SDC

STORAGE DEVELOPER CONFERENCE

SNIA SANTA CLARA, 2017

Persistent Memory over Fabric (PMoF)

Adding RDMA to Persistent Memory

Pawel Szymanski
Intel Corporation

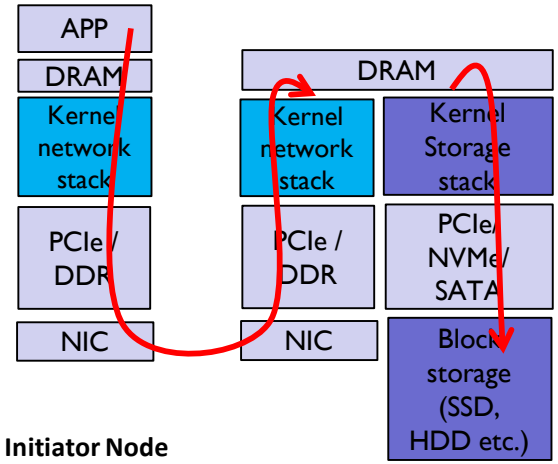
Adding RDMA to Persisteny memory – Agenda

- PMoF Overview
- Comparison with other remote replication technologies
- RPMEM Library Capabilties
- NVML Architecture with RPMEM library
- RPMEM Library API
- Remote replication using PMEMOBJ library
- Future Improvements

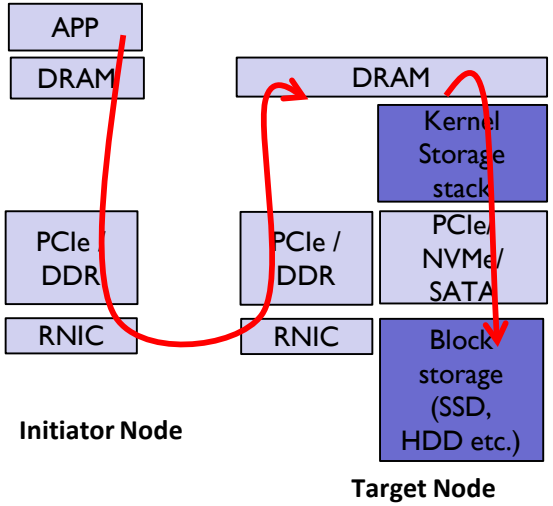
Persistent Memory over Fabric (PMoF) Overview

- Enables low latency and high speed remote replication of persistent memory using various fabrics (IB, RoCE, iWARP, Omnipath, etc)
  - Transport agnostic by using RDMA Verbs
- Many possible non-volatile byte addressable devices are considered in scope (NVDIMMs, 3DXP DIMMs)
- Does not support replication of traditional block-based storage

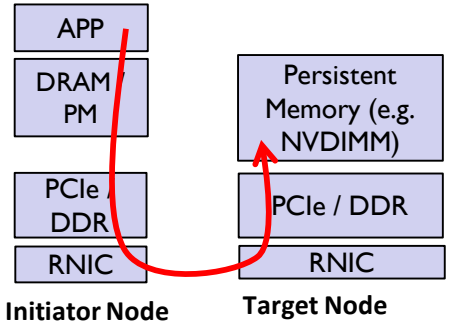
PMoF – comparison with other remote replication storage technologies



TCP/IP based replication to block storage



RDMA based replication to block storage (e.g. NVMEoF)



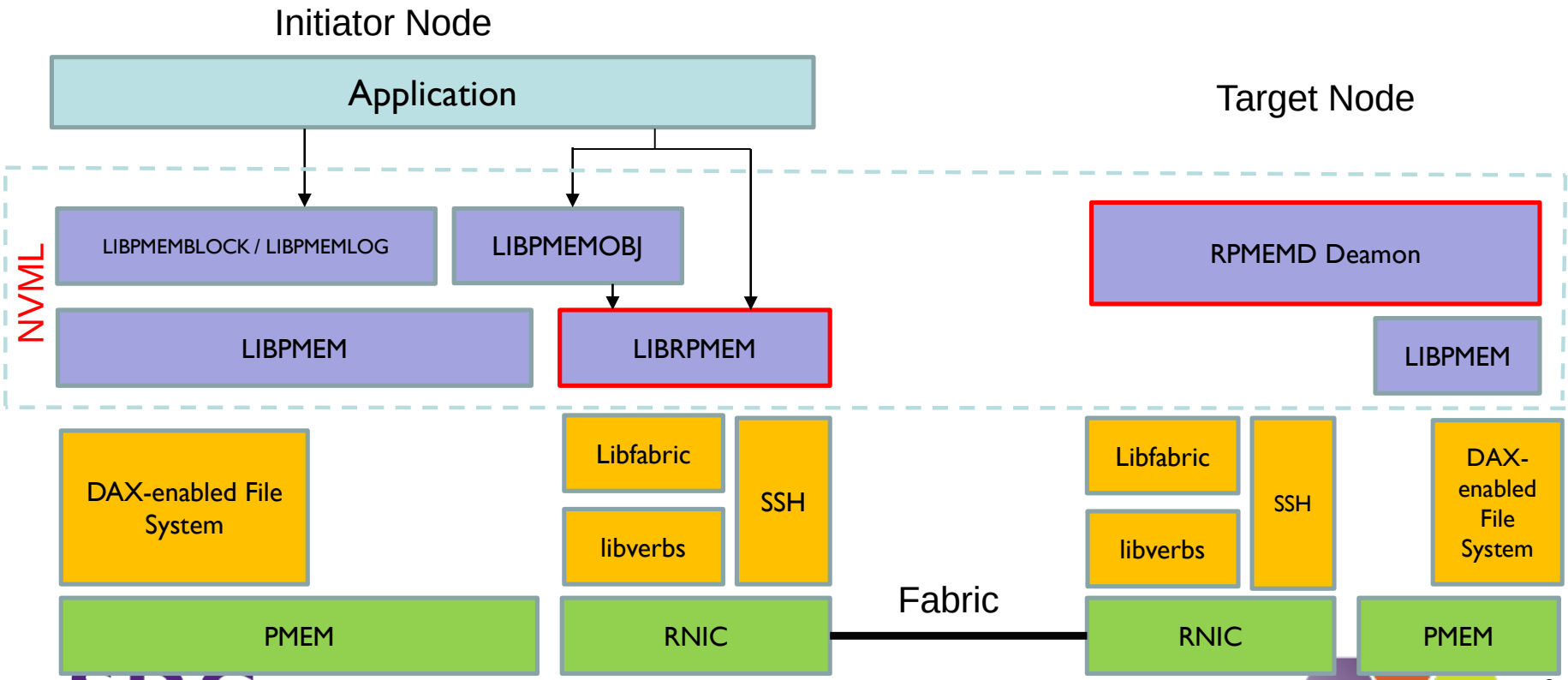
PMoF-based replication to Persistent Memory

Rpmem library – PMoF implementation in NVML

- Fabric agnostic
  - Can use any fabric with libfabric and libverbs support
- Supports GPSPM and APM persistency methods*
  - GPSPM uses RDMA Send
  - APM uses RDMA Read
- Utilizes multiple RDMA queue pairs for highly parallel workload
  - Each application thread can replicate data independently without a need for inter-thread synchronization
- Includes RPMEMD daemon process to be run on remote replica node
  - No need to implement application specific target node service

* For details refer to SNIA SDC 2015 presentation by Chet Douglas:
https://www.snia.org/sites/default/files/SDC15_presentations/persistant_mem/ChetDouglas_RDMA_with_PM.pdf

NVML Architecture with RPMEM



RPMEM Library API
Remote poolset management functions

- `rpmem_create(…)`
  - Starts RPMEMD process on single remote node using SSH
  - Requests remote RPMEMD to create poolset
  - Establishes RDMA connection to remote node
  - Registers local and remote poolset (persistent memory) in libfabric
- `rpmem_open(…)`
  - Same as `rpmem_create` but opens existing poolset and verifies whether it matches local poolset

RPMEM Library API

Poolset management functions

- `rpmem_close(…)`
  - Deregisters local and remote poolset from libfabric/verbs
  - Disconnects RDMA connection
  - Shuts down RPMEMD process on remote node
- `rpmem_remove (…)`
  - Same as `rpmem_close` but also removes poolset on remote node

RPMEM Library API – memory replication functions

- `rpmem_persist(…)`
  - Replicates data from local to remote poolset using RDMA
  - Allows to specify data offset and size within the pool
- `rpmem_read(…)`
  - Copies data from remote poolset to local memory (either local persistent memory or regular DRAM) using RDMA Read
  - Could be used to verify correctness of remote replica or recover local poolset from remote replica
  - Allows to specify data offset and size within the pool
  - Does not persist data locally (no CPU cache flush etc.)

Refer to pmem.io for more detailed description

Remote replication in PMEMOBJ

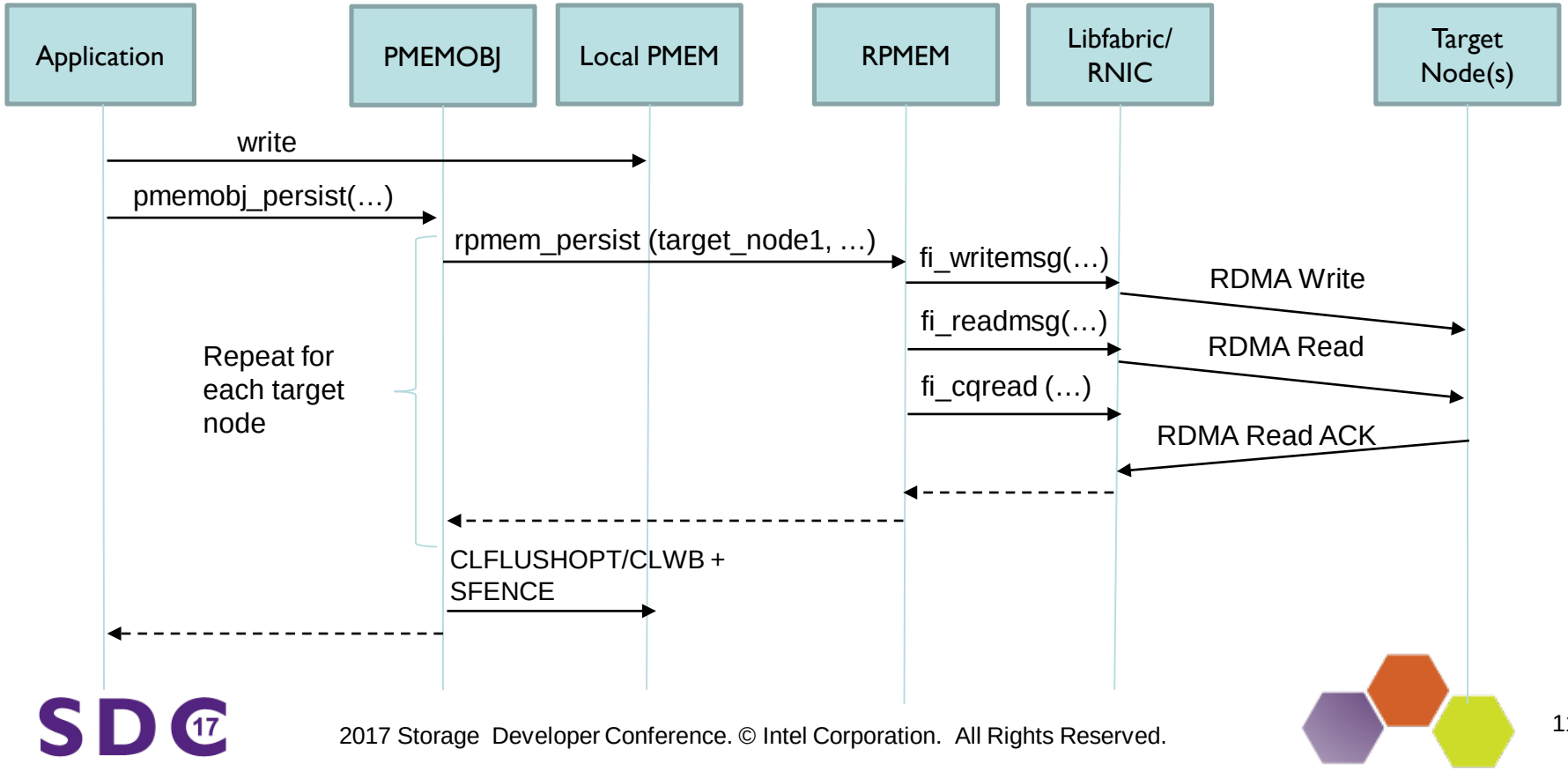
- PMEMOBJ can automatically replicate any writes to persistent memory using RPMEM
- Replication process is transparent to application
- Just add one or more remote replicas to pool set definition file

```
PMEMPOOLSET
100G /mountpoint0/myfile.part0

# local replica
REPLICA 100G /mountpoint3/mymirror.part0

# remote replica
REPLICA user@example.com remote-objpool.set
```


Remote replication in PMEMOBJ



Future improvements and enhancements

- Performance improvements
  - Parallel replication to multiple remote nodes and local replica(s)
  - Use of single RDMA Read/Send for multiple RDMA Write operations – similar to local OPTIMIZED_FLUSH
- Support for Windows OS
- Eventually consistent (aka asynchronous) replication

Key Takeaways

- PMoF improves latency and throughput of remote PMEM replication
- NVML includes RPMEM library implementing PMoF
- Application can either call RPMEM directly or can use PMEMOBJ that use RPMEM for replication
- Start using NVML with RPMEM for remote replication: http://pmem.io