End-to-End Data Protection with SPDK NVMe/TCP Target

05/09/2019

Shuhei Matsumoto

IT Platform Products Management Division
Hitachi, Ltd.

Contents

1. Introduction
2. DIF Support in SPDK NVMe/TCP Target
3. Performance Evaluation
4. Summary and Next Steps

1. Introduction

End-to-End Data Protection

HITACHI
Inspire the Next

- Data corruption can occur anywhere, and silent data corruption must be avoided.
- Data Integrity Field (DIF) provides a standardized end-to-end data protection mechanism that spans transport and protocol boundaries.

Create DIF with data creation ⟷ Check DIF through data path ⟷ Store data and DIF into storage media

- 8-byte DIF is associated with each data block.
  - Guard (GRD) tag contains a CRC of the data block.
  - Application (APP) tag is up to application.
  - Reference (REF) tag normally contains the lower 4 bytes of the associated Logical Block Address.

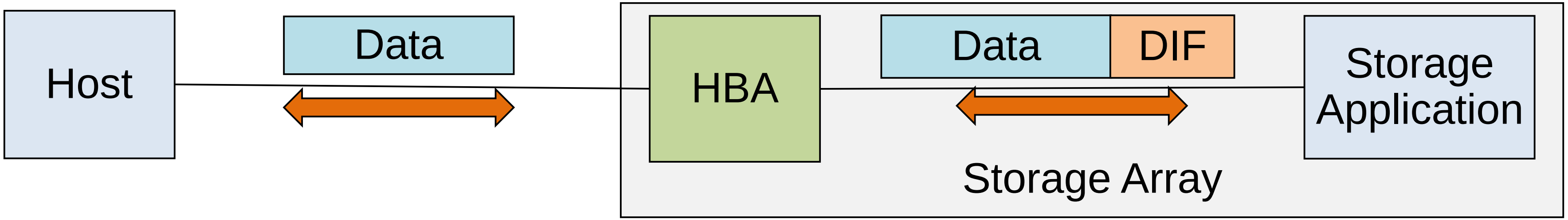
Data Block	Metadata	GRD	APP	REF

- There are many variables about DIF, metadata format, DIF settings, and DIF check types.

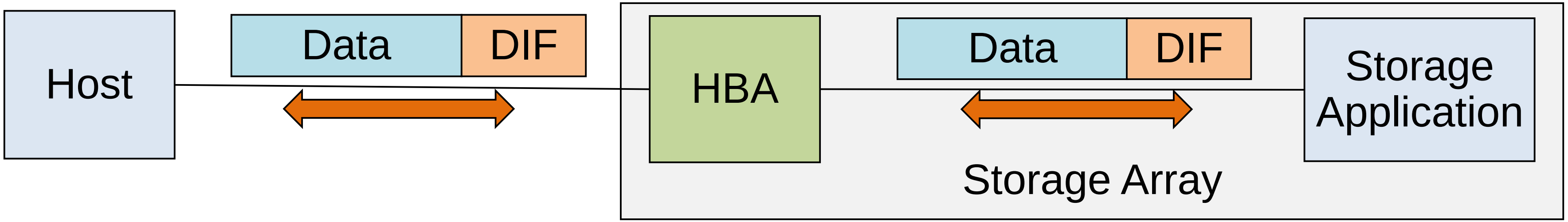
End-to-End Data Protection with Storage Arrays

HITACHI
Inspire the Next

- Dealing with data corruption is important for storage arrays and storage arrays have used DIF extensively.
- For some hosts that are not aware of DIF, iSCSI/FC HBAs have inserted/stripped DIF for write/read I/O.



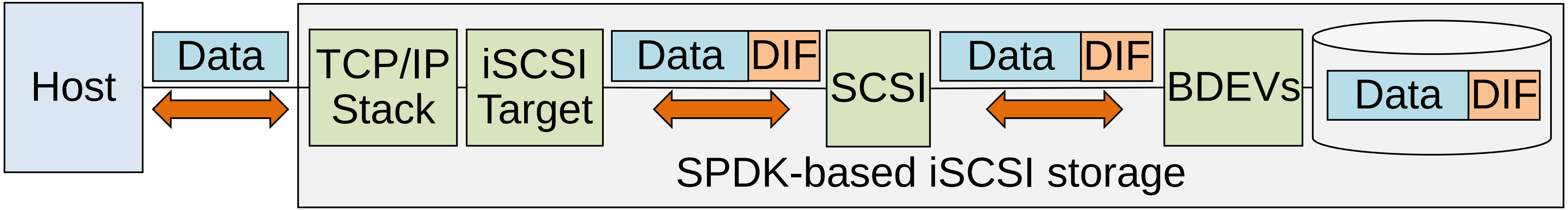
- For some hosts that are aware of DIF, iSCSI/FC HBAs pass data with DIF for read/write I/O.



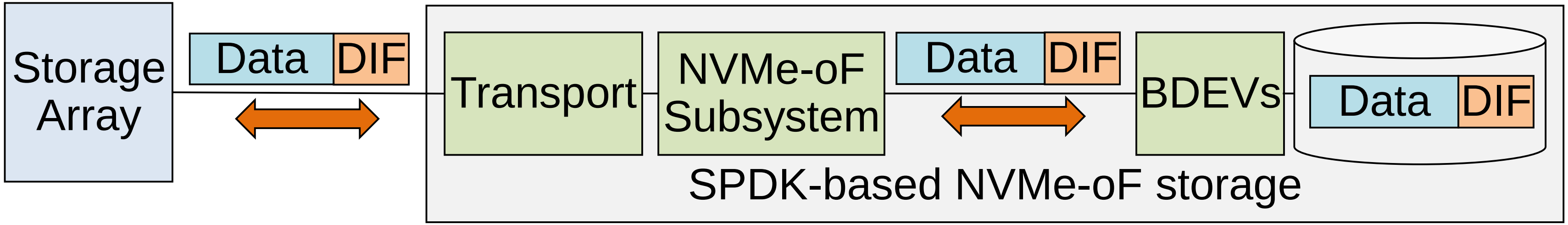
End-to-End Data Protection with SPDK 19.04

HITACHI
Inspire the Next

- DIF insert/strip feature in SPDK iSCSI target.

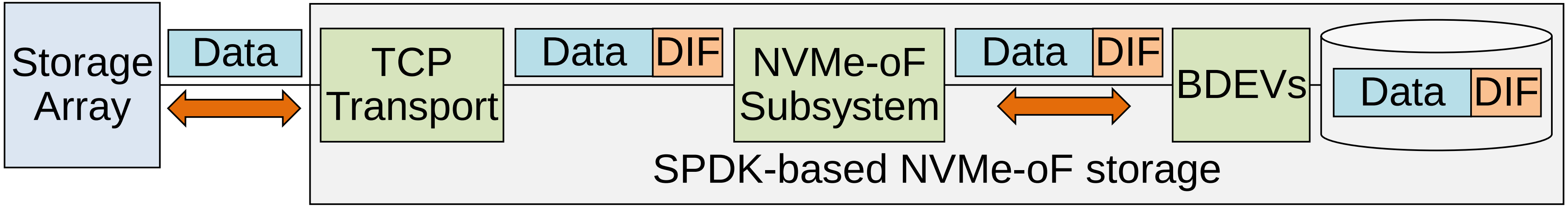


- DIF passthrough feature in SPDK NVMe-oF target

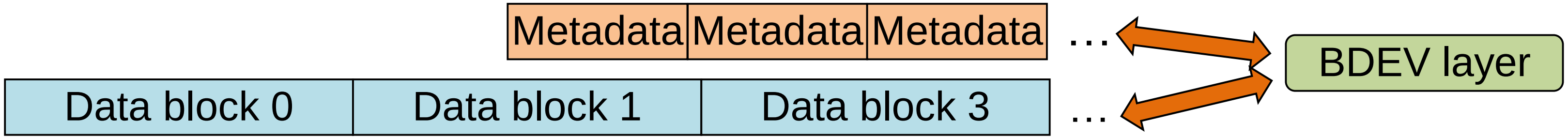


End-to-End Data Protection with SPDK 19.07

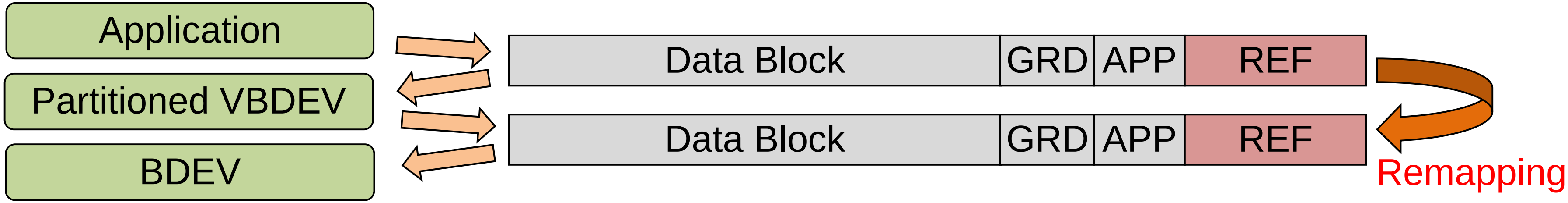
- DIF insert/strip feature in SPDK NVMe-oF target



- BDEV layer support I/O with separate metadata.



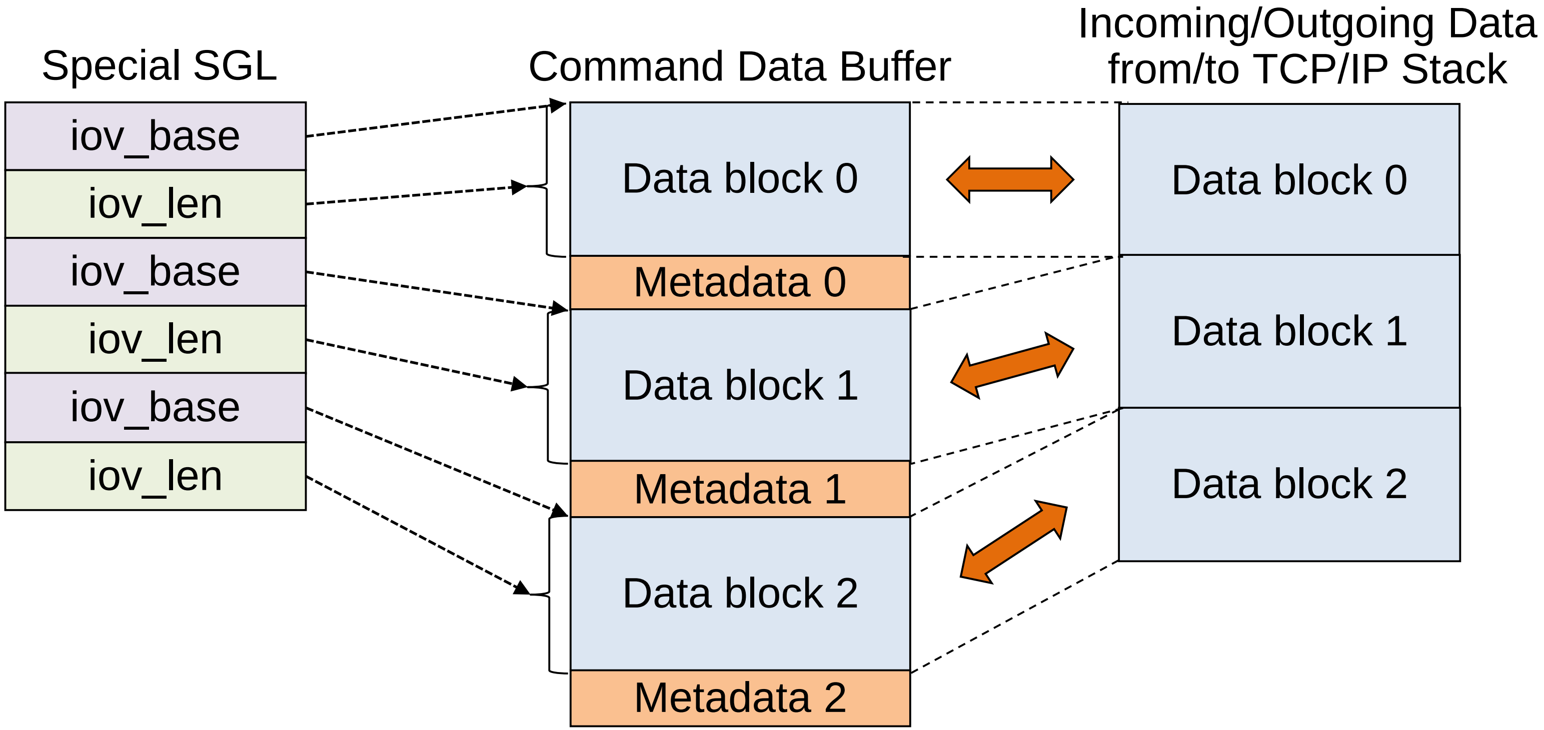
- Partitioned virtual bdev modules support DIF reference tag remapping.



2. DIF Support in SPDK NVMe/TCP Target

Key Idea: Special Scatter-Gather List to Insert or Strip DIF

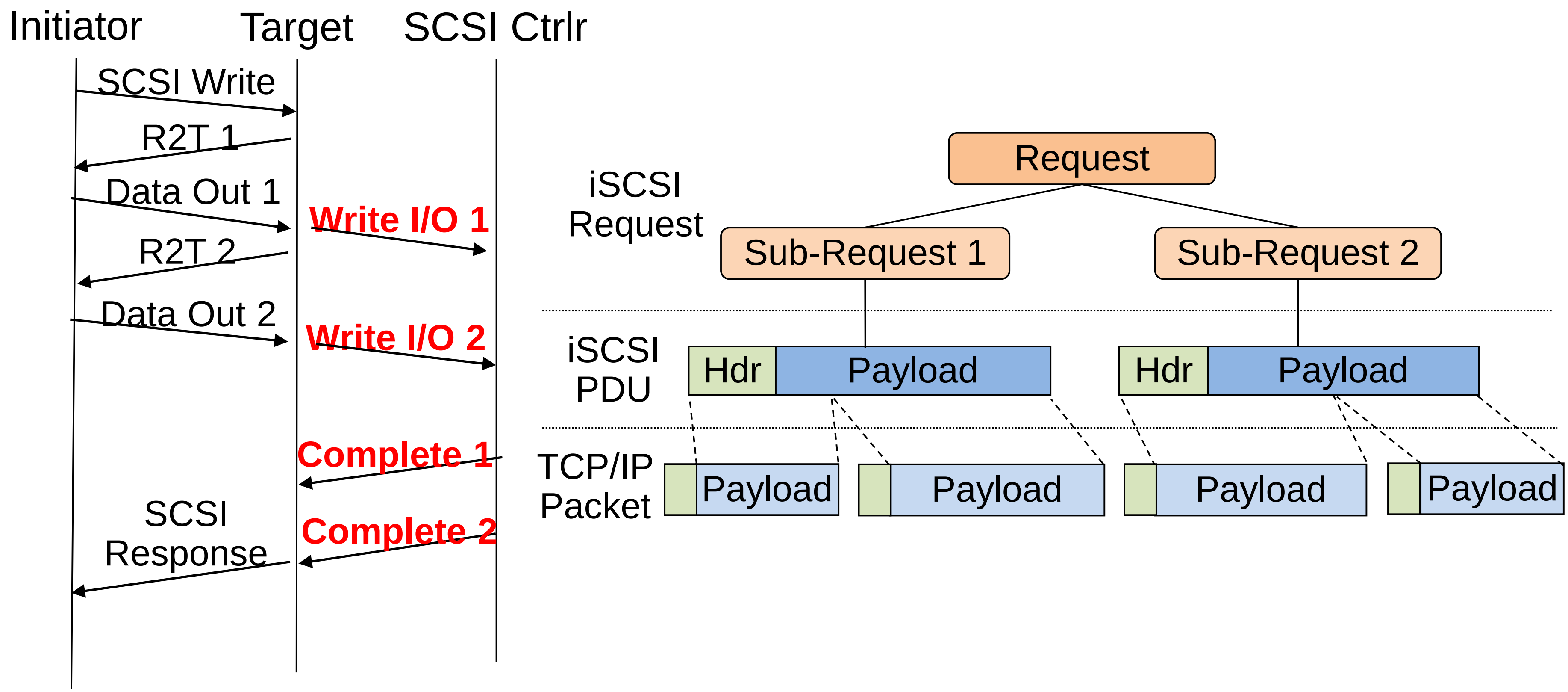
- NVMe/TCP target avoids extra data copy and any bounce buffer by using special Scatter Gather List (SGL).



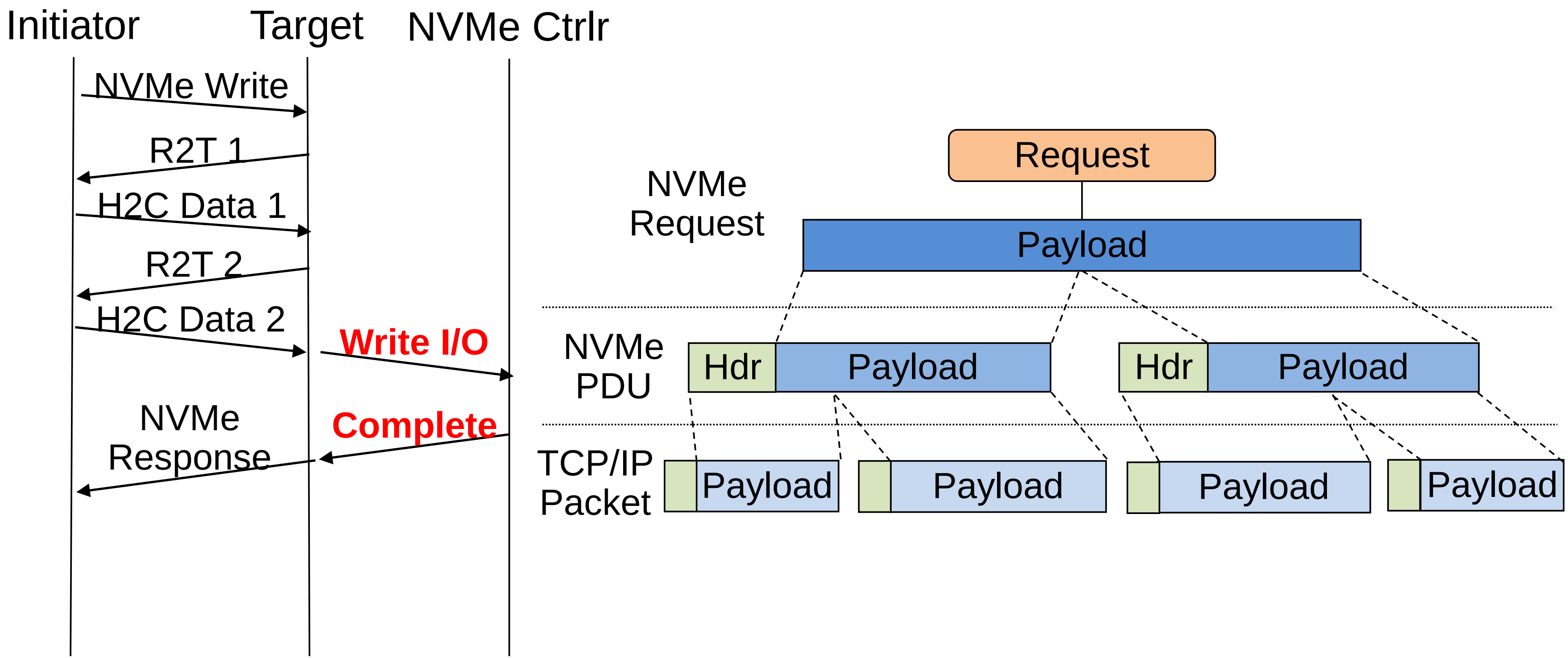
Differences between iSCSI target and NVMe/TCP Target

	iSCSI Target	NVMe/TCP Target
I/O Size Limit.	No. There is an optional feature, VPD Block Limit. Initiator may not implement it.	Yes. There is a mandatory feature, Maximum Data Transfer Size.
Supported DIF Mode	Only local mode (DIF insert/strip)	Both local mode and end-to-end mode (DIF pass-through)

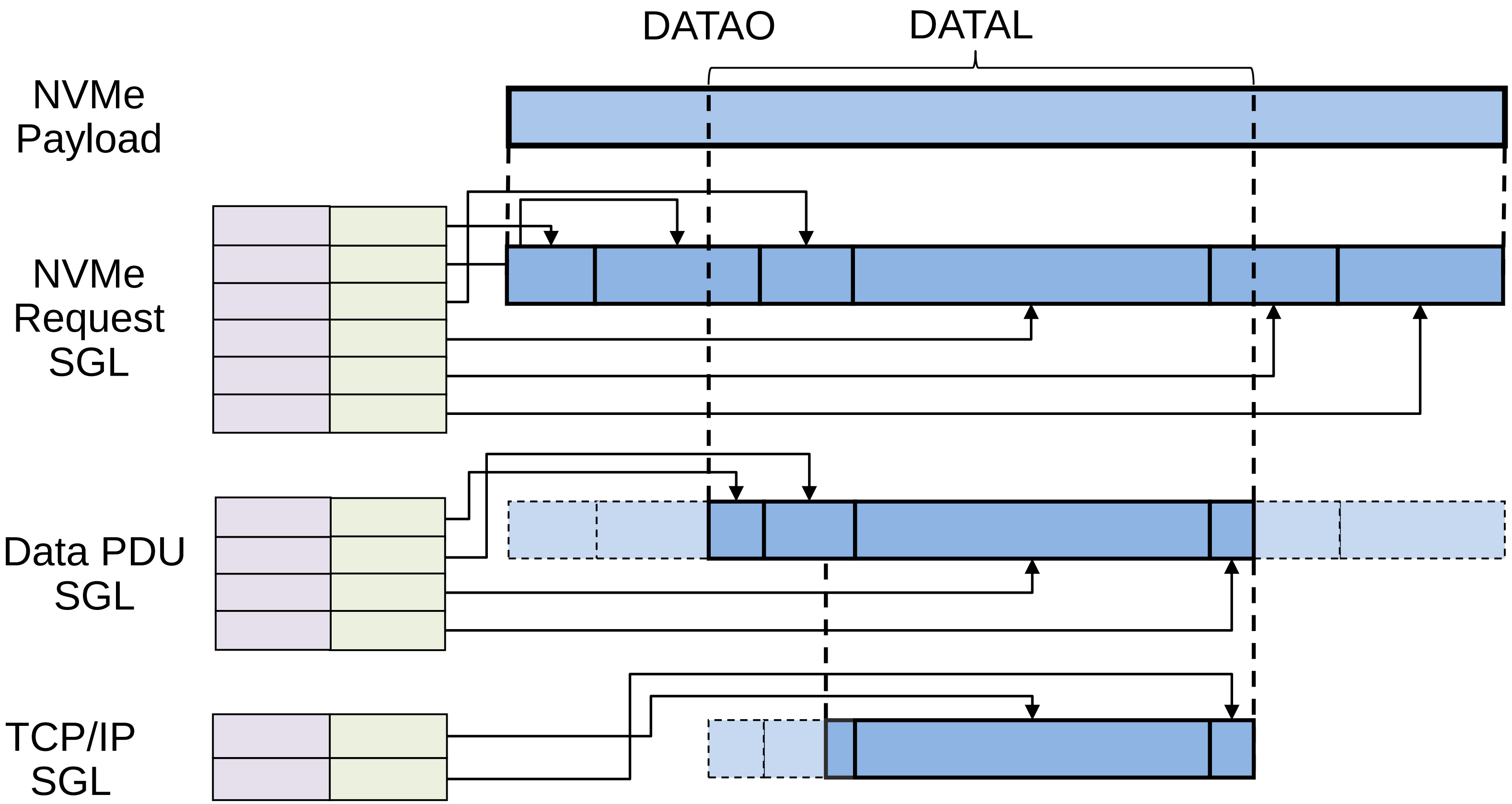
Write I/O Flow and Request Structure for iSCSI Target



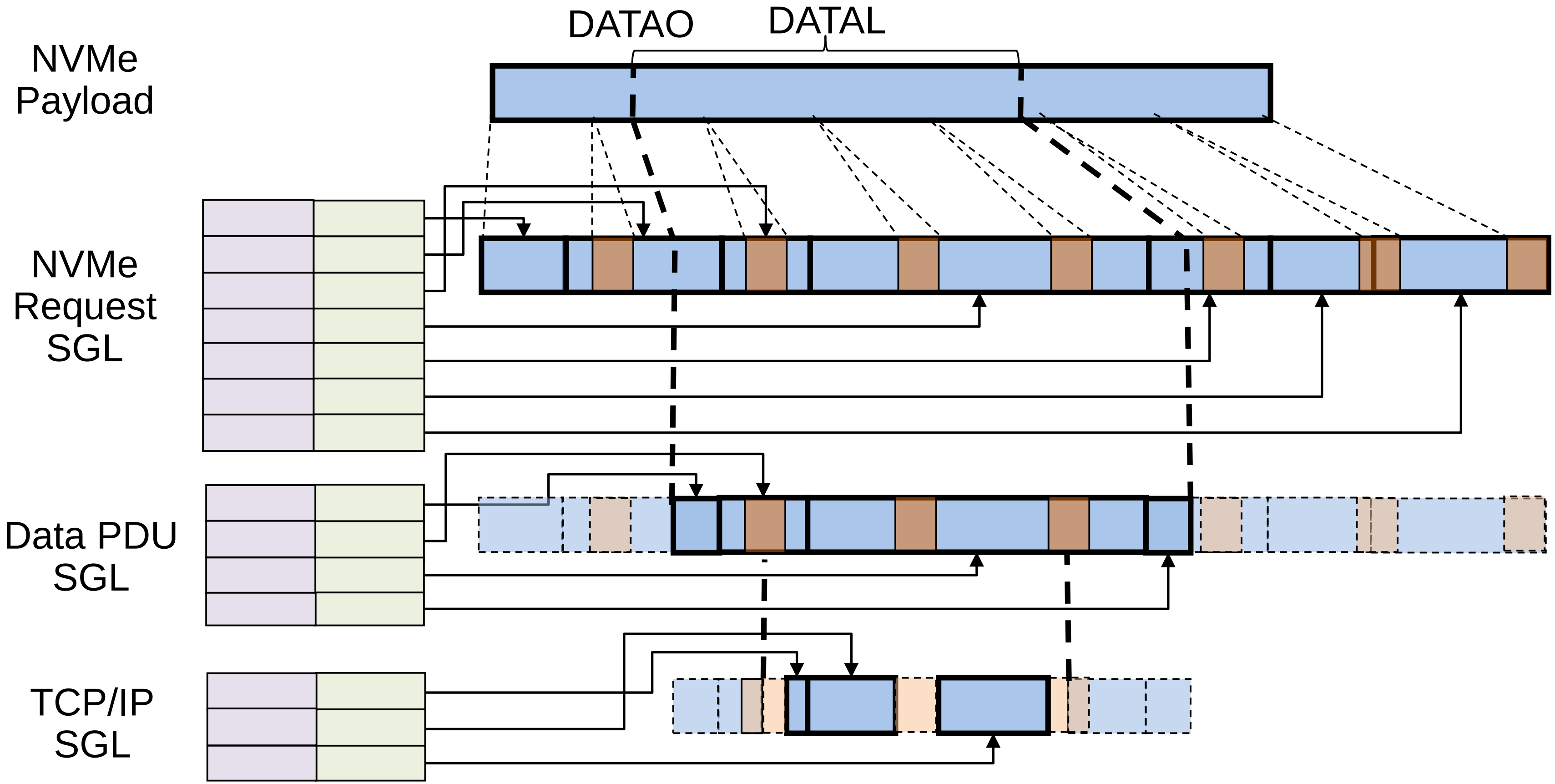
Write I/O Flow and Request Structure for NVMe/TCP Target



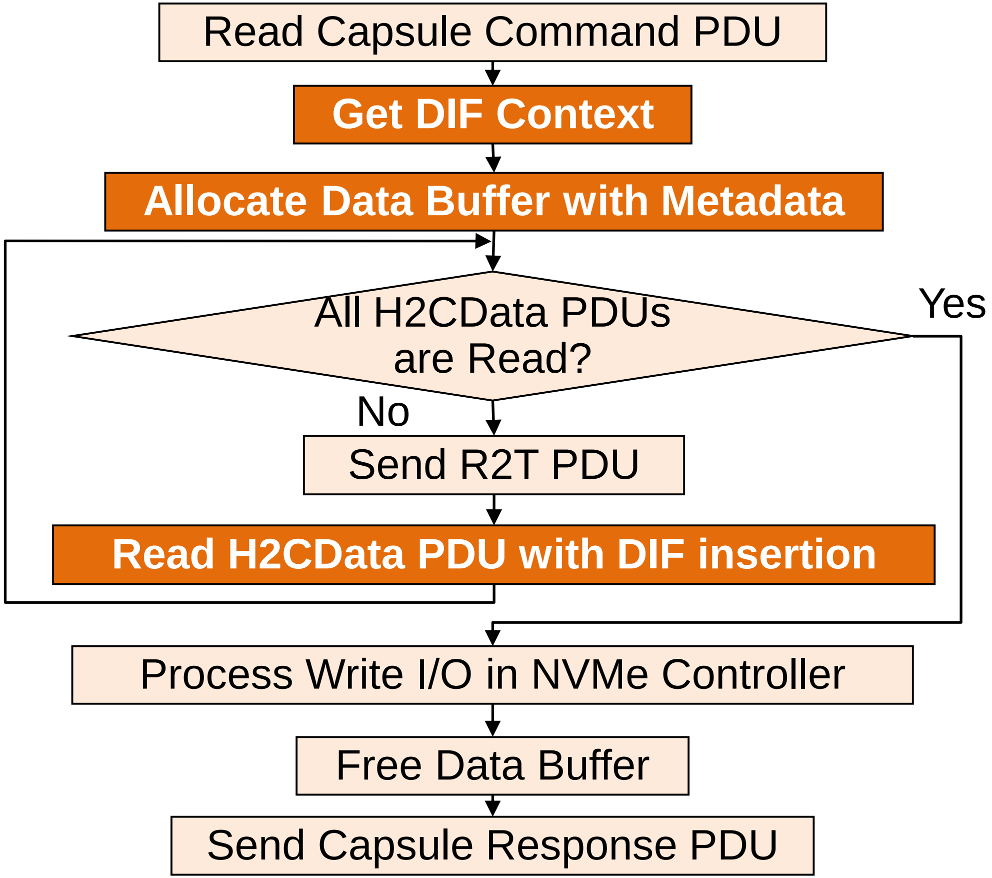
Three-Level SGLs for NVMe/TCP Request



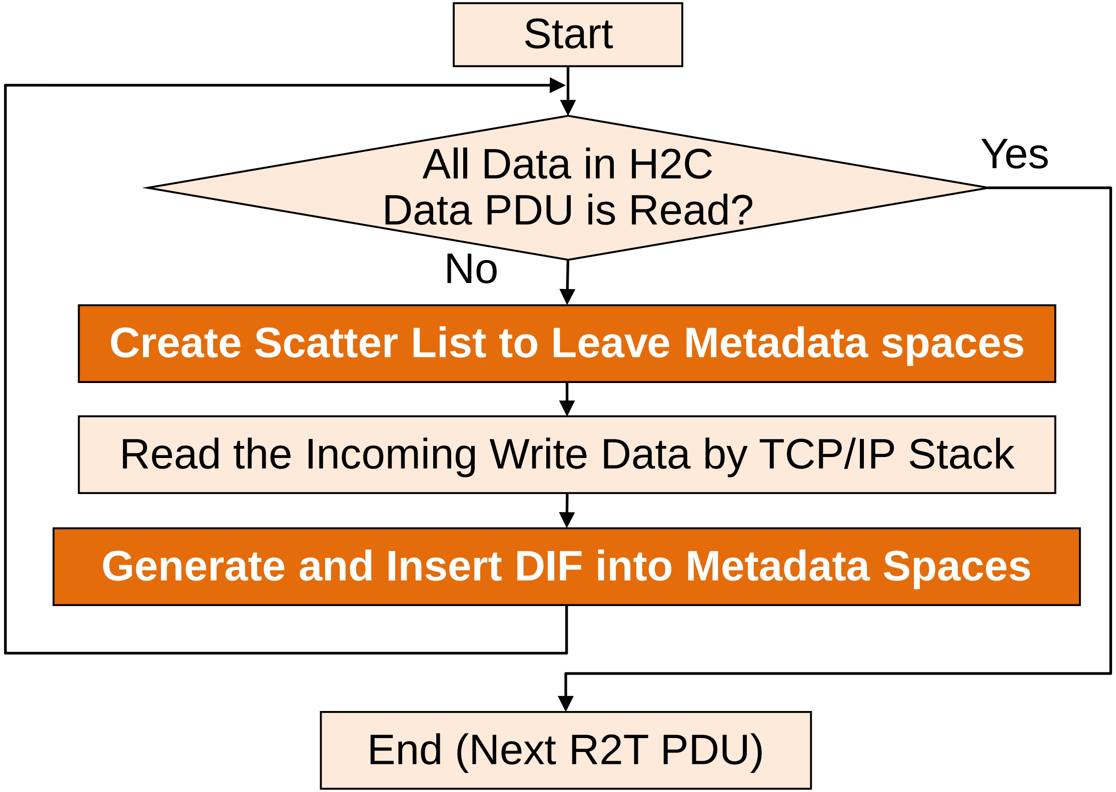
Three-Level SGLs with DIF insertion/strip for NVMe/TCP Req.



Write I/O flow with DIF Insertion

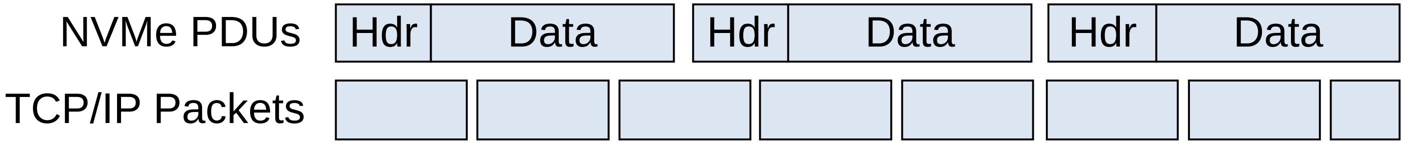


Read H2C Data PDU with DIF Insertion during Write I/O

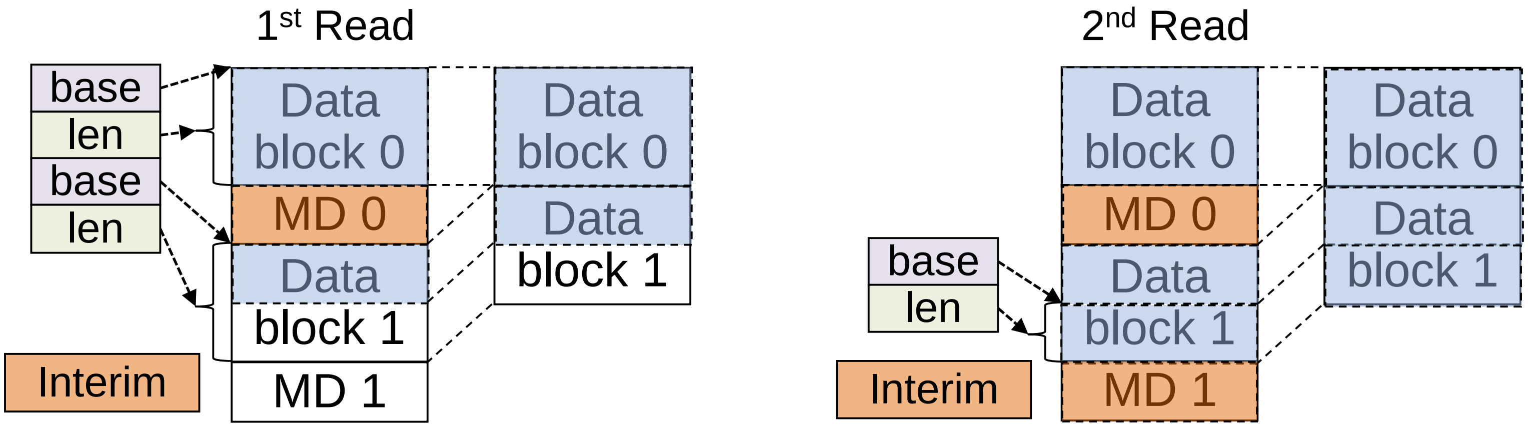


Process the Split Incoming Write Data with DIF Insertion

- Incoming write data may be split into multiple TCP packets.

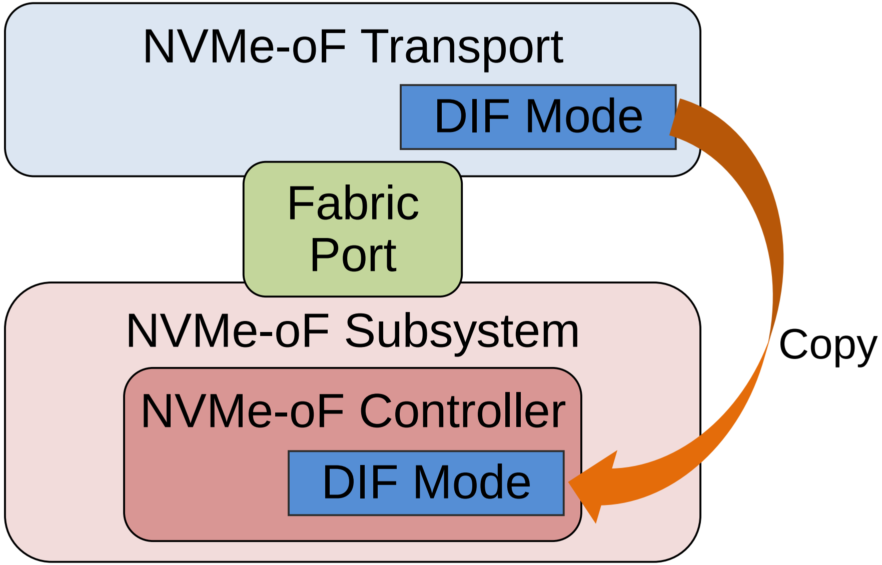


- NVMe/TCP target adjusts scatter-list before every read.
- We can split and merge CRC. Hence If the newly read ends in the middle of the data block, save the computed interim guard value. Then the next read uses it.



Hold DIF Mode Setting in NVMe-oF Controller

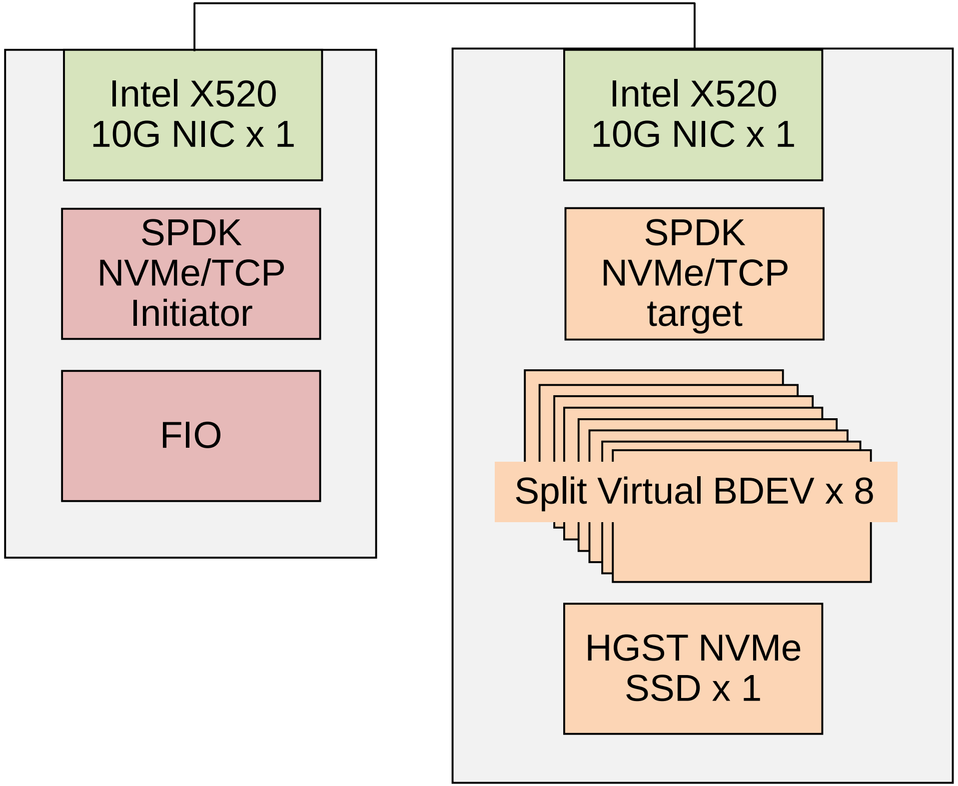
- SPDK 19.07 supports DIF mode per NVMe-oF transport (e.g. RDMA, TCP).
- NVMe-oF Controller controls whether DIF setting is exposed to NVMe-oF initiator according to the DIF mode.
- We may want to control DIF mode with finer granularity (e.g., per host) in future. NVMe-oF controller gets DIF mode from the transport at its initialization.



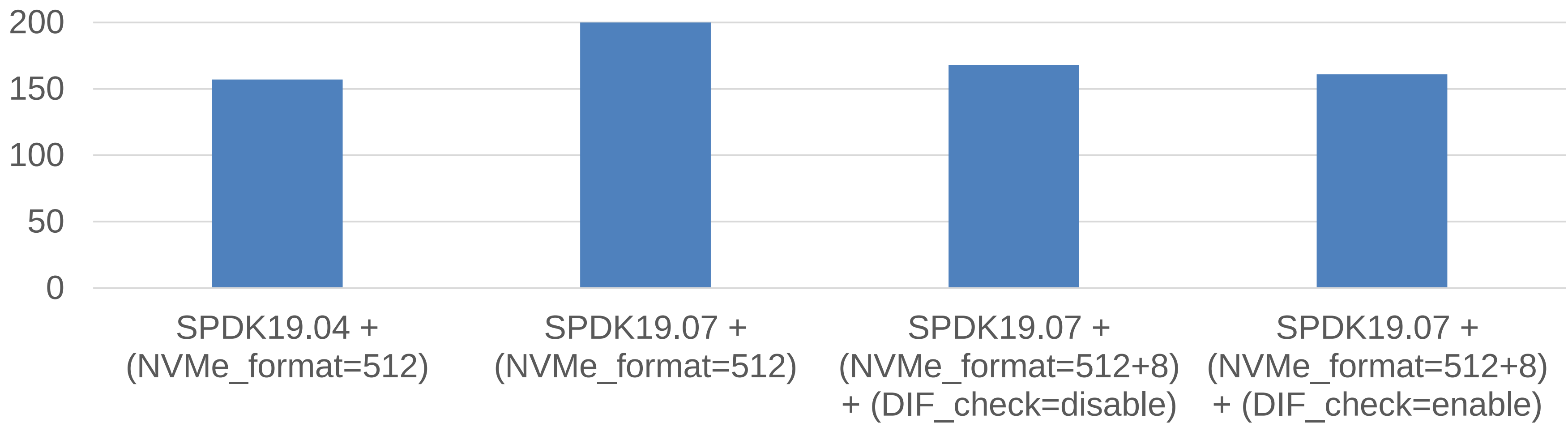
3. Performance Evaluation

System Configuration

- Use two Xeon processor servers
- SPDK NVMe/TCP target used
  - a single CPU core.
  - Linux kernel TCP/IP stack.
  - a single NVMe SSD which supports SGL.
  - 8 split virtual BDEVs on top of the NVMe SSD.
- NVMe/TCP initiator used
  - SPDK NVMe/TCP initiator.
  - Used four CPU cores.
- Use FIO tool and SPDK FIO plugin on a single 10G link.

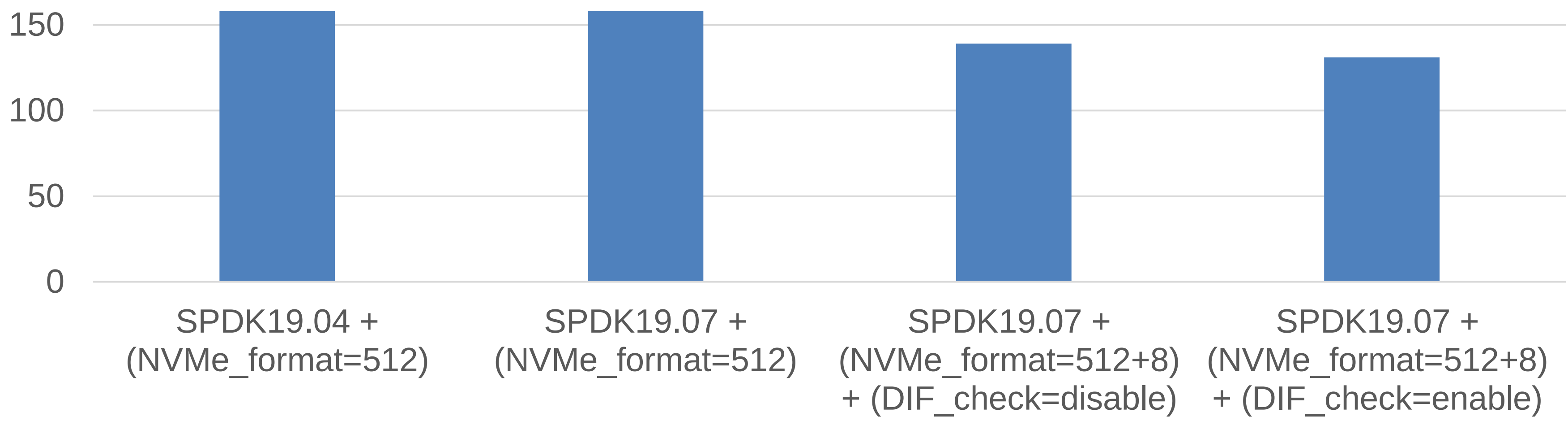


4KB Random Read



Configuration	Throughput (KIOPS)	Overhead
SPDK 19.04 + (NVMe format = 512)	157	22.5%
SPDK 19.07 + (NVMe format = 512)	200	-
SPDK 19.07 + (NVMe format = 512 + 8) + (DIF check = disable)	168	16.0%
SPDK 19.07 + (NVMe format = 512 + 8) + (DIF check = enable)	161	19.5%

4KB Random Write



Configuration	Throughput (KIOPS)	Overhead
SPDK 19.04 + (NVMe format = 512)	178	0%
SPDK 19.07 + (NVMe format = 512)	177	-
SPDK 19.07 + (NVMe format = 512 + 8) + (DIF check = disable)	150	18.0%
SPDK 19.07 + (NVMe format = 512 + 8) + (DIF check = enable)	145	22.0%

4. Summary and Next Steps

Summary and Next Steps

Summary

- SPDK NVMe/TCP target provides DIF insert and strip feature without specialized hardware in SPDK 19.07.
- Performance evaluation showed that the overhead were about 20% both for 4KB random read and 4KB random write.

Next Steps

- Support DIF Insert/Strip in Other Transports (RDMA, FC)
- Performance Evaluation with Vector Packet Processing (VPP)
- Support DIF in Other BDEV Modules (e.g., Crypto, Compress, RAID)
- Additional Enterprise Storage Features
  - Load balancing / Failover of iSCSI Target and NVMe-oF Target
  - RAID1

END

End-to-End Data Protection with SPDK NVMe/TCP Target

05/09/2019

Shuhei Matsumoto

IT Platform Products Management Division

Hitachi, Ltd.

HITACHI
Inspire the Next