Dynamo DB

Sirui Tan st2957

Dynamo DB is a highly available key-value storage system that some of Amazon's core services use to provide 'always-on' experience. It is built on top of Amazon's e-commerce platform and is designed to meet with the following requirements: query model, ACID (Atomicity, Consistency, Isolation, Durability) properties and efficiency. When designing Dynamo DB, there are two most important design considerations: when to perform the process of resolving update conflicts and who to proform the process conflict resolution.

Distributed implementations that are analogous to Dynamo DB include Peer to Peer Systems, distributed file systems and databases like Google File System (GFS). Dynamo DB differs from these decentralized storage systems in terms of its target requirements. Unlike the others, Dynamo DB does not reject updates, assumes trustworthy nodes, does not need hierarchical namespaces and is built for latency sensitive applications.

GFS uses leases to maintain a consistent mutation order across replicas of a chunk. When a client wants to mutate on a chunk (e.g. write some new data), it starts by asking master the location of primary who holds the lease. Then client pushes data to all replicas followed by sending write request to primary. The primary then assigns a serial number to the request and forward it to every other replica. All replicas then mutate themselves in the order of request serial number. Atomic record append is almost identical to lease-based mutation with only a little extra logic at the primary. Mutation guarantees bandwidth usage by pipelining data transfer sequentially. GFS also provides a snapshot operation which makes an instantaneous copy of a file or directory tree.

Dynamo features several core distributed systems techniques: partitioning, replication, versioning, membership, failure handling and scaling. Dynamo's partitioning scheme relies on consistent hashing to distribute the load across multiple storage hosts so that it can dynamically partition the data over the set of nodes in system. To achieve high availability and durability, Dynamo replicates its data on multiple hosts. Specifically, each data item is replicated at N hosts and each key is assigned to a coordinator node. Dynamo also provides eventual consistency, which allows for updates to be propagated to all replicas asynchronously. In order to handle failures, Dynamo uses a 'sloppy quorum' and all real and write operations are performed on the first N healthy nodes from the preference list. Other features of Dynamo includes ring membership, external discovery and failure detection.

In Dynamo, each storage node has three main software components: request coordination, membership and failure detection, and a local persistence engine. All these components are implemented in Java. In Amazon, Dynamo is used by several services with different configurations, including business logic specific reconciliation, timestamp based reconciliation and high performance read engine. Dynamo tunes the internal values to achieve the desired levels of performance, availability and durability. The success of Dynamo shows that an eventual-consistent storage system can be a building block for highly available applications.