Internet Engineering Task Force (IETF)
Request for Comments: 6581
Updates: 5043, 5044
Category: Standards Track
ISSN: 2070-1721

A. Kanevsky, Ed.
Dell Inc.
C. Bestler, Ed.
Nexenta Systems
R. Sharp
Intel
S. Wise
Open Grid Computing

April 2012

Enhanced Remote Direct Memory Access (RDMA) Connection Establishment

Abstract

This document updates RFC 5043 and RFC 5044 by extending Marker Protocol Data Unit (PDU) Aligned Framing (MPA) negotiation for Remote Direct Memory Access (RDMA) connection establishment. The first enhancement extends RFC 5044, enabling peer-to-peer connection establishment over MPA / Transmission Control Protocol (TCP). The second enhancement extends both RFC 5043 and RFC 5044, by providing an option for standardized exchange of RDMA-layer connection configuration.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6581.

Table of Contents

1. Introduction

When used over the Transmission Control Protocol (TCP), the current Remote Direct Data Placement (RDDP) [RFC5041] suite of protocols relies on the MPA [RFC5044] protocol for both connection establishment and for markers for TCP layering.

A typical model for establishing an RDMA connection has the following steps:

- The passive side (responder) Upper Layer Protocol (ULP) listens for connection requests.
- The active side (initiator) ULP submits a connection request using an RDMA endpoint, the desired destination, and the parameters to be used for the connection. Those parameters include both RDMA-layer characteristics, such as the number of simultaneous RDMA Read Requests to be allowed, and application-specific data.
- The passive side ULP receives a connection request that includes the identity of the active side and the requested connection characteristics. The passive side ULP uses this information to decide whether to accept the connection, and if it is to be accepted, how to create and/or configure the local RDMA endpoint.
- If accepting, the responder submits its acceptance of the connection request, which in turn generates the accept message to the initiator. This responder accept operation includes the RDMA endpoint to be used and the connection characteristics (both the RDMA configuration and any application-specific Private Data to be transferred to the initiator).
- The active side receives confirmation that the connection has been accepted, what the configured connection characteristics are, and any application-supplied Private Data.

Currently, MPA only supports a client-server model for connection establishment, forcing peer-to-peer applications to interact as though they had a client-server relationship. In addition, negotiation of some parameters specific to the Remote Direct Memory Access Protocol (RDMAP) [RFC5040] are left to ULP negotiation. Providing an optional ULP-independent format for exchanging these parameters would be of benefit to transport neutral RDMA applications.

1.1. Summary of Changes Affecting RFC 5044

This document enhances the MPA connection setup protocol [RFC5044]. First, it adds exchange and negotiation of the parameters necessary to support RDMA Read Requests. Second, it adds a message that serves as a Ready to Receive (RTR) indication from the initiator to the responder as the last message of connection establishment and adds negotiation of which type of message to use for carrying the RTR indication into MPA Request/Reply Frames.

RTR indications are optional and are carried by existing RDMA message types, specifically a zero-length FULPDU Send message, a zero-length RDMA Read message, or a zero-length RDMA write message. The presence vs. absence of the RTR indication and the type of RDMA message to use are negotiated by control flags in Enhanced RDMA connection establishment data specified by this document (see Section 9). RDMA implementations are often tightly integrated with application libraries and hardware, hence the flexibility to use more than one type of RDMA message enables implementations to choose message types that are less disruptive to the implementation structure. When an RTR indication is used, and MPA connection setup negotiation indicates support for multiple RDMA message types as RTR indications by both the initiator and responder, the initiator selects one of the supported RDMA message types as the RTR indication at the initiator's sole discretion.

1.2. Summary of Changes Affecting RFC 5043

This document enhances [RFC5043] by adding new Enhanced Session Control Chunks that extend the currently defined Chunks with the addition of Inbound RDMA Read Queue Depth (IRD) and Outbound RDMA Read Queue Depth (ORD) negotiation.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Definitions

Active Side: See Initiator.

Consumer: The ULPs or applications that lie above MPA and Direct Data Placement (DDP). The Consumer is responsible for making TCP or Stream Control Transmission Protocol (SCTP) connections, starting MPA and DDP connections, and generally controlling operations. See [RFC5044] and [RFC5043].

CRC: Cyclic Redundancy Check

Completion Queue (CQ): A Consumer-accessible queue where the RDMA device reports completions of Work Requests. A Consumer is able to reap completions from a CQ without requiring per-transaction support from the kernel or other privileged entity. See [RDMAC].

Completion Queue Entry (CQE): Transport- and device-specific representation of a Work Completion. A CQ holds CQEs. See [RDMAC].

FULPDU: Framed Upper Layer Protocol PDU. See FPDU of [RFC5044].

Inbound RDMA Read Request Queue (IRRQ): A queue that is associated with an RDMA connection that tracks active incoming simultaneous RDMA Read Request Messages. See [RDMAC].

Inbound RDMA Read Queue Depth (IRD): The maximum number of incoming simultaneous RDMA Read Request Messages an RDMA connection can handle. See [RDMAC].

Initiator: The endpoint of a connection that sends the MPA Request Frame. The initiator is the active side of the connection establishment. See [RFC5044].

IRD: See Inbound RDMA Read Queue Depth.

MPA Fencing: MPA responder connection establishment logic that ensures that no ULP messages will be transferred until the initiator's first message has been received.

MPA Request Frame: Data sent from the MPA initiator to the MPA responder during the Startup Phase. See [RFC5044].

MPA Reply Frame: Data sent from the MPA responder to the MPA initiator during the Startup Phase. See [RFC5044].

ORD: See Outbound RDMA Read Queue Depth.

Outbound RDMA Read Queue Depth (ORD): The maximum number of simultaneous RDMA Read Requests that can be issued for the RDMA connection. This should be less than or equal to the peer's IRD. See [RDMAC].

Passive Side: See Responder.

Private Data: A block of data exchanged between MPA endpoints during initial connection setup. See [RFC5044].

Queue Pair (QP): A Queue Pair is the set of Work Queues associated exclusively with a single Endpoint (first defined in [VIA]). The Send Queue (SQ), Receive Queue (RQ), and Inbound RDMA Read Queue (IRQ) are considered to be part of the Queue Pair. The potentially shared Completion Queue (CQ) and Shared Receive Queue (SRQ) are not. See [RDMAC].

Remote Peer: The MPA protocol implementation on the opposite end of the connection. Used to refer to the remote entity when describing protocol exchanges or other interactions between two nodes. See [RFC5044].

Responder: The connection endpoint that responds to an incoming MPA connection request (the MPA Request Frame). The responder is the passive side of the connection establishment. See [RFC5044].

Ready to Receive (RTR): RTR is an indication provided by the last connection establishment message sent from the initiator to the responder. An RTR indicates that the initiator is ready to receive messages and that connection establishment is completed.

Startup Phase: The initial exchanges of an MPA connection that serves to more fully identify MPA endpoints to each other and pass connection-specific setup information to each other. See [RFC5044].

Shared Receive Queue (SRQ): A shared pool of Receive Work Requests posted by the Consumer that can be allocated by multiple RDMA endpoints (QP). See [RDMAC].

Tagged (DDP) Message: A DDP Message that targets a Tagged Buffer that is explicitly advertised to the Remote Peer through exchange of an STag (memory handle), offset in the memory region identified by STag, and length [RFC5040].

Untagged (DDP) Message: A DDP Message that targets an Untagged Buffer associated with a queue specified the by Queue Number (QN). [RFC5040].

Work Queue: An element of a QP that allows user-space applications to submit Work Requests directly to network hardware (first defined in [VIA]). Specific Work Queues include the Send Queue (SQ) for transmit requests, Receive Queue (RQ) for receive requests specific to a single endpoint, and Shared Receive Queues (SRQs) for receive requests that can be allocated by one or more endpoints. See [RDMAC].

Work Queue Element (WQE): Transport- and device-specific representation of a Work Request. See [RDMAC].

Work Request: An elementary object used by Consumers to enqueue a requested operation (WQEs) onto a Work Queue. See [RDMAC].

4. Motivations

The goal of this document is two-fold. The first is to extend support from the current client-server model for RDMA connection setup to a peer-to-peer model. The second is to add negotiation of the RDMA Read Queue size for both sides of an RDMA connection.

4.1. Standardization of RDMA Read Parameter Configuration

Most RDMA applications are developed using a transport-neutral Application Programming Interface (API) to access RDMA services based on a "Queue Pair" paradigm as originally defined by the Virtual Interface Architecture [VIA], refined by the Direct Access Programming Library [DAPL], and most commonly deployed with the OpenFabrics API [OFA].

These transport-neutral APIs seek to provide a common set of RDMA services whether the underlying transport is, for example, RDDP over MPA, RDDP over SCTP, or InfiniBand.

The common model for establishing an RDMA connection has the following steps:

- The passive side ULP listens for connection requests.
- The active side ULP submits a connection request using an RDMA endpoint ("Queue Pair"), the desired destination, and the parameters to be used for the connection. Those parameters include both RDMA-layer characteristics, such as the number of simultaneous RDMA Read Requests to be allowed, and application-specific data (typically referred to as "Private Data").
- The passive side ULP receives a connection request, which includes the identity of the active side and the requested connection characteristics. The passive side ULP uses this information to decide whether to accept the connection, and if it is to be accepted, how to create and/or configure the RDMA endpoint.

- If accepting, the passive side ULP submits its acceptance of the connection request. This local accept operation includes the RDMA endpoint to be used and the connection characteristics (both the RDMA configuration and any application-specific Private Data to be returned).

- The active side receives confirmation that the connection has been accepted, what the configured connection characteristics are, and any application-supplied Private Data.

As currently defined, DDP connection establishment requires the ULP to encode the RDMA configuration in the application-specific Private Data. This results in undesirable duplication of logic to cover RDMA characteristics of both InfiniBand and RDDP for each ULP, and to specify for InfiniBand and RDDP the extraction of the RDMA characteristics for each ULP.

Both RDDP and InfiniBand support an initial Private Data exchange; therefore, a standard definition of the RDMA characteristics within the Private Data section would enable common connection establishment APIs to format the RDMA characteristics based on the same API information used when establishing either protocol to form the connection. The application would then only have to indicate that it was using this standard format to enable common connection establishment procedures to apply common code to properly parse these fields and configure the RDMA endpoints accordingly. Exchange of parameters necessary to perform RDMA Read operations is a common usage of the initial Private Data exchange.

One of the RDMA operations that is defined in [RDMAC] is an RDMA Read. RDMA Read operations are performed using an untagged message sent from a Queue Pair (QP) on the local endpoint to a QP on the remote endpoint targeting the Inbound RDMA Read Request Queue (QN=1 or Inbound RDMA Read Request Queue (IRRQ)) associated with the connection. RDMA Read responses transfer data associated with each RDMA Read Request from the remote endpoint to the local endpoint using tagged messages. An inbound RDMA Read Request remains on the IRRQ from the time that it is received until the time that the last tagged message associated with the RDMA request is acknowledged. The IRRQ is associated with a QP but is not a Work Queue. Instead, the IRRQ is a stand-alone queue that is used to manage RDMA Read Requests associated with a QP. See [RDMAC], Section 6 for more information regarding QPs and IRRQ. One of the characteristics that must be configured for a QP is the size of the IRRQ. This parameter is called the Inbound RDMA Read Queue Depth (IRD). Another characteristic of a QP that must be configured is a local limit on the number of simultaneous outbound RDMA Read Requests based on the size of the remote endpoint QP's IRRQ. This parameter is call the

Outbound RDMA Read Queue Depth (ORD). ORD is used to limit the number of simultaneous RDMA Read Requests such that the local endpoint does not overrun the remote endpoint's IRRQ depth or IRD. Note that outbound RDMA Reads are submitted to a QP's Send Queue at the local peer, not to a separate outbound RDMA Read Request queue on the local peer. The local endpoint uses ORD to strictly limit simultaneous Read Requests so that IRRQ overruns do not occur at the remote endpoint.

Determination of the values of the ORD and IRD are left to the ULP by the current RDDP suite of protocols and also by [RDMAC]. Since this negotiation of ORD and IRD is typical, it is desirable to provide a common mechanism as described in this document.

4.2. Enabling MPA Mode

MPA defines encoding of DDP Segments in Framed Upper Layer Protocol PDUs (FULPDUs). Generation of FULPDUs requires the ability to periodically insert MPA Markers and to generate the MPA CRC-32c for each frame. Reception may require parsing/removing the markers after using them to identify MPA Frame boundaries and validation of the MPA-CRC32c.

A major design objective for MPA was to ensure that the resulting TCP stream would be fully compliant for any and all TCP-aware middleboxes. The challenge is that while only some TCP payload streams are a valid stream of MPA FULPDUs, any sequence of bytes is a valid TCP payload stream. The determination that a given stream is in a specific MPA mode cannot be made at the MPA or TCP layer. Therefore, enabling of MPA mode is handled by the ULP.

The MPA protocol can be viewed as having two parts:

- a specification of generation and reception of MPA FULPDUs. This is unchanged by enhanced RDMA connection establishment.
- a pre-MPA exchange of messages to enable a specific MPA mode for the TCP connection. Enhanced RDMA connection establishment extends this protocol with two new features.

In typical implementations, generation and reception of MPA FULPDUs is handled by hardware. The exchange of the MPA Request and Reply Frames is then handled by host software. As will be explained, this implementation split impedes applications that are not compatible with the client-server assumptions in the current MPA Request/Reply exchange.

4.3. Lack of Explicit RTR in MPA Request/Reply Exchange

The exchange of MPA Request and Reply messages to place a TCP connection in MPA mode is specified in [RFC5044]. This protocol provides many benefits to the design of MPA FULPDU hardware:

- The ULP is responsible for specifying the exact MPA Mode (Markers enabled or disabled, CRC-32c enabled or suppressed) and the point in the TCP streams (inbound and outbound) where MPA Frames will begin.
- Before the first MPA Frame is transmitted, all pre-MPA mode TCP payloads will have been acknowledged by the peer. Therefore, it is never necessary to generate a retransmission that mixes pre-MPA and MPA payload.
- Before MPA reception is enabled, all incoming pre-MPA mode TCP payloads will have been acknowledged. Therefore, the host will never receive a TCP segment that mixes pre-MPA and MPA payload.

The limitation of the current MPA Request/Reply exchange is that it does not define a Ready to Receive (RTR) indication that the active side would send, so that the passive side can know that the last non-MPA payload (the MPA Reply) had been received.

Instead, the role of an RTR indication is piggybacked on the first MPA FULPDU sent by the active side. This is actually a valuable optimization for all applications that fit the classic client-server model. The client only initiates the connection when it has a request to send to the server, and the server has nothing to send until it has received and processed the client request.

Even applications where the server sends some configuration data immediately can easily send the same information as application Private Data in the MPA Reply. So the currently defined exchange works for almost all applications.

Many peer-to-peer applications, especially those involving cluster calculations (frequently using Message Passing Interface (MPI) [UsingMPI] or [RDS]), have no natural client or server roles ([PPMPI] [OpenMP]). Typically, one member of the cluster is arbitrarily selected to initiate the connection when the distributed task is launched, while the other accepts it. At startup time, however, there is no way to predict which node will have the first message to actually send. Immediately establishing the connections is valuable because it reduces latency once results are ready to transmit and it validates connectivity throughout the cluster.

The lack of an explicit RTR indication in the MPA Request/Reply exchange forces all applications to have a first message from the connection initiator, whether or not this matches the application communication model.

4.4. Limitations on ULP Workaround

The requirement that the RDMA connection initiator sends the first message does not appear to be onerous on first examination. The natural question is why the application layer would not simply generate a dummy message when there is no other message to submit.

There are three factors that make this workaround unsuitable for many peer-to-peer applications:

- Transport-Neutral APIs.
- Work/Completion Queue Accounting.
- Host-based implementation of MPA Fencing.

4.4.1. Transport-Neutral APIs

Many of these applications access RDMA services using a transport-neutral API such as [DAPL] or [OFA]. Only RDDP over TCP [RFC5044] has a first message requirement. Other RDMA transports, including RDDP over SCTP (see [RFC5043]) and InfiniBand (see [IBTA]), do not.

Application or middleware communications can be expressed as transport-neutral RDMA operations, allowing lower software layers to translate to transport and device specifics. Having a distinct extra message that is required only for one transport undermines the application's goal of being transport neutral.

4.4.2. Work/Completion Queue Accounting

RDMA local APIs conventionally use Work Queues to submit requests (Work Queue elements or WQEs) and to asynchronously receive completions (in Completion Queues or CQs).

Each Work Request can generate a Completion Queue Entry (CQE). Completions for successful transmit Work Requests are frequently suppressed, but the CQ capacity must account for the possibility that each will complete in error. A CQ can receive completions from multiple Work Queues.

CQs are defined to allow hardware RDMA implementations to generate CQEs directly to a user-space-mapped buffer. This enables a user-space RDMA Consumer to reap completions without requiring kernel intervention.

A hardware RDMA implementation cannot reasonably wait for an available slot in the CQ. The queue must be sized such that an overflow will not occur. When an overflow does occur, it is considered a catastrophic error and will typically require tearing down all RDMA connections using that CQ.

This style of interface is very efficient, but places a burden on the application to properly size each CQ to match the Work Queues that feed it.

While the format of both WQEs and CQEs is transport and device dependent, a transport-neutral API can deal with WQEs and CQEs as abstract transport- and device-neutral objects. Therefore, the number of WQEs and CQEs required for an application can be transport and device neutral.

The capacity of the Work Queues and CQs can be calculated in an abstract transport- and device-neutral fashion. If a dummy operation approach is used, it would require lower layers to know the usage model, and would disrupt the calculations by inserting a dummy "operation" Work Request and filtering out the matching completion. The lower layer does not know the usage model on which the queue sizes are built, nor does it know how frequently an insertion will be required.

4.4.3. Host-based Implementation of MPA Fencing

Many hardware implementations of RDDP using MPA/TCP do not handle the MPA Request/Reply exchange in hardware, rather they are handled by the host processor in software. With such designs, it is common for the MPA Fencing to be implemented in the user-space, device-specific library (commonly referred to as a 'User Verbs' library or module).

When the generation and reception of MPA FULPDUs are already dedicated to hardware, a Work Completion can only be generated by an untagged message, since arrival of a message for a tagged buffer does not necessarily generate a completion and is done without any interaction with ULP [RFC5040].

5. Enhanced MPA Connection Establishment

Below we provide an overview of Enhanced Connection Setup. The goal is to allow standard negotiation of the ORD/IRD setting on both sides of the RDMA connection and/or to negotiate the initial data transfer operation by the initiator when the existing 'client sends first' rule does not match application requirements.

The RDMA connection initiator sends an MPA Request, as specified in [RFC5044]; the new format defined here allows for:

- Standardized negotiation of ORD and IRD.
- Negotiation of RTR functionality and the RDMA message type to use as the RTR indication.

The RDMA connection responder processes the MPA Request and generates an MPA Reply, as specified in [RFC5044]; the new format completes the negotiation.

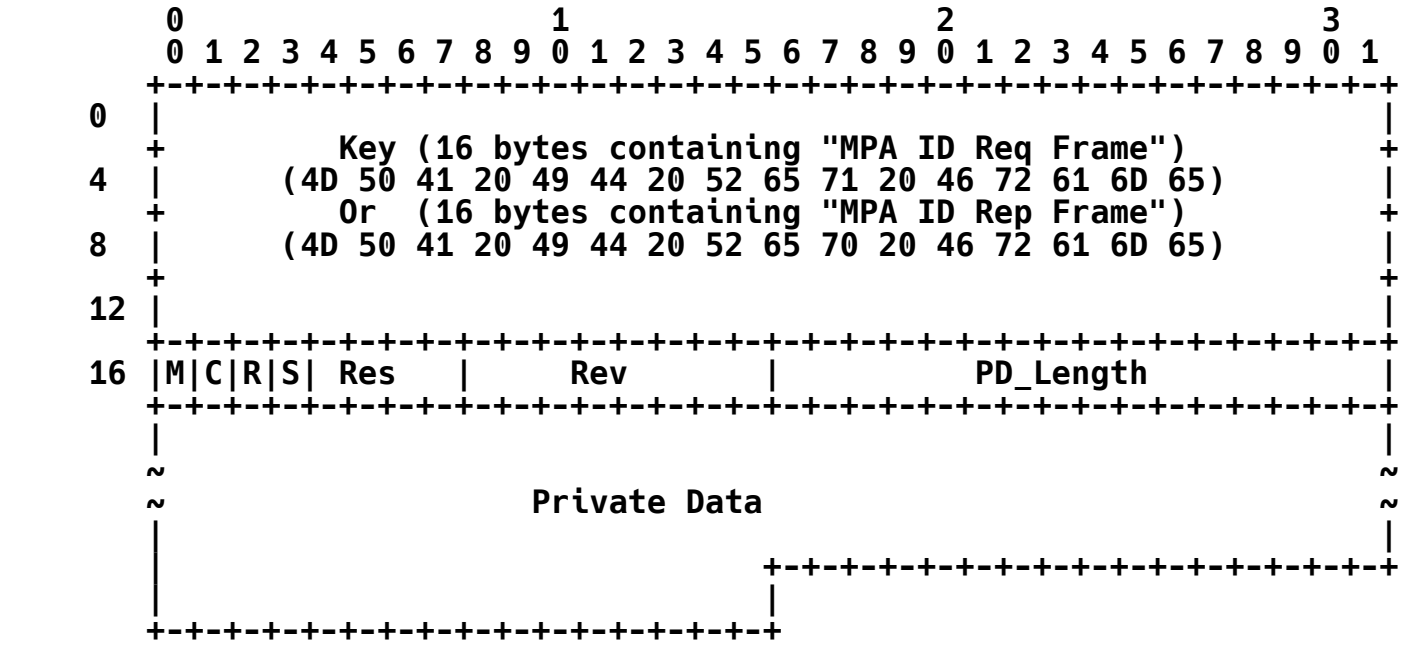
The local interface needs to provide a way for a ULP to request the use of explicit RTR indication on a per-application or per-connection basis when an explicit RTR indication will be required. Piggybacking the RTR on a Client's first message is a valuable optimization for most connections.

The RDMA connection initiator MUST NOT allow any later FULPDUs to be transmitted before the RTR indication. One method to achieve this is to delay notifying the ULP that the RDMA connection has been established until after any required RTR indication has been transmitted.

All MPA exchanges are performed via TCP prior to RDMA establishment, and are therefore signaled via TCP and not via RDMA completion.

6. Enhanced MPA Request/Reply Frames

Enhanced RDMA connection establishment uses an alternate format for MPA Requests and Replies as follows:



Key: Unchanged from [RFC5044].

M: Unchanged from [RFC5044].

C: Unchanged from [RFC5044].

R: Unchanged from [RFC5044].

S: One, if the Private Data begins with the enhanced RDMA connection establishment data; 0 otherwise.

Res: One bit smaller than in [RFC5044]; otherwise unchanged. In [RFC5044], the 'Res' field, in which the newly defined 'S' bit resides, is reserved for future use. [RFC5044] specifies that 'Res' MUST be set to zero when sending and MUST NOT be checked on reception, making use of 'S' bit backwards compatibility with the original MPA Frame format. When the 'S' bit is set to zero, no additional Private Data is used for enhanced RDMA connection establishment; therefore, the resulting MPA Request and Reply Frames are identical to the unenhanced protocol.

Rev: This field contains the revision of MPA. To use any enhanced connection establishment feature, this MUST be set to two or higher. If no enhanced connection establishment features are desired, it MAY be set to one. A host accepting MPA connections MUST continue to accept MPA Requests with version one, even if it supports version two.

PD_Length: Unchanged from [RFC5044]. This is the total length of the Private Data field, including the enhanced RDMA connection establishment data, if present.

Private Data: Unchanged from [RFC5044]. However, if the 'S' flag is set, Private Data MUST begin with enhanced RDMA connection establishment data (see Section 9).

7. Enhanced SCTP Session Control Chunks

Enhanced RDMA connection establishment uses the first 32 bits of the Private Data field for IRD and ORD negotiation in the "DDP Stream Session Initiate" and "DDP Stream Session Accept" SCTP Session Control Chunks.

The type of the SCTP Session Control Chunk is defined by a Function Code (see [RFC4960]). [RFC5043] already defines codes for 'DDP Stream Session Initiate' and 'DDP Stream Session Accept', which are equivalent to an MPA Request Frame and an accepting MPA Reply Frame.

Enhanced RDMA connection establishment requires three additional function codes listed below:

Enhanced DDP Stream Session Initiate: 0x005

Enhanced DDP Stream Session Accept: 0x006

Enhanced DDP Stream Session Reject: 0x007

The Enhanced Reject function code MUST be used to indicate rejection of enhanced DDP stream session for a configuration that would have been accepted for unenhanced DDP stream session negotiation.

The enhanced DDP stream session establishment follows the same rules as the standard DDP stream session establishment as defined in [RFC5043]. ULP-supplied Private Data MUST be included for Enhanced DDP Stream Session Initiate, Enhanced DDP Stream Session Accept, and Enhanced DDP Stream Session Reject messages, and MUST follow the enhanced RDMA connection establishment data in the DDP Stream Session Initiate and the Enhanced DDP Stream Session Accept messages.

Private Data length MUST NOT exceed 512 bytes in any message, including enhanced RDMA connection establishment data.

Private Data MUST NOT be included in the DDP Stream Session TERM message.

Received Extended DDP Stream Session Control messages SHOULD be reported to the ULP. If reported, any supplied Private Data MUST be available for the ULP to examine. For example, a received Extended DDP Stream Session Control message is not reported to ULP if none of the requested RTR indication types are supported by the receiver. In this case, the Provider MAY generate a reject reply message indicating which RTR indication types it supports.

The enhanced DDP stream management MUST use the DDP stream session termination function code to terminate a stream established using enhanced DDP stream session function codes.

[RFC5043] already supports either side sending the first DDP Message since the Payload Protocol Identifier (PPID) already distinguishes between Session Establishment and DDP Segments. The enhanced RDMA connection establishment provides the ULP a transport-independent way to support the peer-to-peer model.

The following additional Legal Sequences of DDP Stream Session messages are defined:

- Enhanced Active/Passive Session Accepted: as with Section 6.2 of [RFC5043], but with the extended opcodes as defined in this document.

- Enhanced Active/Passive Session Rejected: as with Section 6.3 of [RFC5043], but with the extended opcodes as defined in this document.

- Enhanced Active/Passive Session Non-ULP Rejected: as with Section 6.4 of [RFC5043], but with the extended opcodes as defined in this document.

8. MPA Error Reporting

The RDMA connection establishment protocol is layered upon the protocols defined in [RFC5040] and [RFC5041]. Any enhanced RDMA connection establishment error generates an MPA termination message to a peer. [RFC5040] defines a triplet of protocol layers, error types, and error codes for error specification. MPA negotiation for RDMA connection establishment uses the following layer and error type for MPA error reporting:

Layer: 0x2 - LLP Error Type: 0x0 - MPA

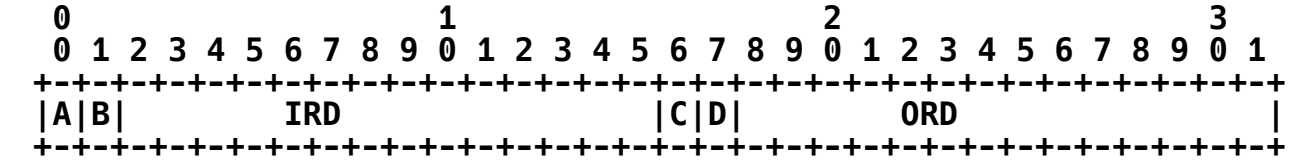
While [RFC5044] defines four error codes, [RFC5043] does not define any. Enhanced RDMA connection establishment extends the error codes defined in [RFC5044] by adding three new error codes. Thus, enhanced RDMA connection establishment is backward compatible with both [RFC5043] and [RFC5044].

The following error codes are defined for enhanced RDMA connection establishment negotiation:

Error Code	Description
0x05	Local catastrophic
0x06	Insufficient IRD resources
0x07	No matching RTR option

9. Enhanced RDMA Connection Establishment Data

Enhanced RDMA connection establishment places the following 32 bits at the beginning of the Private Data field of the MPA Request and Reply Frames or the "DDP Stream Session Initiate" and "DDP Stream Session Accept" SCTP Session Control Chunks. ULP-specified Private Data follows this field. The maximum amount of ULP-specified Private Data is therefore reduced by 4 bytes. Note that this field MUST be sent in network byte order, with the IRD and ORD encoded as 14-bit unsigned integers.



IRD: Inbound RDMA Read Queue Depth.

ORD: Outbound RDMA Read Queue Depth.

A: Control Flag for connection model.

B: Control Flag for use of a zero-length FULPDU (Send) RTR indication.

C: Control Flag for use of a zero-length RDMA Write RTR indication.

D: Control Flag for use of a zero-length RDMA Read RTR indication.

9.1. IRD and ORD Negotiation

The IRD and ORD are used for negotiation of Inbound RDMA Read Request Queue depths for both endpoints of the RDMA connection. The IRD is used to configure the depth of the Inbound RDMA Read Request Queue (IRRQ) on each endpoint. ORD is used to limit the number of simultaneous outbound RDMA Read Requests allowed at any given point in time in order to avoid IRRQ overruns at the remote endpoint. In order to describe the negotiation of both local endpoint and remote endpoint ORD and IRD values, four terms are defined:

Initiator IRD: The IRD value sent in the MPA Request or "DDP Stream Session Initiate" SCTP Session Control Chunk. This is the value of the initiator's IRD at the time of the MPA Request generation. The responder sets its local ORD value to this value or less. The initiator IRD is the maximum number of simultaneous inbound RDMA Read Requests that the initiator can support for the requested connection.

Initiator ORD: The ORD value in the MPA Request or "DDP Stream Session Initiate" SCTP Session Control Chunk. This is the initial value of the initiator's ORD at the time of the MPA Request generation and also a request to the responder to support a responder IRD of at least this value. The initiator ORD is the maximum number of simultaneous outbound RDMA Read operations that the initiator desires the responder to support for the requested connection.

Responder IRD: The IRD value returned in the MPA Reply or "DDP Stream Session Accept" SCTP Session Control Chunk. This is the actual value that the responder sets for its local IRD. This value is greater than or equal to the initiator ORD for successful negotiations. The responder IRD is the maximum number of simultaneous inbound RDMA Read Requests that the responder actually can support for the requested connection.

Responder ORD: The ORD value returned in the MPA Reply or "DDP Stream Session Accept" SCTP Session Control Chunk. This is the actual value that the responder used for ORD and is less than or equal to the initiator IRD for successful negotiations. The responder ORD is the maximum number of simultaneous outbound RDMA Read operations that the responder will allow for the requested connection.

The relationships between these parameters after a successful negotiation is complete are the following:

initiator ORD <= responder IRD

responder ORD <= initiator IRD

The responder and initiator MUST pass the peer's provided IRD and ORD values to the ULP, in addition to using the values as calculated by the preceding rules.

The responder ORD SHOULD be set to a value less than or equal to the initiator IRD. If the initiator ORD is insufficient to support the selected connection model, the responder IRD MAY be increased; for example, if the initiator ORD is 0 (RDMA Reads will not be used by the ULP) and the responder supports use of a zero-length RDMA Read RTR indication, then the responder IRD can be set to 1. The responder MUST set its ORD at most to the initiator IRD. The responder MAY reject the connection request if the initiator IRD is not sufficient for the ULP-required ORD and specify the required ORD in the MPA Reject Frame responder ORD. Thus, the TERM message MUST contain Layer 2, Error Type 0, Error Code 6.

Upon receiving the MPA Accept Frame from the responder, the initiator MUST set its IRD at least to the responder ORD and its ORD at most to the responder IRD. If the initiator does not have sufficient resources for the required IRD, it MUST send a TERM message to the responder indicating insufficient resources and terminate the connection due to insufficient resources. Thus, the TERM message MUST contain Layer 2, Error Type 0, Error Code 6.

The initiator MUST pass the responder provided IRD and ORD to the ULP for both MPA Accept and Reject messages. The initiator ULP can decide its course of action. For example, the initiator ULP may terminate the established connection and renegotiate the responder ORD.

An all ones value (0x3FFF) indicates that automatic negotiation of the IRD or ORD is not desired, and that the ULP will be responsible for it. The responder MUST respond to an initiator ORD value of 0x3FFF by leaving its local endpoint IRD value unchanged and setting the IRD to 0x3FFF in its reply message. The initiator MUST leave its local endpoint ORD value unchanged upon receiving a responder IRD value of 0x3FFF. The responder MUST respond to an initiator IRD value of 0x3FFF by leaving its local endpoint ORD value unchanged, and setting ORD to 0x3FFF in its reply message. The initiator MUST leave its local endpoint IRD value unchanged upon receiving a responder ORD value of 0x3FFF.

9.2. Peer-to-Peer Connection Negotiation

Control Flag A value 1 indicates that a peer-to-peer connection model is being performed, and value 0 indicates a client-server model. Control Flag B value 1 indicates that a zero-length FULPDU (Send) RTR indication is requested for the initiator and supported by the responder, respectively, 0 otherwise. Control Flag C value 1 indicates that a zero-length RDMA Write RTR indication is requested for the initiator and supported by the responder, respectively, 0 otherwise. Control Flag D value 1 indicates that a zero-length RDMA Read RTR indication is requested for the initiator and supported by the responder, respectively, 0 otherwise. The initiator MUST set Control Flag A to 1 for the peer-to-peer model. The initiator MUST set each Control Flag B, C, and D to 1 for each of the options it supports, if Control Flag A is set to 1.

The responder MUST support at least one RTR indication option if it supports Enhanced RDMA connection establishment. If Control Flag A is 1 in the MPA Request message, then the responder MUST set Control Flag A to 1 in the MPA reply message. For each initiator-supported RTR indication option, the responder SHOULD set the corresponding Control Flag if the responder can support that option in an MPA reply. The responder is not required to specify all RTR indication options it supports. The responder MUST set at least one RTR indication option if it supports more than one initiator-specified RTR indication option. The responder MAY include additional RTR indication options it supports, even if not requested by any initiator specified RTR indication options. If the responder does not support any of the initiator-specified RTR indication options, then the responder MUST set at least one RTR indication type option it supports.

Upon receiving the MPA Accept Frame with Control Flag A set to 1, the initiator MUST generate one of the negotiated RTR indications. If the initiator is not able to generate any of the responder-supported RTR indications, then it MUST send a TERM message to the responder indicating failure to negotiate a mutually compatible connection model or RTR option, and terminate the connection. Thus, the TERM message MUST contain Layer 2, Error Type 0, Error Code 7. The ULP can negotiate a ULP-level RTR indication when a Provider-level RTR indication cannot be negotiated.

The initiator MUST set Control Flag A to 0 for the client-server model. The responder MUST set Control Flag A to 0 if Control Flag A is 0 in the request. If Control Flag A is set to 0, then Control Flags B, C, and D MUST also be set to 0. On reception, if Control Flag A is set to 0, then Control Flags B, C, and D MUST be ignored.

9.3. Enhanced Connection Negotiation Flow

The RTR indication type and ORD/IRD negotiation follows the following order:

initiator (MPA Request) --> The initiator sets Control Flag A to 1 to indicate the peer-to-peer connection model and sets its initial IRD/ORD on the local endpoint of the connection. The initiator also sets Control Flags B, C, and D to 1 for each initiator-supported option of RTR indication.

responder (MPA Reply) <-- The responder matches the initiator's Control Flag A value and sets ORD/IRD to its local endpoint values based upon the initiator's initial ORD/IRD values and the number of simultaneous RDMA Read Requests required by the ULP. The responder sets Control Flags B, C, and D to 1 for each responder-supported option of RTR indication options for the peer-to-peer connection model. The responder also sets its IRD/ORD to actual values.

initiator (First RDMA Message) --> After the initiator modifies its ORD/IRD to match the responder's values as stated above, the initiator sends the first message of the negotiated RTR indication option. If no matching RTR indication option exists, then the initiator sends a TERM message.

The initiator or responder MUST generate the TERM message that contains Layer 2, Error Type 0, Error Code 5 when it encounters any error locally for which the special Error Code is not defined in Section 8 before resetting the connection.

10. Interoperability

The initiator requests enhanced RDMA connection establishment by sending an enhanced RDMA establishment request; an enhanced responder is REQUIRED to respond with an enhanced RDMA connection establishment response, whereas an unenhanced responder treats the enhanced request as incorrectly formatted and closes the TCP connection. All responders are REQUIRED to issue unenhanced RDMA connection establishment responses in response to unenhanced RDMA connection establishment requests.

The initiator MUST NOT use the enhanced RDMA connection establishment formats or function codes when no enhanced functionality is desired.

The responder MUST continue to accept unenhanced connection requests.

There are three initiator/responder cases that involve enhanced MPA: both the initiator and responder, only the responder, and only the initiator. The enhanced MPA Frame is defined by field 'S' set to 1.

Enhanced MPA initiator and responder: If the responder receives an enhanced MPA message, it MUST respond with an enhanced MPA message.

Enhanced MPA responder only: If the responder receives an unenhanced MPA message ('S' is set to 0), it MUST respond with an unenhanced MPA message.

Enhanced MPA initiator only: If the responder receives an enhanced MPA message and it does not support enhanced RDMA connection establishment, it MUST close the TCP connection and exit MPA. From a standard RDMA connection establishment point of view, the enhanced MPA Frame is improperly formatted as stated in [RFC5044]. Thus, both the initiator and responder report TCP connection termination to an application locally. In this case, the initiator MAY attempt to establish an RDMA connection using the unenhanced MPA protocol as defined in [RFC5044] if this protocol is compatible with the application, and let the ULP deal with ORD and IRD and peer-to-peer negotiations.

A note for potential future enhancements for connection establishment negotiation: It is possible to further extend formatting of Private Data of the MPA Request and Reply Frames and to use other bits from the "Res" field to indicate additional Private Data formatting.

11. IANA Considerations

IANA has added the following entries to the "SCTP Function Codes for DDP Session Control" registry created by Section 3.5 of [RFC6580]:

0x0005, Enhanced DDP Stream Session Initiate, [RFC6581]

0x0006, Enhanced DDP Stream Session Accept, [RFC6581]

0x0007, Enhanced DDP Stream Session Reject, [RFC6581]

IANA has added the following entries to the "MPA Errors" registry created by Section 3.3 of [RFC6580]:

0x2/0x0/0x05, - MPA Error / Local catastrophic error, [RFC6581]

0x2/0x0/0x06 - MPA Error / Insufficient IRD resources, [RFC6581]

0x2/0x0/0x07 - MPA Error / No matching RTR option, [RFC6581]

12. Security Considerations

The security considerations from RFC 5044 and RFC 5043 apply and the changes in this document do not introduce new security considerations. However, it is recommended that implementations do sanity checking for the input parameters, including ORD, IRD, and the control flags used for RTR indication option negotiation.

13. Acknowledgements

The authors wish to thank Sean Hefty, Dave Minturn, Tom Talpey, David Black, and David Harrington for their valuable contributions and reviews of this document.

14. References

14.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[RFC5040] Recio, R., Metzler, B., Culley, P., Hilland, J., and D. Garcia, "A Remote Direct Memory Access Protocol Specification", RFC 5040, October 2007.

[RFC5041] Shah, H., Pinkerton, J., Recio, R., and P. Culley, "Direct Data Placement over Reliable Transports", RFC 5041, October 2007.

[RFC5043] Bestler, C. and R. Stewart, "Stream Control Transmission Protocol (SCTP) Direct Data Placement (DDP) Adaptation", RFC 5043, October 2007.

[RFC5044] Culley, P., Elzur, U., Recio, R., Bailey, S., and J. Carrier, "Marker PDU Aligned Framing for TCP Specification", RFC 5044, October 2007.

[RFC6580] Ko, M. and D. Black, "IANA Registries for the Remote Direct Data Placement (RDDP) Protocols", RFC 6580, April 2012.

14.2. Informative References

[DAPL] "Direct Access Programming Library", <http://www.datcollaborative.org/uDAPL_doc_062102.pdf>.

[IBTA] "InfiniBand Architecture Specification Release 1.2.1", <http://www.infinibandta.org>.

[OFA] "OFA verbs & APIs", <http://www.openfabrics.org/>.

[OpenMP] McGraw-Hill, "Parallel Programming in C with MPI and OpenMP", 2003.

[PPMPI] Morgan Kaufmann Publishers Inc., "Parallel Programming with MPI", 2008.

[RDMAC] "RDMA Protocol Verbs Specification (Version 1.0)", <http://www.rdmaconsortium.org/home/draft-hilland-iwarp-verbs-v1.0-RDMAC.pdf>.

[RDS] Open Fabrics Association, "Reliable Datagram Socket", 2008, <http://www.openfabrics.org/archives/spring2008sonoma>.

[UsingMPI] MIT Press, "Using MPI-2: Advanced Features of the Message Passing Interface", 1999.

[VIA] Cameron, Don and Greg Regnier, "Virtual Interface Architecture", Intel, April 2002.

Authors' Addresses

Arkady Kanevsky (editor)
Dell Inc.
One Dell Way, MS PS2-47
Round Rock, TX 78682
USA

Phone: +1-512-728-0000
EMail: arkady.kanevsky@gmail.com

Caitlin Bestler (editor)
Nexenta Systems
555 E El Camino Real #104
Sunnyvale, CA 94087
USA

Phone: +1-949-528-3085
EMail: Caitlin.Bestler@nexenta.com

Robert Sharp
Intel
LAD High Performance Message Passing, Mailstop: AN1-WTR1
1501 South Mopac, Suite 400
Austin, TX 78746
USA

Phone: +1-512-493-3242
EMail: robert.o.sharp@intel.com

Steve Wise
Open Grid Computing
4030 Braker Lane STE 130
Austin, TX 78759
USA

Phone: +1-512-343-9196 x101
EMail: swise@opengridcomputing.com