on: Sept. 20, 2012.

[85] Open Fabrics Alliance. [http://www.openfabrics.org/], Last accessed on: Sept. 20, 2012.

[86] Open MPI: Open Source High Performance Computing. [http://www.open-mpi.org/], Last accessed on: Sept. 20, 2012.

[87] S. Patel, S. Philips, A. Strong, "Sun's Next-Generation Multi-threaded Processor - Rainbow Falls," in *Proceedings of the Hot Chips Symposium*, 2009.

[88] J. Pinkerton, E. Deleganes, M. Krause, “Sockets Direct Protocol for iWARP over TCP”, RDMA Consortium, October 2003.

[89] N. S. V. Rao, W. Yu, W. R. Wing, S. W. Poole and J. S. Vetter, "Wide-area performance profiling of 10GigE and InfiniBand technologies," in *Proceedings of the ACM/IEEE Conference on Supercomputing,* 2008, on CD, 12 pages.

[90] M. J. Rashti and A. Afsahi, "10-gigabit iWARP ethernet: Comparative performance analysis with InfiniBand and myrinet-10G," in *Proceedings of the IEEE International Parallel and Distributed Processing Symposium,* 2007, on CD, 8 pages.

[91] M. J. Rashti and A. Afsahi, "A Speculative and Adaptive MPI Rendezvous Protocol Over RDMA-Enabled Interconnects," *International Journal of Parallel Programming,* vol. 37, pp. 223-246. 2009.

[92] M. J. Rashti, R. E. Grant, P. Balaji, A. Afsahi, "iWARP Redefined: Scalable Connectionless Communication over High-Speed Ethernet", in *Proceedings of the High Performance Computing Conference* (HiPC'10), 2010, on CD, 10 pages.

[93] RDMA Consortium. [www.rdmaconsortium.org], Last accessed on: Sept. 20, 2012.

[94] R. Recio, P. Culley, D. Garcia, J. Hilland and B. Metzler. April 2005. An RDMA protocol specification. [http://tools.ietf.org/html/draft-ietf-rddp-rdmap-07.txt], Last accessed on: Sept. 20, 2012.

[95] G. Regnier, S. Makineni, I. Illikkal, R. Iyer, D. Minturn, R. Huggahalli, D. Newell, L. Cline and A. Foong, "TCP onloading for data center servers," *Computer,* vol. 37, pp. 48-58, 2004.

[96] S. A. Reinemo, T. Skeie, T. Sødring, O. Lysne and O. Tørudbakken, "An Overview of QoS Capabilities in InfiniBand, Advanced Switching Interconnect, and Ethernet," *IEEE Communications Magazine,* vol. 44, pp. 32-38, 2006.

[97] B. Rochwerger, J. Caceres, R. Montero, D. Breitgand, E. Elmroth, et al. "The RESERVOIR Model and Architecture for Open Federated Cloud Computing", *IBM Journal of Research and Development*, Vol. 53, pp. 4:1-11, 2009.

[98] H. Schulzrinne, S. Casner, R. Frederick, V. Jacbosen, "RTP: A Trasnport Protocol for Real-Time Applications", Audio-Video Transport Working Group, RFC 3550, July 2003.

[99] H. Shah, J. Pinkerton, R. Recio, P. Culley. "Direct Data Placement over Reliable Transports (version 1.0)". RDMA Consortium, October 2002.

[100] H. V. Shah, C. Pu and R. S. Madukkarumukumana, "High Performance Sockets and RPC over Virtual Interface (VI) Architecture," *Lecture Notes in Computer Science,* vol. 1602, pp. 91-107, 1999.

[101] M. Shah, J. Barreh, J. Brooks, R. Golla, G. Grohoski, N. Gura, R. Hetherington, P. Jordan, M. Luttrell and C. Olson, "UltraSPARC T2: A highly-treaded, power-efficient, SPARC

SOC," in *Proceedings of the 2007 IEEE Asian Solid-State Circuits Conference,* 2007, pp. 22-25.

[102] M. Shah, R. Golla, G. Grohoski, P. Jordan, J. Barreh, J. Brooks, M. Greenberg, G. Levinsky, M, Luttrell, C. Olson, Z. Samoail, M. Smittle, T. Ziaja, "Sparc T4: A Dynamically Threaded Server-on-a-Chip," *Micro, IEEE* , vol. 32, no. 2, pp.8-19, March-April 2012

[103] H. Shan, J.P. Singh, L. Oliker, R. Biswas, "Message Passing and Shared Address Space Parallelism on an SMP Cluster", *Parallel Computing*, 29(2): 167–186, 2003.

[104] G. M. Shipman, R. Brightwell, B. Barrett, J. M. Squyres and G. Bloch, "Investigations on infiniband: Efficient network buffer utilization at scale," *Lecture Notes in Computer Science,* vol. 4757, pp. 178, 2007.

[105] J. F. Shoch and J. A. Hupp, "Measured performance of an Ethernet local network," *Communications of the ACM*, vol. 23, pp. 711-721, 1980.

[106] SLAC National Accelerator Laboratory, "PingER", Aug. 2011; [http://www-iepm.slac.stanford.edu/pinger/], Last accessed on: Sept. 20, 2012.

[107] L. Spracklen and S. G. Abraham, "Chip multithreading: Opportunities and challenges," in *HPCA '05: Proceedings of the International Symposium on High-Performance Computer Architecture,* 2005, pp. 248-252.

[108] H. Subramoni, P. Lai, M. Luo, D. K. Panda, "RDMA over Ethernet - A Preliminary Study", in *Proceedings of the Workshop on High Performance Interconnects for Distributed Computing (HPIDC'09)*, 2009, on CD, 9 pages.

[109] H. Subramoni, P. Lai , S. Sur, D. K. Panda, "Improving Application Performance and Predictability using Multiple Virtual Lanes in Modern Multi-Core InfiniBand Clusters", in *Proceedings of the International Conference on Parallel Processing (ICPP '10)*, 2010, pp. 462-471.

[110] H. Subramoni, G. Marsh, S. Narravula, P. Lai and D. K. Panda, "Design and evaluation of benchmarks for financial applications using advanced message queuing protocol (AMQP) over InfiniBand" in *Proceedings of the Workshop on High Performance Computational Finance (in Conjunction with SC'08),* 2008, on CD, 8 pages.

[111] Sun Microsystems. Java messaging service. [http://java.sun.com/products/jms], Last accessed on: Sept. 20, 2012.

[112] S. Sur, M. J. Koop, L. Chai and D. K. Panda, "Performance analysis and evaluation of Mellanox ConnectX InfiniBand architecture with multi-core platforms," in *Proceedings of the 15th Symposium on Hot Interconnects,* 2007, pp. 125-134.

[113] A. Tirumala, F. Qin, J. Dugan, J. Ferguson and K. Gibbs, “Iperf: The TCP/UDP bandwidth measurement tool,” [http://sourceforge.net/projects/iperf/*],* Last accessed on: Sept. 20, 2012.

[114] A. Totok. "TPC-W-NYU" [http://cs.nyu.edu/~totok/professional/software/tpcw/tpcw.html], Last accessed on: Sept. 20, 2012.

[115] TPC Council, "TPC-W Benchmark" [http://www.tpc.org/tpcw/default.asp], Last accessed on: Sept. 20, 2012.

[116] N. Tuck and D. M. Tullsen, "Initial observations of the simultaneous multithreading pentium 4 processor," in *PACT '03: Proceedings of the 12th International Conference on Parallel Architectures and Compilation Techniques,* 2003, pp. 26-34.

[117] D. M. Tullsen, S. J. Eggers and H. M. Levy, "Simultaneous multithreading: Maximizing on-chip parallelism," in *ISCA '95: Proceedings of the 22nd Annual International Symposium on Computer Architecture,* 1995, pp. 392-403.

[118] K. Vaidyanathan and D. K. Panda, "Benefits of I/O acceleration technology (I/OAT) in clusters," in *the Proceedings of the IEEE International Symposium on Performance Analysis of Systems & Software, (ISPASS 2007),* 2007, pp. 220-229.

[119] VideoLan Project, “VLC Media Player”, Aug. 2011; [http://www.videolan.org/vlc/.]

[120] D. Walker and J. Dongarra, "MPI: a standard Message Passing Interface," *Supercomputer,* vol. 12, pp. 56-68, 1996.

[121] W. Yu, N. S. V. Rao and J. S. Vetter, "Experimental analysis of infiniband transport services on WAN," in *Proceedings of the International Conference on Networking, Architecture, and Storage, 2008. NAS'08.* 2008, pp. 233-240.

[122] 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 *Proceedings of the IFIP International Conference on Network and Parallel Computing Workshops, 2007. (NPC Workshops),* 2007, pp. 602-607.

Appendix I: Publication List

Patents

Ryan E. Grant, Mohammad J. Rashti, Pavan Balaji, Ahmad Afsahi, "Remote Direct Memory Access over Datagrams", U.S. Patent Pending, Dec. 2010.

Journal Publications

Ryan E. Grant, Mohammad J. Rashti, Pavan Balaji, and Ahmad Afsahi, "Scalable Datagram-Based RDMA over Ethernet", under review by the International Journal of Parallel and Distributed Computing (JPDC)

Ryan E. Grant, Pavan Balaji, and Ahmad Afsahi, "High Speed Sockets for Data center Applications", under review by the Journal of Computation and Concurrency: Practice and Experience (CCPE)

Conference Publications

Ryan E. Grant, Mohammad J. Rashti, Pavan Balaji, and Ahmad Afsahi, "RDMA Capable iWARP over Datagrams", in *Proceedings of the25th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2011)*, Anchorage, Alaska, USA, May 16-20, 2011, pp. 628-639. (Acceptance rate 19.6%, 12 pages IEEE dual-column format)

Mohammad J. Rashti, **Ryan E. Grant**, Pavan Balaji, and Ahmad Afsahi, "iWARP Redefined: Scalable Connectionless Communication over High-Speed Ethernet", in *Proceedings of the 17th International Conference on High Performance Computing (HiPC 2010)*, Goa, India, December 19-22, 2010, pp. 1-10. (Acceptance rate 20.5%, 10 pages IEEE dual-column format)

Ryan E. Grant, Pavan Balaji, and Ahmad Afsahi, "A Study of Hardware Assisted IP over InfiniBand and its Impact on Data Center Performance", In *Proceedings of the 2010 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS 2010)*, White Plains, NY, USA, March 28-30, 2010, pp 144-153. (Acceptance rate 34%, 10 pages IEEE dual-column format)

Ryan E. Grant, Ahmad Afsahi, and Pavan Balaji, "An Evaluation of ConnectX Virtual Protocol Interconnect for Data Centers", In *Proceedings of the15th International Conference on Parallel and Distributed Systems (ICPADS 2009)*, Shenzhen, China, December 8-11, 2009, pp. 57-64. (Acceptance rate 29.8%, 10 pages IEEE dual-column format)

Ryan E. Grant, Mohammad J. Rashti, and Ahmad Afsahi, "An Analysis of QoS Provisioning for Sockets Direct Protocol vs. IPoIB over Modern InfiniBand Networks", In *Proceedings of the 37th International Conference on Parallel Processing (ICPP 2008)*, Portland, Oregon, USA, September 12, 2008, pp. 79-86. (8 pages IEEE dual-column format).