NVMe Cancel and FPIN Support

LSF/MM 2024

John Meneghini
jmeneghi@redhat.com

Fibre-channel FPIN support

- Currently supported by DM-MP with SCSI/FCP
- Hannes' “FPIN” patches add support for FPIN LI to NVMe Native multipathing with FC-NVMe
  - https://lore.kernel.org/linux-nvme/20240402093031.146342-1-hare@kernel.org/
- Currently being tested by Red Hat with the same nvme-fc multipath configurations used to test “queue-depth” and “latency” IO Policies
- Current testing has revealed bugs that are being worked out
  - Goal is to merge this patch into nvme-6.11

FPIN Testing Switch Details

- Following are the details w.r.t. Brocade switches (to get started on generating FPINs)
  - G720 and G620 32G Brocade Switches
  - Brocade v9.0.x FOS
  - Login must be root (i.e., root/Qlogic01)
  - Must have root access. If root access is not possible, request help from Broadcom
- Red Hat testing with with DM-MP and SCSI, adding NVMe-FC

Link Integrity Initiator Tests

- send_link_fpin_initiator "/fabos/cliexec/ftc test --fpin $initpid -li -unknown"
- send_link_fpin_initiator "/fabos/cliexec/ftc test --fpin $initpid -li -link_failure"
- send_link_fpin_initiator "/fabos/cliexec/ftc test --fpin $initpid -li -loss_sync"
- send_link_fpin_initiator "/fabos/cliexec/ftc test --fpin $initpid -li -loss_signal"
- send_link_fpin_initiator "/fabos/cliexec/ftc test --fpin $initpid -li -primitive_error"
- send_link_fpin_initiator "/fabos/cliexec/ftc test --fpin $initpid -li -itw"
- send_link_fpin_initiator "/fabos/cliexec/ftc test --fpin $initpid -li -crc"
- send_link_fpin_initiator "/fabos/cliexec/ftc test --fpin $initpid -li -dev_specific"

Link Integrity Target Tests

- send_link_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -li -unknown"
- send_link_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -li -link_failure"
- send_link_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -li -loss_sync"
- send_link_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -li -loss_signal"
- send_link_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -li -primitive_error"
- send_link_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -li -itw"
- send_link_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -li -crc"
- send_link_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -li -dev_specific"

Link Congestion FPIN Tests

- Send Peer Congestion commands on remote/target PID (SW trgtpid)
- send_cong_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -c -lost-credit"
- send_cong_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -c -credit-stall"
- send_cong_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -c -oversubscription"
- send_cong_fpin_remote "/fabos/cliexec/ftc test --fpin $trgtpid -c -clear"
- Send Congestion commands on Initiator PID (SW initpid)
- send_cong_fpin_init "/fabos/cliexec/ftc test --fpin $initpid -c -lost-credit"
- send_cong_fpin_init "/fabos/cliexec/ftc test --fpin $initpid -c -credit-stall"
- send_cong_fpin_init "/fabos/cliexec/ftc test --fpin $initpid -c -oversubscription"
- send_cong_fpin_init "/fabos/cliexec/ftc test --fpin $initpid -c -clear"

NVMe Cancel Support

- Maurizio Lombardi’s NVMe Cancel patches
  - https://lore.kernel.org/linux-nvme/20240510163026.786898-1-mlombard@redhat.com/T/
- Adds support for TP4097
  - For a full explaination of TP4097 see:
    https://www.youtube.com/watch?v=vRrAD1U0IRw
  -

NVMe Cancel - Open issues

- These are preliminary patches
- Need to add support for all transports (pci and fc)
- Need to provide nvmet patches with blktests
  - We can’t rely upon hardware as a test target
- Need reserved tags (x2) for each IO queue
- Need to impove Cancel algorythim
  - Cancel single IO vs. Cancel IO-Queue

NVMe Cancel testing

```
[83718.210831] nvme nvme5: I/O tag 2 (0002) type 4 opcode 0x2 (Read) QID 8 timeout
[83718.210842] nvme nvme5: I/O tag 5 (8005) type 4 opcode 0x1 (Write) QID 8 timeout
[83718.210844] nvme nvme5: Cancel command failed!
[83718.210845] nvme nvme5: starting error recovery
[83718.210850] nvme nvme5: I/O tag 47 (002f) type 4 opcode 0x1 (Write) QID 8 timeout
[83718.210871] nvme nvme5: I/O tag 48 (f030) type 4 opcode 0x2 (Read) QID 8 timeout
[83718.210874] nvme nvme5: I/O tag 50 (f032) type 4 opcode 0x1 (Write) QID 8 timeout
[83718.210876] nvme nvme5: I/O tag 52 (e034) type 4 opcode 0x2 (Read) QID 8 timeout
[83718.210881] nvme nvme5: I/O tag 97 (f061) type 4 opcode 0x1 (Write) QID 8 timeout
[83718.210883] nvme nvme5: I/O tag 98 (d062) type 4 opcode 0x2 (Read) QID 8 timeout
[83718.210884] nvme nvme5: I/O tag 99 (d063) type 4 opcode 0x1 (Write) QID 8 timeout
[83718.210885] nvme nvme5: I/O tag 100 (9064) type 4 opcode 0x1 (Write) QID 8 timeout
…
[83718.210950] nvme nvme5: I/O tag 105 (6069) type 4 opcode 0x1 (Write) QID 10 timeout
[83718.211091] nvme nvme5: Cancel status: 0x371 imm abrts = 10561 def abrts = 0
[83718.211451] nvme nvme5: Reconnecting in 10 seconds...
[83718.722876] nvme nvme6: I/O tag 87 (f057) type 4 opcode 0x2 (Read) QID 11 timeout
[83718.722889] nvme nvme6: I/O tag 88 (f058) type 4 opcode 0x1 (Write) QID 11 timeout
[83718.722891] nvme nvme6: Cancel command failed!
[83718.722893] nvme nvme6: starting error recovery
[83718.722896] nvme nvme6: I/O tag 89 (f059) type 4 opcode 0x1 (Write) QID 11 timeout
[83718.722919] nvme nvme6: failed to send request -32
[83718.722923] nvme nvme6: Cancel status: 0x370 imm abrts = 0 def abrts = 0
```

Cancel Command failed!

```
+	cancel_req = blk_mq_alloc_request_hctx(rq->q, nvme_req_op(&c),
+					BLK_MQ_REQ_NOWAIT |
+					BLK_MQ_REQ_RESERVED,
+					sqid - 1);
+	if (IS_ERR(cancel_req)) {
+		dev_warn(ctrl->device, "Cancel command failed!\n");
+		return PTR_ERR(cancel_req);
+	}
+
+	nvme_init_request(cancel_req, &c);
+	cancel_req->end_io = nvme_cancel_endio;
+	cancel_req->end_io_data = ctrl;
+
+	blk_execute_rq_nowait(cancel_req, false);
```

```
diff --git a/drivers/nvme/host/core.c b/drivers/nvme/host/core.c
index 9bedbfbe5d3f..d69bb7c62f3d 100644
--- a/drivers/nvme/host/core.c
+++ b/drivers/nvme/host/core.c
@@ -4496,6 +4496,7 @@ int nvme_alloc_io_tag_set(struct nvme_ctrl *ctrl, struct blk_mq_tag_set *set,
                unsigned int cmd_size)
 {
        int ret;
+       u32 effects = le32_to_cpu(ctrl->effects->iocs[nvme_cmd_cancel]);

        memset(set, 0, sizeof(*set));
        set->ops = ops;
@@ -4506,9 +4507,13 @@ int nvme_alloc_io_tag_set(struct nvme_ctrl *ctrl, struct blk_mq_tag_set *set,
         */
        if (ctrl->quirks & NVME_QUIRK_SHARED_TAGS)
                set->reserved_tags = NVME_AQ_DEPTH;
+       else if  (effects & NVME_CMD_EFFECTS_CSUPP)
+                /* Reserve 2 X io_queue count for NVMe Cancel */
+                set->reserved_tags = (2 * ctrl->queue_count);
        else if (ctrl->ops->flags & NVME_F_FABRICS)
                /* Reserved for fabric connect */
                set->reserved_tags = 1;
+
        set->numa_node = ctrl->numa_node;
        set->flags = BLK_MQ_F_SHOULD_MERGE;
        if (ctrl->ops->flags & NVME_F_BLOCKING)
```

NVMe Target Changes

https://lore.kernel.org/linux-nvme/20240510163026.786898-1-mlombard@redhat.com/T/#mcc5bb6f800cb9faf75947b68d3618841ec2bcde0

```
diff --git a/drivers/nvme/target/core.c b/drivers/nvme/target/core.c
index 6bbe4df0166c..b08b01ee4117 100644
--- a/drivers/nvme/target/core.c
+++ b/drivers/nvme/target/core.c
@@ -769,12 +769,33 @@ static void __nvmet_req_complete(struct nvmet_req *req, u16 status)
 void nvmet_req_complete(struct nvmet_req *req, u16 status)
 {
        struct nvmet_sq *sq = req->sq;
+       unsigned long flags;
+
+       spin_lock_irqsave(&sq->state_lock, flags);
+
+       if (unlikely(req->aborted))
+               status = NVME_SC_ABORT_REQ;
+       else
+               list_del(&req->state_list);
+
+       spin_unlock_irqrestore(&sq->state_lock, flags);

        __nvmet_req_complete(req, status);
        percpu_ref_put(&sq->ref);
```

NVMe Target Changes

```
#ifdef CONFIG_BLK_DEV_NULL_BLK_FAULT_INJECTION
/*
 * For more details about fault injection, please refer to
 * Documentation/fault-injection/fault-injection.rst.
 */
static char g_timeout_str[80];
module_param_string(timeout, g_timeout_str, sizeof(g_timeout_str), 0444);
MODULE_PARM_DESC(timeout, "Fault injection. timeout=<interval>,<probability>,<space>,<times>");

static char g_requeue_str[80];
module_param_string(requeue, g_requeue_str, sizeof(g_requeue_str), 0444);
MODULE_PARM_DESC(requeue, "Fault injection. requeue=<interval>,<probability>,<space>,<times>");

static char g_init_hctx_str[80];
module_param_string(init_hctx, g_init_hctx_str, sizeof(g_init_hctx_str), 0444);
MODULE_PARM_DESC(init_hctx, "Fault injection to fail hctx init. init_hctx=<interval>,<probability>,<space>,<times>");
#endif

"drivers/block/null_blk/main.c" [readonly] line 96 of 2138 --4%-- col 38
```

- THE END