HyperLoop: NIC Offloaded Primitives to Accelerate Replicated Transactions in Multi-tenant Storage Systems

Daehyeok Kim

Aimersaman Memaripour, Anirudh Badam, Yibo Zhu, Hongqiang Harry Liu, Jitendra Padhye, Shachar Raindel, Steven Swanson, Vyas Sekar, Srinivasan Seshan

Presented at ACM SIGCOMM 2018

Carnegie Mellon University  Microsoft  UC San Diego
Multi-tenant Storage Systems

• Replicated transactions
  ▪ Data availability and integrity
  ▪ Consistent and atomic updates
  ▪ *e.g.*, Chain replication

• Multiple replica instances are co-located on the same server
Problem: Large and Unpredictable Latency

- Both average and tail latencies increase
- Gap between average and tail latencies increases
CPU Involvement on Replicas

- CPU involvement for **executing** and **forwarding** transactional operations
- Arbitrary CPU scheduling → Unpredictable latency
- Replicas’ CPU utilization hits 100%

Critical path of operations
Our Goal

Today’s storage system

Run by CPUs

Critical path of operations

Removing replica CPUs from the critical path!
Our Work: HyperLoop

• Framework for building fast chain replicated transactional storage systems enabled by three key ideas:
  1. RDMA NIC + NVM
  2. Leveraging the programmability with RDMA NICs
  3. APIs covering key transactional operations

• Minimal modifications for existing applications
  • e.g., 866 lines for MongoDB out of ~500K lines of code

• Up to 24x tail latency reduction in storage applications
Outline

• Motivation

• HyperLoop Design

• Implementation and Evaluation
Idea 1: RDMA NIC + NVM

- RDMA (Remote Direct Memory Access) NICs
  - Enables direct *memory access* from the network without CPUs
- NVM (Non-Volatile Memory)
  - Provides a *durable* storage medium for RDMA NICs
Roadblock 1: Operation Forwarding

- RDMA NICs can execute the logging operation
- CPUs are still involved to request the RNICs to forward operations
• CPUs are involved to **execute** and **forward** operations
  • RDMA NIC primitives do not support some key transactional operations (e.g., locking, commit)
Can We Avoid the Roadblocks?

Today’s storage system

Pushing replication primitives to RNICs!

Offload?
Idea 2: Leveraging the Programmability of RNICs

• Commodity RDMA NICs are not fully programmable

• Opportunity: **RDMA WAIT** operation
  • Supported by commodity RDMA NICs
  • Allows a NIC to wait for a completion of an event (*e.g.*, receiving)
  • Triggers the NIC to perform an operation upon the completion
**Bootstrapping – Program the NICs**

- **Step 1:** Frontend library collects the base addresses of memory regions registered to replica RNICs

- **Step 2:** HyperLoop library programs replica RNICs with RDMA WAIT and the template of target operation
Forwarding Operations

- **Idea:** Manipulating parameter regions of programmed operations
- **Replica NICs can** **forward** operations with **proper parameters**
Idea 3: APIs for Transactional Operations

<table>
<thead>
<tr>
<th>Transactional Operations</th>
<th>Memory Operations</th>
<th>HyperLoop Primitives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logging</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Commit</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Locking/Unlocking</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

See our SIGCOMM paper for details!
Transactions with HyperLoop Primitives

1. Update log (Group Log)
2. Grab a lock (Group Lock)
3. Commit the log (Group Commit)
4. Release the lock (Group Unlock)

Replica NICs can execute and forward operations!
Outline

• Motivation

• HyperLoop Design

• Implementation and Evaluation
Implementation and Case Studies

• HyperLoop library
  • C APIs for HyperLoop group primitives
  • Modify user-space RDMA verbs library and NIC driver

• Case Studies
  • **RocksDB:** Add the replication logic using HyperLoop library
    (modify 120 LOCs)
  • **MongoDB:** Replace the existing replication logic with HyperLoop library
    (modify 866 LOCs)
Result Highlights

• Latency reduction for memory operations on a group:
  • Write: ~801.8x
  • Memory copy: ~848x

• Tail latency reduction for RocksDB: 5.7 – 24.2x

• Latency reduction regardless of group sizes

• Zero CPU utilization in the data path
Summary

• Predictable low latency is lacking in replicated storage systems
  • Root cause: CPU involvement on the critical path

• Our solution: HyperLoop
  – Offloads transactional operations to commodity RNICs + NVM
  – Minimal modifications for existing applications

• Result: up to 24x tail latency reduction in in-memory storage apps

• More opportunities
  – Other data center workloads
  – Efficient remote memory utilization