100G NVMe-oF TCP

Chelsio T6: Bandwidth, IOPS and Latency Performance

NVMe over Fabrics specification extends the benefits of NVMe to large fabrics, beyond the reach and scalability of PCIe. NVMe over Fabrics (NVMe-oF) based on TCP is a new technology which enables the use of NVMe-oF over existing Datacenter IP networks. Chelsio's TOE (TCP Offload Engine) is fully capable of offloading TCP/IP processing to hardware at 100Gbps and provides a low latency, high throughput, plug-and-play Ethernet solution for connecting high performance NVMe SSDs over a scalable, congestion controlled and traffic managed fabric, with no special configuration needed. The unique ability of a *TOE* to perform the full transport layer functionality in hardware is essential to obtaining tangible benefits. The vital aspect of the transport layer is process-to-process communication, i.e. the data passed to the *TOE* comes straight from the application process, and the data delivered by the *TOE* goes straight to the application process.

The Terminator series adapters provide a powerful zero copy capability for regular TCP connections, requiring no changes to the sender or receiver applications, to deliver line rate performance at minimal CPU and memory utilization, interrupts and context switches. This paper presents the significant performance benefits of Chelsio T6 NVMe-oF over 100GbE TOE solution. Chelsio's T6 adapter delivers line-rate throughput and more than 2.6 Million IOPS at 4K I/O size. In addition, with only 12 µs delta latency between remote and local storage, Chelsio's solution proves to be best-in-breed providing the next generation, scalable storage network over standard and cost-effective Ethernet infrastructure with an efficient processing path.

Test Results

The following graph compares NVMe-oF READ, WRITE IOPS and throughput results over Chelsio TOE & regular NIC (TCP) modes using Null Block devices. The results are collected using the **fio** tool with I/O size varying from 4 to 512 KBytes with an access pattern of random READs and WRITEs.

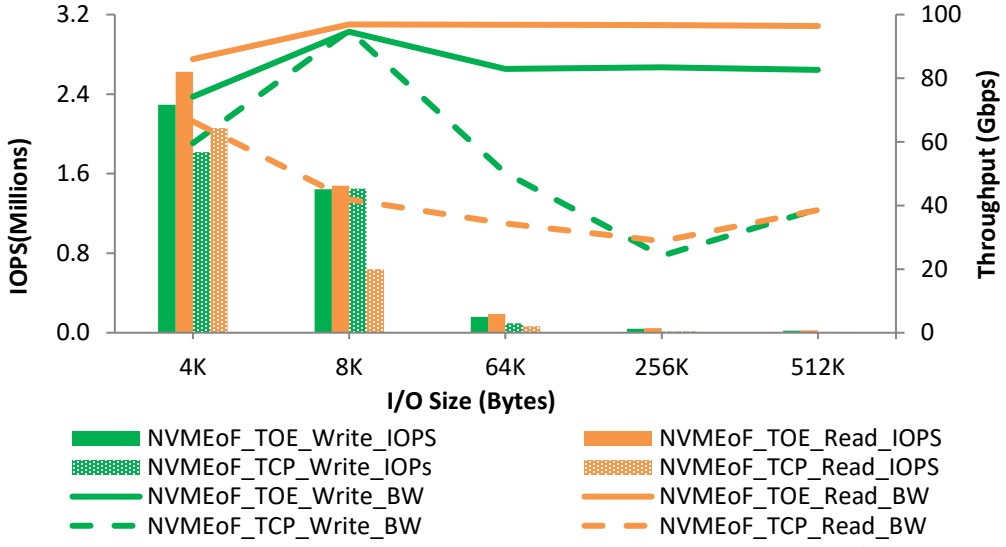


Figure 1 – NVMe-oF TOE vs TCP, IOPS and Throughput vs. I/O size

As evident from the graph above, T6 NVMe-oF TOE solution delivers line-rate throughput for READ and a stable throughput for WRITE. In NIC Mode, the throughput is not consistent and stable, and fails to reach line-rate. In fact, this is one of the most important benefits of a hardware offloaded TCP solution – it isolates the host applications performance from the performance spikes caused by network traffic. No longer does the application have to get swapped out, so that the TCP stack can be brought back into the cache to retransmit or reorder a packet. An NVMe-TOE solution delivers a fully ordered, reliable data stream to the host. Chelsio's TOE solution is required to achieve 100 Gb/s and higher line-rate throughputs. READ IOPS exceeds 2.6 Million and Write IOPS exceed 2.2 Million at 4K I/O size which are far exceed compared to NIC mode.

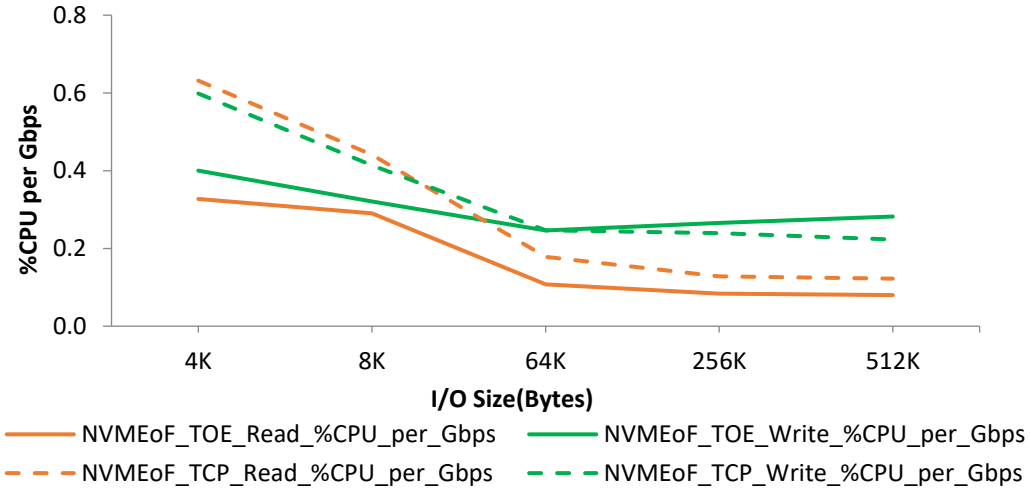


Figure 2 - NVMe-oF TOE vs TCP, %CPU per Gbps vs. I/O size

As seen from the graph above, TOE CPU consumption per Gbps on target is low compared to TCP for both READ and WRITE operations.

The following graph presents the NVMe-oF TOE latency numbers at 4K I/O size varying the number of connections.

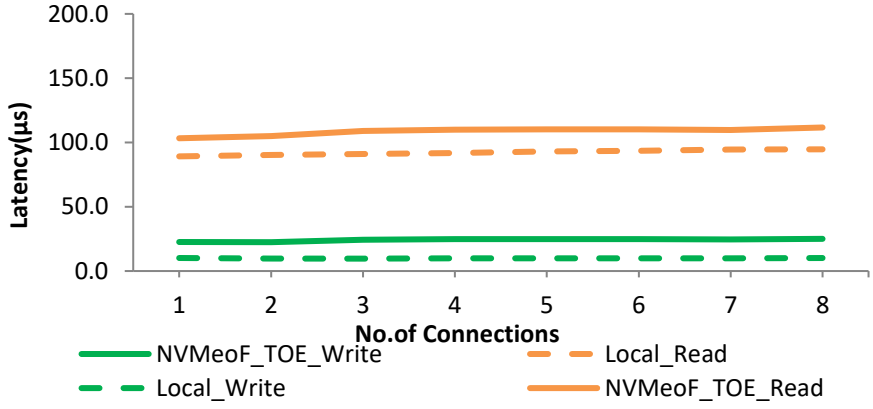
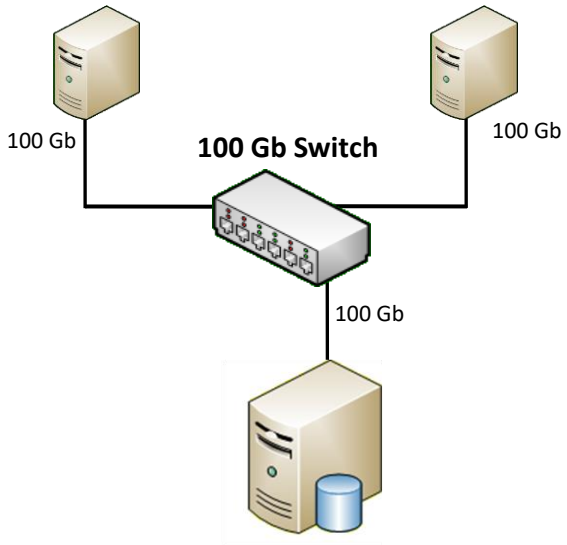


Figure 3 - Latency vs. #. Connections

From the graph above, observe that remote versus local NVMe device access adds only 12-14 µs latency at 4K I/O for both READ and WRITE operations.

The Demonstration



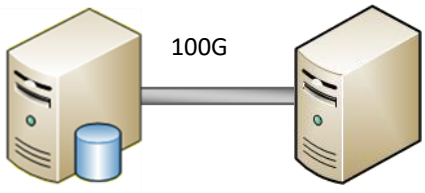
- Supermicro X10SRA-F Initiators with T62100-CR adapter
- 1 Intel Xeon CPU E5-1620 v4 4-core @ 3.50GHz (HT enabled)
- 32GB RAM
- RHEL 7.3 (4.18.0-rc6 kernel)

- Supermicro X10DRG-Q Target with T62100-CR adapter
- 2 Intel Xeon CPUs E5-2687W v4 12-core @ 3.00GHz (HT disabled)
- 128GB RAM
- RHEL 7.3 (4.18.0-rc6 kernel)

Figure 4 – Bandwidth/IOPs Test setup

The BW/IOPS test setup consists of an NVMe target machine connected to two initiator machines through a 100GbE switch using single port on each system. An MTU of 9000B is used on the ports under test.

- Supermicro X10DRG-Q Target with T62100-CR.
- 2 Intel Xeon CPUs E5-2687W v4 12-core @ 3.00GHz (HT disabled)
- 128GB RAM
- 4 INTEL 1.6 TB PCIe NVME SSDS



- Supermicro X10SRA-F Initiator with T62100-CR
- 1 Intel Xeon CPU E5-1620 v4 4-core @ 3.50GHz (HT disabled),
- 32GB RAM

Figure 5 – Latency Test setup

The Latency test setup consists of an NVMe target machine connected to a single initiator connected back to back using single port on each system. MTU of 9000B is used.

Storage configuration

For BW/IOPS test, 16 NVMe-oF TCP targets are created on the target server using NULL BLOCK devices, each 1GB in size. Each initiator connects to 8 targets using 4 connections per target.

For latency test, the target is configured with 1 - 8 LUNs using 4 Intel 1.6TB NVMe SSD's. One initiator connects to the target using one connection.

Setup Configuration

Target/Initiator Configuration

i. Disable virtualization, c-state technology, VT-d, Intel I/O AT, Hyperthreading and SR-IOV in system BIOS.
ii. Compile and install kernel http://git.infradead.org/nvme.git/shortlog/refs/heads/nvme-tcp and reboot the machine into the newly installed kernel.
iii. Install Chelsio Unified Wire drivers v3.10.0.0 drivers.

```
[root@host~]# make install
```

iv. For latency test, add *idle=poll* to the kernel command line.
v. Set the below tuned-adm profile for BW/IOPS test:

```
[root@host~]# tuned-adm profile network-throughput
```

Set the below tuned-adm profile for Latency test:

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

vi. Set the below sysctl parameters.

```
[root@host~]# sysctl -w net.ipv4.tcp_timestamps=0
[root@host~]# sysctl -w net.core.netdev_max_backlog=250000
[root@host~]# sysctl -w net.core.rmem_max=4194304
[root@host~]# sysctl -w net.core.wmem_max=4194304
[root@host~]# sysctl -w net.core.rmem_default=4194304
[root@host~]# sysctl -w net.core.wmem_default=4194304
[root@host~]# sysctl -w net.ipv4.tcp_rmem="4096 1048576 4194304"
[root@host~]# sysctl -w net.ipv4.tcp_wmem="4096 1048576 4194304"
```

vii. Load the Chelsio TOE driver and bring up interface with IPv4 address and MTU 9000.

```
[root@host~]# modprobe t4_tom
[root@host~]# ifconfig ethX <IP address> mtu 9000 up
[root@host~]# mount -t configfs none /sys/kernel/config
```

viii. Load the NVMe drivers.

```
[root@host~]# modprobe nvmet (On Target)
[root@host~]# modprobe nvmet-tcp (On Target)
[root@host~]# modprobe nvme-tcp (On Initiator)
```

ix. CPU affinity was set for BW/IOPs test.

```
[root@host~]# t4_perftune.sh -n -Q ofld -N
```

Target Configuration

BW/IOPs test:

i. Create 16 Null Block devices, each of 1GB size.

```
[root@host~]# modprobe null_blk nr_devices=16 gb=1 use_per_node_hctx=Y
```

ii. Configure the target using the below script:

```
#!/bin/bash
IPPORT="4420"
IPADDR="10.1.1.149"  # the ipaddress of your target TCP interface
NAME="nvme-nullb"    # Use "nvme-ssd" while configuring SSDs
DEV="/dev/nullb"     # Use "/dev/nvme" while configuring SSDs

for i in `seq 0 15`; do   # For latency test, use `seq 0 7`
mkdir /sys/kernel/config/nvmet/subsystems/${NAME}${i}
mkdir -p /sys/kernel/config/nvmet/subsystems/${NAME}${i}/namespaces/1
echo -n ${DEV}${i}
>/sys/kernel/config/nvmet/subsystems/${NAME}${i}/namespaces/1/device_path
# use 'echo -n ${DEV}${i}n1' while configuring SSDs
echo 1 > /sys/kernel/config/nvmet/subsystems/${NAME}${i}/attr_allow_any_host
echo 1 > /sys/kernel/config/nvmet/subsystems/${NAME}${i}/namespaces/1/enable
done

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

for i in `seq 0 15`; do   # For latency test, use `seq 0 7`
                ln -s /sys/kernel/config/nvmet/subsystems/${NAME}${i}
/sys/kernel/config/nvmet/ports/1/subsystems/${NAME}${i}
done
echo '...Done'
```

Initiator Configuration

Target was discovered:

```
[root@host~]# nvme discover -t tcp -a <target_ip> -s 4420
```

BW/IOPs test:

i. Initiator1 connected to Target.

```
[root@host1~]# for i in `seq 0 7`; do /root/nvme-cli/nvme connect -t tcp -s
4420 -a 10.1.1.149 -n nvme-nullb${i} -i 4; done
```

ii. Initiator2 connected to Target.

```
[root@host2~]# for i in `seq 8 15`; do /root/nvme-cli/nvme connect -t tcp -s
4420 -a 10.1.1.149 -n nvme-nullb${i} -i 4; done
```

iii. *fio* tool was run on both initiators.

```
[root@host~]# fio --rw=randwrite/randread --ioengine=libaio --name=random  --
size=400m --invalidate=1 --direct=1 --runtime=30 --time_based --
fsync_on_close=1 --group_reporting --filename= <device list> --iodepth=64 --
numjobs=16 --bs=<value>
```

Latency test:

i. Single Initiator connects to the target:

```
[root@host~]# for i in `seq 0 $((N))`; do /root/nvme-cli/nvme connect -t tcp -
a 10.1.1.149 -n nvme-ssd${i} -i 1; done
```

where N specifies the number of target devices.

ii. *fio* tool was run on the initiator.

```
[root@host~]# fio --rw=randwrite/randread --ioengine=libaio --name=random  --
size=400m --invalidate=1 --direct=1 --runtime=30 --time_based --
fsync_on_close=1 --group_reporting --filename= <device list> --iodepth=1 --
numjobs=1 --bs=4K
```

iii. After collecting latency for single target device, initiator was disconnected and logs into multiple target devices for latency collection. This procedure is repeated from 1 to 8 target devices.

Conclusion

This paper showcases the remote storage access performance capabilities of Chelsio T6 NVMe-oF over 100GbE TOE solution and proves significant benefits over regular TCP (NIC mode). Using TOE enables the NVMe storage devices to be shared, pooled and managed more effectively across a low latency, high performance network. The results show that Chelsio's TOE:

- delivers line-rate throughput for READ and stable BW for WRITE.
- reaches more than 2.6 Million READ IOPS and 2.2 Million WRITE IOPS at 4K I/O size.
- remote versus local NVMe device access adds only 12-14 µs latency at 4K I/O for both READ and WRITE operations.
- Isolates the host application performance from performance spikes caused by the network.

TOE improves performance for all TCP applications while freeing up CPU resources to be used for the application processing. Using a Chelsio adapter along with the Unified Wire Software package available as part of the Chelsio solution, users can create and maintain a true Converged Fabric cluster where all storage and networking traffic runs over a single 25/100Gb network, rather than having to build and maintain multiple networks, resulting in significant acquisition and operational savings.

Related Links

100G SPDK NVMe over Fabrics
NVMe-oF with iWARP and NVMe/TCP
NVMe over Fabrics Performance JBOF
NVMe over Fabrics Performance for AMD EPYC
High Performance NVMe-oF with T6 100G iWARP RDMA
NVMe Over Fabrics Performance for Qualcomm ARM