Byzantine Fault Tolerance (BFT)

- Traditional BFT protocols
  - Tolerating arbitrary (Byzantine) faults
  - $3f + 1$ nodes to tolerate $f$ faults
  - TCP/IP-based communication

  High reliability and consistency

Remote Direct Memory Access

- Direct access to memory on remote systems with without CPU involvement
- Zero-copy data transfer
- Communication resources
  - Queue based-communication
  - Memory registration & buffer management
- Richer data transfer primitives
  - One-sided Read/Write
  - Two-sided Send/Receive

Towards building RDMA-based BFT

How can we build a secure scalable RDMA-based BFT?

- Efficient use of RDMA one-sided and two-sided communication
  - Clients use Send primitive to not saturate the leader
  - Dynamically Connected Transport (DCT) for better scalability
  - Replicas perform a direct RDMA Write into remote memory

Problem:

- Malicious replicas have access to remote nodes’ memory
- Memory RDMA keys are not secure
- Implement counter-measures

Basis BFT protocol: Hybster [Behl et al., EuroSys’17]

- Use of two-sided to avoid security issues
  - Preliminary approach
  - Design an RDMA-based selector replacing the Java NIO selector

Example Applications: Blockchain and Coordination Services

- Building a coordination service with a similar interface as ZooKeeper
  - Strong consistency and availability
- Implementing BFT-ordering service
  - Benefit from improved performance of BFT

Related Work

- DARE is an RDMA-based Raft protocol optimized using RDMA features
- APUS is scalable RDMA-based Paxos protocol performing replication using RDMA Write

Ines Messadi, TU Braunschweig, Germany. Advisor: Rüdiger Kapitza
messadi@ibr.cs.tu-bs.de, rrrkapitz@ibr.cs.tu-bs.de