NVMe/TCP PDU Offload Demonstration with T7

Using Kioxia SSDs & Chelsio T7

Overview

The Terminator 7 (T7) ASIC from Chelsio Communications, Inc. is a seventh generation, high performance 1/10/25/40/50/100/200/400 Gbps, Unified Wire Data Processing Unit (DPU) that offers offload support for a wide range of Storage (NVMe/TCP, NVMe-oF, iSCSI, iSER, S2D, SMB), Network (TCP/IP, UDP/IP, RDMA – iWARP and RoCEv2) and Crypto (IPsec, TLS/SSL) protocols. Designed for industry-leading performance and efficiency, Chelsio adapters unburden communication responsibilities and processing overhead from host servers and storage systems resulting in a dramatic increase in application performance with a minimum of CPU cycles. With concurrent support for offloading multiple protocols and crypto operations, Chelsio has taken the Unified Wire solution to the next level.

This paper describes the NVMe/TCP PDU Offload capabilities of T7 and demonstrates interoperability with Kioxia NVMe SSDs.

NVMe/TCP PDU Offload Solution

The NVMe over Fabrics (NVMe-oF) specification extends the benefits of NVMe to large fabrics beyond the reach and scalability of traditional in-server physical PCIe. NVMe/TCP is a technology that transfers data and commands between NVMe-oF hosts and controllers over existing, standard, robust, reliable, and scalable TCP/IP networks without having to purchase special/separate RDMA-capable hardware.

Chelsio's T7 NVMe/TCP PDU Offload solution is fully capable of offloading TCP/IP, NVMe/TCP PDU Header, and Data digests processing to the hardware at 400 Gbps rate. The driver supports Direct Data Placement (DDP), which allows the data to be DMAed directly to the host memory buffer (shown with Orange arrows in Figure 1). T7's peer-to-peer (P2P) capability DMAes the data directly between the network and SSD CMB without the use of host memory and CPU resources, resulting in significant CPU savings and low latency (shown with Green arrows in Figure 1).

In Transmit, T7 will segment the NVMe/TCP PDUs and will send them on the wire. In Receive, T7 will DMA the data directly to the host memory buffers. Additionally, T7 can DMA the data to/from the NVMe SSD CMB.

T7 will insert the header/data digest in the transmit and can verify them in the Receive.

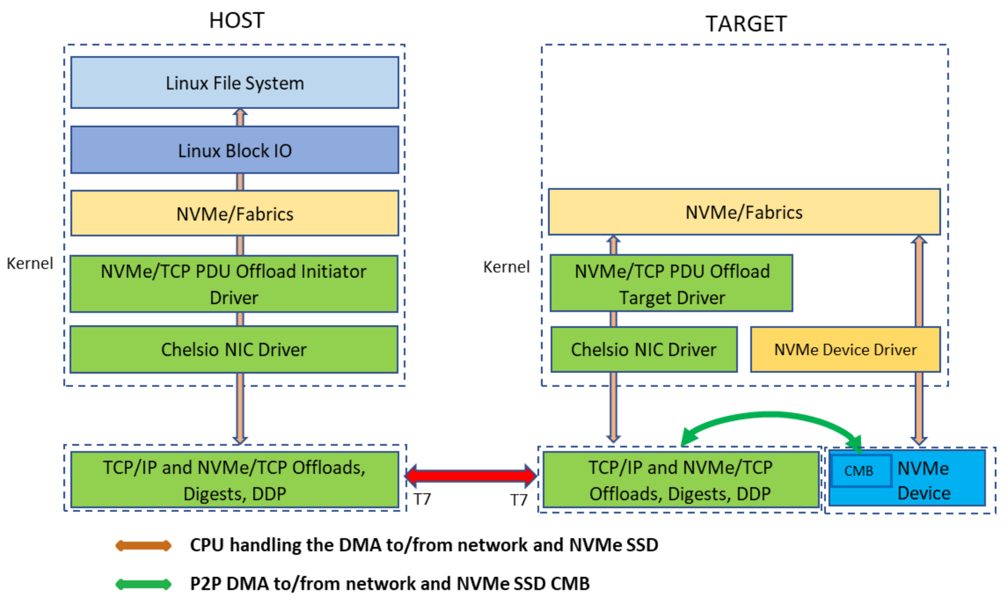
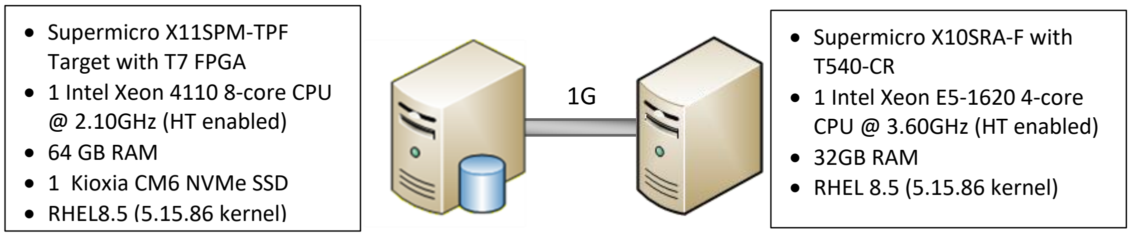

HOST
TARGET
Linux File System
Linux Block IO
NVMe/Fabrics
Kernel
NVMe/TCP PDU Offload Initiator Driver
Chelsio NIC Driver
NVMe/Fabrics
Kernel
NVMe/TCP PDU Offload Target Driver
Chelsio NIC Driver
NVMe Device Driver
TCP/IP and NVMe/TCP Offloads, Digests, DDP
T7
T7
TCP/IP and NVMe/TCP Offloads, Digests, DDP
CMB
NVMe Device
CPU handling the DMA to/from network and NVMe SSD
P2P DMA to/from network and NVMe SSD CMB

Figure 1 – NVMe/TCP PDU Offload with T7 P2P DMA

The Demonstration



- Supermicro X11SPM-TPF Target with T7 FPGA
- 1 Intel Xeon 4110 8-core CPU @ 2.10GHz (HT enabled)
- 64 GB RAM
- 1 Kioxia CM6 NVMe SSD
- RHEL8.5 (5.15.86 kernel)

- Supermicro X10SRA-F with T540-CR
- 1 Intel Xeon E5-1620 4-core CPU @ 3.60GHz (HT enabled)
- 32GB RAM
- RHEL 8.5 (5.15.86 kernel)

Figure 2 – Test set-up

The test setup consists of a storage target machine having 1 Kioxia CM6 3.84 TB SSD (with CMB support) direct attached, connecting to a single host/initiator machine using a single port on each system. The host/initiator connects to the target using one NVMe/TCP connection. A standard MTU of 1500 is used.

Set-up Configuration

Kernel NVMe/TCP Target Configuration

i. Disable virtualization, c-state technology, VT-d, Intel I/O AT, and SR-IOV in system BIOS.
ii. Download the 5.15.86 kernel and enable the below in the kernel configuration file. Compile and install the kernel.

```
CONFIG_PCI_P2PDMA=y
```

iii. Reboot the machine into the newly installed kernel.

iv. Configure the T7 emulation platform with the required images and firmware.

```
[root@target~]# lspci | grep -i che
03:00.0 Ethernet controller: Chelsio Communications Inc Device d000
03:00.1 Ethernet controller: Chelsio Communications Inc Device d000
```

v. Set the below tuned-adm profile.

```
[root@target~]# tuned-adm profile latency-performance
```

vi. Load the Chelsio NIC driver (*cxgb4*) and bring up the interface with an IPv4 address.

```
[root@target~]# modprobe cxgb4
[root@target~]# ifconfig ethX <IPv4 address> up
```

vii. Load the NVMe/TCP PDU Offload drivers.

```
[root@target~]# modprobe nvmet
[root@target~]# modprobe cnvmet-tcp
```

viii. Verify that the CMB is enabled in the Kioxia SSD.

```
[root@target~]# lspci -s 0d:00.0 -vvv
0d:00.0 Non-Volatile memory controller: KIOXIA Corporation NVMe SSD
Controller Cx6 (rev 01) (prog-if 02 [NVM Express])
        Subsystem: KIOXIA Corporation Generic NVMe CM6 RI 3.84TB

[root@target~]# nvme show-regs /dev/nvme0 -H
...
cmbloc  : 4
        Offset                   (OFST): 0x0 (See cmbsz.szu for granularity)
        CMB Queue Dword Alignment     (CQDA): 0
        CMB Data Metadata Mixed Memory Support (CDMMMS): Enforced
        CMB Data Pointer and Command Independent Locations Support
        (CDPCILS): Enforced
        CMB Data Pointer Mixed Locations Support (CDPMLS): Enforced
        CMB Queue Physically Discontiguous Support (CQPDS): Enforced
        CMB Queue Mixed Memory Support(CQMMS): Enforced
        Base Indicator Register (BIR): 0x4

cmbsz   : 800001f
        Size                       (SZ): 32768
        Size Units                (SZU): 4 KB
        Write Data Support        (WDS): Write Data and metadata transfer in
        Controller Memory Buffer is Supported
        Read Data Support         (RDS): Read Data and metadata transfer in
        Controller Memory Buffer is Supported
        PRP SGL List Support   (LISTS): PRP/SG Lists in Controller Memory
        Buffer is Supported
        Completion Queue Support (CQS): Admin and I/O Completion Queues in
        Controller Memory Buffer is Supported
        Submission Queue Support (SQS): Admin and I/O Submission Queues in
        Controller Memory Buffer is Supported
```

ix. Configure the target with Kioxia SSD using the below script.

```
#!/bin/bash
```

```
nvmetcli clear > /dev/null 2>$1
mount -t configfs none /sys/kernel/config > /dev/null 2>$1

IPPORT="4420"
IPADDR="102.11.11.45" # the ipaddress of your target tcp interface
NAME="nvme-ssd"  # this can be whatever you want
DEV="/dev/nvme0n1"     # you can use any block device

mkdir /sys/kernel/config/nvmet/subsystems/${NAME}0
mkdir /sys/kernel/config/nvmet/subsystems/${NAME}0/namespaces/1
echo -n ${DEV}
>/sys/kernel/config/nvmet/subsystems/${NAME}0/namespaces/1/device_path
echo 1 > /sys/kernel/config/nvmet/subsystems/${NAME}0/attr_allow_any_host
echo 1 > /sys/kernel/config/nvmet/subsystems/${NAME}0/namespaces/1/p2pmem
echo 1 > /sys/kernel/config/nvmet/subsystems/${NAME}0/namespaces/1/enable

mkdir /sys/kernel/config/nvmet/ports/1
echo "ipv4" > /sys/kernel/config/nvmet/ports/1/addr_adrfam
echo "tcp" > /sys/kernel/config/nvmet/ports/1/addr_trtype
echo $IPPORT > /sys/kernel/config/nvmet/ports/1/addr_trsvcid
echo $IPADDR > /sys/kernel/config/nvmet/ports/1/addr_traddr

ln -s /sys/kernel/config/nvmet/subsystems/${NAME}0
/sys/kernel/config/nvmet/ports/1/subsystems/${NAME}0
```

Kernel NVMe/TCP Host Configuration

i. Disable virtualization, c-state technology, VT-d, Intel I/O AT, and SR-IOV in system BIOS.
ii. Install the latest Chelsio Unified Wire v3.18.0.1 for Linux.
iii. Set the below tuned-adm profile.

```
[root@host~]# tuned-adm profile latency-performance
```

iv. Load the Chelsio NIC driver (*cxgb4*) and bring up the interface with an IPv4 address.

```
[root@host~]# modprobe cxgb4
[root@host~]# ifconfig ethX <IPv4 address> up
[root@host~]# ethtool -s ethX speed 1000 duplex full autoneg off
```

v. Load the NVMe/TCP drivers.

```
[root@host~]# modprobe nvme
[root@host~]# modprobe nvme-tcp
```

vi. Connect to the target.

```
[root@host~]# nvme connect -n nvme-ssd0 -t tcp -a <target_ip> -s 4420 -i 1 -d
```

vii. Run the *fio* tool.

```
[root@host~]# fio --rw=randread --ioengine=libaio --name=random --
invalidate=1 --direct=1 --runtime=60 --time_based --group_reporting --
filename= /dev/nvme0n1 --iodepth=1 --numjobs=1 --bs=4K --eta-newline=1 --
readonly --ramp_time=2
...
  read: IOPS=3671, BW=14.3MiB/s (15.0MB/s)(860MiB/60001msec)
    slat (nsec): min=2098, max=14484, avg=3700.25, stdev=262.44
    clat (usec): min=212, max=1582, avg=268.18, stdev=106.25
     lat (usec): min=215, max=1585, avg=271.96, stdev=106.26
    clat percentiles (usec):
     |  1.00th=[  215],  5.00th=[  215], 10.00th=[  215], 20.00th=[  217],
     | 30.00th=[  217], 40.00th=[  243], 50.00th=[  251], 60.00th=[  265],
     | 70.00th=[  277], 80.00th=[  289], 90.00th=[  302], 95.00th=[  318],
     | 99.00th=[ 1090], 99.50th=[ 1123], 99.90th=[ 1123], 99.95th=[ 1139],
     | 99.99th=[ 1156]
   bw (  KiB/s): min=12848, max=15800, per=100.00%, avg=14711.01,
stdev=850.78, samples=119
   iops        : min= 3212, max= 3950, avg=3677.75, stdev=212.70,
samples=119
  lat (usec)   : 250=48.43%, 500=48.71%, 750=1.73%, 1000=0.02%
  lat (msec)   : 2=1.11%
  cpu          : usr=0.35%, sys=1.16%, ctx=440499, majf=0, minf=58
  ...

Run status group 0 (all jobs):
   READ: bw=14.3MiB/s (15.0MB/s), 14.3MiB/s-14.3MiB/s (15.0MB/s-15.0MB/s),
io=860MiB (902MB), run=60001-60001msec

Disk stats (read/write):
  nvme0n1: ios=0/0, merge=0/0, ticks=0/0, in_queue=0, util=0.00%
```

NOTE: These numbers are only preliminary results on the emulation platform. They will further improve with an actual adapter.

Conclusion

The Chelsio T7 NVMe/TCP PDU Offload solution enables Kioxia NVMe SSDs to be shared, pooled, and managed more effectively across a low-latency, high-performance network. The Chelsio offload solution frees up significant CPU resources for application processing. This means all storage and networking traffic runs over a single network, rather than building and maintaining multiple networks, resulting in significant acquisition and operational cost savings. T7 and Kioxia combination delivers a unique CMB feature for performance while enabling a number of storage wire protocols or Ethernet or other inline protocols for computational storage space – all with a single ASIC, and a single FW.

Related Links

T7 Product Brief
NVMe/TCP Offload Demonstration on T7 Emulation Platform