RDMA Capable iWARP over Datagrams

Ryan E. Grant, Mohammad J. Rashti, Ahmad Afsahi
Electrical and Computer Engineering,
Queen's University,
Kingston, ON, Canada
{ryan.grant, mohammad.rashti, ahmad.afsahi}@queensu.ca

Pavan Balaji
Mathematics and Computer Science,
Argonne National Laboratory,
Argonne, IL, USA
balaji@mcs.anl.gov

***Abstract*- iWARP is a state of the art high-speed connection-based RDMA networking technology for Ethernet networks to provide InfiniBand-like zero-copy and one-sided communication capabilities over Ethernet. Despite the benefits offered by iWARP, many datacenter and web-based applications, such as stock-market trading and media-streaming applications, that rely on datagram-based semantics (mostly through UDP/IP) cannot take advantage of it because the iWARP standard is only defined over reliable, connection-oriented transports. This paper presents an RDMA model that functions over reliable and unreliable datagrams. The ability to use datagrams significantly expands the application space serviced by iWARP and can bring the scalability advantages of a connectionless transport to iWARP. In our previous work, we had developed an iWARP datagram solution using send/receive semantics showing excellent memory scalability and performance benefits over the current TCP-based iWARP. In this paper, we demonstrate an improved iWARP design that provides true RDMA semantics over datagrams.**

Specifically, because traditional RDMA semantics do not map well to unreliable communication, we propose RDMA Write-Record, the first and the only method capable of supporting RDMA Write over both unreliable and reliable datagrams. We demonstrate through a proof-of-concept software implementation that datagram-iWARP is feasible for real-world applications. Our proposed RDMA Write-Record method has been designed with data loss in mind and can provide superior performance under conditions of packet loss. It is shown through micro-benchmarks that by using RDMA capable datagram-iWARP a maximum of 256% increase in large message bandwidth and a maximum of 24.4% improvement in small message latency can be achieved over traditional iWARP. For application results we focus on streaming applications, showing a 24% improvement in memory usage and up to a 74% improvement in performance, although the proposed approach is also applicable to the HPC domain.

Keywords - iWARP; RDMA; Networking; Datagrams; High performance networks

I. INTRODUCTION

Ethernet is the undisputed technology of choice for modern commercial data centers. Unfortunately, many of the technical enhancements in use in the high performance networking area have not been incorporated into Ethernet. Efforts have been made utilizing *Transmission Control Protocol over IP* (TCP/IP) offloading schemes such as stateless offloads (checksumming, segmentation etc.) and stateful TCP offload engines, with stateless offloading being implemented more frequently than stateful offloading in commercial devices. *iWARP* (Internet Wide Area RDMA Protocol) [23] was proposed to offer *Remote Direct Memory Access* (RDMA) functionality over Ethernet. This allowed for significantly lower latencies and reduced CPU utilization by offering kernel bypass, zero-copy based communications, and non-interrupt based asynchronous communication [1].

Despite the benefits offered by iWARP, many datacenter and web-based applications, such as stock-market trading and media-streaming applications, that rely on datagram-based semantics (mostly through the *User Datagram Protocol over IP* (UDP/IP)) cannot take advantage of it because the iWARP standard is only defined over reliable, connection-oriented transports. Moreover, currently one-sided RDMA operations (such as RDMA Write) are only defined on reliable and connected transports. This effectively limits the number of applications that could utilize iWARP and its RDMA capabilities by excluding UDP, which according to [2] could comprise more than 90% of all Internet consumer traffic by 2014.

Connection-based iWARP has a number of limitations that make it inappropriate for large-scale systems. First, the scalability of the current iWARP is limited since the hardware needs to keep data for each and every connection in hardware or host memory. This limits its effectiveness for applications that are required to service a very large number of clients at a single time. In addition, the current iWARP standard suffers from numerous overheads associated with connection-based transports such as TCP and *Stream Control Transmission Protocol* (SCTP) [17]. High overhead reliability and flow-control measures in TCP and SCTP protocols impose the burden of unnecessary communication processing on applications running on low error-rate networks (such as *High Performance Computing* (HPC) and data center clusters) as well as applications which do not require reliability.

Moreover, the complexities and overhead associated with packet marking, which is required to adapt the message-oriented iWARP stack over the stream-oriented TCP protocol, further reduce the overall message rate that can be achieved with the current TCP-based iWARP standard.

In [22], we proposed send/recv datagram support for iWARP for use in high-performance computing environments, with corresponding reliability mechanisms. It has been demonstrated that in such 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 paper extends the work in [22] 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, in this paper 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. We believe that 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 [12].

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 *Real-time Transport Protocol* (RTP), 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) [20]. 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. We will concentrate on sockets-based application in this paper.

Through our verbs level tests, we find that RDMA Write-Record over datagrams has a maximum bandwidth improvement of over 256% compared to *Reliable Connection* (RC) RDMA Write. We also see over a 24% improvement in small message latency using UD RDMA Write-Record over RC RDMA Write. Application results using our socket interface show a 24.1% memory usage improvement for a SIP server application and performance improvements for SIP applications (43.1%) and media streaming (VLC) applications (74.1%).

This paper is organized as follows. In Section II, we discuss the current iWARP standard. Section III reviews related scholarly work. Section IV, details the changes required to iWARP to include datagram functionality, as well as details about how we can provide one-sided RDMA operations over datagrams. Section V discusses the design of our software implementation. Section VI details the experimental platform and our evaluation results. Section VII provides our conclusions and discusses future work.

II. iWARP Background

Proposed by the RDMA Consortium [23] in 2002 to the IETF [13], iWARP is a specification that uses a user level stack over either TCP or SCTP protocols over Ethernet. It uses an interface referred to as verbs in order to interact with the stack [10]. These verbs are different in semantics than a traditional socket interface as iWARP was designed as an alternative to the operating system networking stack. Figure 1 shows the iWARP stack next to the traditional OS TCP/IP stack for comparison, and illustrates how the iWARP stack can completely bypass the kernel's networking stack.

Kernel bypass has several advantages, it can prevent extra copies of the data being made during processing (zero copy) and it completely offloads the processing overhead onto the Network Interface Card (NIC) hardware away from the CPU. In order to successfully offload this task, the NIC needs to have both stateless and stateful offloading capabilities. This allows it to perform all of the tasks normally done in the OS networking stack in hardware.

The iWARP stack consists of three unique fundamental layers underneath the verbs interface, the *Remote Direct Memory Access Protocol* (RDMAP) layer, the *Direct Data Placement* (DDP) layer and the *Marker PDU Alignment* (MPA) layer, as shown in Figure 1. A NIC that implements these layers on top of TOE is called and *RDMA-enabled NIC* or RNIC.

The RDMAP layer [24] is responsible for providing the basic communication primitives, namely the send, receive, RDMA write and RDMA read functions. It directly uses the

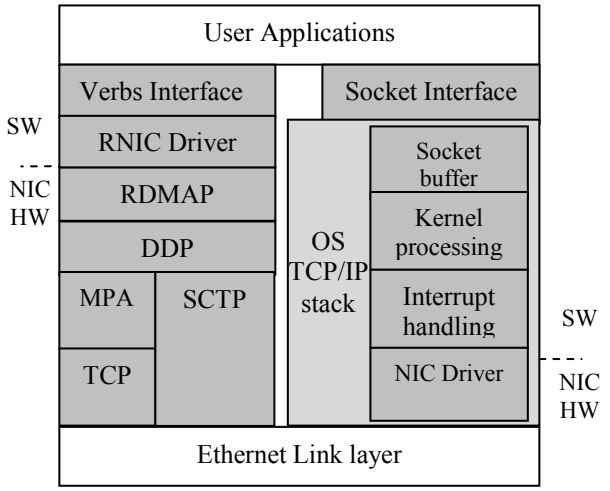

Figure 1. iWARP standard stack compared to host-based TCP/IP

DDP layer to accomplish this and is a relatively lightweight layer. All requests to the RDMAP layer from the verbs interface are delivered in order to the DDP layer.

The DDP layer [25] allows for data to be directly fetched from or written to memory without any intermediary copies. For the untagged model (send/recv), it places incoming data by matching the incoming packet's header to an existing sink. The receiver side handles all of the buffer management and determines where incoming data will be placed. The communication between both sender and target utilizes queues at each side, and is referred to as a *Queue Pair* (QP). In the tagged model (RDMA Write/Read), the header contains a steering tag (STag) that indicates the memory location where the data should be placed. This STag is known by the sender of the data via a previous advertisement of the available buffers on the receiving node. Data to be written or read from advertised buffers are accompanied by an offset value and a length, in order to be properly placed. The receiving machine enforces the requirement that the requested memory location must be registered with the device as a valid memory region before placing the data.

The MPA layer is responsible for placing markers in the outgoing messages and removing them from incoming messages [4]. These markers point to the appropriate header for a given message. This is required in a stream-based protocol as the messages can be segmented by intermediate devices, which requires that there be some way to determine what message an incoming packet belongs to. Such functionality is not needed for datagrams as they have defined message boundaries and are not segmented by intermediary devices. This allows message-based protocols to avoid this costly activity of inserting and removing markers from packets.

III. Related Work

Evaluations of 10 Gigabit Ethernet NICs offload engines [7], and a comparison of the high-speed networks discussed in [21] have been previously performed. The implementation of the current iWARP standard in software, originated from a project by OSC [18] that provides both user-space [5] and kernel space [6] implementations. The user-space part of this implementation is the code-base for implementing datagram-iWARP. The SoftRDMA project at IBM Zurich laboratory has finished a software iWARP solution [14] to be integrated into Open Fabrics Enterprise Distribution stack [19].

Beside the iWARP solution, there have been other approaches with the goal of improving Ethernet efficiency. The Open-MX project is an open-source implementation of Myrinet Express over Ethernet (MXoE) [9]. Local Area Network (LAN) RDMA technology such as MXoE [16] and InfiniBand [12] over Ethernet (IBoE) have shown the effectiveness of RDMA over LANs. IBoE, also called RDMA over Ethernet (RDMAoE) is designed to take advantage of InfiniBand's RDMA stack by replacing InfiniBand's link layer with Ethernet [27]. This technology encapsulates InfiniBand reliable and unreliable services (only send/recv is defined over unreliable datagrams) inside Ethernet frames.

A new set of standards referred to as *Converged Enhanced Ethernet* (CEE) has opened up issues revolving around providing advanced features over Ethernet networks. Some industry vendors and researchers [3] are also proposing to include RDMA functionality over CEE (RDMAoCEE).

Our work in [22] introduced send/recv datagram-iWARP in an HPC context over Message Passing Interface (MPI) [15]. We have defined a send/recv iWARP datagram solution that works over both unreliable and reliable datagram transports. We analyzed the performance of such a scheme within a high-performance computing context using an MPI interface. We found that the datagram based iWARP solution increased bandwidth by up to 20% and decreased latency by up to 16% while providing a 30% improvement in application memory usage and a 40% reduction in runtime for scientific applications. This work expands on that previous work by introducing RDMA Write-Record and our sockets interface, as well as studying the performance of datagram-iWARP in a data center context and under conditions of packet loss.

IV. Datagram-iWARP

Datagram-iWARP expands on the iWARP standard, to add support for UDP alongside TCP and SCTP. There are several applications in which UDP is preferable over TCP or SCTP. This section will discuss the benefits of datagram-iWARP as well as the design of datagram-iWARP for both tagged (one-sided) and untagged (send/recv) operations.

A. Motivation

Datagram iWARP brings the advantages of high-speed networking technology to applications that are currently bound by the limitations of traditional Ethernet. One of the most important advantages is the increased performance that using datagrams provides. The high overhead reliability and flow control methods used in TCP [11] are not present in UDP, which allows it to function with reduced latency and

increased throughput. TCP and SCTP both guarantee in-order delivery of data. Therefore, in the case of packet loss, there can be a significant delay in delivering data that have already arrived but is blocked by an incomplete message at the head of the receive queue, causing network jitter. The goal of using UDP as opposed to SCTP or TCP is to provide a very low overhead lightweight solution that provides a transport to applications capable of handling an unreliable transport, such as applications that handle large amounts of time sensitive data, like online gaming. In this application, there is little benefit to taking advantage of reliability or in-order delivery, as the data are only valid for very short periods. In addition, such applications are very computationally intensive so offloading communication is very desirable.

In addition, because datagram-iWARP has defined message boundaries via the datagrams, it avoids the requirement of having to mark packets to determine message boundaries. Packet marking which is used to correct the semantic mismatch between message-based iWARP and stream-based TCP, is a high overhead activity and is very expensive to implement in hardware [1]. Therefore, avoiding it in datagram-based UDP is a significant advantage over TCP. SCTP also has defined message boundaries, but 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 as such is still in the process of being tuned for performance on various platforms.

The other shortcoming of connection-based transports is that they limit resource sharing among different connections. Therefore, resource usage such as memory consumption will exponentially increase as the scale of the system increases. Although methods such as shared receive queues (SRQ) slow down the connections' memory usage trend to some extent, such methods are still not as scalable as true datagrams.

A datagram-based iWARP avoids many complexities associated with connections. It avoids the overhead of MPA layer and complicated reliability measures in TCP and SCTP. It also requires a smaller memory footprint than the current connection based standard over TCP or SCTP, since it does not have to keep information regarding connections [22]. When utilizing OS bypass, such memory savings are important, as state information needs to be held in memory on the RNIC, or be accessed on the system through memory accesses. In addition, by offloading such tasks onto a NIC, and using RDMA, we can leverage a common method of significantly reducing the CPU requirements for maintaining high bandwidth data streams while bringing this technology into existing data centers that provide services that take advantage of UDP. This allows for businesses to concentrate their investment directly into networking technologies and capacity instead of the CPU hardware required to support such large bandwidth or low latency requirements, a common benefit of Wide Area Network (WAN) capable RNICS.

The lowered complexity of datagram-iWARP means that if fabricated as a stand-alone solution, it can be fabricated at much lower cost than traditional iWARP. Alternatively, it also provides an inexpensive addition to existing iWARP silicon. With the increase in datagram based Internet traffic [2], it is important to improve the performance, and reduce the server side overhead of such traffic using the existing infrastructure. At the consumer level, Voice over IP (VOIP) applications, on-demand video, rich streaming media content, high quality Internet radio, and low-latency multi-player online gaming are all applications that can benefit from using an RDMA enabled datagram-based Ethernet. Broadcast and multicast support are also attractive features of using datagrams. In particular, a multicast capable iWARP solution would be useful in providing high bandwidth media while leveraging the other benefits of datagram-iWARP.

B. 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 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 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 2. It should be noted that datagram-iWARP does not require the MPA [4] layer. This is due to the fact that datagram packets are never segmented by intermediate routers, and therefore no middle-box fragmentation problem exists with datagrams, as it does with stream-based protocols. 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. Such changes are:

1. In the DDP standard [25], 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 [25], Section 5 item 8, states that if an error happens on an 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 [24] 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.

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.

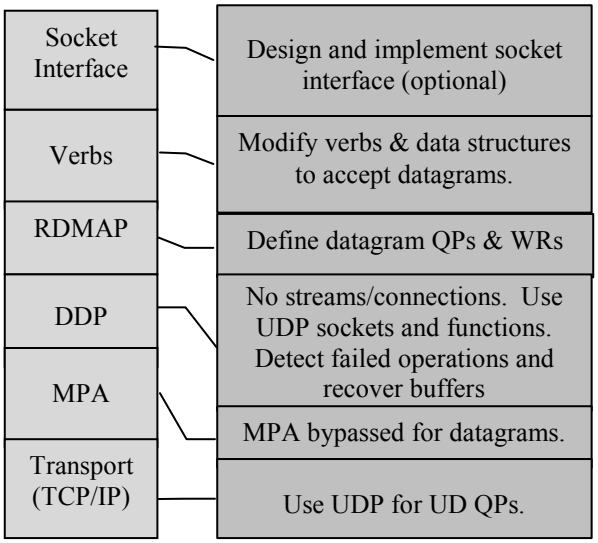

Figure 2. Changes for datagram-iWARP

5. Datagram-iWARP does not require packet marking for either UD or RD modes, therefore the MPA layer [4] 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, 64KB. Although for UD RDMA operations it is possible to perform reassembly of larger messages in the iWARP stack.

1) Send/Recv over Datagrams - The design for send/recv operations is discussed in detail in [22]. It is capable of using many of the existing designs in the iWARP stack. The iWARP stack needs to be altered such that it can handle a connection-less based traffic flow, and that it can report the source of incoming traffic back to the calling application. To support this several new verbs have been added, such as datagram versions of the send and receive posting request verbs, although such verbs could be integrated with the existing verbs. Our implementation was also adapted to allow for opening iWARP sockets that use datagrams. These changes are in line with those detailed in Section IV.B.

Datagram-iWARP matches incoming packets at the DDP layer with the appropriate receive WR. It does not guarantee that the order in which the incoming packets are completed is the order in which the send side actually transmitted the data. As such, the applications must be able to handle out of order data, or a reliable UDP implementation must be used that is capable of ensuring in order delivery. In order to prevent polling on operations that will never complete (in the event that incoming data are lost and no more incoming data are expected) it is essential that the completion queue be polled with a defined timeout period. This allows for the failure to receive a given packet.

The send/recv verb semantics match those used for sockets very well. As such, send/recv functionality is integrated into our iWARP socket interface discussed in Section V.A.

2) RDMA Write over Datagrams - There is currently no method proposed that allows for RDMA operations over unreliable datagrams. 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 generate 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 [24]. 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.

3) RDMA Write-Record: We propose a new method called RDMA Write-Record 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 3.

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). 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.

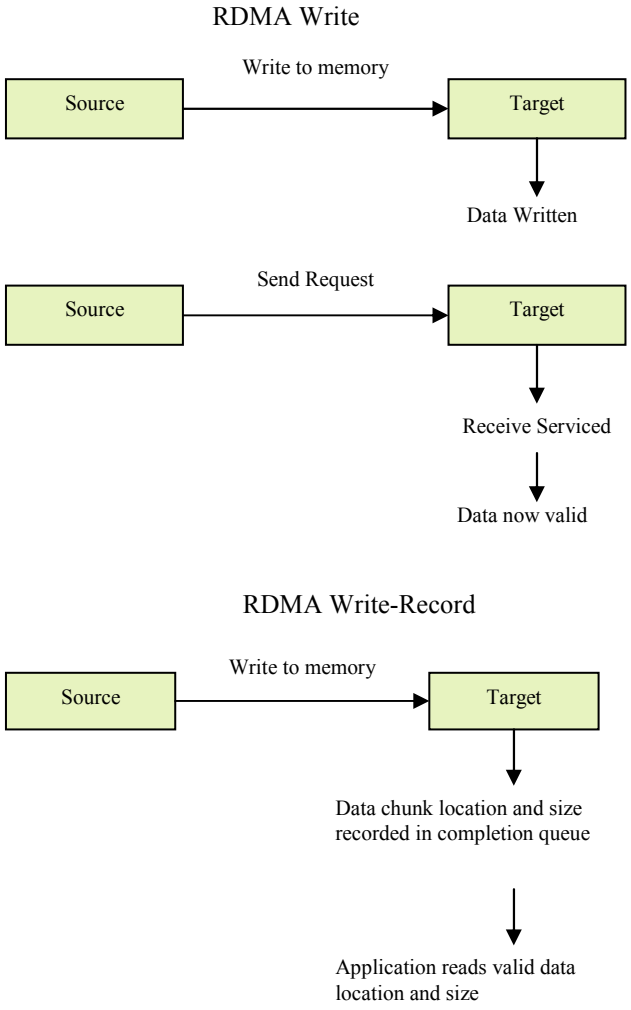

Figure 3. Comparison of RDMA Write over RC and RDMA Write-Record over UD

4) Packet Loss Design Considerations - 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 VI.A.

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 KB. 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.

V. Software Implementation

The software implementation of datagram-iWARP was developed using a TCP-based iWARP implementation from [5]. A socket interface was added, which is described in more detail in Section V.A.

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 4. The changes required to the verbs, RDMAP, and DDP layers as described in Section IV.B 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.

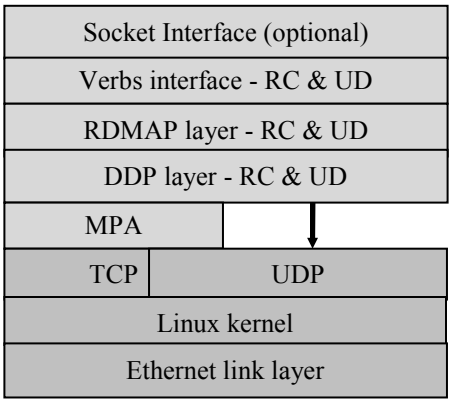

Figure 4. The software implementation of datagram-iWARP

In addition, it 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.

A. 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 [20]. 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 vs. RC results, support for both UD and RC operations has been included in our socket interface implementation.

1) Design - Our socket interface works by dynamically preloading it 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 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.

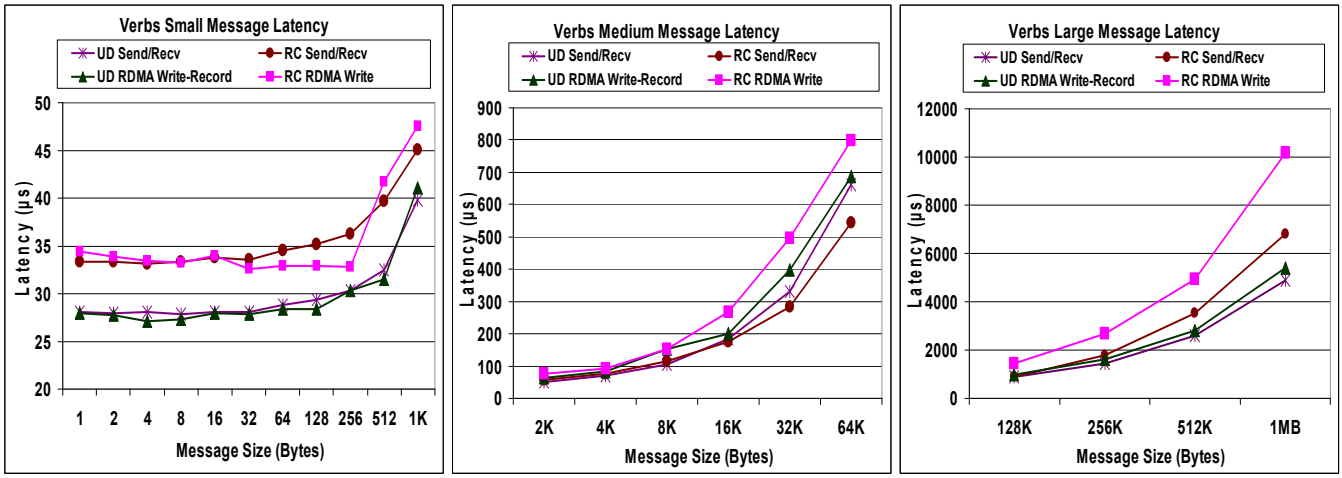

Figure 5. UD iWARP vs. RC iWARP Verbs latency

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.

VI. 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 microbenchmarks for all of the discussed modes of operation, and investigates the performance of two applications, a media streaming application VLC [28] and a SIP [8] application/benchmark SIPp.

The experimental results were run on two nodes, each with two quad-core 2GHz AMD Opteron processors with a 512KB L2 cache per core and 8MB shared L3 cache per processor chip, 8GB RAM and a NetEffect 10-Gigabit Ethernet (10GE) card connected through a Fujitsu 10-Gigabit Ethernet switch. The OS used was Fedora Core 12 (kernel 2.6.31).

A. Microbenchmark Performance Results

1) Latency and Bandwidth - The results of the verbs latency of datagram iWARP are shown in Figure 5, for send/recv, RC RDMA write and UD RDMA Write-Record. 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 and RC RDMA Write, which has latency around 33µs for messages under 128 bytes long. For message sizes up to 2KB, UD send/recv offers a 18.1% improvement in latency over RC send/recv iWARP, while UD RDMA Write-Record offers a 24.4% improvement over RC RDMA write. For messages between 16KB to 64KB we can observe that the performance of RC send/recv is slightly better than that of UD RDMA Write-Record and UD send/recv, but the UD based iWARP solution (send/recv and Write-Record) have better latencies for larger message sizes.

Figure 6 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 1KB and 1.5KB is of great importance as such message sizes are the most likely to be used in the delivery of media. For 1KB 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.

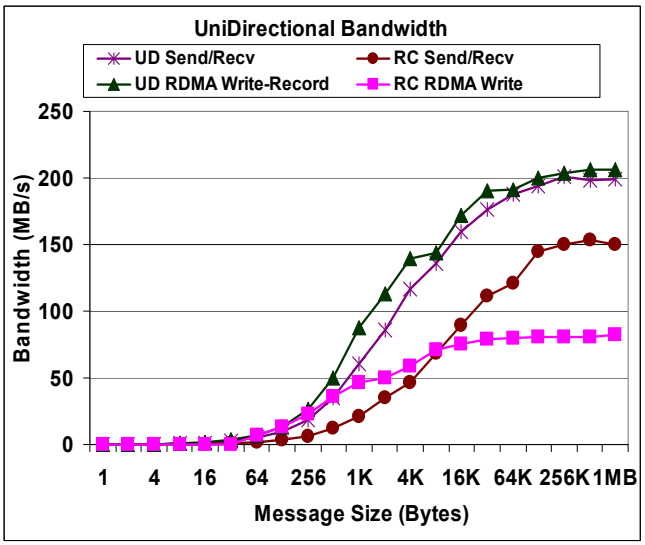

Figure 6. Unidirectional Verbs Bandwidth

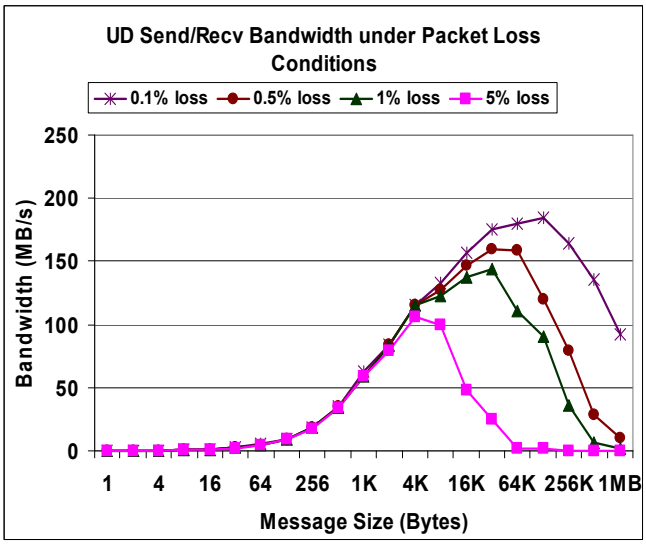

Figure 7. UD Send/Recv Bandwidth Under Packet Loss Conditions

For messages larger than 1.5KB, 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 128KB) in Figure 6, 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 256KB messages. Most importantly, RDMA Write-Record has a significant 256% advantage at message sizes of 512KB over RC RDMA Write.

2) Packet Loss and Performance - In order to study the proposed system under packet loss conditions, the Linux traffic control provisions. 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 7, 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 [26]. 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 improves performance over the required full message delivery used in our send/recv method.

Figure 8 illustrates that the partial RDMA Write-Record method performs better in low packet loss conditions even for larger message sizes. We observe a drop at 64KB messages as these messages exceed the maximum sized MTU of the UDP layer, requiring multiple UDP messages. For messages under 64KB and greater than the network MTU of 1500 bytes, multiple packets are segmented at the sender 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 64KB) message being dropped. For messages larger than 64KB, 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 64KB UDP segments some of the overall message can be saved even if some 64KB 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.

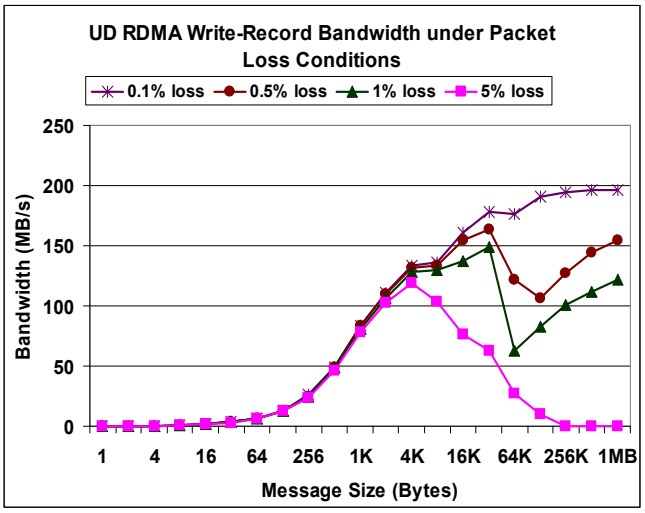

Figure 8. UD RDMA Write-Record Bandwidth Under Packet Loss Conditions

B. *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 [28] a popular media streaming application as well as SIPp [8], a testing framework for load testing SIP servers.

1) VideoLan's VLC media player - VLC [28] 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 9, 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. However, it should be noted that there is more inherent overhead involved in the HTTP based 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.

2) SIPp Server/Client - 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 10. 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 [8] 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 11 details the memory savings that are possible when using datagram-iWARP on a SIP server compared to RC in such a scenario. 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) predict that such a case would result in a 28.1% memory improvement over RC. The resulting 4% difference can be directly attributed to the application's memory usage, which would require some additional book keeping 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.

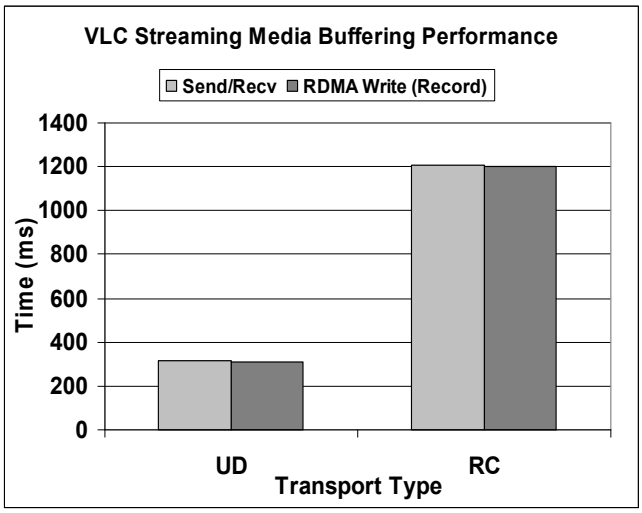

Figure 9. VLC UD Streaming vs. RC-based HTTP Streaming

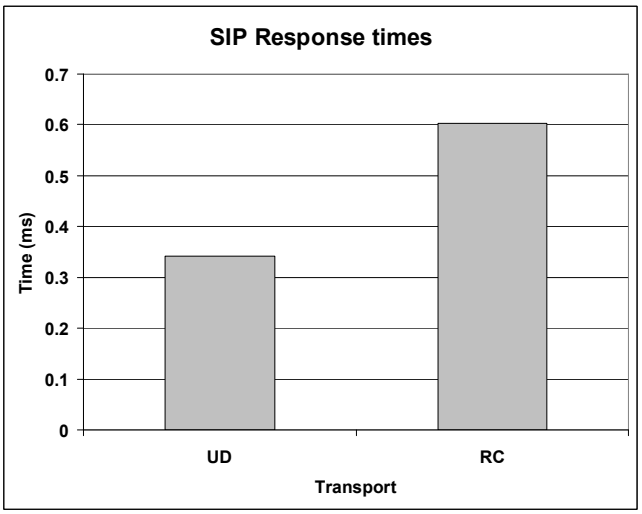

Figure 10. SIP Response Times

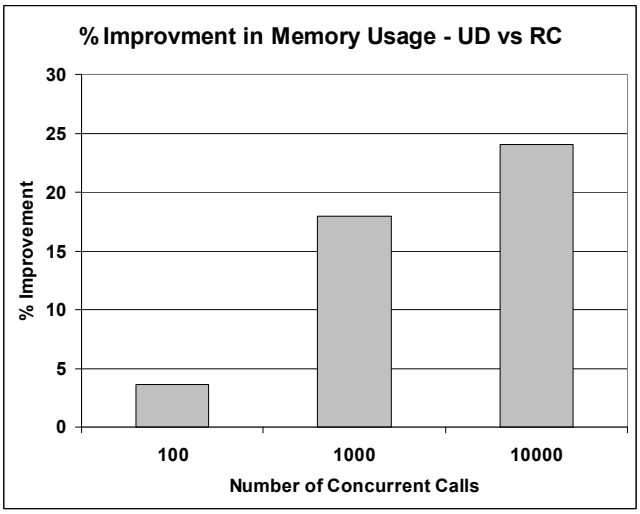

Figure 11. SIP Improvement in memory usage using datagram-iWARP over traditional iWARPConclusions and Future Work

Datagram-iWARP has been shown to be effective in both HPC [22] and datacenter 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, the inclusion of datagram support for modern high performance interconnects is an important step in broadening the application space for RDMA-enabled network technologies. This paper 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. As unreliable datagrams become an ever-growing majority of Internet traffic this proposal provides the opportunity to expand RDMA technologies to a majority of applications and traffic, particularly high bandwidth applications that are likely to use datagrams.

In this paper we have provided a design for a full featured datagram-iWARP solution. We have 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.

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.

In the future we would like to examine the performance and memory scalability of our proposal on larger systems and with different applications. We would like to develop a protocol similar in concept to SDP for datagrams based on our observations in this paper. It would also be useful to expand the proposed concepts by utilizing a reliable datagram transport, to provide datagram support for a larger range of possible applications. We would also like to extend this work by creating an interface to allow MPI to take advantage of the new RDMA Write-Record over datagram-iWARP and also propose UD-based RDMA Read for use in HPC applications.

ACKNOWLEDGMENT

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

REFERENCES

[1] P. Balaji, W. Feng, S. Bhagvat, D. K. Panda, R. Thakur, W. Gropp, "Analyzing the Impact of Supporting Out-of-Order Communication on In-order Performance with iWARP" Proceedings of the ACM International Supercomputing Conference (SC07), Reno, Nevada, November 2007.

[2] Cisco Systems Inc. (June 2, 2010). Hyperconnectivity and the Approaching Zettabyte Era, [http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/VNI_Hyperconnectivity_WP.pdf].

[3] 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", 17th IEEE symposium on High Performance Interconnects, New York, NY, August 2009.

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

[5] D. Dalessandro, A. Devulapalli, P. Wyckoff, "Design and Implementation of the iWARP Protocol in Software" Parallel and Distributed Computing and Systems (PDCS'05), Phoenix, AZ, November 2005.

[6] D. Dalessandro, A. Devulapalli, P. Wyckoff, "iWARP Protocol Kernel Space Software Implementation", 20th IEEE International Parallel & Distributed Processing Symposium (IPDPS'06), Rhodes, Greece, 2006.

[7] W. Feng, P. Balaji, L. N. Bhuyan, D. K. Panda, "Performance Characterization of a 10-Gigabit Ethernet TOE", 13th International Symposium on High Performance Interconnects, Stanford, CA, August 2005.

[8] R. Gayraud et al., "SIPp Traffic Generator", Nov. 2009; http://sipp.sourceforge.net/.

[9] Brice Goglin, "Design and Implementation of Open-MX: High-Performance Message Passing over Generic Ethernet Hardware", 22nd IEEE International Parallel & Distributed Processing Symposium (IPDPS'08), Miami, FL, April 2008.

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

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

[12] InfiniBand Trade Association, "InfiniBand Architecture Specification" Vol. 1, Release 1.2.1, November 2007.

[13] Internet Engineering Task Force: http://www.ietf.org.

[14] 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.

[15] MPI Forum, 'MPI: A Message Passing Interface Standard," v2.2, September 2009.

[16] Myricom homepage: http://www.myri.com.

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

[18] Ohio Supercomputer Center, "Software Implementation and Testing of iWARP Protocol": http://www.osc.edu/research/network_file/projects/iwarp/iwarp_main.shtml.

[19] OpenFabrics Alliance: http://www.openfabrics.org/.

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

[21] M. J. Rashti, A. Afsahi, "10-Gigabit iWARP Ethernet: Comparative performance analysis with InfiniBand and Myrinet-10G", 21st IEEE International Parallel and Distributed Processing Symposium (IPDPS'07), Long Beach, CA, 2007.

[22] M. J. Rashti, R. E. Grant, P. Balaji, A. Afsahi, "iWARP Redefined: Scalable Connectionless Communication over High-Speed Ethernet", High Performance Computing Conference (HiPC'10), Goa, India, December 2010.

[23] RDMA Consortium: http://www.rdmaconsortium.org.

[24] R. Recio, P. Culley, D. Garcia, J. Hilland, "An RDMA Protocol Specification (version 1.0)", RDMA Consortium, October 2002.

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

[26] SLAC National Accelerator Laboratory, "PingER", Aug. 2010; http://www-iepm.slac.stanford.edu/pinger/.

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

[28] VideoLan Project, "VLC Media Player", Aug. 2010; http://www.videolan.org/vlc/.