YA DRO

РОССИЙСКИЕ
ИНФОРМАЦИОННЫЕ
ТЕХНОЛОГИИ
НА БАЗЕ ЛУЧШЕГО
МИРОВОГО ОПЫТА

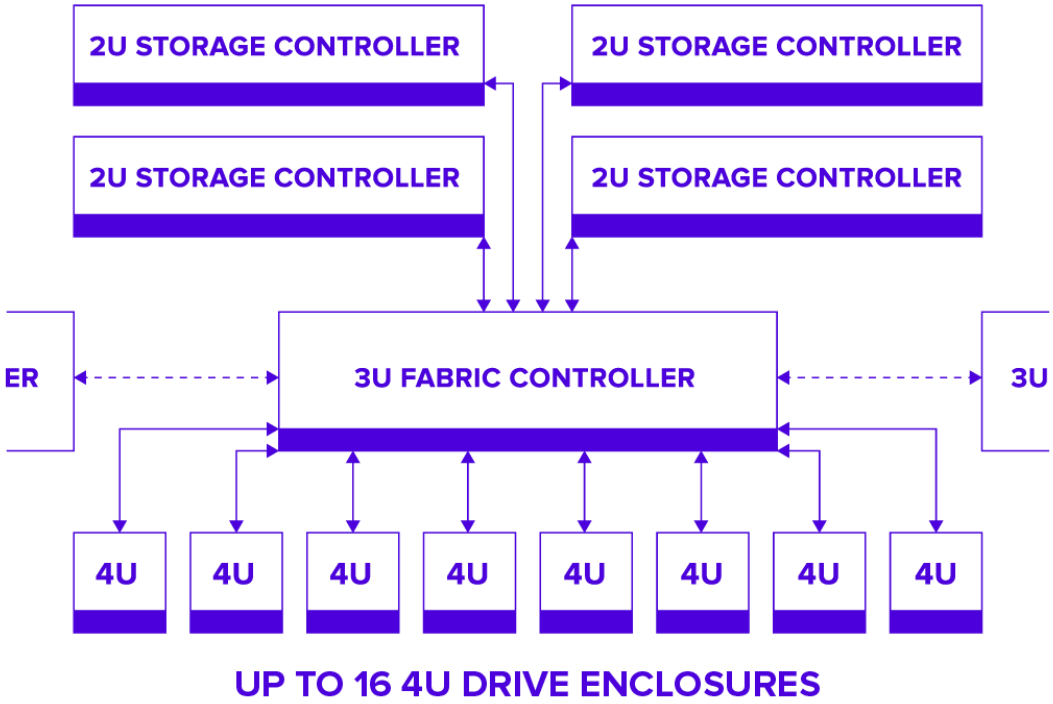
# Software RDMA: RoCE, RXE, SLE

Mikhail Malygin

# What we do

- Hybrid storage
- Unified
- PCIe fabric

# What we do


2U STORAGE CONTROLLER
2U STORAGE CONTROLLER
2U STORAGE CONTROLLER
2U STORAGE CONTROLLER
ER
3U FABRIC CONTROLLER
3U
4U
4U
4U
4U
4U
4U
4U
4U
UP TO 16 4U DRIVE ENCLOSURES

# What we use

- OpenPOWER storage controllers
- SUSE Linux Enterprise (kernel 4.4.x LTS)


# What we need

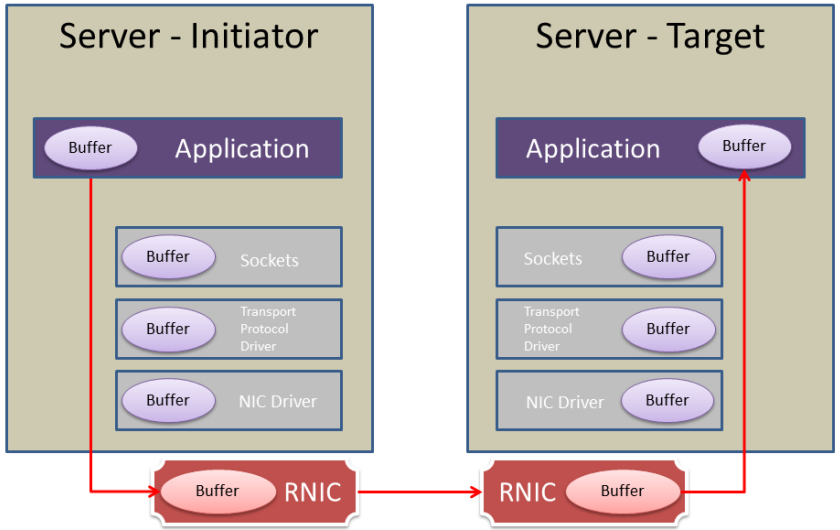
- Need high performance low latency communication channel
- Available from kernel space
- Options:
    - PCIe NTB
    - Something on TCP
    - RDMA

# PCIe NTB

- PCIe over wire card
  - Communication between controller and fabric
  - PLX PEX8734 switch
  - 16 PCIe lines
  - No DMA support for NTB

# RDMA

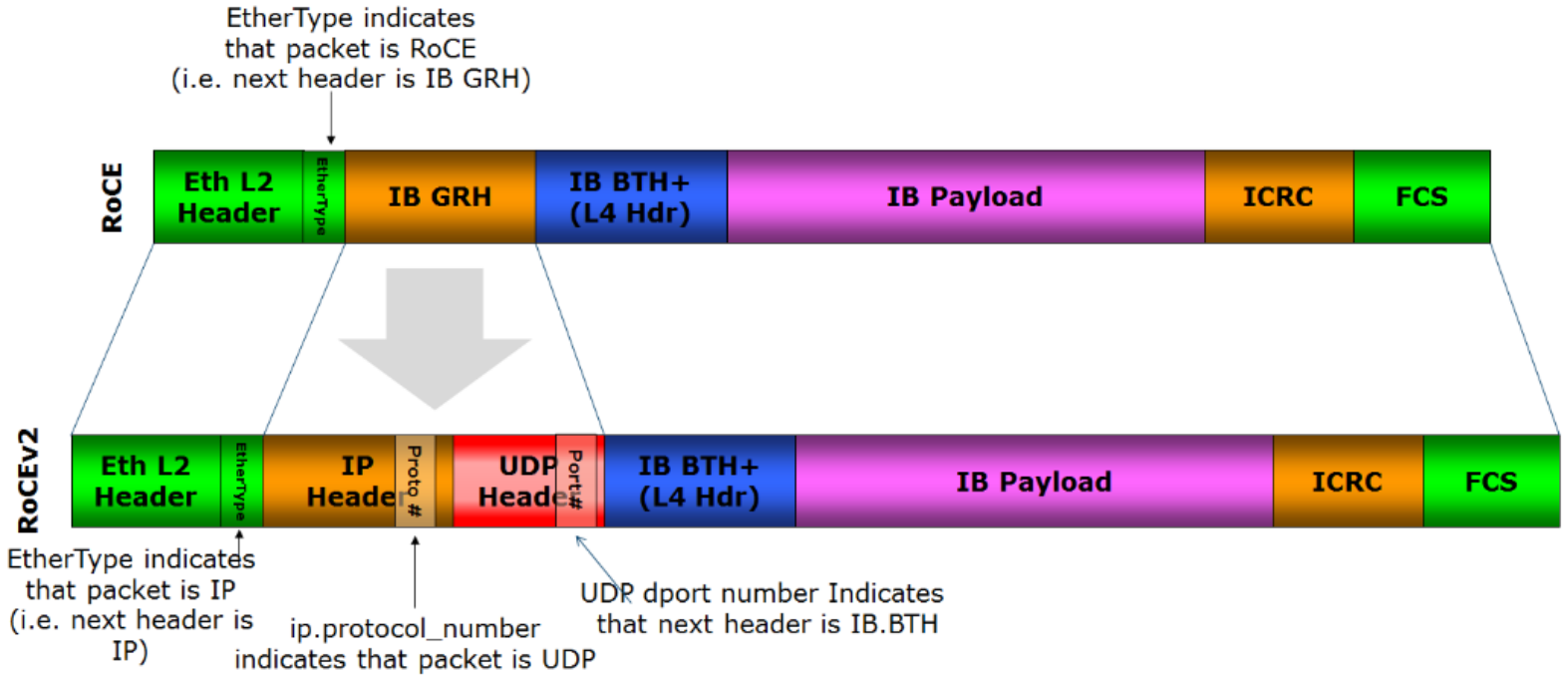
- Remote Direct Memory Access
- Low latency transport
- High throughput
- CPU offload
- InfiniBand, RoCE, iWARP
- Requires rNIC (RDMA enabled NIC)



# RoCE

- RDMA over Converged Ethernet
- Use common Ethernet fabric
- InfiniBand frames encapsulated in Ethernet or UDP
- Requires lossless Ethernet (DCB)
    - IEEE 802.1Qbb – Priority based flow control

# RoCE


EtherType indicates
that packet is RoCE
(i.e. next header is IB GRH)
RoCE
Eth L2
Header
EtherType
IB GRH
IB BTH+
(L4 Hdr)
IB Payload
ICRC
FCS
RoCEv2
Eth L2
Header
EtherType
IP
Header
Proto #
UDP
Header
Port#
IB BTH+
(L4 Hdr)
IB Payload
ICRC
FCS
EtherType indicates
that packet is IP
(i.e. next header is
IP)
ip.protocol_number
indicates that packet is UDP
UDP dport number Indicates
that next header is IB.BTH

# What we had for PoC

- Mellanox ConnectX-2 adaptors
- PPC64le SLES 12SP2
- Tested with:
    - rdma_server/rdma_client
    - ib_*_bw tools
- https://community.mellanox.com/docs/DOC-2086

# How to use in kernel

- Communication management (RDMA CM)
    - rdma/rdma_cm.h
    - Establish channel between peers
- IB verbs
    - rdma/ib_verbs.h
    - Perform data operations
    - send/receive/read/write/atomic

# How to use: connection

| Client | Server |
| --- | --- |
| rdma_create_id() | rdma_create_id() |
| rdma_resolve_addr() | rdma_listen() |
| rdma_resolve_route() | |
| <prepare QP> | <prepare QP> |
| rdma_connect() | rdma_accept() |

# How to use: connection

- Before connect or accept the following needs to be done:
    - Create protection domain (PD)
    - Create send/receive completion queues (CQ)
    - Create queue pairs (QP)
    - Start polling for events
    - Your are ready to go!

# How to use: verbs

- Verbs define RDMA operation:
    - IB_WR_RDMA_WRITE, IB_WR_SEND, IB_WR_RDMA_READ, IB_WR_REG_MR, etc.
- Working with verbs:
    - Prepare work request (WR)
    - Assign memory buffers (DMA mapped)
    - Post WR to QP
    - Wait for work completion (WC)

# How to use: memory registration

- RDMA memory must be registered:
    - pin pages
    - DMA map
    - get remote key (rkey) and I/O virtual address (iova)
- rkey, iova, length are required to perform remote memory access

# How to use: memory registration

- 4 different types
    - Memory registration (MR)
    - Fast memory registration (FMR)
    - Memory window (MW)
    - Fast memory registration mode (FRWR)
        - Allocate MR, dma map, post IB_WR_REG_MR

# How to use: complete example

- krping project
- Kernel module
    - Test multiple verbs (read/write/send/receive/invalidate)
- Several versions in the internet, current:
    - https://github.com/larrystevenwise/krping
    - works with 4.4+ kernels with some modifications

# How to develop for RDMA?

- Need a way to develop and run tests:
    - On laptop
    - In virtual machines
    - Separate hardware is not an option
        - SR-IOV HCA may help here

# RXE

- Mellanox driver for software-only RoCE
- Works on every Ethernet card
- Including virtual machines (even virtio)
- Works with RDMA HCA on the same wire (I did not check)
- In mainline since 4.8

# RXE: it’s what we need!

- Bad news
    - We still on 4.4 from SLES 12SP2
- Good news:
    - SP3 already has it backported!
- So far we can backport it on SP2…

# RXE: backport

- Take sources from 4.10
    - Most recent code similar to 4.4
- Make it compile on 4.4
- Test passed! (ib_*_bw, rdma_server/rdma_client)
- Almost works, except…

# RXE works, except…

- Krping does not work:

# RXE works, except…

- Krping does not work:
  - Caused by failed DMA mappings
  - Need to use ib_dma_map* instead of dma_map_* functions
    - A thin wrapper around dma_map_* that allows drivers to override mapping behavior

# RXE works, except…

- Can’t map more than one page via ib_dma_map_page()
    - But can map same memory via ib_dma_map_single()

drivers/infiniband/sw/rxe/rxe_dma.c

```
static u64 rxe_dma_map_single(struct ib_device *dev, void *cpu_addr,
                              size_t size, enum dma_data_direction direction)
{
	WARN_ON(!valid_dma_direction(direction));
	return (uintptr_t)cpu_addr;
}

static u64 rxe_dma_map_page(struct ib_device *dev, struct page *page, unsigned long offset,
                            size_t size, enum dma_data_direction direction)
{
	u64 addr;
	WARN_ON(!valid_dma_direction(direction));

	if (offset + size > PAGE_SIZE) {
		addr = DMA_BAD_ADDER;
		goto done;
	}

	addr = (uintptr_t)page_address(page);

	if (addr)
		addr += offset;

done:
	return addr;
}
```

drivers/infiniband/sw/rxe/rxe_dma.c

```
static u64 rxe_dma_map_single(struct ib_device *dev, void *cpu_addr,
                              size_t size, enum dma_data_direction direction)
{
        WARN_ON(!valid_dma_direction(direction));
        return (uintptr_t)cpu_addr;
}

static u64 rxe_dma_map_page(struct ib_device *dev, struct page *page, unsigned long offset,
                            size_t size, enum dma_data_direction direction)
{
        u64 addr;
        WARN_ON(!valid_dma_direction(direction));

        if (offset + size > PAGE_SIZE) {
                addr = DMA_BAD_ADDER;
                goto done;
        }

        addr = (uintptr_t)page_address(page);

        if (addr)
                addr += offset;

done:
        return addr;
}
```

# RXE works, except…

- Can’t map more than one page via ib_dma_map_page()
  - But can map same memory region via ib_dma_map_single()
  - This code is missing in 4.11+
  - Just remove it ☺

# RXE works, except…

# RXE works, except…

- Writes larger than 1k are failing (error 24, RESPST_ERR_LENGTH)
    - reads are fine
    - responder moves to error state, requester got stuck

- Simple test:
  - 1k RDMA write
  - 2k RDMA write

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 3 | 0.081913 | 10.0.3.101 | 10.0.3.100 | RRoCE | 322 | CM: ConnectRequest |
| 4 | 0.086187 | 10.0.3.100 | 10.0.3.101 | RRoCE | 322 | CM: ConnectReply |
| 5 | 0.202151 | 10.0.3.101 | 10.0.3.100 | RRoCE | 322 | CM: ReadyToUse |
| 6 | 0.202171 | 10.0.3.101 | 10.0.3.100 | RRoCE | 82 | RC Send Only QP=0x000011 |
| 7 | 0.202184 | 10.0.3.100 | 10.0.3.101 | RRoCE | 62 | RC Acknowledge QP=0x000011 |
| 8 | 0.202236 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1102 | RC RDMA Write Only Immediate QP=0x000011 |
| 9 | 0.268380 | 10.0.3.101 | 10.0.3.100 | RRoCE | 62 | RC Acknowledge QP=0x000011 |
| 10 | 2.312123 | 10.0.3.101 | 10.0.3.100 | RRoCE | 82 | RC Send Only QP=0x000011 |
| 11 | 2.312150 | 10.0.3.100 | 10.0.3.101 | RRoCE | 62 | RC Acknowledge QP=0x000011 |
| 12 | 2.312201 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1098 | RC RDMA Write First QP=0x000011 |
| 13 | 2.312203 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1086 | RC RDMA Write Last Immediate QP=0x000011 |
| 14 | 2.396112 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1098 | RC RDMA Write First QP=0x000011 |
| 15 | 2.396140 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1086 | RC RDMA Write Last Immediate QP=0x000011 |
| 16 | 4.545418 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1098 | RC RDMA Write First QP=0x000011 |
| 17 | 4.545448 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1086 | RC RDMA Write Last Immediate QP=0x000011 |

# RXE works, except…

- Writes larger than 1k are failing (error 24, RESPST_ERR_LENGTH)
  - reads are fine
  - responder moves to error state, requester got stuck
  - affected writes larger than RMTU (1k/2k/4k)

drivers/infiniband/sw/rxe/rxe_resp.c

```
static enum resp_states check_rkey(struct rxe_qp *qp, struct rxe_pkt_info *pkt)
{
/* A lot of lines skipped here... */
      if (pkt->mask & RXE_WRITE_MASK) {
            if (resid > mtu) {
                  if (pktlen != mtu || bth_pad(pkt)) {
                        state = RESPST_ERR_LENGTH;
                        goto err;
                  }

                  qp->resp.resid = mtu;
            } else {
                  if (pktlen != resid) {
                        state = RESPST_ERR_LENGTH;
                        goto err;
                  }
                  if ((bth_pad(pkt) != (0x3 & (-resid)))) {
                        /* This case may not be exactly that
                         * but nothing else fits.
                         */
                        state = RESPST_ERR_LENGTH;
                        goto err;
                  }
            }
      }
} /* The end is skipped as well */
```

drivers/infiniband/sw/rxe/rxe_resp.c

```
static enum resp_states check_rkey(struct rxe_qp *qp, struct rxe_pkt_info *pkt)
{
/* A lot of lines skipped here... */
      if (pkt->mask & RXE_WRITE_MASK) {
            if (resid > mtu) {
                  if (pktlen != mtu || bth_pad(pkt)) {
                        state = RESPST_ERR_LENGTH;
                        goto err;
                  }

                  qp->resp.resid = mtu;
            } else {
                  if (pktlen != resid) {
                        state = RESPST_ERR_LENGTH;
                        goto err;
                  }
                  if ((bth_pad(pkt) != (0x3 & (-resid)))) {
                        /* This case may not be exactly that
                         * but nothing else fits.
                         */
                        state = RESPST_ERR_LENGTH;
                        goto err;
                  }
            }
      }
} /* The end is skipped as well */
```

- Fixed in 4.12+

```
From d52418502e288b5c7e9e2e6cf1de5f1d3d79d2e1 Mon Sep 17 00:00:00 2001
From: Johannes Thumshirn <jthumshirn@suse.de>
Date: Thu, 6 Apr 2017 14:49:44 +0200
Subject: IB/rxe: Don't clamp residual length to mtu

diff --git a/drivers/infiniband/sw/rxe/rxe_resp.c b/drivers/infiniband/sw/rxe/rxe_resp.c
index ec11a9c..2303976 100644
--- a/drivers/infiniband/sw/rxe/rxe_resp.c
+++ b/drivers/infiniband/sw/rxe/rxe_resp.c
@@ -481,8 +481,6 @@ static enum resp_states check_rkey(struct rxe_qp *qp,
                                state = RESPST_ERR_LENGTH;
                                goto err;
                        }
-
-                       qp->resp.resid = mtu;
                } else {
                        if (pktlen != resid) {
                                state = RESPST_ERR_LENGTH;
```

- Simple test:
  - 1k RDMA write
  - 2k RDMA write

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 18 | 214.892918 | 10.0.3.101 | 10.0.3.100 | RRoCE | 322 | CM: ConnectRequest |
| 19 | 214.894570 | 10.0.3.100 | 10.0.3.101 | RRoCE | 322 | CM: ConnectReply |
| 20 | 214.954438 | 10.0.3.101 | 10.0.3.100 | RRoCE | 322 | CM: ReadyToUse |
| 21 | 214.954463 | 10.0.3.101 | 10.0.3.100 | RRoCE | 82 | RC Send Only QP=0x000011 |
| 22 | 214.954479 | 10.0.3.100 | 10.0.3.101 | RRoCE | 62 | RC Acknowledge QP=0x000011 |
| 23 | 214.954654 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1098 | RC RDMA Write First QP=0x000011 |
| 24 | 214.954681 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1086 | RC RDMA Write Last Immediate QP=0x000011 |
| 25 | 214.955231 | 10.0.3.101 | 10.0.3.100 | RRoCE | 62 | RC Acknowledge QP=0x000011 |
| 26 | 214.955236 | 10.0.3.101 | 10.0.3.100 | RRoCE | 82 | RC Send Only QP=0x000011 |
| 27 | 214.955241 | 10.0.3.100 | 10.0.3.101 | RRoCE | 62 | RC Acknowledge QP=0x000011 |
| 28 | 214.955279 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1098 | RC RDMA Write First QP=0x000011 |
| 29 | 214.955282 | 10.0.3.100 | 10.0.3.101 | RRoCE | 1086 | RC RDMA Write Last Immediate QP=0x000011 |
| 30 | 214.955785 | 10.0.3.101 | 10.0.3.100 | RRoCE | 62 | RC Acknowledge QP=0x000011 |

# RXE: some more to backport…

| Age | Commit message (Expand) | Author | Files | Lines |
|---|---|---|---|---|
| 2017-08-28 | IB/rxe: Handle NETDEV_CHANGE events | Andrew Boyer | 1 | -1/+6 |
| 2017-08-28 | IB/rxe: Avoid ICRC errors by copying into the skb first | Andrew Boyer | 1 | -6/+6 |
| 2017-08-28 | IB/rxe: Another fix for broken receive queue draining | Andrew Boyer | 1 | -1/+3 |
| 2017-08-28 | IB/rxe: Remove unneeded initialization in prepare6() | Andrew Boyer | 1 | -1/+1 |
| 2017-08-28 | IB/rxe: Fix up rxe_qp_cleanup() | Andrew Boyer | 1 | -7/+2 |
| 2017-08-28 | IB/rxe: Add dst_clone() in prepare_ipv6_hdr() | Andrew Boyer | 1 | -1/+1 |
| 2017-08-28 | IB/rxe: Fix destination cache for IPv6 | Andrew Boyer | 2 | -1/+7 |
| 2017-08-28 | IB/rxe: Fix up the responder's find_resources() function | Andrew Boyer | 1 | -1/+1 |
| 2017-08-28 | IB/rxe: Remove dangling prototype | Andrew Boyer | 1 | -2/+0 |
| 2017-08-28 | IB/rxe: Disable completion upcalls when a CQ is destroyed | Andrew Boyer | 4 | -0/+24 |
| 2017-08-28 | IB/rxe: Move refcounting earlier in rxe_send() | Andrew Boyer | 1 | -3/+5 |
| 2017-08-24 | IB/rxe: Make rxe_counter_name static | Kamal Heib | 1 | -1/+1 |
| 2017-08-18 | IB/rxe: Remove unneeded check | Yuval Shaia | 1 | -5/+0 |
| 2017-08-18 | IB/rxe: Convert pr_info to pr_warn | Yuval Shaia | 1 | -1/+1 |
| 2017-07-24 | IB/rxe: Constify static rxe_vm_ops | Kamal Heib | 1 | -1/+1 |
| 2017-07-24 | IB/rxe: Use __func__ to print function's name | Kamal Heib | 2 | -5/+5 |
| 2017-07-24 | IB/rxe: Use DEVICE_ATTR_RO macro to show parent field | Kamal Heib | 1 | -3/+3 |
| 2017-07-24 | IB/rxe: Prefer 'unsigned int' to bare use of 'unsigned' | Kamal Heib | 3 | -3/+3 |
| 2017-07-24 | IB/rxe: Use "foo *bar" instead of "foo * bar" | Kamal Heib | 1 | -1/+1 |
| 2017-07-18 | Merge tag 'for-linus' of git://git.kernel.org/pub/scm/linux/kernel/git/dledfo... | Linus Torvalds | 43 | -290/+366 |
| 2017-07-17 | IB/rxe: Set dma_mask and coherent_dma_mask | yonatanc | 1 | -0/+2 |
| 2017-07-17 | IB/rxe: Fix kernel panic from skb destructor | Yonatan Cohen | 1 | -0/+3 |
| 2017-07-12 | IB/rxe: do not copy extra stack memory to skb | Kees Cook | 1 | -1/+3 |

# RXE works, except…

- It does not send error response to the requester
    - responder moves to error state
    - requester get stuck waiting for response
    - for several types of errors

# Example: read invalid MR

| | | | | | | |
|---|---|---|---|---|---|---|
| 143 | 13.673404 | 10.0.3.101 | 10.0.3.100 | RRoCE | 82 | RC Send Only QP=0x000011 |
| 144 | 13.673434 | 10.0.3.100 | 10.0.3.101 | RRoCE | 62 | RC Acknowledge QP=0x000011 |
| 145 | 13.673489 | 10.0.3.100 | 10.0.3.101 | RRoCE | 74 | RC RDMA Read Request QP=0x000011 |
| 146 | 15.823286 | 10.0.3.100 | 10.0.3.101 | RRoCE | 74 | RC RDMA Read Request QP=0x000011 |
| 147 | 17.975418 | 10.0.3.100 | 10.0.3.101 | RRoCE | 74 | RC RDMA Read Request QP=0x000011 |
| 148 | 20.136010 | 10.0.3.100 | 10.0.3.101 | RRoCE | 74 | RC RDMA Read Request QP=0x000011 |
| 149 | 22.287033 | 10.0.3.100 | 10.0.3.101 | RRoCE | 74 | RC RDMA Read Request QP=0x000011 |
| 150 | 24.453816 | 10.0.3.100 | 10.0.3.101 | RRoCE | 74 | RC RDMA Read Request QP=0x000011 |
| 151 | 26.599277 | 10.0.3.100 | 10.0.3.101 | RRoCE | 74 | RC RDMA Read Request QP=0x000011 |

drivers/infiniband/sw/rxe/rxe_resp.c

```
int rxe_responder(void *arg)
{
/* lines skipped */
	case RESPST_ERR_RNR:
		if (qp_type(qp) == IB_QPT_RC) {
			/* RC - class B */
			send_ack(qp, pkt, AETH_RNR_NAK | (~AETH_TYPE_MASK &
					qp->attr.min_rnr_timer), pkt->psn);
		} else {
			/* not related to RC */
		}
	state = RESPST_CLEANUP;
	break;

	case RESPST_ERR_RKEY_VIOLATION:
		if (qp_type(qp) == IB_QPT_RC) {
			/* Class C */
			do_class_ac_error(qp, AETH_NAK_REM_ACC_ERR, IB_WC_REM_ACCESS_ERR);
			state = RESPST_COMPLETE;
		} else {
			/* not related to RC */
		}
	break;
/* lines skipped */
}
```

drivers/infiniband/sw/rxe/rxe_resp.c

```
int rxe_responder(void *arg)
{
/* lines skipped */
        case RESPST_ERR_RNR:
                if (qp_type(qp) == IB_QPT_RC) {
                        /* RC - class B */
                        send_ack(qp, pkt, AETH_RNR_NAK | (~AETH_TYPE_MASK &
                                        qp->attr.min_rnr_timer), pkt->psn);
                } else {
                        /* not related to RC */
                }
        state = RESPST_CLEANUP;
        break;

        case RESPST_ERR_RKEY_VIOLATION:
                if (qp_type(qp) == IB_QPT_RC) {
                        /* Class C */
                        do_class_ac_error(qp, AETH_NAK_REM_ACC_ERR, IB_WC_REM_ACCESS_ERR);
                        state = RESPST_COMPLETE;
                } else {
                        /* not related to RC */
                }
        break;
/* lines skipped */
}
```

# RXE works, except…

- It does not send error response to the requester
    - responder moves to error state
    - requester get stuck waiting for response
    - for several types of errors
    - sending NACK with error code works fine

# Example: read invalid MR – fixed!

```
136 2.167902    10.0.3.101          10.0.3.100          RRoCE     82 RC Send Only QP=0x000011
137 2.167931    10.0.3.100          10.0.3.101          RRoCE     62 RC Acknowledge QP=0x000011
138 2.168060    10.0.3.100          10.0.3.101          RRoCE     74 RC RDMA Read Request QP=0x000011
139 2.189452    10.0.3.101          10.0.3.100          RRoCE     62 RC Acknowledge QP=0x000011
140 5.000298    PcsCompu_84:6f:15   PcsCompu_ac:95:6d   ARP       42 Who has 10.0.3.100? Tell 10.0.3.101
141 5.000324    PcsCompu_ac:95:6d   PcsCompu_84:6f:15   ARP       42 10.0.3.100 is at 08:00:27:ac:95:6d

▶ Frame 139: 62 bytes on wire (496 bits), 62 bytes captured (496 bits)
▶ Ethernet II, Src: PcsCompu_84:6f:15 (08:00:27:84:6f:15), Dst: PcsCompu_ac:95:6d (08:00:27:ac:95:6d)
▶ Internet Protocol Version 4, Src: 10.0.3.101, Dst: 10.0.3.100
▶ User Datagram Protocol, Src Port: 49152, Dst Port: 4791
▼ InfiniBand
  ▶ Base Transport Header
  ▼ AETH - ACK Extended Transport Header
      Syndrome: 98
      Message Sequence Number: 65
    Invariant CRC: 0x92143728
```

# RXE works, except…

- It still requires reliable Ethernet

# RXE works, except…

- It still requires reliable Ethernet
    - Not a case for VirtualBox
    - May be caused by missing TX flow control on Ethernet devices
    - However openvswitch + kvm seems to work fine

# What we've got

- RXE is good option for development and testing
- Starting from 4.12 ☺
- Works on SLES as well
    - Requires a lot of patches from upstream
- Works on Power
- Wrong ethernet configuration causes issues

YADRO

РОССИЙСКИЕ
ИНФОРМАЦИОННЫЕ
ТЕХНОЛОГИИ
НА БАЗЕ ЛУЧШЕГО
МИРОВОГО ОПЫТА

# Questions?