Dynamo – Review

Yixing Chen, yc3094

The paper *Dynamo: Amazon's Highly Available Key-value Store* introduced Dynamo – a highly available key-value storage system that some of Amazon's core services use to provide an "always-on" experience. To achieve this level of availability, Dynamo sacrifices consistency under certain failure scenarios. It makes extensive use of object versioning and application-assisted conflict resolution in a manner that provides a novel interface for developers to use.

Amazon runs a world-wide e-commerce platform that serves tens of millions customers at peak times using tens of thousands of servers located in many data centers around the world. There are strict operational requirements on Amazon's platform – performance, reliability and efficiency, and to support continuous growth the platform needs to be highly scalable. The reliability and scalability of a system is dependent on how its application state is managed. To achieve these features, Dynamo provides a simple primary-key only interface to meet the requirements of applications.

Before clarifying the architecture of Dynamo, we should know some system assumptions and requirement. The storage system for this class of services has the following requirements: Query Model (simple read and write operations to a data item that is uniquely identified by a key), ACID Properties (ACID – Atomicity, Consistency, Isolation, Durability, is a set of properties that guarantee that database transactions are processed reliably.), Efficiency (The system needs to function on a commodity hardware infrastructure) and so on.

The system architecture is very complex, several techniques are used in Dynamo to achieve desirable performance. Consistent Hashing is used on incremental scalability, vector clocks with reconciliation during reads are used to achieve high availability for writes. Sloppy Quorum and hinted handoff is used to provide high availability and durability guarantee. Anti-entropy is used to recover from partition. Gossip-based membership protocol and failure detection are used to preserves symmetry and avoids having a centralize registry for storing membership and node livens information.

Then the writer talks about the experience and lessons learned in Dynamo. Dynamo is used by several services with different configurations, it uses Business logic specific reconciliation, Timestamp based reconciliation and High performance read engine. The main advantage of Dynamo is that its client applications can tune the values of N, R and W to achieve their desired levels of performance, availability and durability. Unlike popular commercial data stores, Dynamo exposes data consistency and reconciliation logic issues to the developers. Besides, Dynamo make many important tradeoffs on several aspect: balancing performance and durability, ensuring uniform load distribution, client-driven or server-driven Coordination and balancing background vs. foreground tasks.

In my opinion, it's a great paper to introduce the Dynamo to us. It illustrates background, related work, system design, implementation and experiences and insights gained by running Dynamo in production. Dynamo has provided the desired levels of availability of and performance and has been successful in handling server failures, data center failures and network partitions. Its success in one of the most challenging application environments shows that an eventual consistent storage system can be a building block for highly available applications.