iWARP Redefined: Scalable Connectionless Communication over High-Speed Ethernet

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

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

***Abstract*—iWARP represents the leading edge of high performance Ethernet technologies. By utilizing an asynchronous communication model, iWARP brings the advantages of OS bypass and RDMA technology to Ethernet. The current specification of iWARP is only defined over connection-oriented transports such as TCP. The memory requirements of many connections along with TCP's flow and reliability controls lead to scalability and performance issues for large-scale HPC and datacenter applications. In this research, we propose guidelines to extend iWARP over datagrams to provide better scalability and performance. While the proposed extension is designed for use in both HPC and datacenters, the emphasis of this paper is on HPC applications. We present our software implementation of datagram-iWARP over UDP and MPI over datagram-iWARP. Our microbenchmark and MPI application results show performance and memory usage benefits for MPI applications, promoting the use of datagram-iWARP for large-scale HPC applications.**

Keywords- iWARP; Ethernet; datagram; Message Passing Interface

I. INTRODUCTION

Despite recognized performance inefficiencies, Ethernet currently accounts for more than half of the interconnection networks in the top 500 supercomputers [30]. It is due to its easy deployment and low cost of ownership that Ethernet is ubiquitously used in commercial and research clusters, serving High Performance Computing (HPC) and datacenter systems.

The large overhead that Gigabit and 10-Gigabit Ethernet network protocol processing puts on the CPU cores has led to critical CPU availability and performance issues [8]. For this, a wide range of efforts started to boost Ethernet efficiency, especially targeting its latency for HPC. The first major attempt was offloading TCP/IP processing using stateless offload (e.g. offloading checksum, segmentation and reassembly, etc.) and stateful TCP Offload Engines (TOE) [8].

Another major approach on top of TOE has been equipping Ethernet with techniques such as *Remote Direct Memory Access* (RDMA) and zero-copy communication that have traditionally been associated with other high performance interconnects such as InfiniBand [12]. iWARP (Internet Wide Area RDMA Protocol) [25] was the first standardized protocol to integrate such features into Ethernet, effectively reducing Ethernet latency and increasing host CPU availability by taking advantage of RDMA, kernel bypass capabilities, zero copy and non-interrupt based asynchronous communication [1][24]. Rather than the traditional kernel level socket API, iWARP provides a user-level interface on top of TCP/IP stack that can be used in both LAN and WAN environments, thus, efficiently bypassing kernel overheads such as data copies, synchronization and context switching.

Despite its contributions to improving Ethernet efficiency, the current specification of iWARP lacks functionality to support the whole spectrum of Ethernet based applications. The current iWARP standard is only defined on reliable connection-oriented transports. Such a protocol suffers from scalability issues in large-scale applications due to memory requirements associated with multiple inter-process connections. In addition, some applications and data services do not require the reliability overhead and implementation complexity and cost associated with connection-oriented transports such as TCP. For example, HPC applications running on a system area network do not require the complexities associated with TCP. A connectionless protocol is a lighter weight protocol that can improve the communication performance as well.

In this paper, we propose to extend the iWARP standard on top of the User Datagram Protocol (UDP) in order to utilize the inherent scalability, low implementation cost and the minimal overhead of datagram protocols. We provide guidelines and discuss the required extensions to different layers of the current iWARP standard in order to support the connectionless UDP transport. Our proposal is designed to co-exist with and to be consistent and compatible with the current connection-oriented iWARP. While the proposed extension is designed to be used in datacenter and HPC clusters, the emphasis of this paper is on HPC applications.

Our implementation of datagram-iWARP in software reveals performance benefits that can be potentially achieved

when using datagrams in iWARP-based Ethernet clusters. Our verbs level microbenchmark results show that the datagram-iWARP improves the communication latency up to 30%. Our MPI level results also show up to 14% small message latency reduction and up to 20% large message bandwidth improvement when using Message Passing Interface (MPI) [17] on top of datagram-iWARP. We also observe that MPI applications can substantially benefit in performance and memory resource usage when running on datagram-iWARP, compared to the connection-based iWARP; more than 30% less memory usage and more than 40% runtime reduction for some HPC applications on a 64-core cluster.

The rest of this paper is organized as follows. In Section II, we provide background about the current iWARP standard and discuss some of its shortcomings for HPC and datacenter applications. In Section III, we propose guidelines for changes to the iWARP for datagram support. Section IV describes our implementation of iWARP over UDP. Section V and Section VI include the experimental platform and evaluation results respectively. Section VII discusses some related scholarly work and finally, Section VIII concludes the paper and points to future directions.

II. iWARP STANDARD

Proposed by RDMA Consortium [25] in 2002 to the IETF [13], the iWARP specification defines a multi-level processing stack on top of standard TCP/IP over Ethernet. The stack is designed to decouple the processing of *Upper Layer Protocol* (ULP) data from the operating system (OS) and reduce the host CPU utilization by avoiding intermediate copies during data transfer (zero copy). To achieve these goals, iWARP needs to be fully offloaded, for example on top of stateless or stateful TOE.

As illustrated in Fig. 1, at the top layer, iWARP provides a set of descriptive user-level interfaces called iWARP verbs [10]. The verbs interface bypasses the OS kernel and is defined on top of an RDMA enabled stack. A network interface card (NIC) that supports the RDMA stack as described in iWARP standard is called an *RDMA-enabled NIC* or RNIC. An RNIC implements both iWARP stack and TOE functionality in hardware.

The *RDMA protocol* (RDMAP) layer supplies communication primitives for verbs layer [26]. The data transfer primitives are Send, Receive, RDMA Write and RDMA Read that are passed as work requests (WR) to a Queue Pair (QP) data structure. The WRs are processed asynchronously by the RNIC, and the completion is notified either by polled Completion Queue (CQ) entries or by event notification [10].

Verbs layer WRs are delivered in order from RDMAP to the lower layers. The Send and RDMA Write operations require a single message for data transfer, while the RDMA Read needs a request by the consumer (data sink), followed by a response from the supplier (data source) [26]. RDMAP is designed as a stream-based layer. Operations in the same RDMAP stream are processed in the order of their submission.

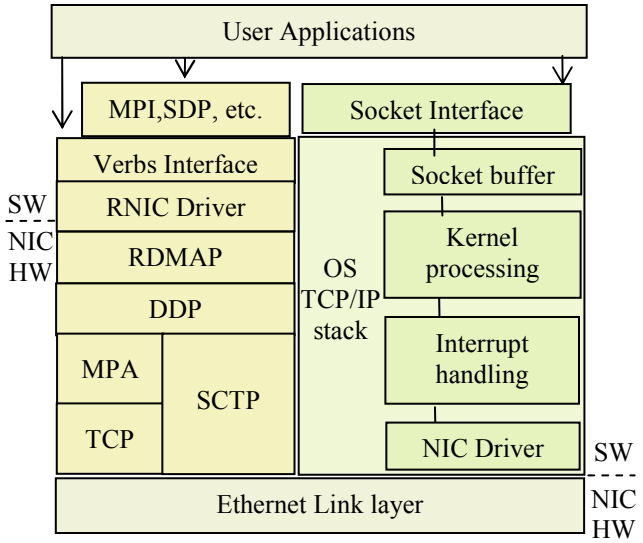

Figure 1. iWARP standard stack compared to host-based TCP/IP

The *Direct Data Placement* (DDP) layer is designed to directly transfer data between the user buffer and the RNIC without intermediate buffering [27]. The packet based DDP layer matches the data sink at the RDMAP layer with the incoming data segments based on two types of data placements models: tagged and untagged. The tagged model, used for one-sided RDMA Write and Read operations, has a sender-based buffer management in which the initiator provides a pre-advertised reference to the data buffer address at the remote side. The untagged model uses a two-sided Send/Receive semantic, where the receiver both handles buffer management and specifies the receive buffer address [27].

Due to DDP being a message-based protocol, out-of-order placement of message segments is possible, therefore DDP assures delivery of a complete message upon arrival of all segments. In the current iWARP specification, DDP assumes that the lower layer provides in order and correct delivery of messages.

The lower layer protocol (LLP) on which the iWARP stack is running can be either TCP or SCTP [21]. Due to the message-oriented nature of DDP, the iWARP protocol requires an adaptation layer to put boundaries on DDP messages transferred over the stream oriented TCP protocol. The *Marker PDU Alignment* (MPA) protocol [5] inserts markers into DDP data units prior to passing them to the TCP layer. It also re-assembles marked data units from the TCP stream and removes the markers before passing them to the DDP. The MPA layer is not needed on top of message-oriented transports such as SCTP because intermediate devices do not fragment message-based packets as they would with stream-based ones, removing the middle-box fragmentation issue that the MPA layer solves.

A. Shortcomings of the Current Standard

The current iWARP standard offers a range of capabilities that increase the efficiency of Ethernet in modern

HPC and datacenters clusters. Taking advantage of the well-known reliable transports in the TCP/IP protocol suite is one of its key advantages. Reliability has in fact been a major force for designing the current iWARP standard on top of connection-oriented transports. The LLP for DDP and MPA is assumed to be a point-to-point reliable stream, established prior to iWARP communication. This requirement makes it easy for the ULP to assume reliable communication of user data. In addition, the independence of individual streams makes iWARP able to enforce error management on a per stream basis.

Such a standard is a fit for applications that require strict reliability at the lower layer, including data validation, flow control and in order delivery. Examples for such applications are reliable datacenter services such as database servers, file services, financial applications and policy enforcement systems (e.g. security applications, etc.).

On the other hand, there is a growing demand for applications such as voice and video streaming that find the strict connection-based semantics of iWARP unnecessary. For such cases, the current iWARP standard imposes barriers to application scalability in large systems, due to its explicit connection oriented nature and reliability measures. The following subsections point to the shortcomings of the current standard and their relevant implications. As such, there are strong motivations for extending the iWARP standard with datagram transport.

1) Memory usage. The pervasiveness of the Ethernet in modern clusters places a huge demand on the scalability of the iWARP standard. The scale of high performance clusters is increasing rapidly and can soon reach to a million cores. A similar trend can be observed for datacenters. An obvious drawback of the connection-oriented iWARP is the connection memory usage that can exponentially grow with the number of processes. This dramatically increases the application's memory footprint, unveiling serious scalability issues for large-scale applications.

As the number of required connections increases, memory usage grows proportionally at different network stack layers. In a software implementation of iWARP at the TCP/IP layer, each connection will require a set of socket buffers allocated, in addition to the data structure required to maintain the connection state information. Although the socket buffers are not required in a hardware implementation of iWARP due to zero-copy on the fly processing of data, making a lot of connections will have other adverse effects. Due to limited RNIC cache for connection state information, maintaining out-of-cache connections will require extra memory requests by the RNIC, which implies extra overhead on communciation time.

The other major place of memory usage is the application layer. Specifically, the communication libraries such as MPI pre-allocate memory buffers per connection to be used for fast buffering and communication management [14].

2) Performance. In addition to memory usage problem, connection oriented protocols such as TCP, with their inherent flow-control and congestion management limit performance [11]. HPC applications running on a local cluster do not require the complexities of TCP flow and congestion management. UDP offers a much lighter weight protocol that can significantly reduce the latency of individual messages, closing the latency gap between iWARP and other high speed interconnects. In addition, many datacenter applications such as those using media streaming protocols over WAN are currently running on top of unreliable datagram transports such as UDP. Due to such semantic discrepancies, the current connection-oriented specification of iWARP makes it impossible for such applications to take advantage of iWARP's major benefits such as zero copy and kernel bypass.

3) Fabrication cost. The complexities associated with stream based LLPs such as TCP and SCTP translate into expensive and non-scalable hardware implementations. This becomes especially important with modern multi-core systems where multiple processes could utilize the offloaded stack. A heavyweight protocol such as SCTP or even TCP can partially support multiple parallel on-node requests, due to implementation costs associated with hardware level parallelism [24]. This means that a small portion of many cores available on the node will be able to simultaneously utilize the actual hardware on the NIC. This can lead to serialization of the communication.

4) Hardware level operations. iWARP lacks useful operations such as hardware level multicast and broadcast. These operations, if supported, can be utilized in applications with MPI collectives and also media streaming services. iWARP does not support such operations primarily because the underlying TCP protocol is not able to handle multicast and broadcast operations. An extension of iWARP to datagrams will boost iWARP's position as a leading solution for high performance Ethernet. Next section presents our proposal for such an extension.

III. DATAGRAM-IWARP

The current iWARP standard and its main layers (RDMAP and DDP) are explicitly designed for connection-oriented LLPs. Therefore, there are semantic discrepancies with datagram protocols such as UDP that need to be addressed in our design. There are implications in the standard that make the definition of unreliable and datagram services viable in the current iWARP framework (for example Sections 3.2 and 8.2 of DDP specification [27]).

In this proposal, we try to keep the current well-developed specification of iWARP, while extending its functionality to support datagram traffic. In the first step, we highlight parts of the standard at different layers that are incompatible with datagram semantics. Then we propose guidelines to address such incompatibilities. In this paper we cover the untagged model of the DDP layer and the tagged

model will be covered in our future research. It is important to note that these proposals should not be considered as exact modifications of the standard. We rather point to major places of the standard for modification to support datagram transport. Fig. 2 presents major changes required at each layer of the current standard. Categorized details can be found in the subsequent sections.

A. Modifications to the Verbs Layer

We do not necessarily require introducing new verbs for the datagram mode. The existing set of iWARP verbs can be adapted to accept datagram related input and act according to the datagram service. Here we point to some major parts of the verbs specification that need to be changed to support datagram transport:

- Currently, there is only one type of QP, the connection-based QP. Thus, there has been no need for QP type definition. With the new extension, new QP type(s) must be added to distinguish datagram-iWARP from connection-based iWARP. More details will be discussed in part B of Section III.
- *QP creation* and its input modifiers need to be changed. For example, to specify the transport type (connected or datagram) a new input modifier should be added to the QP attribute structure.
- Specification of the *QP modify* verb needs to change, to accommodate the new definition of the QP states and the required input data for datagram QPs. As an example, the datagram QPs need a pre-established datagram socket to be passed to modify the QP into the *Ready To Send* (RTS) state [10].
- An *address handle* is required for each send WR posted to the datagram QP to specify the receiver's IP address and UDP port related to the remote QP.
- *Completion notifications structure* needs to be changed to accommodate the new WR structure. In particular, the work completion structure should be changed to include the source address.

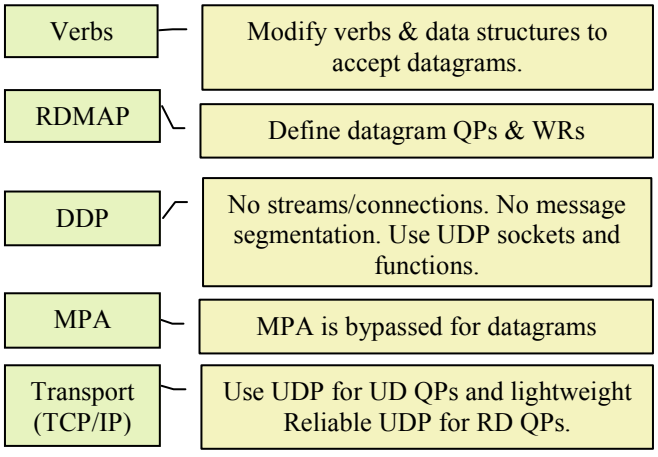

Figure 2. Extensions to the stack for datagram-iWARP

B. Reliable, In-order Delivery

Reliable service is a fundamental assumption in the current iWARP standard. This assumption is not necessarily in opposition to the use of datagrams. In datagram-iWARP design we introduce two types of datagram services, *Unreliable Datagram* (UD) and *Reliable Datagram* (RD). Subsequently QP types need to be defined at the verbs and RDMAP layers. The defined QP types are: unreliable datagram and reliable datagram for UDP and *Reliable Connection* (RC) for the current TCP-based QPs.

The datagram-iWARP over UD transport assumes no reliability or order of delivery from the LLP. In the untagged model which is the focus of this paper, the incoming messages will be matched to the posted receive WRs at the data sink, in order of their arrival at the DDP layer which is not necessarily their order of issue at the data source. While keeping the iWARP data integrity checksum mechanism (e.g. CRC), the rest of reliability measures are left to the application protocol. Such a service is very useful for applications with high error resiliency (such as media streaming applications in datacenters) and applications that can efficiently provide their own required level of reliability. An example can be the applications running in low error rate environments such as closed Local or System Area Networks where standard reliability measures impose too much performance overhead.

For the RD service, the LLP is assumed to guarantee that messages from a specific data source are delivered correctly and in-order. Such a definition implies a logical pseudo-connection between the local QP and the remote QP. However, DDP/RDMAP layers are not required to keep state information for such a logical connection. Similar to UD, DDP and RDMAP for RD service are required to pass the messages in the order they have received them. To keep the scalability advantages of using a connectionless transport, the LLP reliability service is assumed to be lightweight and should require no or minimal buffering for individual remote sockets. The way the LLP (here, a reliable form of UDP) provides reliability and the mechanism by which such a service is configured is outside of the scope of the extended iWARP specification.

C. Streams and Connections

Currently, the RDMAP and DDP layer streams are assigned to underlying LLP connections that are assumed to be pre-established. Since connections are conceptually not supported in a datagram (connectionless) model, no connection establishment and teardown is required. For datagram-iWARP, a previously created UDP socket is required for each QP. In this case, the ULP transitions the QP into iWARP mode after creating the QP and assigning the lower layer socket. This operation is done locally without negotiating any parameters (such as CRC settings) with any other peer. Such parameters need to be pre-negotiated by the ULP. This also implies that the ULP is no longer required to configure both sides for iWARP at the same time.

Transport error management and connection teardown/termination requirements in the current standard will be the responsibility of the datagram LLP, if a reliable

service is being used (i.e. RD mode). Error management at the higher layers (e.g. DDP or RDMAP) needs to be modified to suit the datagram service. For example, the current standard requires an abortive termination of a stream at all layers and an abortive error must be surfaced to the ULP, should an error be detected at the RDMAP layer for that stream [26]. Since such a requirement does not apply to datagrams, an abortive error must be surfaced to the ULP and the QP must simply go into the Error state without requiring any stream termination. This makes the QP unable to communicate with any other pair, until the QP is reset and modified to the RTS state by the application. Instead of the stream termination message sent to the other side, a simple error message should be transferred, identifying the erroneous message number using the *Message Sequence Number* (MSN in the DDP header). The error message can be placed into an *Error* queue that replaces the *Terminate* queue of the connected mode.

D. QP Identification

For the untagged model, the DDP layer provides a queue number (QN) in the DDP header to identify the destination QP [27], which is currently not fully utilized by the RDMAP. The RDMAP only uses its first 2 bits [26]. In the datagram-iWARP, we currently assume assignment of a single datagram QP to a UDP socket. In such a model, no QN is required to identify the source and destination QPs. An optional model of the datagram service can assign multiple datagram QPs to a single socket, similar to multiple streams per LLP connection in the current iWARP. Such a case benefits from the QN field in the DDP header.

E. Message Segmentation

Unlike TCP byte-oriented service, UDP datagrams will arrive at the LLP in their entirety and thus the concept of message segmentation and out-of-order placement at the DDP layer is irrelevant to the datagram service. This implies that the DDP layer does not need its provisions for segmented message arrival over the datagram transport (including message offset (MO) and even MSN for some cases). For messages larger than maximum datagram size (64KB), segmentation and reassembly is done at the application layer.

F. Completion of WRs

In the connected mode, a WR is considered complete when the LLP layer can guarantee its reliable delivery. The same semantics can be used for RD transport. However, for the UD transport we no longer require an LLP guarantee. Thus, a WR should be considered complete as soon as it is accepted by the LLP for delivery.

G. MPA Layer

Since each DDP message will be encapsulated into one UDP datagram, no markers are required for iWARP over UDP. Therefore, the MPA layer (specifically the marker functionality) is not needed for the datagram service. This will improve the performance of the datagram transport since MPA processing has shown to impose significant overhead on the performance of iWARP due to marker placement complexities [1], in addition to increasing the overall size of the required data transmission.

IV. Software Implementation

To evaluate the proposed datagram extension to the iWARP standard we have developed a software implementation of datagram-iWARP. Fig. 3 shows the layered stack of this implementation which is built on top of an available software-based iWARP code from Ohio Supercomputer Center (OSC) [22]. Our implementation can be used on top of both reliable and unreliable UDP protocols.

Our evaluation in this paper is on top of unreliable (regular) UDP. To assess our implementation in a standard way, we have completed an OpenFabrics (OF) verbs interface [23] on top of the native software iWARP verbs. We have also used the OF verbs interface to adapt an existing MPI implementation [14] on top of our iWARP stack. The next subsections discuss some features of our implementation at both iWARP and MPI levels.

A. Software Datagram-iWARP

As mentioned above, we have used the OSC software iWARP implementation as our code base and extended that code in the datagram domain. Here we list a number of features for our implementation:

- Complete implementation and integration of iWARP over UDP into the TCP-based iWARP stack from OSC. This has been done by introducing new native verbs to support datagram semantics.
- Using CRC error checking at the lower DDP layer for datagrams.
- Using a round-robin polling method on operating sockets, to ensure a fair service to all QPs in the software RNIC.

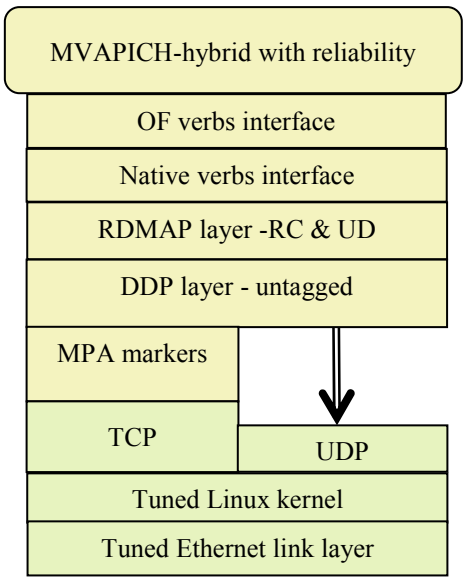

Figure 3. The software implementation of datagram-iWARP

- Using I/O vectors for UDP communication (similar to TCP) to avoid extra sender and receiver side copies, for improved performance and CPU availability. In I/O vector calls (sendmsg and recvmsg), message data and header can be gathered/scattered from/to non-contiguous data buffers to/from the datagram. Therefore an intermediate copy is not required to make the datagram.
- Avoiding segmentation of DDP messages into MTU-size datagrams. This option, which is possible due to message-oriented nature of UDP, positively contributes to the performance of datagram-iWARP.
- Implementation of standard OF verbs: These verbs were originally designed for InfiniBand (called OpenIB verbs). Currently they are known as OF verbs and are utilized to implement iWARP verbs abstraction as well. We use the native verbs to implement the OF verbs. The OF verbs are utilized at the MPI layer.

B. MPI over Datagram-iWARP

To evaluate the performance and memory usage of a datagram based iWARP for HPC applications, we have adapted MVAPICH [20] on top of OF verbs over software iWARP. We have used the hybrid channel from the MVAPICH-Aptus over InfiniBand [14]. MVAPICH-Aptus is an available MPI implementation that offers a hybrid (UD and RC) channel over OF verbs. The hybrid channel offers a dynamic channel management over InfiniBand's UD and RC transports, meant to offer scalability for MPI applications on large scale InfiniBand clusters. The channel starts with UD-based QPs for each process, and based on a set of policies, establishes RC connections to a selected set of other processes, up to a maximum number of RC QPs. This strategy makes applications scale better by limiting the resource-greedy RC connections and putting most of the communication on UD QPs. Reliability has been added for UD communication at the MPI layer, using acknowledgments and timeouts [14].

The MVAPICH code has been modified in several ways to adapt it over the iWARP standard and our software implementation. Here is a list of some modifications made to the implementation:

- Transforming MVAPICH UD-based connection management to the datagram-iWARP: This includes establishing datagram sockets and relevant address handles to be used as the underlying LLP (UDP) sockets required by datagram-iWARP.
- Transforming MVAPICH InfiniBand RC-based connection management: The MVAPICH hybrid channel uses on-demand RC connection management [14]. Due to semantic differences between TCP and InfiniBand RC, the handshaking steps for MVAPICH dynamic connection establishment have been modified. The new arrangement also piggybacks some new required information and in addition, performs the socket connections at the very last handshake stage.
- Changing or disabling parts of the code relying on incompatibilities between iWARP and InfiniBand. This includes functions unsupported in the iWARP implementation such as immediate data, DDP tagged model, GRH headers, Shared Receive Queues (SRQ), eXtended Reliable Connections (XRC), Service Levels (SL), LIDs and LID mask control.

V. Experimental Platform

We use two different clusters for our experiments. Cluster C1 is a set of four nodes, each with two quad-core 2GHz AMD Opteron processors, 8GB RAM, 512KB L2 cache per core and 8MB shared L3 cache per processor chip. The nodes are interconnected through NetEffect 10GE cards connected to a Fujitsu 10GE switch. The OS on C1 cluster nodes is Fedora 12 (kernel 2.6.31).

Cluster C2 contains 16 nodes, each with two dual-core 2.8GHz Opteron processors, with 1MB L2 cache per core, 4GB RAM and a Myricom 10GE adapter [18] connected to a Fulcrum 10GE switch. The OS on C2 cluster nodes is Ubuntu with kernel version 2.6.27.

The reason for using two clusters for the evaluation of this work is to show how application performance and memory usage scale using datagram-iWARP on two different architectures. In particular, the number of cores per node for C1 and C2 is different. With C2 having half of the C1 core-per-node ratio and twice the number of cores in total, its inter-node communication share will be four times that of the C1. This is expected to yield more application performance and scalability, since the proposed extension only affects MPI inter-node communications.

VI. Experimental Results and Analysis

We assess the performance of our UD-based datagram-iWARP implementation using verbs and MPI level microbenchmarks over cluster C1. We also evaluate the effect of datagram-iWARP on the performance and memory of some MPI applications on both C1 and C2 clusters.

A. Microbenchmark Performance Results

We use microbenchmarks to test the performance of the UDP-based iWARP compared to that of the standard TCP-based iWARP. At the verbs layer, we present latency results for both native verbs and OF verbs on top of them. Fig. 4 shows the verbs layer ping-pong latency results. We clearly observe that in most cases the UD latency is lower than that of the RC, primarily due to the following reasons:

- Due to no reliability measures, communication over UDP offers a lighter and consequently faster network processing path, compared to the TCP-based communication.
- Markers are a significant source of overhead on all message sizes. UDP path is bypassing the MPA layer markers, while TCP-based communication requires markers due to the stream oriented nature of TCP.
- The closed dedicated cluster provides an almost error-free environment where strict reliability measures of the TCP protocol are considered purely overhead compared to the unreliable UDP.

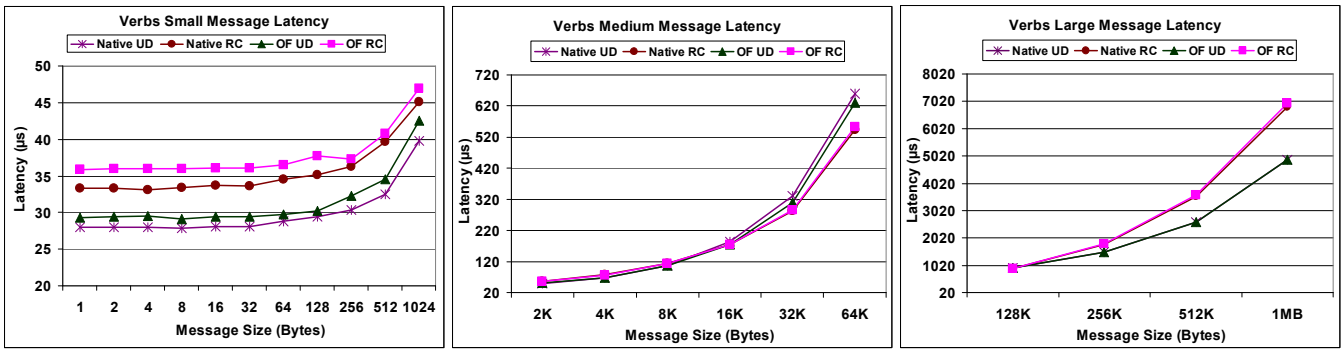

Figure 4. Verbs ping-pong latency

The reason for UD latency being significantly more than RC latency at 64KB is that it is exceeding the maximum datagram size and the benchmark needs to segment the messages into 64KB chunks. The plots in Fig. 4 also show a small overhead for the OF verbs implementation on top of the native verbs.

At the MPI layer, we present microbenchmark results for latency and bandwidth. For all datagram tests we use the MPI level reliability provisions that exist in MVAPICH code for the UD transport [14]. These provisions include sequence numbers, acknowledgements sent for every 50 messages, and timeouts (a fraction of a second) in case acknowledgements are not received or out-of-sequence packets are received. Such provisions are satisfactory for the relatively error-free local area networks that are used for these tests, while not adding unnecessary overhead.

Fig. 5 includes the ping-pong latency comparison of MPI over datagram and connection based iWARP for different message sizes. Results show the superiority of the datagram-mode MPI performance over the connection mode, which is mainly carried from the verbs performance benefits.

Fig. 6 shows the bidirectional bandwidth results at the MPI level. For this test, we use two pairs of processes on two nodes, communicating in the opposite directions. In each pair, one of the processes posts a window of non-blocking receive calls. The other process in the pair posts a window of non-blocking send calls. Synchronization then occurs at the end. As observed, MPI-UD offers a higher bidirectional bandwidth for most of the message sizes, meaning that we can better saturate the network using datagrams. The improvement is about 20% for large messages. Lighter protocol processing and minimal reliability measures are the advantages of UD-based communication that make the benchmark capable of pushing more data on the wire in each direction.

B. *MPI Application Results*

1) Application Performance. Fig. 7 presents MPI application performance results, including total communication time and application runtime. For measuring communication time we aggregate the time spent in communication primitives: MPI blocking and non-blocking send and receive and MPI wait calls.

The results are reported for class B of CG, MG and LU benchmarks from NAS Parallel Benchmark (NPB) suit version 2.4 [19], as well as Radix [28] and SMG2000 [3] applications. All results are presented for 4, 8, 16, 32 and 64 processes (64-process results are only on C2 cluster).

The results for both communication time and application runtime clearly show that we can expect considerable performance benefits when using datagram communication. In addition to reasons for superiority of datagram-based communication discussed above, the lowered complexity of UDP should theoretically create the availability of more CPU cycles for applications' computation phases, which would lead to lower overall runtimes.

2) Application Memory Usage. One of the strongest motivations to extend iWARP standard to datagram domain is to improve its memory usage in order to make it scalable for large scale parallel jobs.

Socket buffers are the most contributors to the memory usage at the OS level. However, in many operating systems including Linux, the *Slab* allocation system [2] is used in which a pool of buffers are pre-allocated and assigned to the sockets when data is being communicated. This mechanism that is primarily used to alleviate the memory fragmentation effects hides the contribution of socket buffer sizes to the overall application memory usage. Therefore, the socket buffer allocation is not reflected in the total memory of the system, unless the pre-allocated slab buffers are filled and new buffers are reallocated due to high instantaneous network usage.

At the MPI layer, MVAPICH pre-allocates a number of general buffer pools with different sizes for each process. For the datagram QP that is established in both connection and datagram based modes, a number of buffers are picked from these pools and pre-posted as receive buffers to the QP. Once a new connection-based QP is established, a default number of 95 receive buffers are picked from the pools and posted to the QP. With a default size of 8KB for each buffer, a rough estimate of 800KB or 200 memory pages of 4KB size are required per connection for each process.

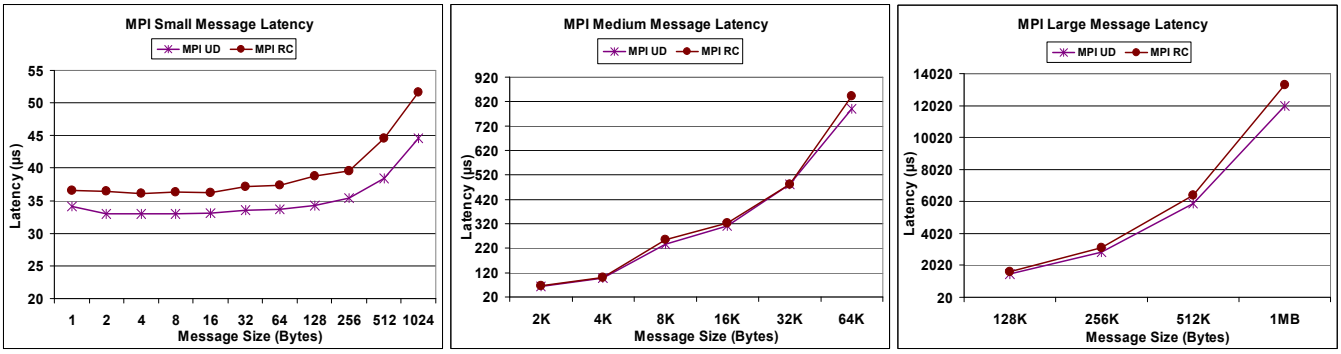

Figure 5. MPI ping-pong latency

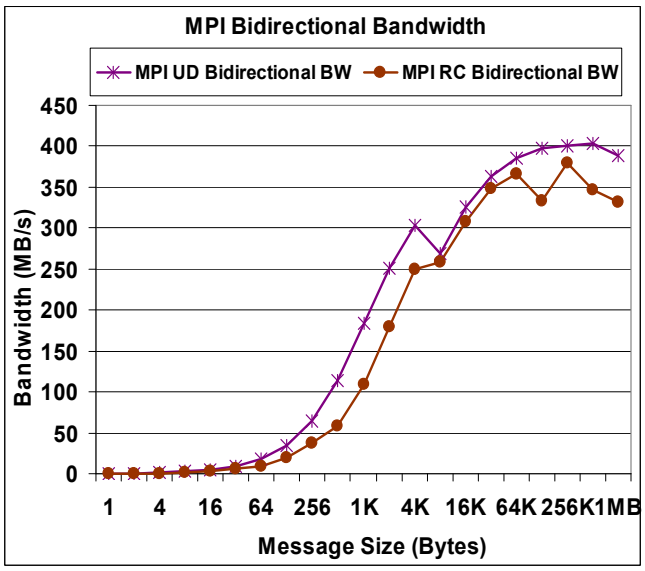

Figure 6. MPI bidirectional bandwidth

To measure the memory usage for the software-based iWARP, we use the total number of memory pages allocated to each MPI job in Linux, reported by the Linux *proc* files system. Fig. 8 shows the improvement percentage for datagram-iWARP over connection-oriented iWARP. As observed, the overall memory usage of MPI applications can benefit from using datagrams. For most cases, the results also show an increasing trend in application memory saving, from 4 processes to 64 processes. This trend clearly implies higher memory saving on a larger cluster.

The benefit for some applications in NPB (such as CG) are relatively low and do not scale very well with the number of processes. This is primarily because each process in such applications usually communicates with a few partners. Therefore, the number of connections for each process will not scale exponentially with the number of job processes. This means that the memory benefit can decrease or stay at the same level (the results are correspondingly better for the C2 cluster due to greater inter-node communication).

This is however not the case for SMG2000 and Radix. In these applications a process may communicate with all of the other processes, and therefore the number of dynamically allocated connections will increase exponentially with the number of processes in the job. This is why we see significant increase in memory saving when the scale of the MPI job increases.

An obvious observation in both performance and memory usage benchmarks is that the C2 cluster results are significantly better than that of the C1 cluster. As discussed in Section V, with the same number of processes, the communication between the nodes is quadrupled in C2. This also translates in more inter-process connections, which implies more memory saving on C2. The results lead to this conclusion that when the amount of inter-node communication rises, so do the benefits of using datagram-iWARP.

VII. Related Work

The implementation of the current iWARP standard in software, originated from a project by OSC [22] that provides both user-space [6] and kernel space [7] implementations. The user-space part of this software implementation is the code-base for implementing datagram-iWARP.

Another project that has recently completed its main functionality is the SoftRDMA project at IBM Zurich laboratory. This work is meant to be integrated into the Open Fabrics Enterprise Distribution (OFED) [23] stack as a software iWARP solution [16]. Our proposal in this paper is the first and the only work in this area that extends the iWARP standard to datagram transport and utilizes it in HPC applications.

Beside the iWARP solution, there have been other approaches with the goal of improving Ethernet efficiency using modern user-level libraries of other high performance interconnects. One is the Myrinet Express (MX) over Ethernet (MXoE) [18] to provide the high-performance functionality of Myrinet MX user-level library on top of Ethernet networks. Open-MX project [9] is an open-source implementation of MXoE.

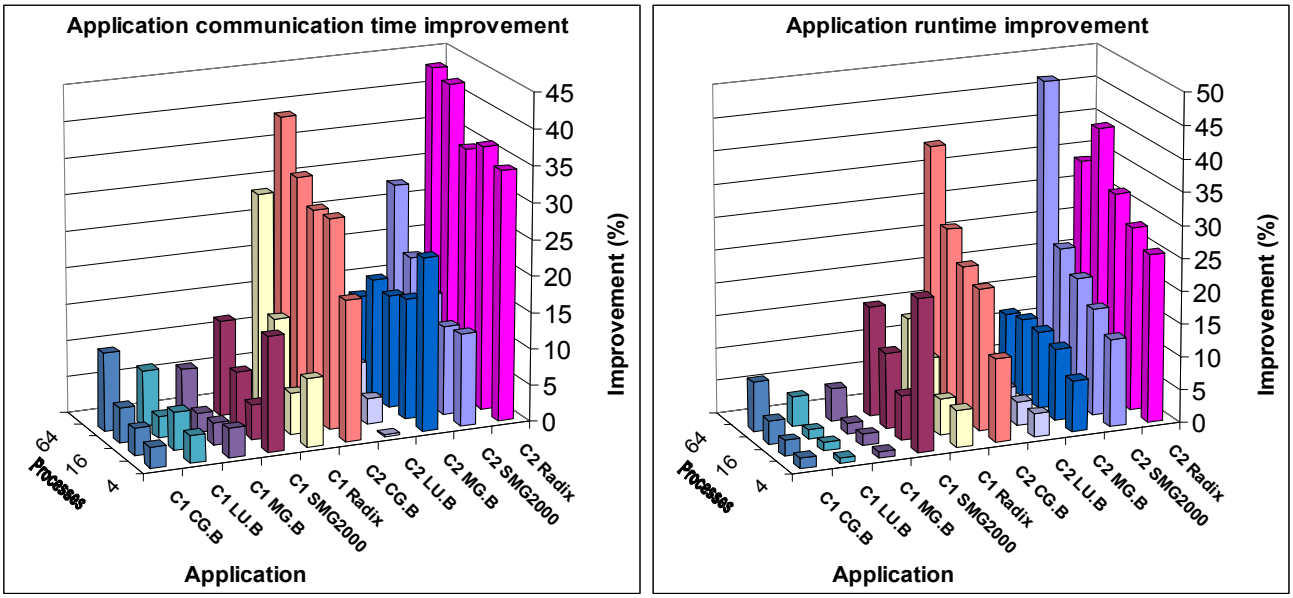

Figure 7. MPI application communication time and rumtime benefits

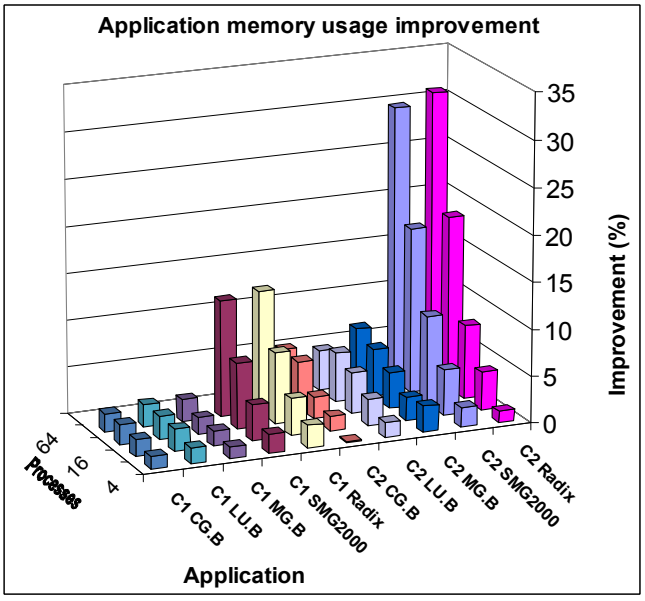

Figure 8. Memory usage improvement and scalability trend

Another work in the similar direction is InfiniBand over Ethernet (IBoE) that is also called RDMA over Ethernet (RDMAoE) and is designed to take advantage of InfiniBand's RDMA stack while simply replacing InfiniBand's link layer with Ethernet. This technology encapsulates InfiniBand reliable and unreliable service data inside Ethernet frames. An evaluation of RDMAoE can be found in [29].

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 [4] are also proposing to include RDMA functionality over CEE (RDMAoCEE).

There has been some past work on the InfiniBand network to equip MPI with the InfiniBand UD transport for scalability purposes on large scale clusters. In MVAPICH-UD project [15] an InfiniBand UD-based channel is designed for MVAPICH MPI implementation which has shown considerable memory usage benefits over the RC-based channel. The MVAPICH-Aptus, which is used as the base of our MPI work in this paper, is a continuation of the MVAPICH-UD work in [15].

VIII. Conclusions and Future Directions

In this paper we discussed some challenges facing the current iWARP standard and tried to address them by extending the standard to the datagram domain. We implemented the iWARP over UD in software to assess its potential benefits for HPC applications. Our experiments reveal that a datagram–enabled iWARP increases the scalability of large-scale HPC applications, while potentially improving their performance at the same time. In addition, verbs level microbenchmark results clearly show the potential benefits that other kinds of applications, such as datacenter services can receive from datagram-iWARP.

Our verbs level microbenchmark results show that the datagram-iWARP improves the communication latency up to 30%. Our MPI level results also show up to 14% small message latency reduction and up to 20% large message bandwidth improvement when using MPI on top of datagram-iWARP. The application results also show more than 40% runtime improvement and more than 30% application memory usage reduction for some MPI applications on a 64-core cluster. In addition, the runtime improvement and memory usage reduction trend for most of

the applications imply more application memory savings and runtime improvement on larger clusters.

This work presents a first step in standardization of datagram-iWARP and can be continued in a number of directions, including reliable datagram (reliable-UDP), tagged model, and socket interface for datacenter applications.

ACKNOWLEDGEMENT

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.

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," in Proceedings of the 2007 IEEE/ACM International Supercomputing Conference (SC07), Reno, Nevada, November 2007.

[2] J. Bonwick, "The Slab Allocator: An Object-Caching Kernel Memory Allocator," in Proceedings of the 1994 USENIX Summer Technical Conference (USTC'94), Boston, Massachusetts, 1994.

[3] 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.

[4] 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 (HotI'09), New York, NY, August 2009.

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

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

[7] D. Dalessandro, A. Devulapalli, P. Wyckoff, "iWARP Protocol Kernel Space Software Implementation," 6th Workshop on Communication Architecture for Clusters (CAC'06), in Proceedings of the 20th IEEE International Parallel & Distributed Processing Symposium (IPDPS'06), Rhodes, Greece, April 2006.

[8] W. Feng, P. Balaji, L. N. Bhuyan, D. K. Panda, "Performance Characterization of a 10-Gigabit Ethernet TOE," in Proceedings of the 13th International Symposium on High Performance Interconnects (HotI'05), Stanford, CA, August 2005.

[9] Brice Goglin, "Design and Implementation of Open-MX: High-Performance Message Passing over Generic Ethernet Hardware," 8th Workshop on Communication Architecture for Clusters (CAC'08), in Proceedings of the 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] M. Koop, T. Jones, D. K. Panda, "MVAPICH-Aptus: Scalable High-Performance Multi-Transport MPI over InfiniBand," in Proceedings of the 22nd IEEE International Parallel and Distributed Processing Symposium (IPDPS'08), Miami, FL, April 2008.

[15] 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), Seattle, WA, June 2007.

[16] 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.

[17] MPI Forum, "MPI: A Message Passing Interface Standard,": http://www.mpi-forum.org/docs/mpi-11-html/mpi-report.html.

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

[19] NAS Parallel Benchmarks, version 2.4: http://www.nas.nasa.gov/Resources/Software/npb.html.

[20] Network-Based Computing Laboratory, "MVAPICH: MPI over InfiniBand, iWARP and RDMAoE," Ohio State University: http://mvapich.cse.ohio-state.edu/.

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

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

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

[24] M. J. Rashti, A. Afsahi, "10-Gigabit iWARP Ethernet: Comparative Performance Analysis with InfiniBand and Myrinet-10G," 7th Workshop on Communication Architecture for Clusters (CAC'07), in Proceedings of the 21st IEEE International Parallel and Distributed Processing Symposium (IPDPS'07), Long Beach, CA, April 2007.

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

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

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

[28] 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.

[29] 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.

[30] Top 500 Supercomputer Sites: http://www.top500.org/.