SNIA | Data, Storage & Networking

Everything You Wanted to Know About RDMA But Were Too Proud to Ask

Live Webinar
March 26, 2025
10:00 am PT / 1:00 pm ET

Today's Presenters

Erik Smith
Distinguished Engineer
Dell Technologies
Moderator

Michal Kalderon
Distinguished Engineer,
Software Architect
Marvell

Rohan Mehta
Senior Software Engineer
Microsoft

The SNIA Community

200
industry leading organizations

2,000
active contributing members

50,000
IT end users & storage pros worldwide

SNIA | Data, Storage & Networking

What We Do

Drive the awareness and adoption of a broad set of technologies, including:

- Storage Protocols (Block, File, Object)
- Traditional and software-defined storage
- Disaggregated, virtualized and hyperconverged
- AI, including storage and networking considerations
- Edge implementation opportunities and factors
- Storage and networking security
- Acceleration and offloads
- Programming frameworks
- Sustainability

How We Do It

By delivering:

- Expert webinars and podcasts
- White papers
- Articles in trade journals
- Blogs
- Social Media
- Presentations at industry events

SNIA | DSN A SNIA Community

Logistics

- The slides are available under the attachments tab at the bottom of your console.
- Questions are welcome!
- Please rate the session and provide feedback!
- Want more sessions like this or other topics, let us know!
  - JOIN US! We meet on Thursday mornings at 10 AM eastern.
  - Email dsn-chair@snia.com if you have questions.

SNIA Legal Notice

- The material contained in this presentation is copyrighted by SNIA unless otherwise noted.
- Member companies and individual members may use this material in presentations and literature under the following conditions:
    - Any slide or slides used must be reproduced in their entirety without modification
    - SNIA must be acknowledged as the source of any material used in the body of any document containing material from these presentations.
- This presentation is a project of SNIA.
- Neither the author nor the presenter is an attorney and nothing in this presentation is intended to be, or should be construed as legal advice or an opinion of counsel. If you need legal advice or a legal opinion please contact your attorney.
- The information presented herein represents the author's personal opinion and current understanding of the relevant issues involved. The author, the presenter, and the SNIA do not assume any responsibility or liability for damages arising out of any reliance on or use of this information.

  NO WARRANTIES, EXPRESS OR IMPLIED. USE AT YOUR OWN RISK.

Agenda

- What is RDMA ?
- RDMA – Technical Details
- Use Cases
- Key Takeaways

What is (R)DMA?

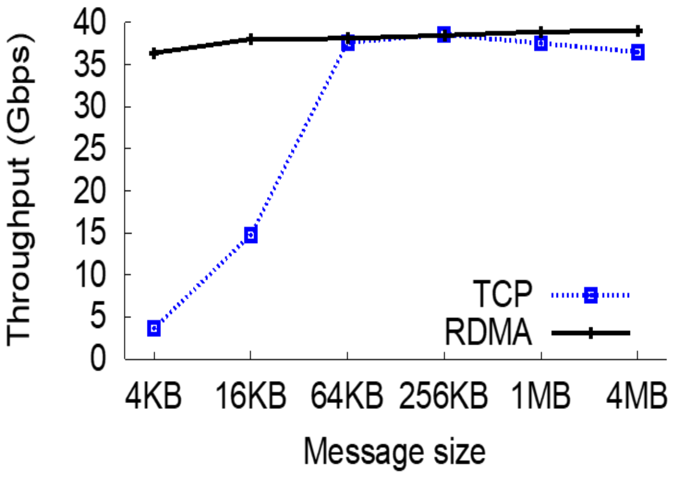
DMA - Direct Memory Access: Allow hardware components to directly read from and write to the main memory without involving the CPU

RDMA - Remote Direct Memory Access: Extends DMA capabilities over a network, enabling one computer to directly access the memory of another computer without involving their CPUs, cache, or OS

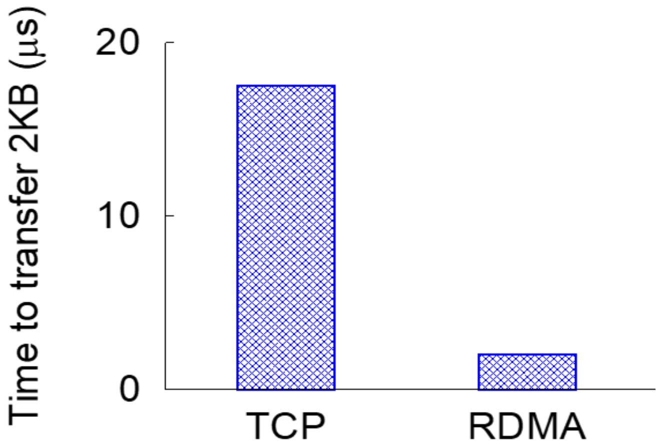
Benefits of RDMA

- Zero copy data
- Bypasses the CPU
- Bypasses the OS kernel
- Message based transactions
- Low Latency
- High Bandwidth

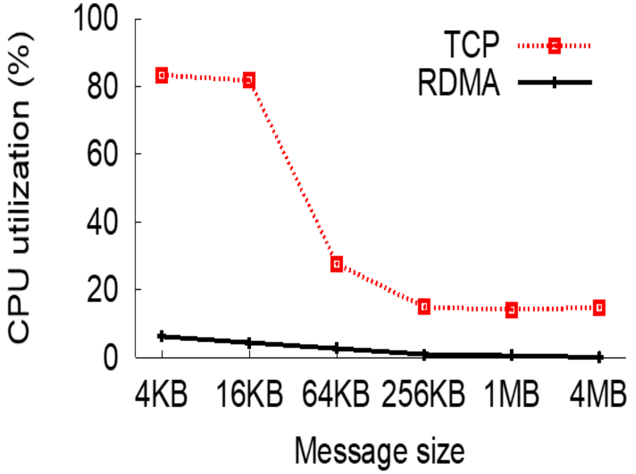
Performance Improvements



1 **Increase throughput up to 86%**



2 **Reduced latency up to 88%**



3 **Reduced CPU usage up to 65%**

https://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p523.pdf

RDMA – A Historical Timeline

Early Days (1990s)

1993: RDMA concept outlined in HP patent

Mid-1990s: Early RDMA research and development for HPC

1995: Paper on parallel computing using standard servers laid the groundwork

InfiniBand Era (2000s)

1999: InfiniBand Trade Association (IBTA) formed

2000: InfiniBand Architecture Specification v1.0 released

Early 2000s: InfiniBand gains traction in HPC (Mellanox a key player)

Mid-2000s: Intel and Microsoft shift focus to PCIe

Ethernet Integration (2010s)

2010: RoCE (RDMA over Converged Ethernet) introduced

2014: RoCE v2 released with improved performance

Late 2010s: RoCE adoption grows in data centers

Commercial Deployments & Growth (2010s - Present)

Early 2010s: Early adopters in cloud and finance

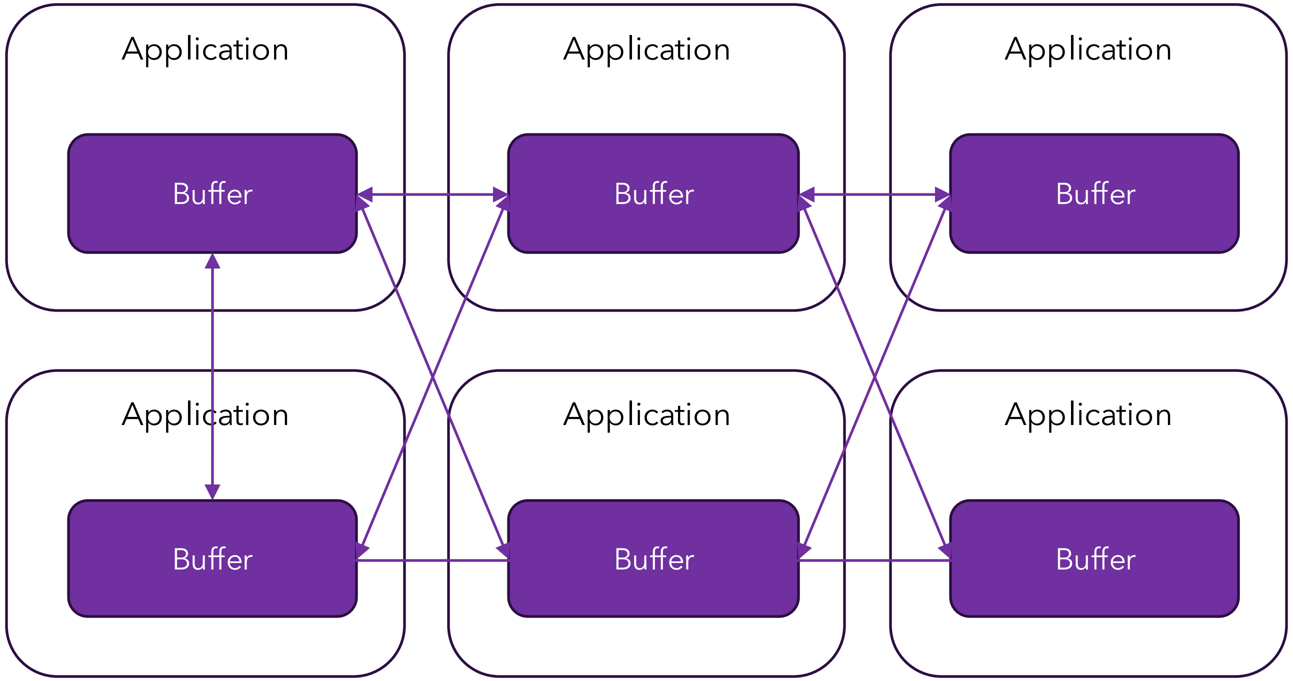
Mid-2010s: NVMe over Fabrics (NVMe-oF) drives further adoption

Late 2010s: Essential for AI and machine learning

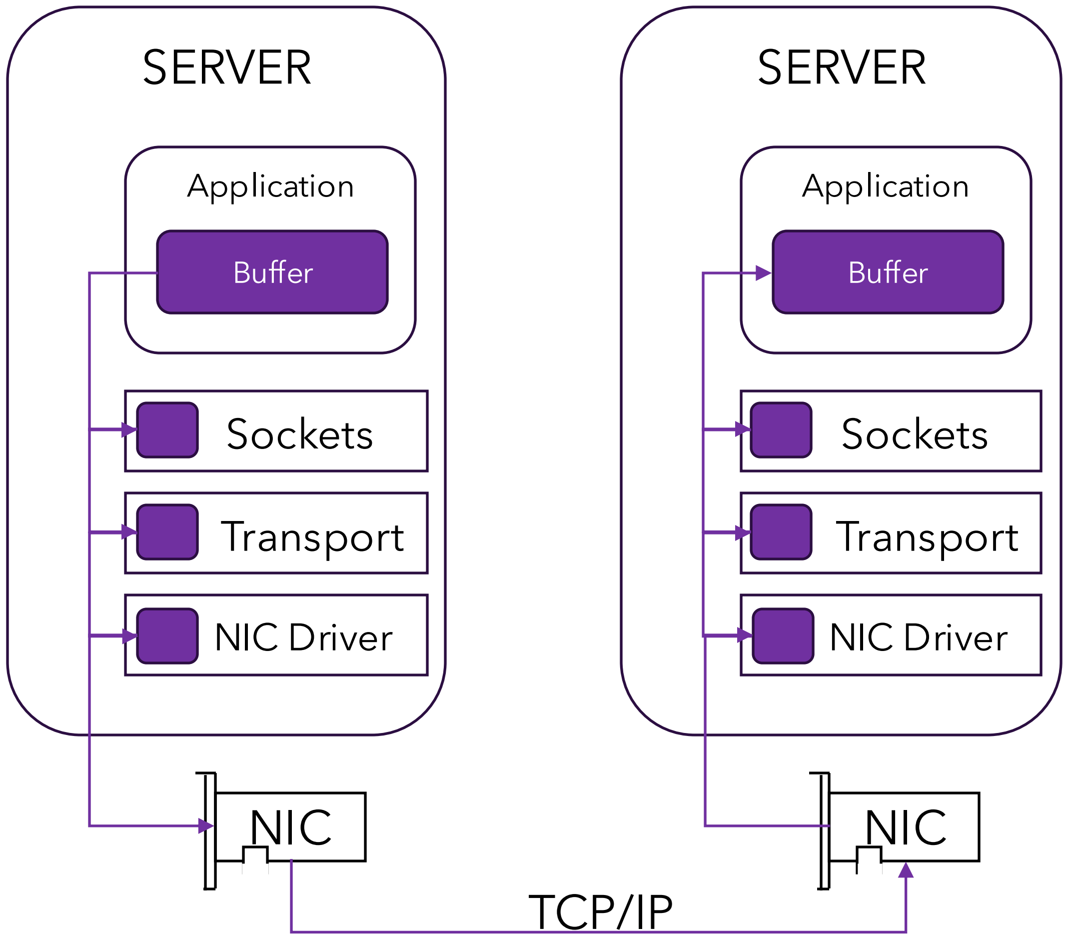
2019: NVIDIA acquires Mellanox

Present: Continued evolution and key role in high-performance computing, cloud, and AI

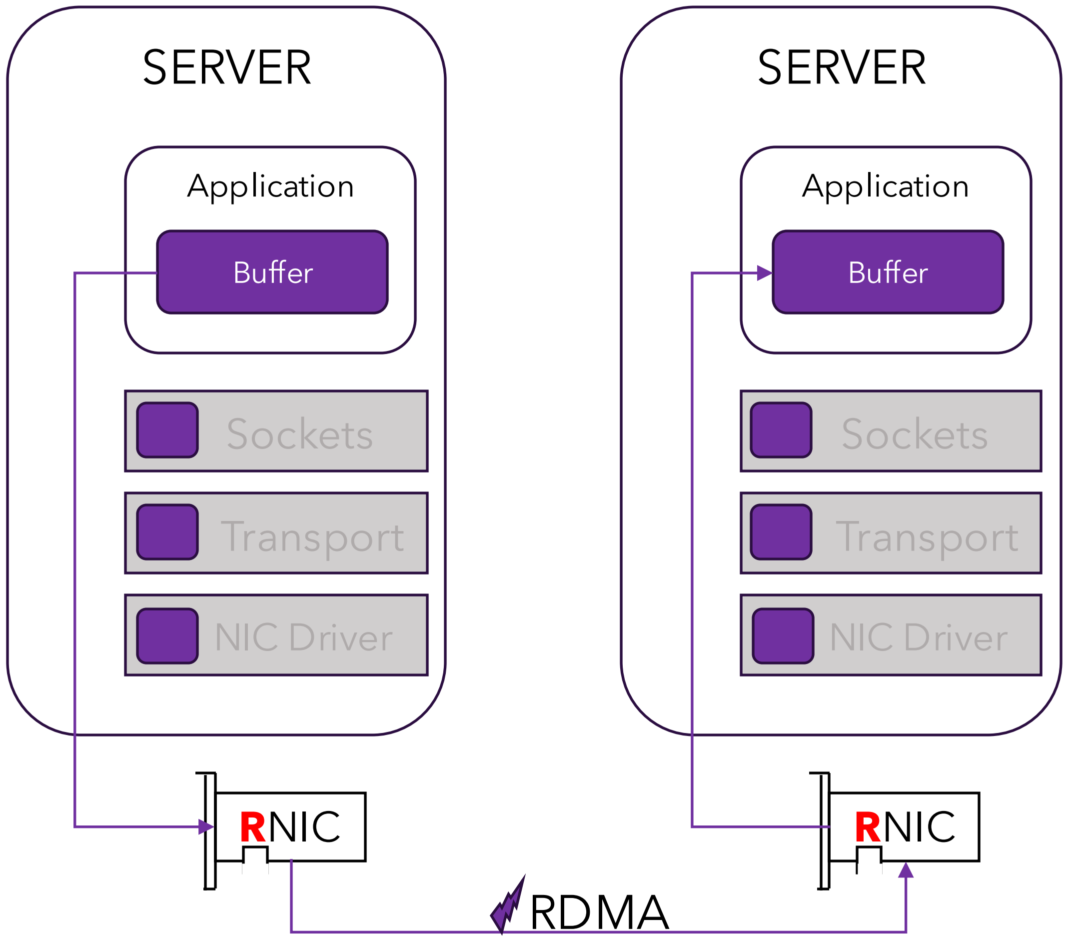
The Message Passing Programming Model



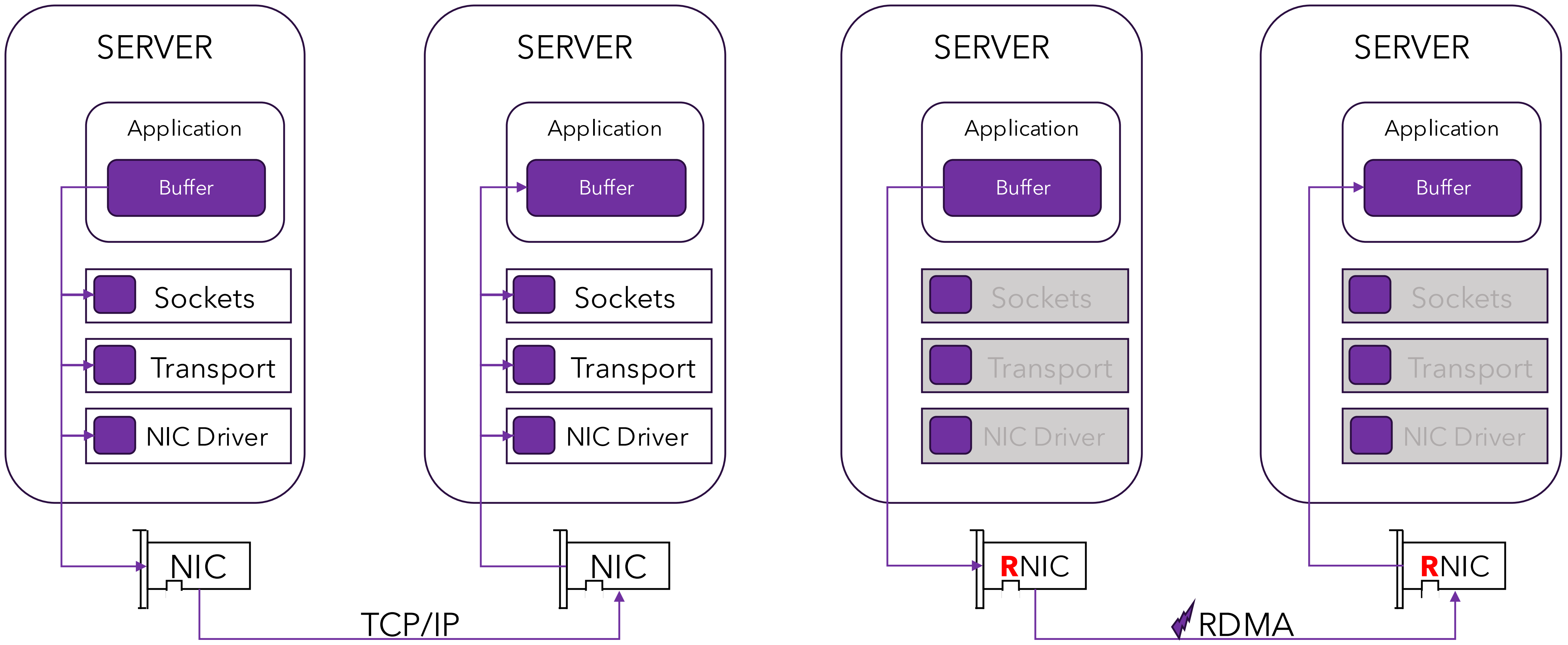
Message Passing over TCP



Message Passing over RDMA



TCP vs. RDMA


SERVER
Application
Buffer
Sockets
Transport
NIC Driver
NIC
SERVER
Application
Buffer
Sockets
Transport
NIC Driver
NIC
TCP/IP
SERVER
Application
Buffer
Sockets
Transport
NIC Driver
RNIC
SERVER
Application
Buffer
Sockets
Transport
NIC Driver
RNIC
RDMA

RDMA – Technical Details

Objects, Connection Management, Verbs, On the Wire

SNIA | DSN A SNIA Community

RDMA Operations

Channel Semantics: Send / Recv

Send Operation: Sender sends a message to the receiver.

Receive Operation: Receiver needs a corresponding operation to handle the incoming data.

Memory Semantics: Read / Write / Atomic

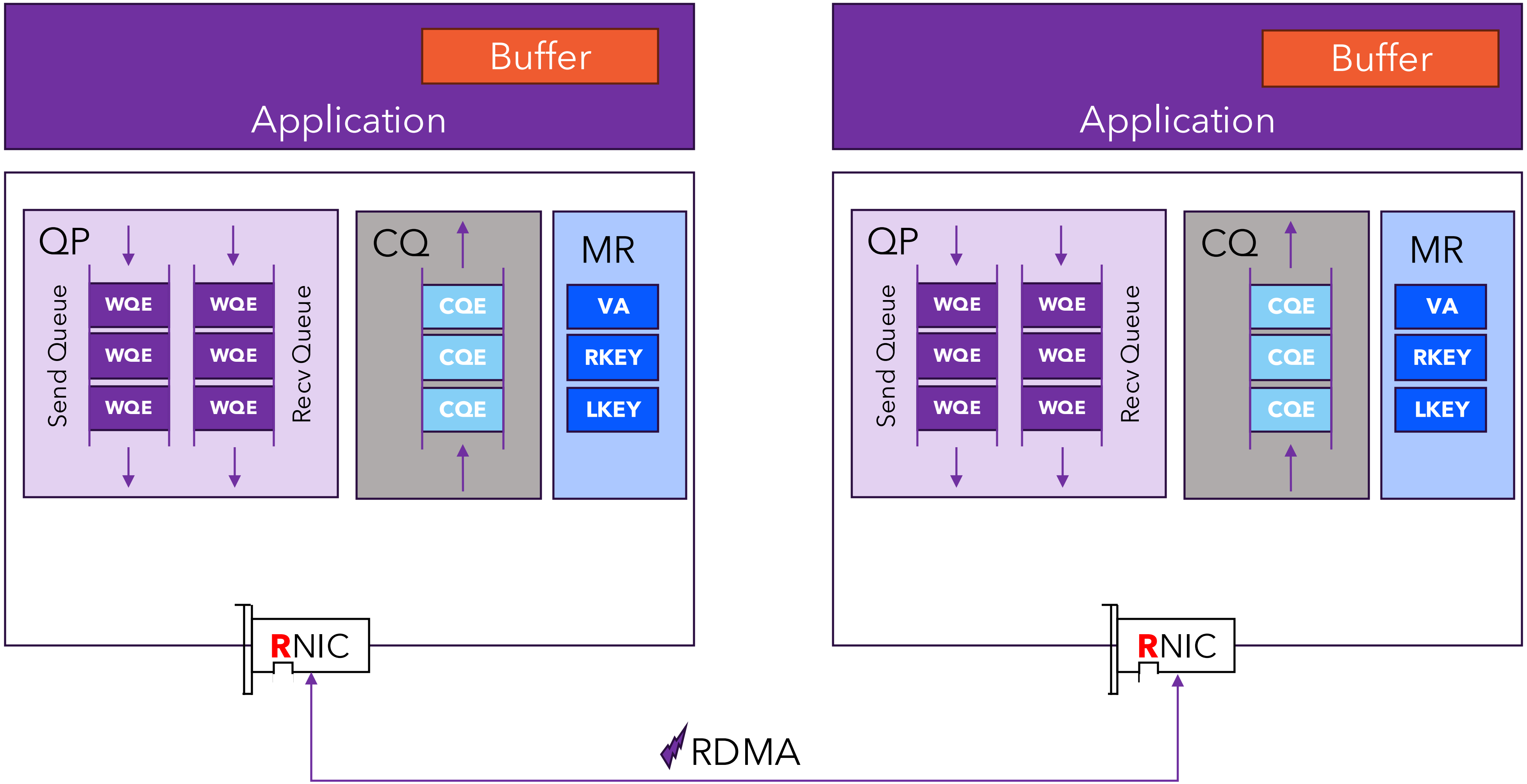
Read: Reads data from the remote memory into the local memory.

Write: Writes data directly to a specified location in the remote memory.

Atomic Operations: Performs atomic read-modify-write operations on remote memory.

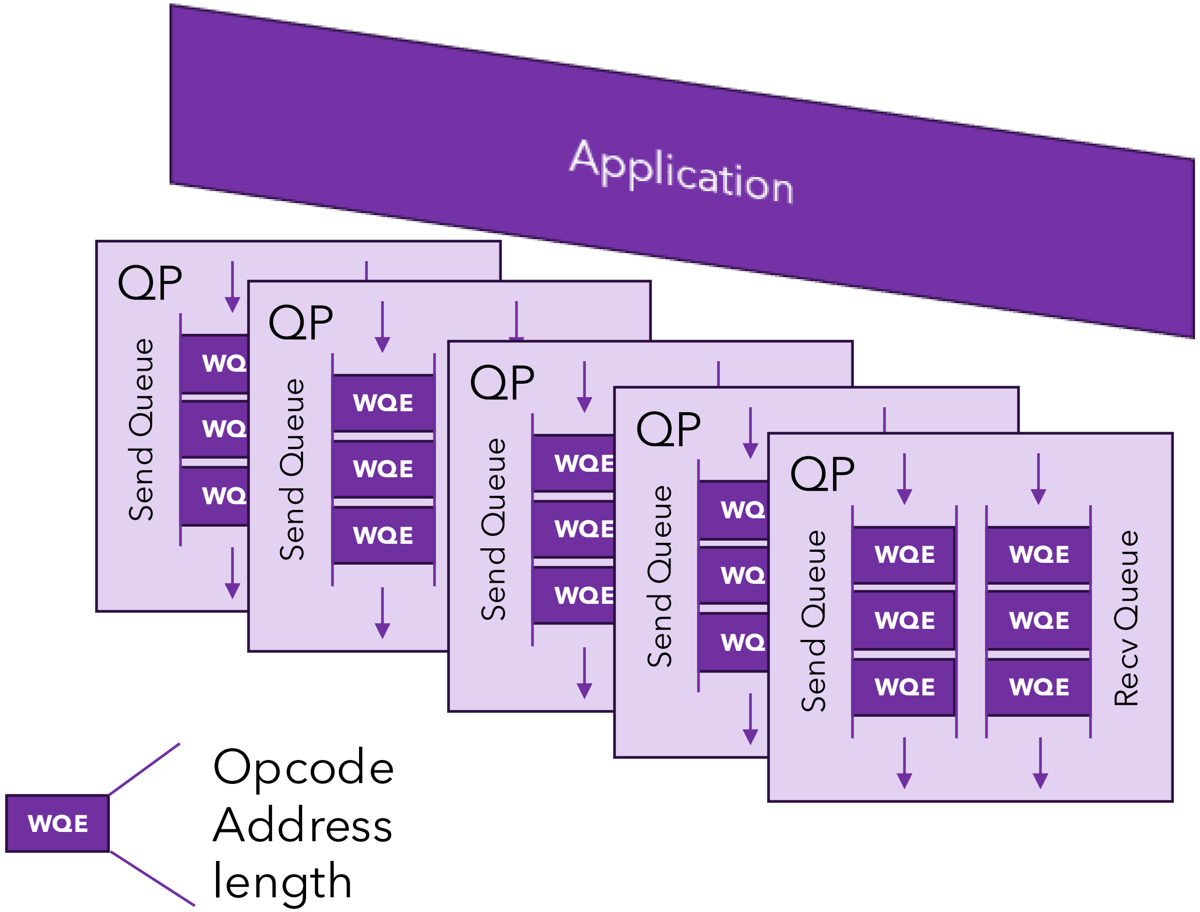
SNIA | DSN A SNIA Community

RDMA Main Objects



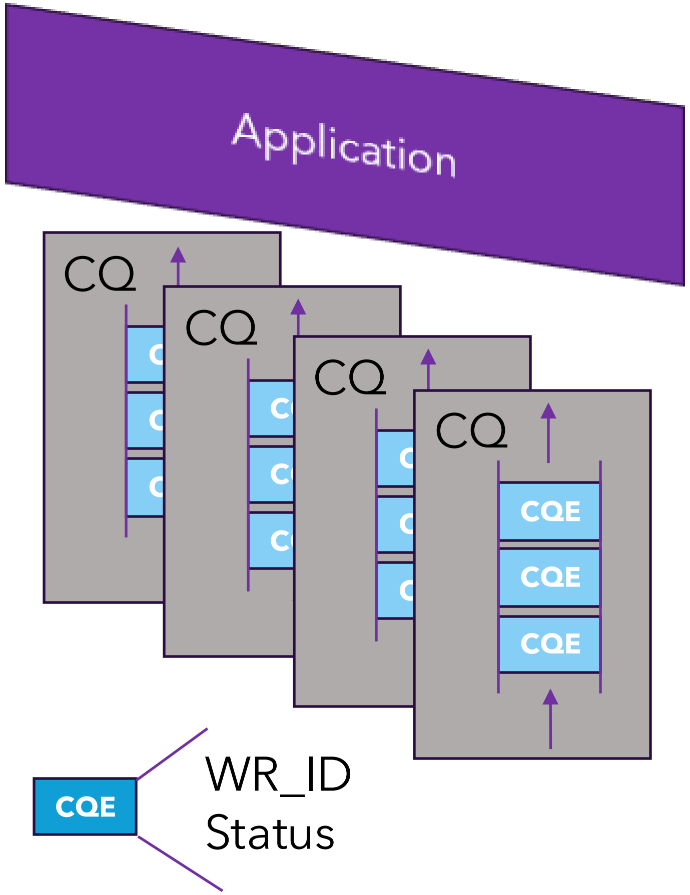
Queue-Pairs (QP)

- Used by consumer (application) to submit operations to the RNIC
- QP consists of
  - Send-Queue (SQ)
  - Receive-Queue (RQ)
- Each consumer can have multiple QPs
- WQEs – Work Queue Elements posted on the SQ / RQ



Completion Queues (CQs)

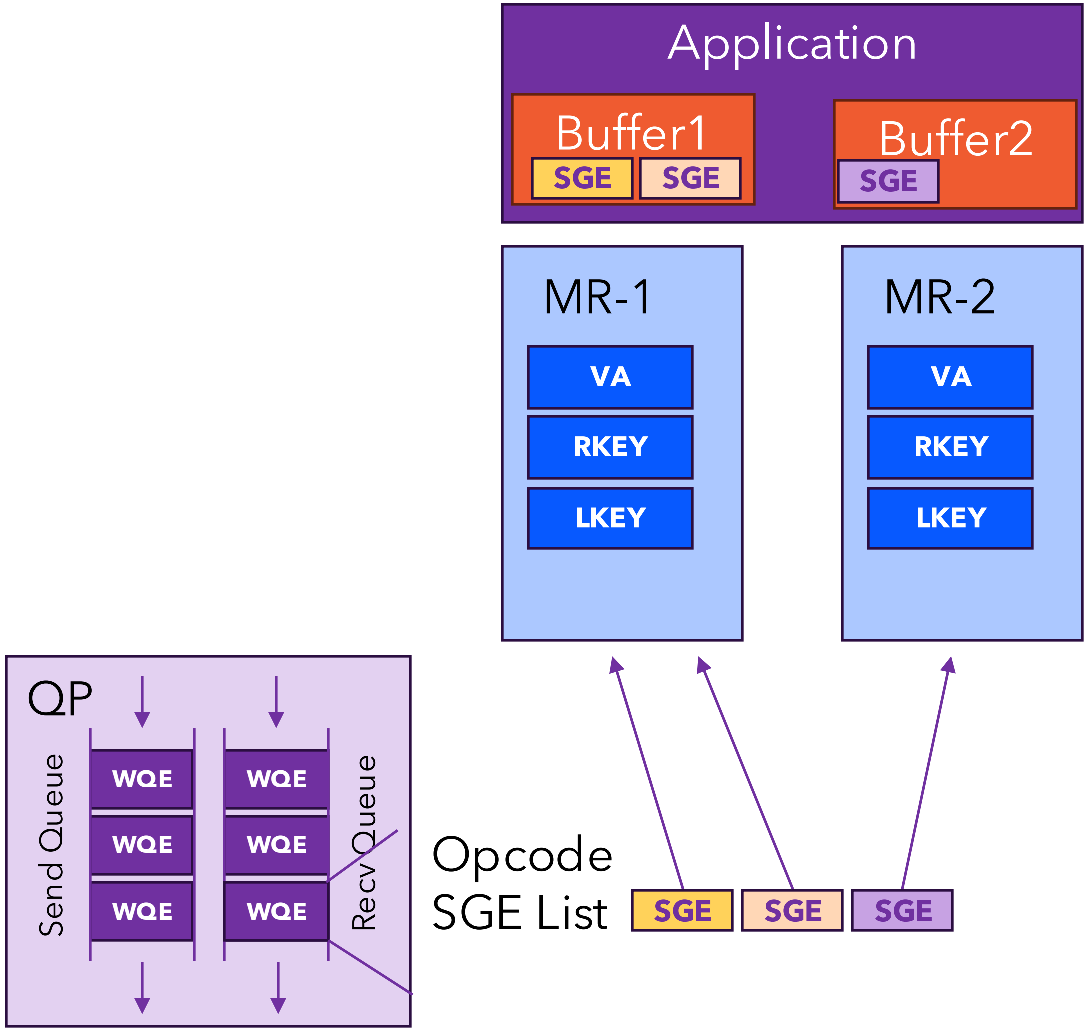
- CQs indicate completions for WQEs placed on SQ and RQ
- Consumer can have multiple CQ-s
- SQs and RQs are mapped to a CQ
- CQ can serve multiple SQs/RQs **NOT** a 1:1 mapping
- Two methods for processing CQs:
  - **Polling mode**: Low latency, high throughput but CPU intensive
  - **Interrupt-based mode**: Reduces CPU usage and power. Higher latency complexity



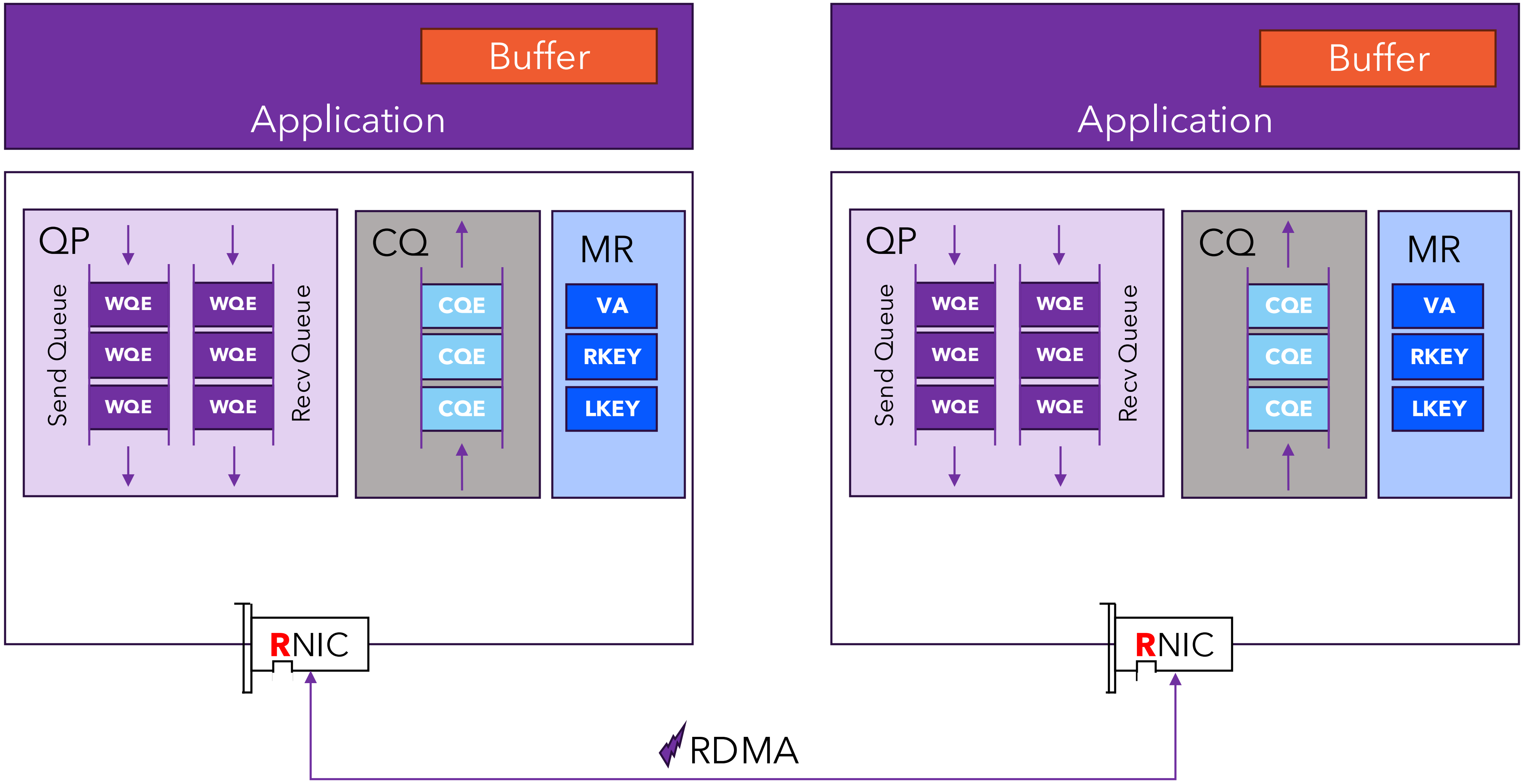
SNIA DSN A SNIA Community

Memory Region (MR)

- Application register region of memory – MR
- Allows RNIC to read/write from this memory
- Registration pins the memory location
- RNIC returns L-KEY and R-KEY
  - L-KEY – used by local APP
  - R-KEY – used by remote APP
- L_Key / R_Key, with the offset in the MR and length, identify the location from which to read / write.



Revisiting…

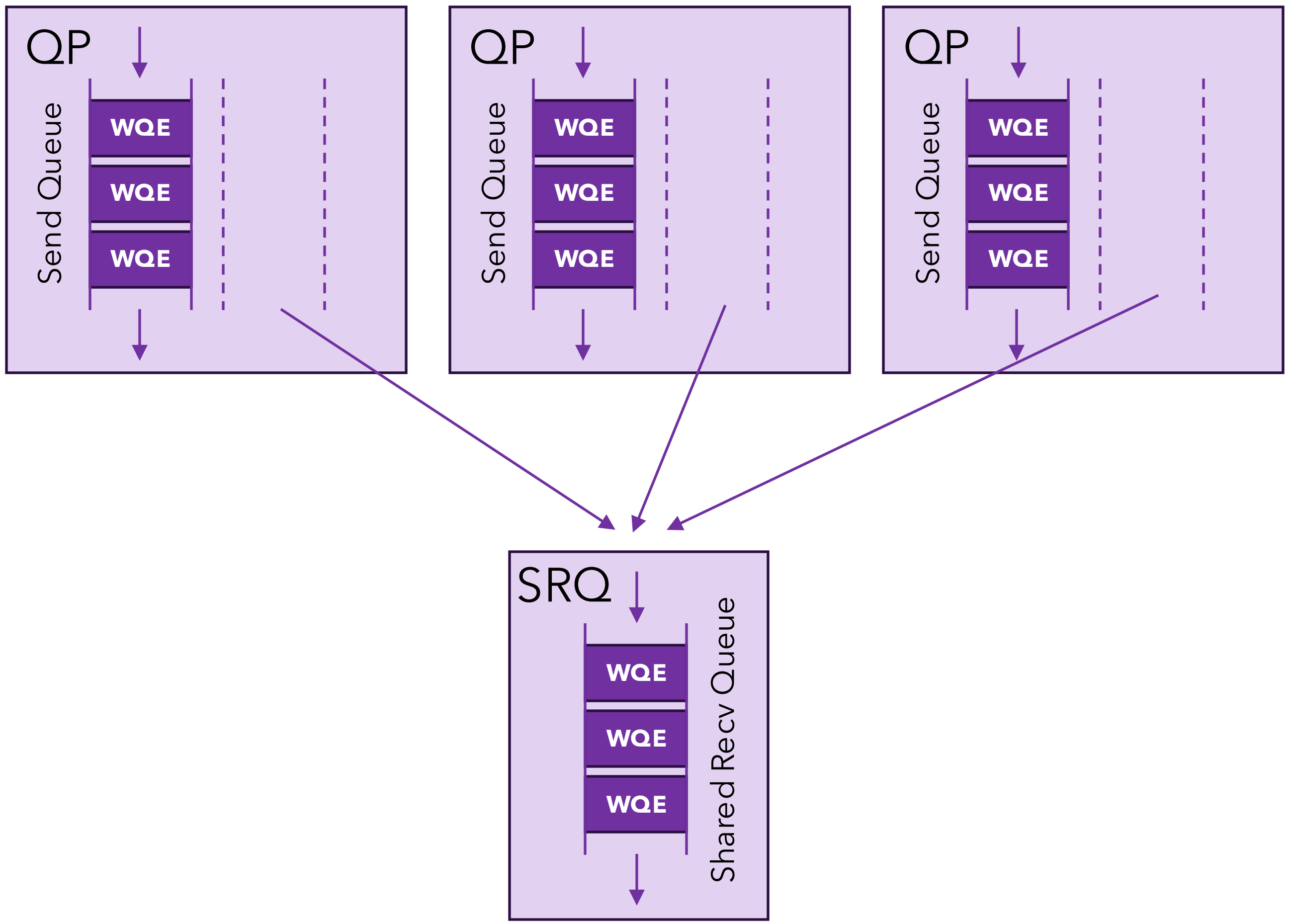

Buffer
Application
QP
Send Queue
WQE
WQE
WQE
WQE
WQE
WQE
Recv Queue
CQ
CQE
CQE
CQE
MR
VA
RKEY
LKEY
RNIC
Buffer
Application
QP
Send Queue
WQE
WQE
WQE
WQE
WQE
WQE
Recv Queue
CQ
CQE
CQE
CQE
MR
VA
RKEY
LKEY
RNIC
RDMA

Protection Domains (PD)

- Group objects that can work together
- Associated with QPs, MRs, SRQs MWs.
- RNIC validates that the MR has the same PD domain as the QP on which it is posted / received.
  - If not, there will be a completion with error

Shared Receive Queue (SRQ)

- Shared Receive Queue between QPs
- QP can be created with SRQ instead of RQ
- More efficient in memory consumption



Connection Management

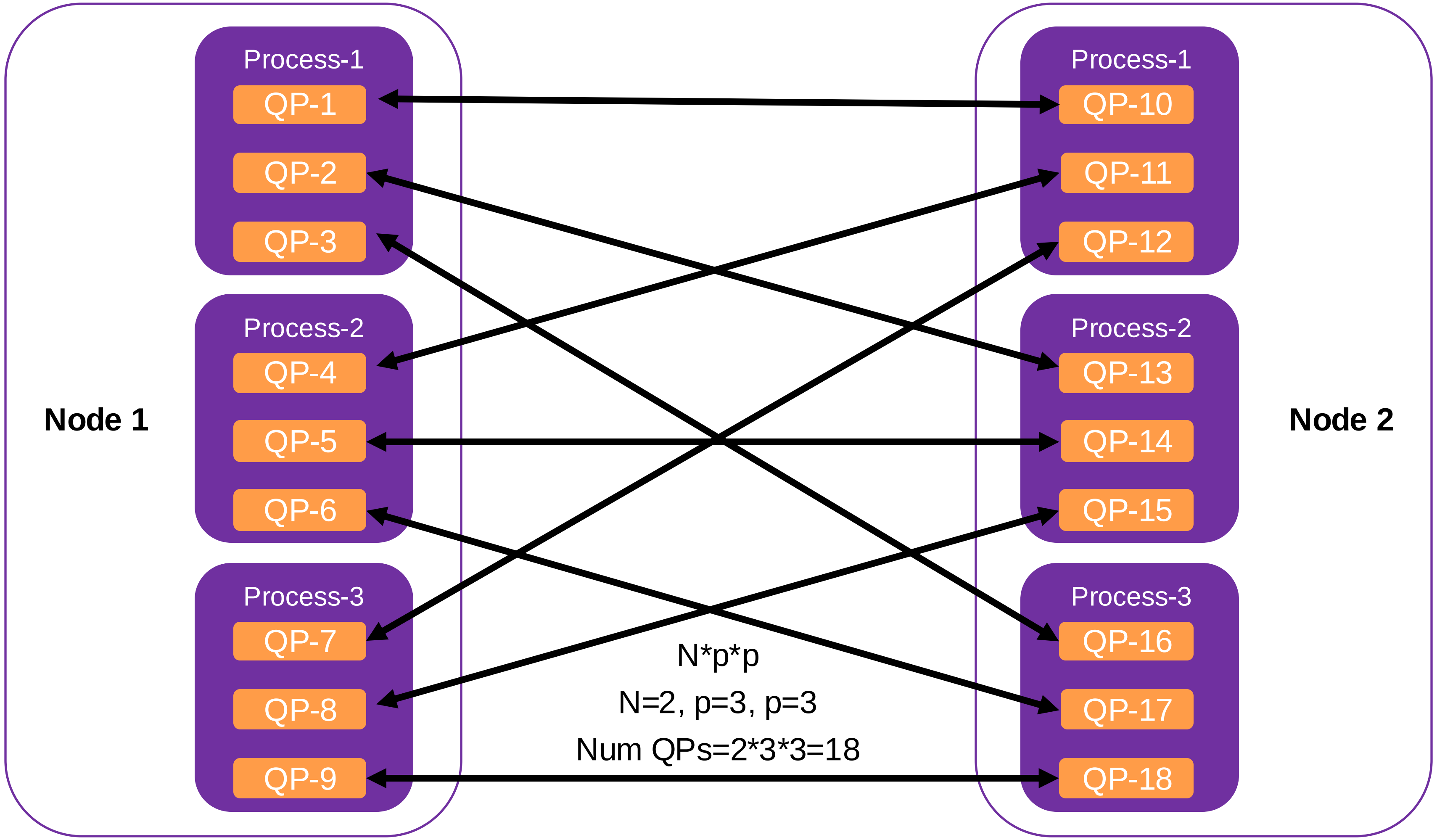
- Four types of QP-s are supported:
  - Reliable connection (RC) – mostly used (somewhat comparable to TCP)
  - Unreliable connection (UC)
  - Reliable datagram (RD)
  - Unreliable datagram (UD) (somewhat comparable to UDP)
- Connections are established out-of-band.
  - Each wire protocol uses a different scheme.
    - Infiniband defines protocol running over UD.
    - iWARP defines a protocol running over TCP.
    - Since it's out-of-band can be emulated over sockets

SNIA | DSN A SNIA Community

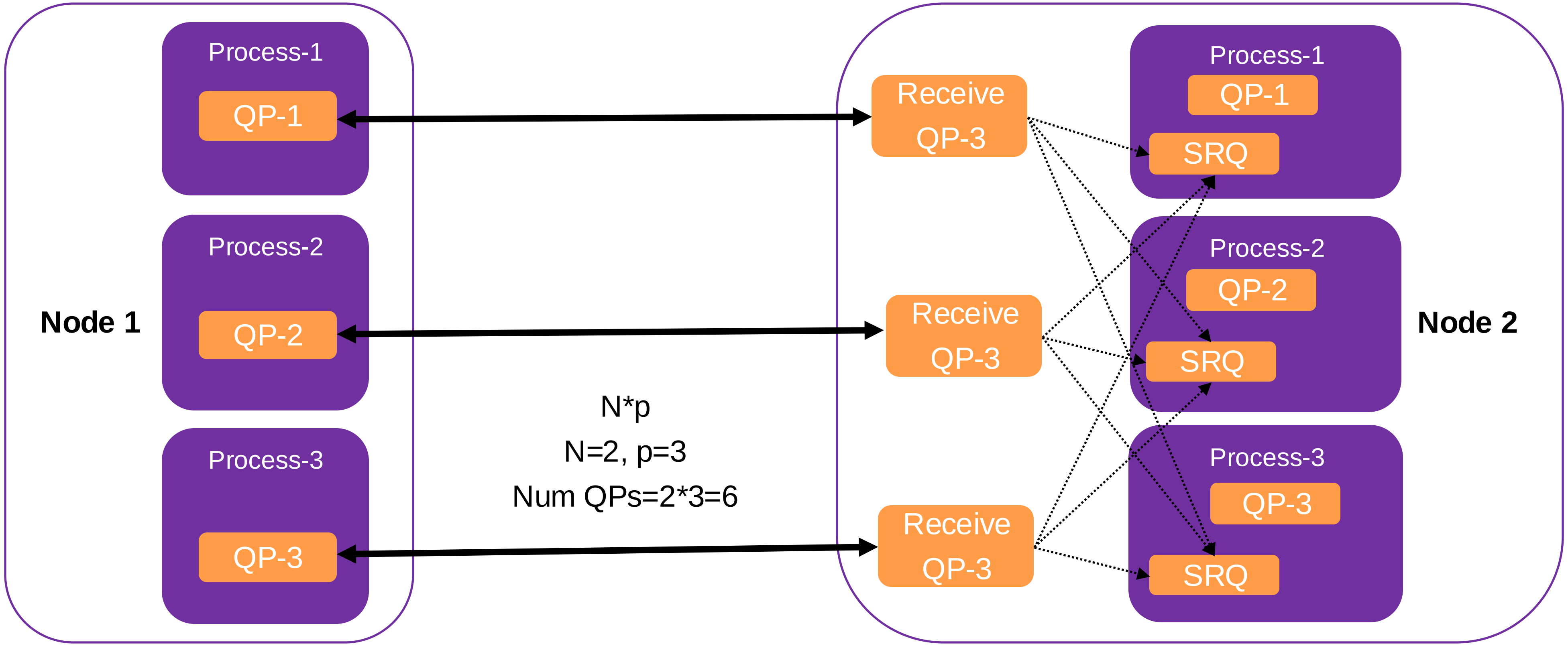
XRC – Extended Reliable Connection

- XRC allows significant savings in the number of QPs required to establish all to all process connectivity in large clusters.
- Single XRC Initiator QP a process in one node can communicate with ALL processes on one remote node, thus reducing by a factor of p the number of overall QPs required for full connectivity (as compared RC QPs)
- $QPs = N_{nodes} \times N_{processes}$
- Decrease from $N_{nodes} \times N_{processes}^{2}$

QP – RC Connection Example



QP – XRC Connection Example



Shown in one direction for simplicity, but the QPs on each side are symmetrical

RDMA Verbs

Verbs provide an abstract definition of the functionality provided to a host by a RDMA NIC.

An operating system may expose some or all of the verb functionality through its programming interface.

Same Verbs for all wire-protocols.

Slow-Path Verbs Operations

Slow-path operations involve privileged driver, Main verbs:

Device related	Open / close device
PD (Protection Domain)	Allocate / De-allocate
MR (Memory Region)	Register / de-register
QP (Queue Pair)	Create / destroy / modify
CQ (Completion Queue)	Create / resize / destroy
SRQ (Shared Queue Pair)	Create / resize / destroy

Fast-Path / Data-Path Verbs

Fast-path operations can be done from either user-space or a privileged driver.

Send operations	Send operation consumes a RQ entry in the peer All memory used in transfers **must be registered**
Receive operations	Post RQ receive user application must ensure a send is not posted on one side before a corresponding recv has been posted on the other side
RDMA operations	RDMA write, RDMA read Atomic operations (fetch and add, compare and swap)
Memory operations	Bind MW, Fast register MR Local invalidate

Putting it all together:
How does an app look?
What goes on the wire?

Typical App Stages Overview

HOSTS INITIALIZE CONTEXT AND REGISTER MEMORY REGIONS

ESTABLISH CONNECTION

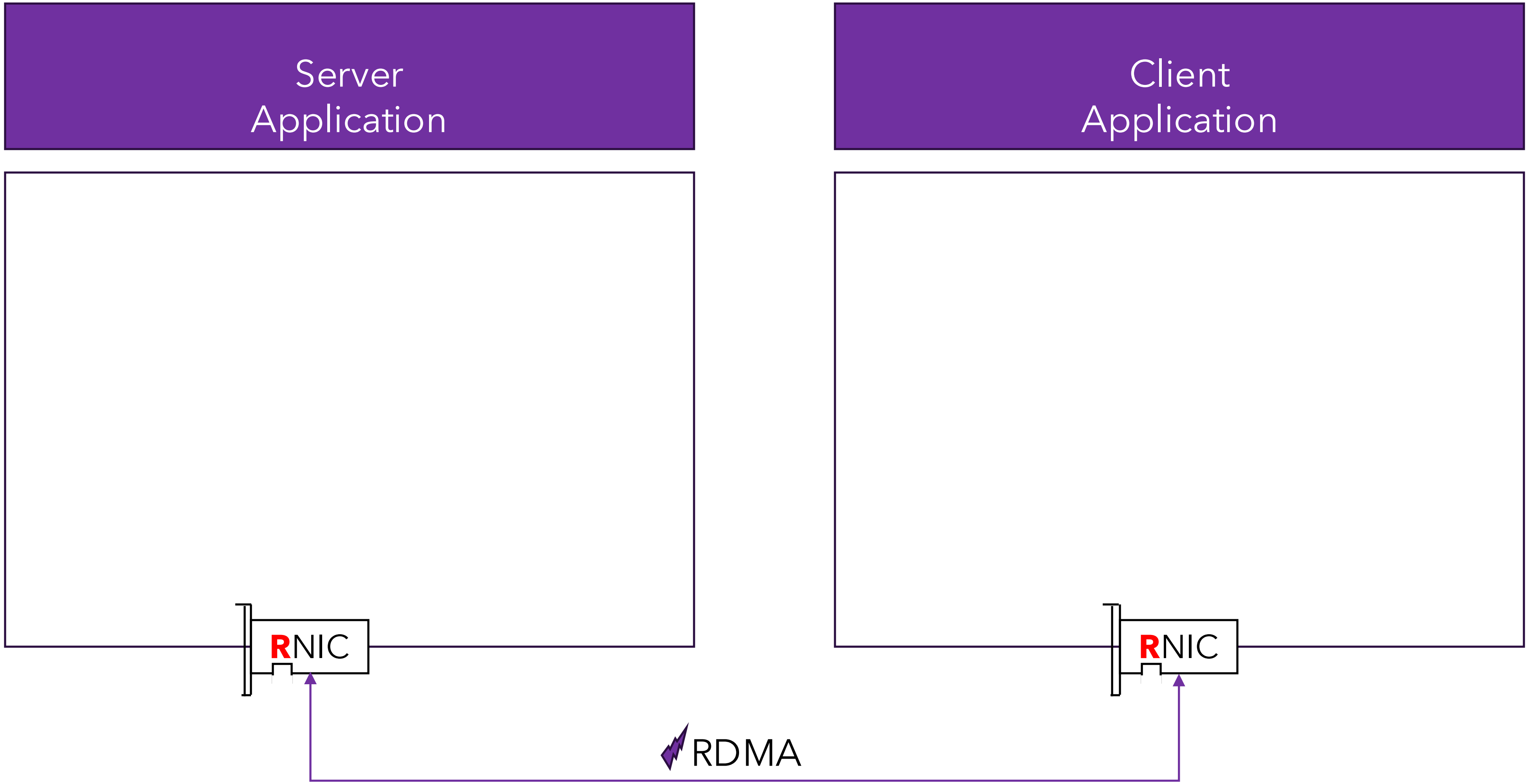
USE SEND/RECEIVE MODEL TO EXCHANGE MEMORY REGION KEYS BETWEEN PEERS

POST READ/WRITE OPERATIONS

DISCONNECT

Initialization

HOSTS INITIALIZE CONTEXT AND REGISTER MEMORY REGIONS



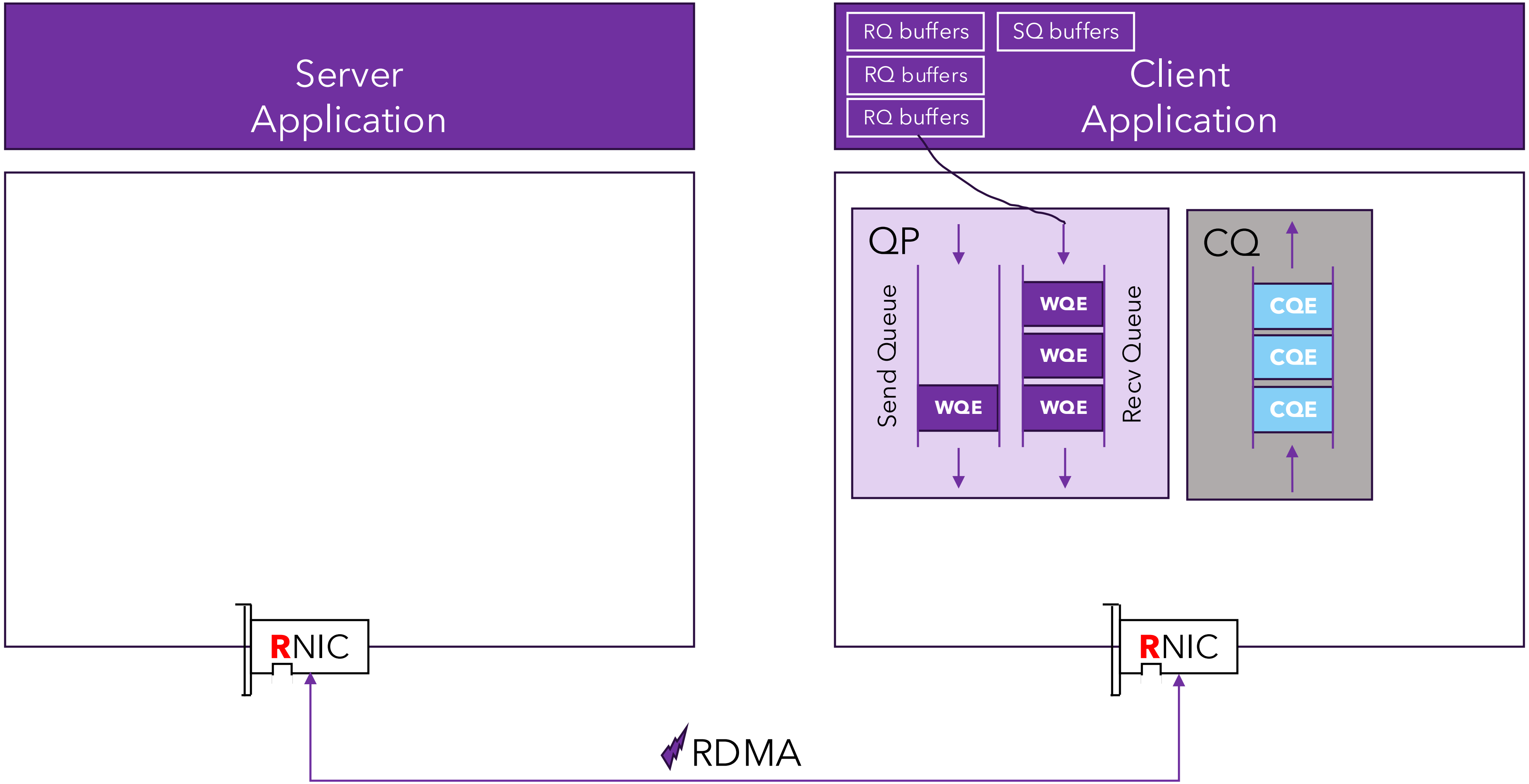
Initialization

HOSTS INITIALIZE CONTEXT AND REGISTER MEMORY REGIONS

Target (Server)	Initiator (Client)
Create an event channel to receive rdmacm events connection-request connection-established notifications	Create an event channel to receive rdmacm events address-resolved route-resolved connection-established notifications
Bind to an address	Create a connection identifier
Create a listener and return the port/address	Resolve the peer's address, which binds the connection identifier to a local RDMA device
Wait for a connection request	Resolve the route to the peer
	Create a PD, CQ, QP

Initialization

HOSTS INITIALIZE CONTEXT AND REGISTER MEMORY REGIONS

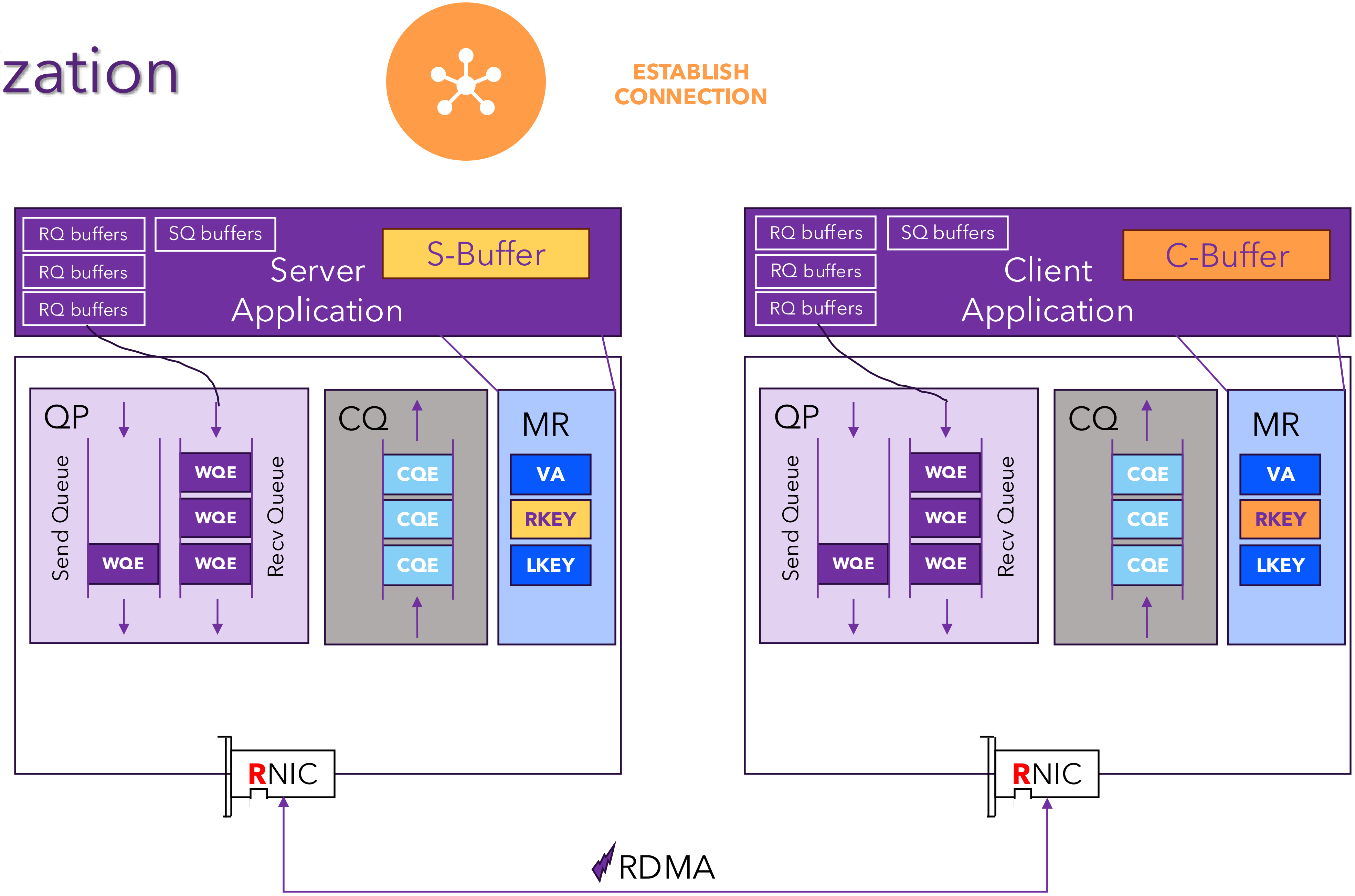


Initialization

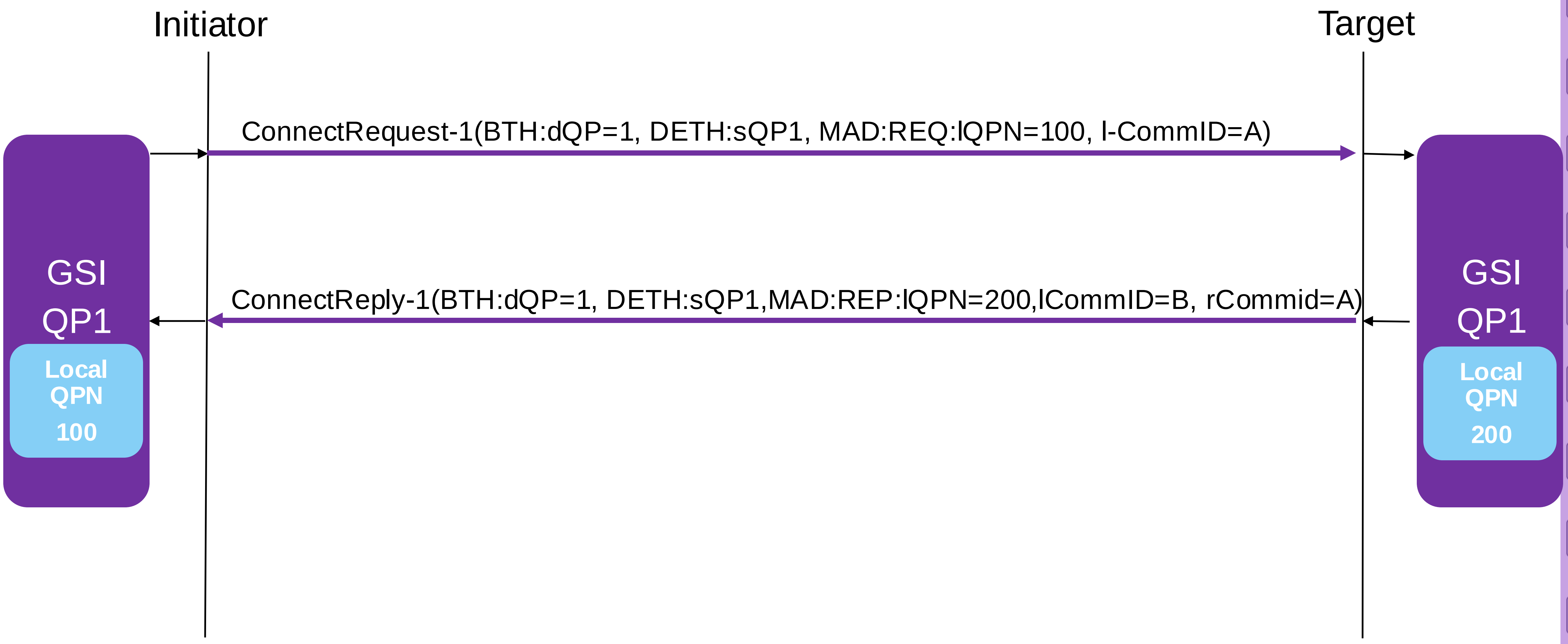
ESTABLISH CONNECTION

SERVER (Passive)	Client (Active)
Create an event channel to receive rdmacm events connection-request connection-established notifications	Create an event channel to receive rdmacm events address-resolved route-resolved connection-established notifications
Bind to an address	Create a connection identifier
Create a listener and return the port/address	Resolve the peer's address, which binds the connection identifier to a local RDMA device
Wait for a connection request	Create a PD, CQ, QP
Create a PD, CQ, QP	Resolve the route to the peer
Accept the connection request	Connect
Wait for connection to be established	Wait for connection to be established
Post operations as appropriate	Post operations as appropriate

Initialization

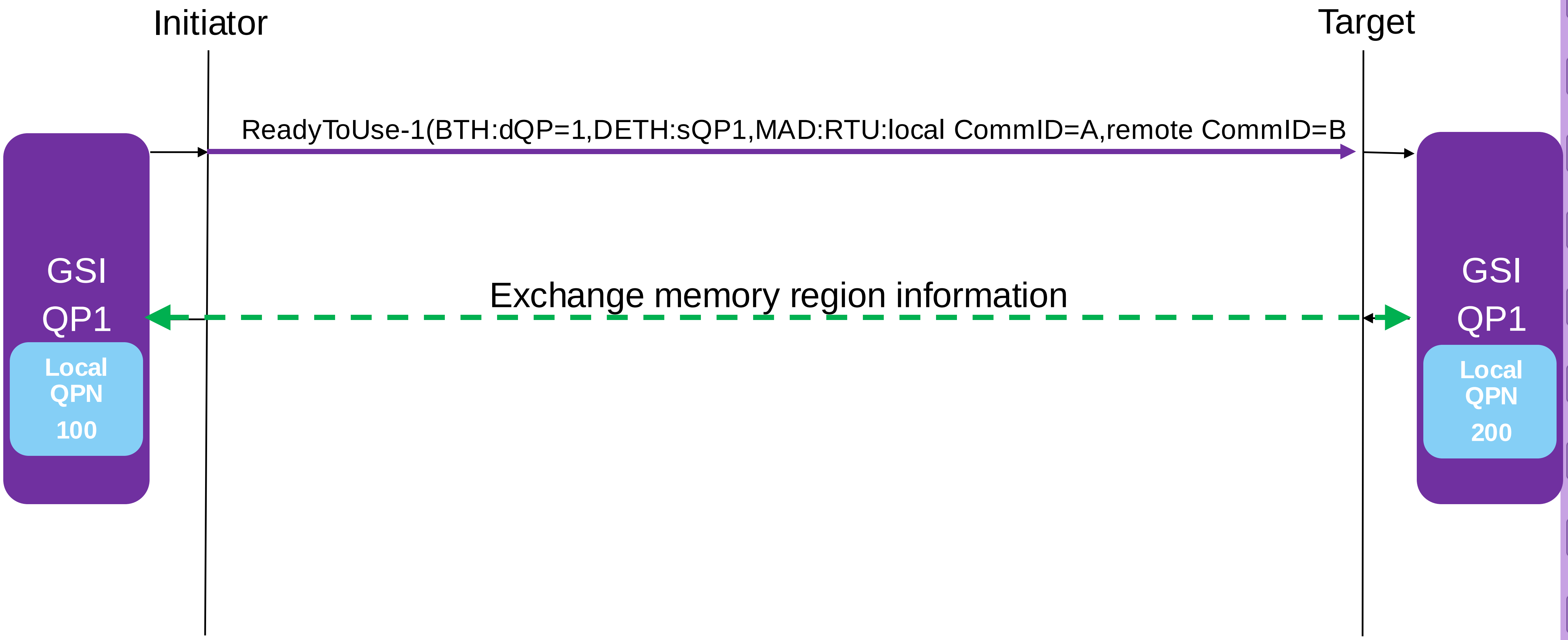


Connection Setup



Example IB Initialization & Connection Setup-1-Create “Admin” Connection & QP

Connection Setup - Ready-to-Use



Example IB Initialization & Connection Setup-2-Send Ready To Use & Exchange Info

On the Wire…

27 4.191361	50.50.50.1	50.50.50.2	RRoCE	322 CM: ConnectRequest
28 4.193920	50.50.50.2	50.50.50.1	RRoCE	322 CM: ConnectReply
29 4.196004	50.50.50.1	50.50.50.2	RRoCE	322 CM: ReadyToUse

ConnectRequest

```
> Internet Protocol Version 4, Src: 50.50.50.1, Dst: 50.50.50.2
> User Datagram Protocol, Src Port: 60036, Dst Port: 4791
v InfiniBand
  > Base Transport Header
  > DETH - Datagram Extended Transport Header
  > MAD Header - Common Management Datagram
  > CM ConnectRequest
    Invariant CRC: 0x525d5604
```

ConnectReply

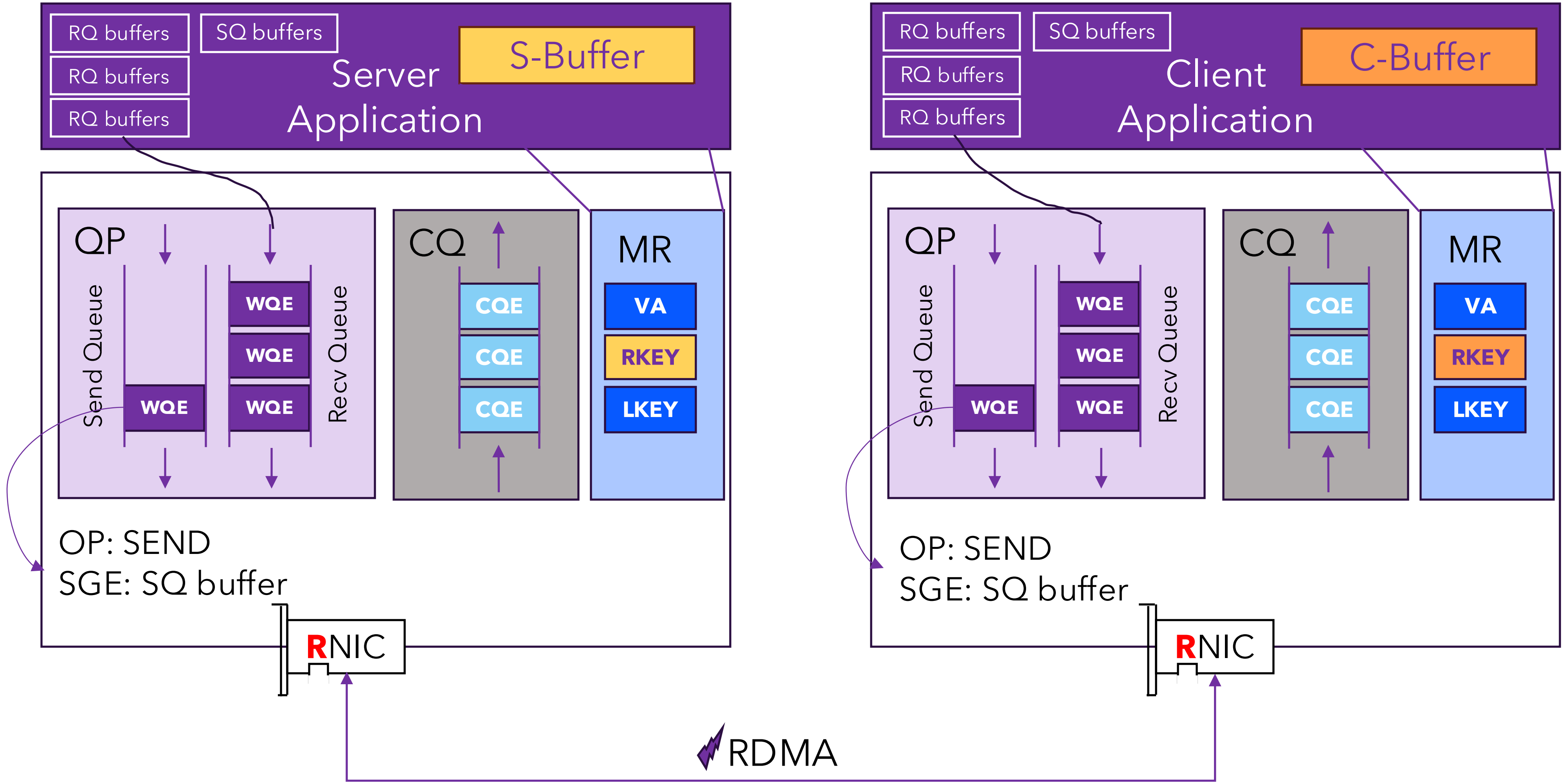
```
> Internet Protocol Version 4, Src: 50.50.50.2, Dst: 50.50.50.1
> User Datagram Protocol, Src Port: 60036, Dst Port: 4791
v InfiniBand
  > Base Transport Header
  > DETH - Datagram Extended Transport Header
  > MAD Header - Common Management Datagram
  > CM ConnectReply
    Invariant CRC: 0xea633b35
```

ReadyToUse

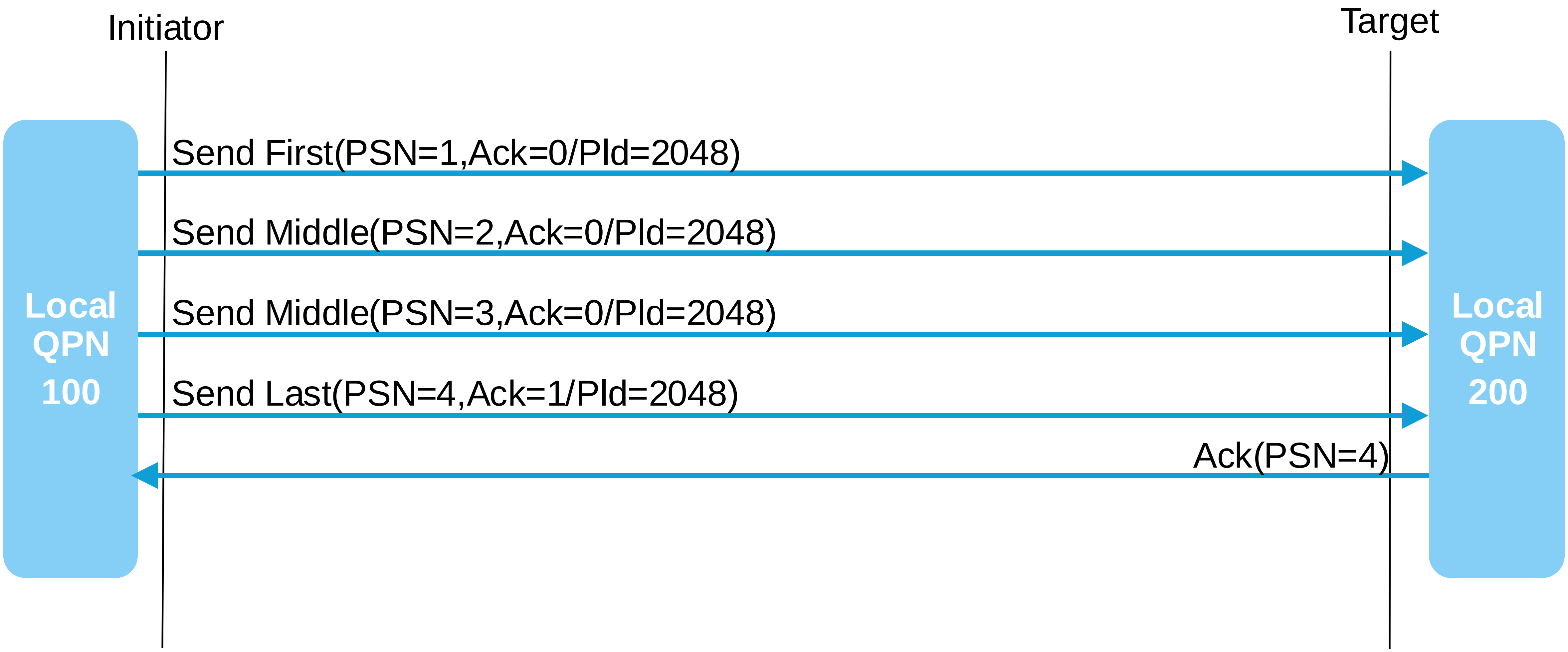
```
> Internet Protocol Version 4, Src: 50.50.50.1, Dst: 50.50.50.2
> User Datagram Protocol, Src Port: 60036, Dst Port: 4791
v InfiniBand
  > Base Transport Header
  > DETH - Datagram Extended Transport Header
  > MAD Header - Common Management Datagram
  > CM ReadyToUse
    Invariant CRC: 0x6e7c38ff
```


Send / Recv Model

USE SEND/RECEIVE MODEL TO EXCHANGE MEMORY REGION KEYS BETWEEN PEERS



Example Send – 8k (MTU=2K)

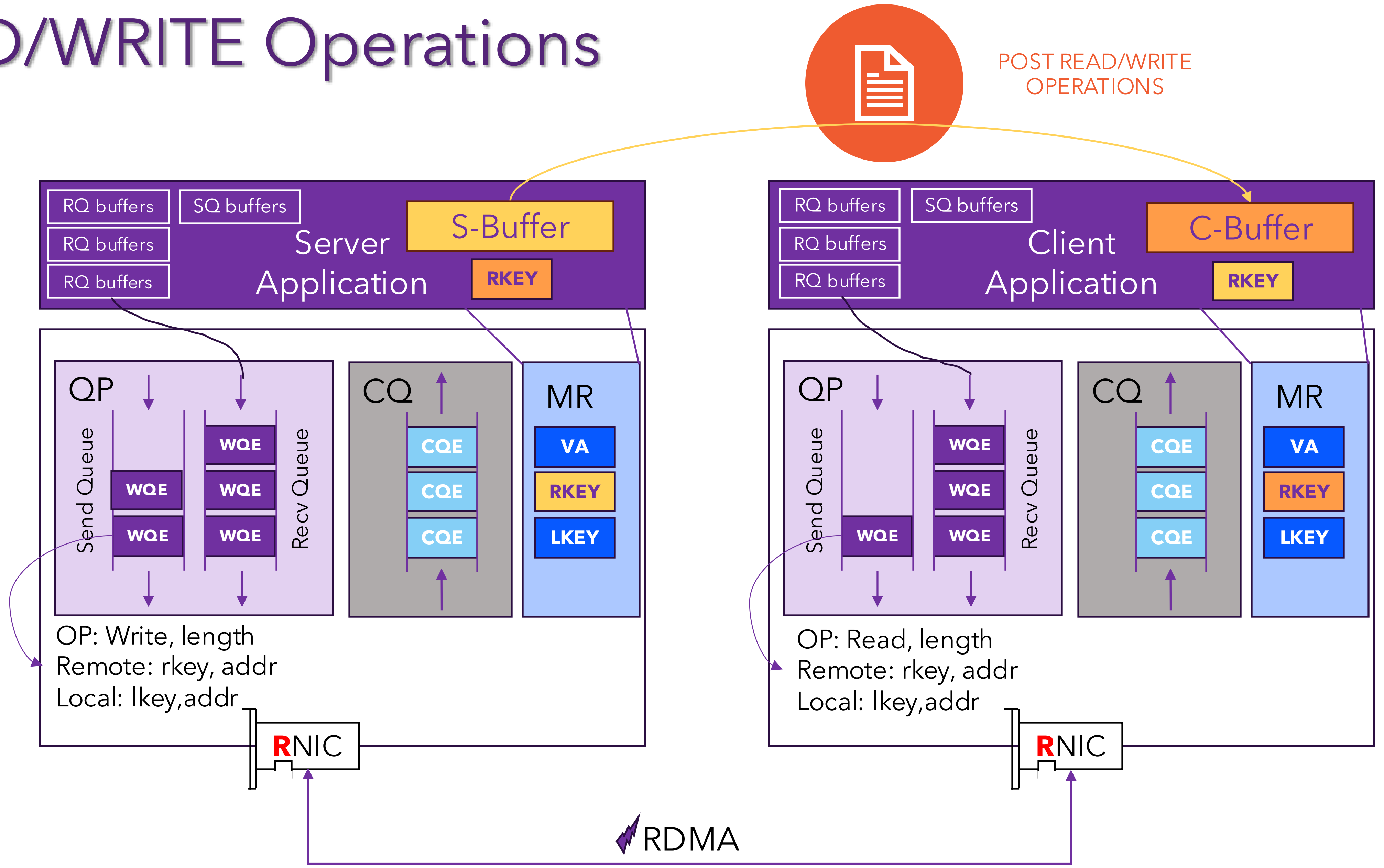


On the Wire Send – 8K (MTU=1K)

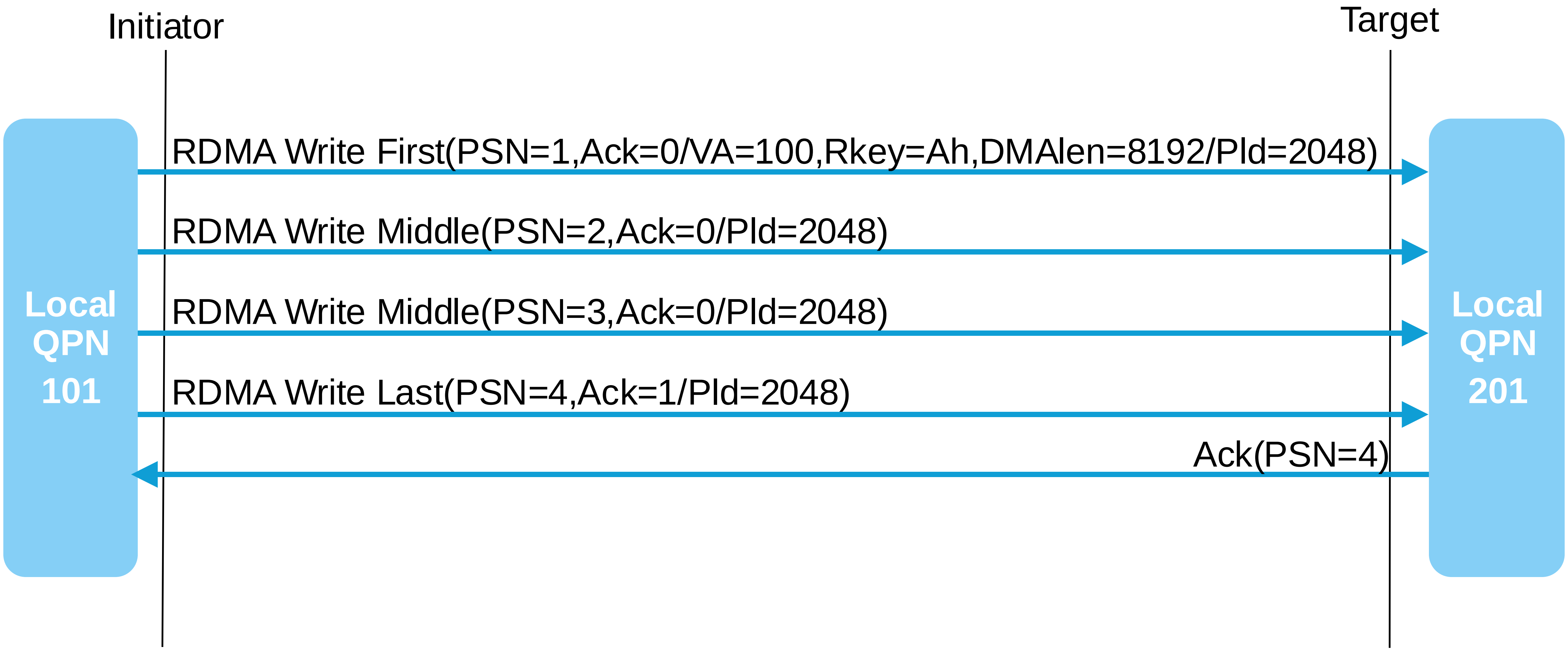
```
> Internet Protocol Version 4, Src: 50.50.50.1, Dst: 50.50.50.2
v User Datagram Protocol, Src Port: 60036, Dst Port: 4791
    Source Port: 60036
    Destination Port: 4791
    Length: 1048
  > Checksum: 0x0000 [zero-value ignored]
    [Stream index: 2]
    [Stream Packet Number: 3]
  > [Timestamps]
    UDP payload (1040 bytes)
v InfiniBand
  v Base Transport Header
      Opcode: Reliable Connection (RC) - SEND First (0)
      0... .... = Solicited Event: False
      .1.. .... = MigReq: True
      ..00 .... = Pad Count: 0
      .... 0000 = Header Version: 0
      Partition Key: 65535
      Reserved: 00
      Destination Queue Pair: 0x0001b4
      1... .... = Acknowledge Request: True
      .000 0000 = Reserved (7 bits): 0
      Packet Sequence Number: 10255594
    Invariant CRC: 0xed5a3375
  [Reassembled PDU in frame: 57]
> Data (1024 bytes)
```

50	4.462568	50.50.50.1	50.50.50.2	RRoCE	1082	RC Send First QP=0x0001b4
51	4.462569	50.50.50.1	50.50.50.2	RRoCE	1082	RC Send Middle QP=0x0001b4
52	4.462570	50.50.50.1	50.50.50.2	RRoCE	1082	RC Send Middle QP=0x0001b4
53	4.462571	50.50.50.1	50.50.50.2	RRoCE	1082	RC Send Middle QP=0x0001b4
54	4.462572	50.50.50.1	50.50.50.2	RRoCE	1082	RC Send Middle QP=0x0001b4
55	4.462572	50.50.50.1	50.50.50.2	RRoCE	1082	RC Send Middle QP=0x0001b4
56	4.462573	50.50.50.1	50.50.50.2	RRoCE	1082	RC Send Middle QP=0x0001b4
57	4.462574	50.50.50.1	50.50.50.2	RRoCE	1082	RC Send Last QP=0x0001b4
58	4.462577	50.50.50.2	50.50.50.1	RRoCE	62	RC Acknowledge QP=0x000063
59	4.462578	50.50.50.2	50.50.50.1	RRoCE	62	RC Acknowledge QP=0x000063

READ/WRITE Operations



Example RDMA Write – 8k (MTU=2K)



On the Wire RDMA Write (MTU=1K)

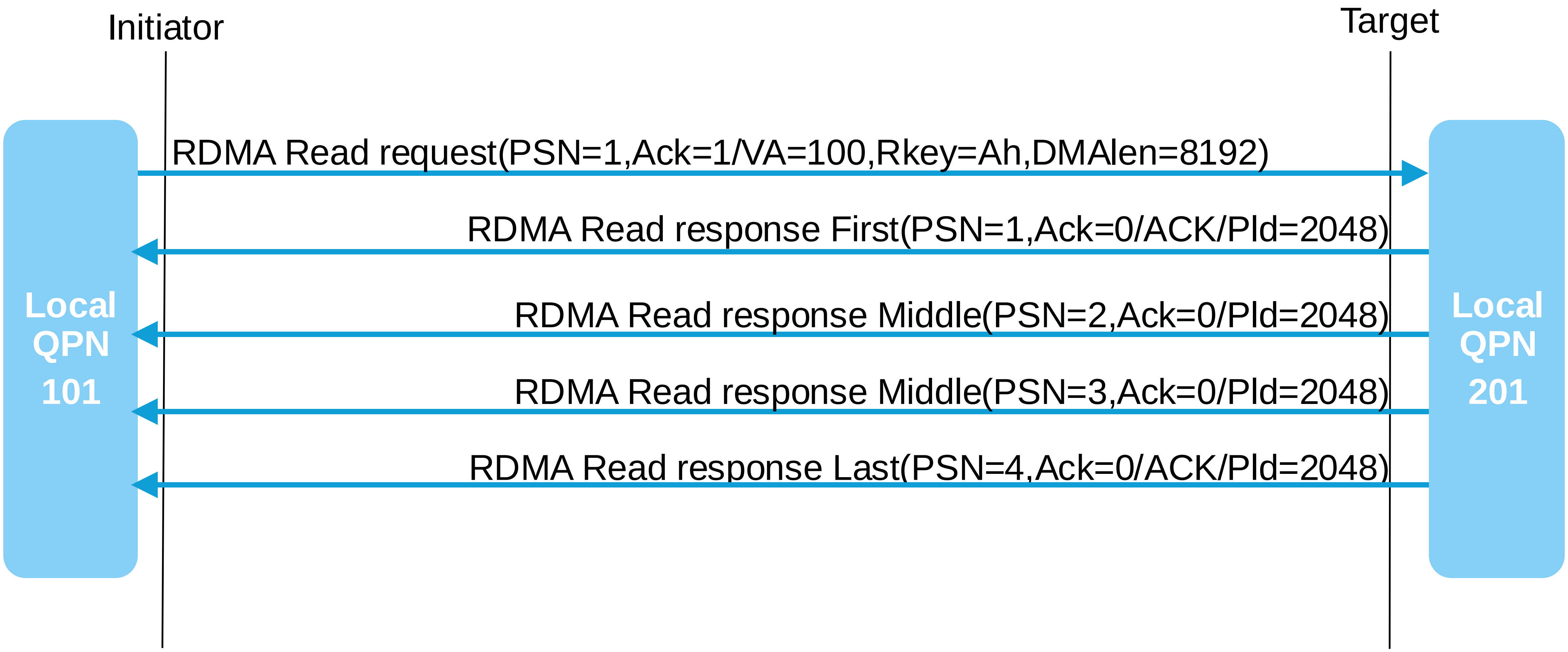
```
InfiniBand
  Base Transport Header
    Opcode: Reliable Connection (RC) - RDMA WRITE First (6)
    0... .... = Solicited Event: False
    .1.. .... = MigReq: True
    ..00 .... = Pad Count: 0
    .... 0000 = Header Version: 0
    Partition Key: 65535
    Reserved: 00
    Destination Queue Pair: 0x000065
    0... .... = Acknowledge Request: False
    .000 0000 = Reserved (7 bits): 0
    Packet Sequence Number: 8939302
  RETH - RDMA Extended Transport Header
    Virtual Address: 0x000055eabc942000
    Remote Key: 0x00004705
    DMA Length: 8192 (0x00002000)
  Invariant CRC: 0x57f30169
Data (1024 bytes)
```

```
InfiniBand
  Base Transport Header
    Opcode: Reliable Connection (RC) - RDMA WRITE Middle (7)
    0... .... = Solicited Event: False
    .1.. .... = MigReq: True
    ..00 .... = Pad Count: 0
    .... 0000 = Header Version: 0
    Partition Key: 65535
    Reserved: 00
    Destination Queue Pair: 0x000065
    0... .... = Acknowledge Request: False
    .000 0000 = Reserved (7 bits): 0
    Packet Sequence Number: 8939303
  Invariant CRC: 0x4a966453
Data (1024 bytes)
```

x6

```
InfiniBand
  Base Transport Header
    Opcode: Reliable Connection (RC) - RDMA WRITE Last (8)
    0... .... = Solicited Event: False
    .1.. .... = MigReq: True
    ..00 .... = Pad Count: 0
    .... 0000 = Header Version: 0
    Partition Key: 65535
    Reserved: 00
    Destination Queue Pair: 0x000065
    1... .... = Acknowledge Request: True
    .000 0000 = Reserved (7 bits): 0
    Packet Sequence Number: 8939309
  Invariant CRC: 0xe68e300e
Data (1024 bytes)
```


Example RDMA Read – 8k (MTU=2K)



On the Wire Read (MTU=1K)

```
v InfiniBand
  v Base Transport Header
      Opcode: Reliable Connection (RC) - RDMA READ Request (12)
      0... .... = Solicited Event: False
      .1.. .... = MigReq: True
      ..00 .... = Pad Count: 0
      .... 0000 = Header Version: 0
      Partition Key: 65535
      Reserved: 00
      Destination Queue Pair: 0x0001b8
      1... .... = Acknowledge Request: True
      .000 0000 = Reserved (7 bits): 0
      Packet Sequence Number: 6681459
  v RETH - RDMA Extended Transport Header
      Virtual Address: 0x0000559fcc462000
      Remote Key: 0x001824fe
      DMA Length: 8192 (0x00002000)
    Invariant CRC: 0x9e2137d4
```

48	1.915354	50.50.50.1	50.50.50.2	RRoCE	74 RC RDMA Read Request QP=0x0001b8
49	1.915377	50.50.50.2	50.50.50.1	RRoCE	1086 RC RDMA Read Response First QP=0x000067
50	1.915378	50.50.50.2	50.50.50.1	RRoCE	1082 RC RDMA Read Response Middle QP=0x000067
51	1.915379	50.50.50.2	50.50.50.1	RRoCE	1082 RC RDMA Read Response Middle QP=0x000067
52	1.915379	50.50.50.2	50.50.50.1	RRoCE	1082 RC RDMA Read Response Middle QP=0x000067
53	1.915380	50.50.50.2	50.50.50.1	RRoCE	1082 RC RDMA Read Response Middle QP=0x000067
54	1.915381	50.50.50.2	50.50.50.1	RRoCE	1082 RC RDMA Read Response Middle QP=0x000067
55	1.915382	50.50.50.2	50.50.50.1	RRoCE	1082 RC RDMA Read Response Middle QP=0x000067
56	1.915383	50.50.50.2	50.50.50.1	RRoCE	1086 RC RDMA Read Response Last QP=0x000067

Transports and Congestion Control

RDMA Transports

Application

Verbs

InfiniBand	RoCEv1	RoCEv2	iWARP
IB Transport	IB Transport	IB Transport	iWARP
IB Network	IB Network	UDP/IP	TCP/IP
IB Link	Eth Link	Eth Link	Eth Link

InfiniBand	LRH	GRH	BTH+	RDMA Data	iCRC	vCRC	
RoCEv2	Eth	IP	UDP	BTH+	RDMA Data	iCRC	FCS
iWARP	Eth	IP	TCP	iWARP* (MPA,DDP)	RDMA Data	FCS	

SNIA | DSN A SNIA Community

RoCE – RDMA over Converged Ethernet

Upper layers are the same as in Infiniband	link and physical layers are replaced with Ethernet. EtherType indicates RoCE (0x8915).
RoCE versions	RoCEv1 doesn't contain an IP header, therefore it is not routable. RoCEv2 over UDP (a.k.a "routable RoCE")
Ethernet subnet management means are used.	Uses the ARP (Address Resolution Protocol) to get remote MAC address. Requires a network interface on the same port.
Requires lossless operation – i.e. PAUSE / PFC.	Same as InfiniBand, but disadvantage comparing to iWARP.

SNIA | DSN A SNIA Community

Congestion Control

PFC (Priority Flow Control) pause frames are used to signal congestion to the source and throttle.

DCQCN (Data Center Quantized Congestion Notification) employs ECN (Explicit Congestion Notification) for signaling congestion feedback. Preferred for Storage workloads.

RoCC (Robust Congestion Control) utilizes switch queue size for fair data rate signaling.

Shaped-Quota is receiver-driven, optimizing bandwidth allocation. It abandons use of PFC.

RTTCC (Round Trip Time Congestion Control) uses RTT as a feedback signal in hardware for congestion control. Preferred for HPC and AI workloads.

Use Cases

Use Cases

High-Performance Computing (HPC)

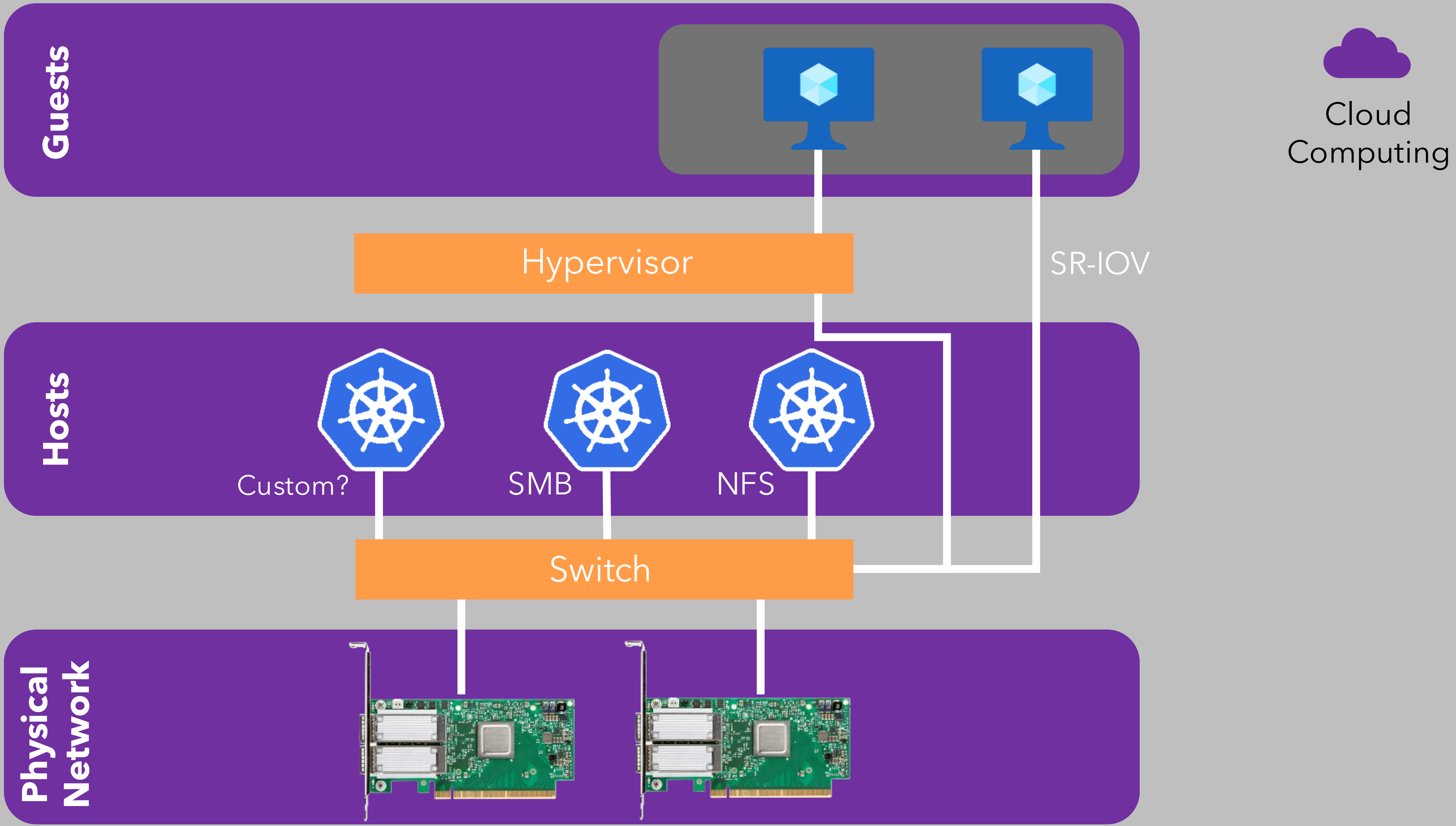
Data Centers

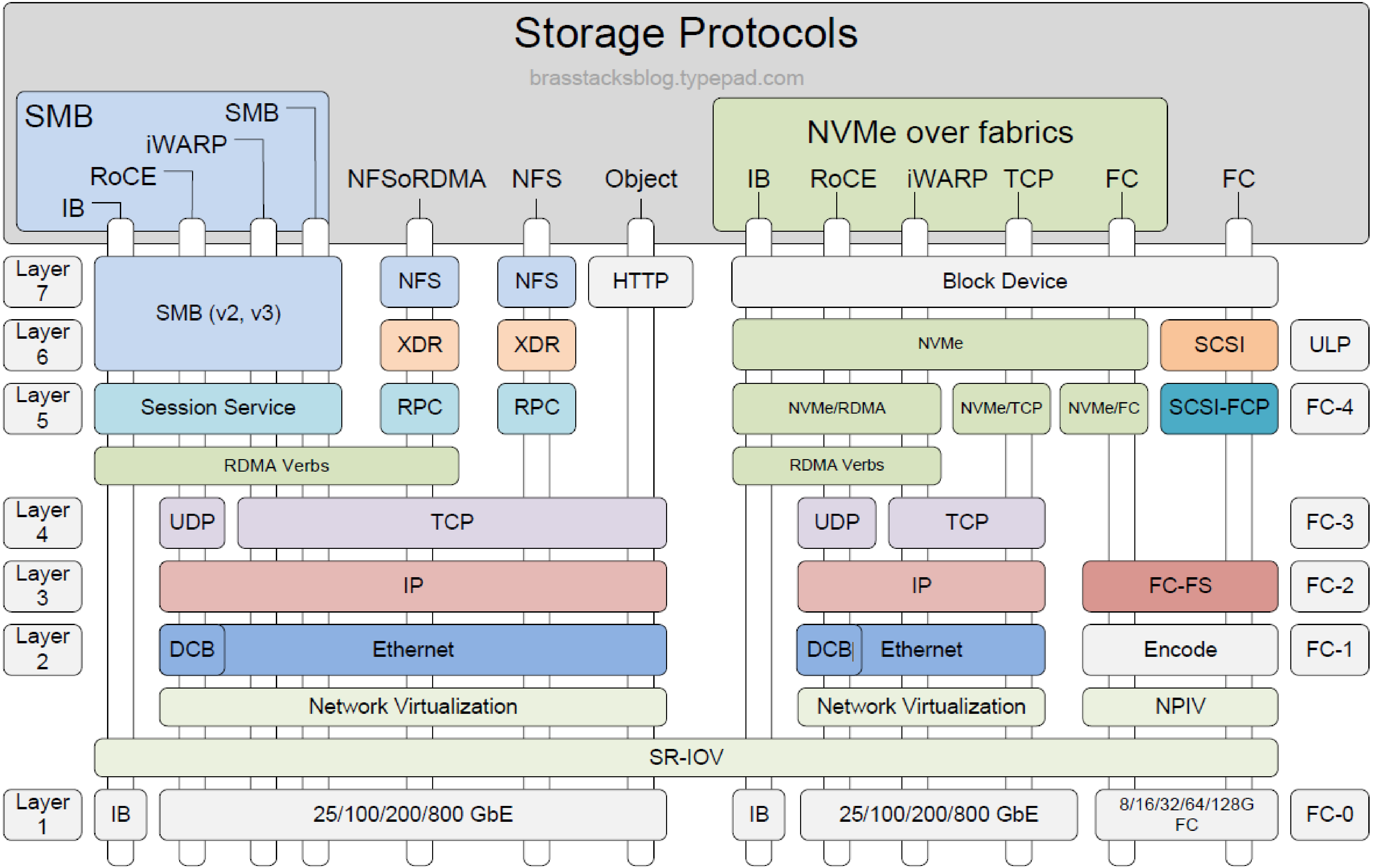
Cloud Computing

Storage Solutions

Financial Services

AI/ML Training / Inference


Guests
Hypervisor
SR-IOV
Hosts
Custom?
SMB
NFS
Switch
Physical Network
Cloud Computing



Storage Solutions

Key Takeaways

- RDMA Benefits: Low latency, high throughput, Kernel bypass, Zero Copy
- RDMA Objects
- Typical Application
- RDMA on the wire
- RDMA Use Cases

Q&A

After this Webinar

- Please rate this webinar and provide us with your feedback
- This webinar and a copy of the slides are available at the SNIA Educational Library snia.org/educational-library
- A Q&A from this webinar, including answers to questions we couldn't get to today, will be posted on our blog at sniablog.org
- Follow us on X/Twitter @SNIA

SNIA | Data, Storage & Networking

Thank you!