Hermes: Fault-tolerant Replication with Strong Consistency and High Performance

Antonios Katsarakis, Vasilis Gavrielatos, Siavash Katebzadeh
Arpit Joshi, Boris Grot, Vijay Nagarajan
University of Edinburgh

Aleksandar Dragojevic
Microsoft Research

ABSTRACT

Datastores underpin today's large scale online services and thus need to provide high throughput to keep up with the ever-increasing demand of those services. To guarantee high availability in the presence of failures, datastores replicate the data across several nodes. This is accomplished with the help of a fault-tolerant replication protocol that is responsible for maintaining the replicas consistent by ensuring that each write is propagated to the replicas. Replication protocols that provide strong consistency are preferred over weaker consistency protocols that cannot guarantee an intuitive behavior for clients. In this work, we observe that existing replication protocols offering high availability and strong consistency are unable to maximize performance. We identify two endemic sources of performance loss: lack of concurrency and/or inability to offer local reads.

Meanwhile, it is well known that shared memory multiprocessors offer both strong consistency and high performance by using invalidating coherence protocols that support local reads (cache hits) and offer high concurrency on reads and writes to different memory locations across threads. However, invalidating coherence protocols are not resilient.

Based on these observations, we propose Hermes, a fault-tolerant replication protocol that guarantees strong consistency and high performance. Similarly to multiprocessor coherence protocols, Hermes uses an invalidation-based mechanism to achieve linearizability. In contrast to the multiprocessor, Hermes achieves fault tolerance by avoiding single points of failure and guaranteeing that any uncommitted write can be replayed. Using an RDMA-based datastore with five replicas, we show that Hermes outperforms two state-of-the-art replication protocols by more than 4$\times$ under a workload with 5% writes while offering strong consistency in the presence of faults.