Agenda

• What is SPDK?
• Accelerated NVMe-oF via SPDK
• Conclusion
Agenda

• What is SPDK?
• Accelerated NVMe-oF via SPDK
• Conclusion
Storage Performance Development Kit

**Scalable and Efficient Software Ingredients**
- User space, lockless, polled-mode components
- Up to millions of IOPS per core
- Designed to extract maximum performance from non-volatile media

**Storage Reference Architecture**
- Optimized for *latest generation CPUs and SSDs*
- Open source composable building blocks (BSD licensed)
- Available via [spdk.io](https://spdk.io)
Benefits of using **SPDK**

**SPDK**

- More performance from Intel CPUs, non-volatile media, and networking

**Up to 10X MORE** IOPS/core for NVMe-oF* vs. Linux kernel

**Up to 8X MORE** IOPS/core for NVMe vs. Linux kernel

**Up to 350% BETTER** Tail Latency for RocksDB workloads

**FASTER TTM/LESS RESOURCES** than developing components from scratch

Provides **Future Proofing** as NVM technologies increase in performance

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to [http://www.intel.com/performance](http://www.intel.com/performance)
### Architecture

#### Storage Protocols
- NVMe-of* Target
- iSCSI Target
- vhost-scsi Target
- vhost-blk Target
- Linux nbd

#### Storage Services
- **Block Device Abstraction (bdev)**
  - 3rd Party
  - Logical Volumes
  - GPT
  - DPDK Encryption
  - QoS
  - BlobFS
  - Blobstore

- **NVMe Devices**
  - NVMe-of* Initiator
  - NVMe+ PCIe Driver

- **Drivers**
  - Intel® QuickData Technology Driver

#### Integration
- Cinder
- VPP TCP/IP
- RocksDB
- Ceph
- QEMU

#### Core
- Application Framework

#### Released
- New release 18.01
- 1H’18

---

*Note: NVMe-of* denotes NVMe over Fiber Channel.*
SPDK ENVIRONMENT ABSTRACTION
WHY AN ENVIRONMENT ABSTRACTION?

FLEXIBILITY FOR USER
ENVIRONMENT ABSTRACTION

- Memory allocation (pinned for DMA) and address translation
- PCI enumeration and resource mapping
- Thread startup (pinned to cores)
- Lock-free ring and memory pool data structures
Configurable:

```
./configure --with-env=...
```

Interface defined in spdk/env.h

Default implementation uses **DPDK** (lib/env_dpdk)

**FLEXIBILITY: DECOUPLING AND DPDK ENHANCEMENTS**
APPLICATION FRAMEWORK
HOW DO WE COMBINE SPDK COMPONENTS?

THE SPDK APP FRAMEWORK PROVIDES THE GLUE
APP FRAMEWORK COMPONENTS

Reactor
Poller
Event
I/O Channel
**Poller**

Essentially a “task” running on a reactor
Primarily checks hardware for async events
Can run periodically on a timer

Example: poll completion queue
Callback runs to completion on reactor thread
Completion handler may send an event
EVENT
EVENT

Cross-thread communication
Function pointer + arguments
One-shot message passed between reactors
Multi-producer/single-consumer ring
Runs to completion on reactor thread
I/O CHANNEL

Abstracts hardware I/O queues
Register I/O devices
Create I/O channel per thread/device combination
Provides hooks for driver resource allocation
I/O channel creation drives poller creation
Pervasive in SPDK
NVMe OVER FABRICS TARGET EXAMPLE

Acceptor network poller handles connect events
Connection event registers new poller
NVME OVER FABRICS TARGET EXAMPLE

Acceptor network poller handles connect events
Connection event registers new poller
I/O request arrives over network
I/O submitted to storage
Storage device poller checks completions
Response sent

ALL ASYNCHRONOUS WORK IS DRIVEN BY POLLERS
Agenda

• What is SPDK?
• Accelerated NVMe-oF via SPDK
• Conclusion
SPDK NVMe-oF Components

NVMe over Fabrics Target

- Released July 2016 (with spec)
- **Hardening:**
  - Intel test infrastructure
  - Discovery simplification
  - Correctness & kernel interop
- **Performance improvements:**
  - Read latency improvement
  - Scalability validation (up to 150Gbps)
  - Event Framework enhancements
  - Multiple connection performance improvement

NVMe over Fabrics Host (Initiator)

- New component added in Dec 2016
- **Performance improvements**
  - Eliminate copy: now true zero-copy
  - SGL (single SGL element)
SPDK NVMe-oF transport work

Existing work: RDMA transport

- DPDK components used which is encapsulated in libspdk_env_dpdk.a, e.g.,
  - PCI device management
  - CPU/thread scheduling
  - Memory management (e.g., lock free rings)
  - Log management

Upcoming work: TCP transport

- Kernel based TCP transport
- VPP/DPDK based user space TCP transport
  - Use DPDK Ethernet PMDs
  - Use user space TCP/IP stack (e.g., VPP)
SPDK reduces NVMe over Fabrics software overhead up to 10x!

**NVMe* over Fabrics Target Features**

<table>
<thead>
<tr>
<th>Target Features</th>
<th>Realized Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utilizes NVM Express* (NVMe)</td>
<td>Reduced overhead per NVMe I/O</td>
</tr>
<tr>
<td>Polled Mode Driver</td>
<td></td>
</tr>
<tr>
<td>RDMA Queue Pair Polling</td>
<td>No interrupt overhead</td>
</tr>
<tr>
<td>Connections pinned to CPU cores</td>
<td>No synchronization overhead</td>
</tr>
</tbody>
</table>

**SPDK vs. Kernel NVMe-oF I/O Efficiency**

- **Rate**
  - SPDK: 3 IOPS, 30 Cores
  - Kernel: 3x 50GbE Line

**System Configuration**

- **Target System**: Supermicro SYS-2028U-TN24R4T+, 2x Intel® Xeon® E5-2699v4 (HT off), Intel® Speed Step enabled, Intel® Turbo Boost Technology enabled, 8x 8GB DDR4 2133 MT/s, 1 DIMM per channel, 12x Intel® P3700 NVMe SSD (800GB) per socket, 10G P2P Network, Mellanox® ConnectX-4 Lx 2x 25Gb RDMA, direct connection between initiators and target, Initiator OS: CentOS® Linux® 7.2, Linux kernel 4.10.0, Target OS (SPDK): Fedora 25, Linux kernel 4.9.11, Target OS (Linux kernel): Fedora 25, Linux kernel 4.9.11

Performance measured by: fio, 4KB Random Read I/O, 2 RDMA QP per remote SSD, Numjobs=4 per SSD, Queue Depth: 32/job. SPDK commit ID: 4163626c5c
NVM Express* Driver Software Overhead

SPDK reduces NVM Express* (NVMe) software overhead up to 10x!

System Configuration: 2x Intel® Xeon® E5-2695v4 (HT off), Intel® Speed Step enabled, Intel® Turbo Boost Technology disabled, 8x 8GB DDR4 2133 MT/s, 1 DIMM per channel, CentOS* Linux* 7.2, Linux kernel 4.7.0-rc1, 1x Intel® P3700 NVMe SSD (800GB), 4x per CPU socket, FW 8DV10102, I/O workload 4KB random read, Queue Depth: 1 per SSD, Performance measured by Intel using SPDK overhead tool, Linux kernel data using Linux AIO

<table>
<thead>
<tr>
<th>Kernel Source of Overhead</th>
<th>SPDK Approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interrupts</td>
<td>Asynchronous Polled Mode</td>
</tr>
<tr>
<td>Synchronization</td>
<td>Lockless</td>
</tr>
<tr>
<td>System Calls</td>
<td>User Space Hardware Access</td>
</tr>
<tr>
<td>DMA Mapping</td>
<td>Hugepages</td>
</tr>
<tr>
<td>Generic Block Layer</td>
<td>Specific for Flash Latencies</td>
</tr>
</tbody>
</table>
NVM Express* Driver Throughput Scalability

- Systems with multiple NVM Express* (NVMe) SSDs capable of millions of I/O per second
- Results in many cores of software overhead with kernel-based interrupt-driven driver model
- SPDK enables:
  - more CPU cycles for storage services
  - lower I/O latency

SPDK saturates 8 NVMe SSDs with a single CPU core!

System Configuration: 2x Intel® Xeon® E5-2695v4 (HT off), Intel® Speed Step enabled, Intel® Turbo Boost Technology disabled, 8x 8GB DDR4 2133 MT/s. 1 DIMM per channel, CentOS* Linux* 7.2, Linux kernel 4.10.0, 8x Intel® P3700 NVMe SSD (800GB), 4x per CPU socket, FW 8DV101H0, I/O workload 4KB random read, Queue Depth: 128 per SSD, Performance measured by Intel using SPDK perf tool, Linux kernel data using Linux AIO.
SPDK Host + Target vs. Kernel Host + Target

**Kernel NVMe-oF Target / Kernel NVMe-oF Initiator**
- 4K Random Read
- 4K Random Write

**SPDK NVMe-oF Target / SPDK NVMe-oF Initiator**
- 4K Random Read
- 4K Random Write

**Kernel vs. SPDK NVMe-oF Stacks**
Coldstream, Perf, qd=1

Latency (usec)

 Avg. I/O Round Trip Time

SPDK reduces Optane NVMe-oF latency by 44%, write latency by 36%!

System Configuration:
- 2x Intel® Xeon® E5-2695v4 (HT on, Intel® Speed Step enabled, Intel® Turbo Boost Technology enabled, 64GB DDR4 Memory, 8x 8GB DDR4 2400 MT/s, Ubuntu 16.04.1, Linux kernel 4.10.1, 1x 25GbE Mellanox 2P CX-4, CX-4 FW= 14.16.1020, mlx5_core= 3.0-1 driver, 1 ColdStream, connected to socket 0, 4KB Random Read I/O. 1 initiators, each initiator connected to bx NVMe-oF subsystems using 2P 25GbE Mellanox.

Performance measured by Intel using SPDK perf tool, 4KB Random Read I/O, Queue Depths: 1/NVMe-oF subsystem, numjobs=1, 300 sec runtime, direct=1, norandommap=1, FIO-2.12, SPDK commit # 42eade49.
NVMe-oF IO Latency Model, 4KB Random Read (Intel Optane SSD DC P4800X)

SPDK Target + Kernel NVMe-oF Initiator

- 20 usec round trip time measured from NVMe-oF initiator
- Out of 20 usec, ~7 usec spent in NVMe controller
- 12-13 usec measured time in the fabric and kernel NVMe-oF initiator
- SPDK NVMe target adds just 100-200 nsec to fabric overhead
SPDK Target + SPDK NVMe-oF Initiator

IO Read from SPDK NVMe-oF Initiator

Start: 0 usec

14 usec

- nvmf_read I/O start
- nvmf_read I/O complete

2 usec

9.15 usec

SPDK NVMe Target

- spdk_lib_read_start
- spdk_lib_read_complete

2.05 usec

9.05 usec

SPDK NVMe lib

Optane Controller

- SPDK NVMe-oF IO Latency Model, 4KB Random Read (Intel Optane SSD DC P4800X)

- 14 usec round trip time measured from NVMf client
- Out of 14 usec, ~7 usec spent in NVMe controller
- 7 usec measured time in the fabric and SPDK NVMe-oF initiator
- SPDK NVMe target adds just 100–200 nsec to fabric overhead

Disclaimer: Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. ©Other names and brands may be claimed as the property of others.

System Configuration: 2x Intel® Xeon® E5-2695v4 (HT on, Intel® Speed Step enabled, Intel® Turbo Boost Technology enabled, 64GB DDR4 Memory, 8x 8GB DDR4 2400 MT/s, Ubuntu 16.04.1, Linux kernel 4.16.1, Mellanox 2P CX-4, CX-4 FW= 14.16.1020, mlx5_core= 3.0-1 driver, 1 ColdStream, connected to socket 0, 4KB Random Read I/O. 1 initiators, each initiator connected to bx NVMe-oF subsystems using 2P 25GbE Mellanox. Performance measured by Intel using SPDK perf tool, 4KB Random Read I/O, Queue Depths: 1/NVMe-oF subsystem. numjobs=1, 300 sec runtime, direct=1, norandommap=1, FIO-2.12, SPDK commit #42eade49.
Agenda

• What is SPDK?
• Accelerated NVMe-oF via SPDK
• Conclusion
Conclusion

- In this presentation, we introduce
  - SPDK library
  - The accelerated NVMe-oF target built from SPDK library
  - SPDK proves to be useful to accelerate storage applications equipped with NVMe based devices
- Call for action:
  - Welcome to use SPDK in storage area (similar as using DPDK in network) and contribute into SPDK community.