My experience With Redis replica

We configure "leader-follower" or "master-slave" replication for redis cluster.

Slave Redis instances are the exact copies of master instances. The slave will automatically reconnect to the master every time the link breaks and sync to become exact copy of master.

This system works using three main mechanisms:

1. When a master and a slave instances are well-connected, the master keeps the slave updated by sending a stream of commands to the slave.

2. When the link between the master and the slave breaks, for network issues or because a timeout is sensed in the master or the slave, the slave reconnects and attempts to proceed with a partial resynchronization: it means that it will try to just obtain the part of the stream of commands it missed during the disconnection.

3. When a partial resynchronization is not possible, the slave will ask for a full resynchronization.

Redis uses by default asynchronous replication, which being low latency and

High performance, is the natural replication mode for the vast majority of Redis use cases.

Some very important facts about Redis replication:

> Redis uses asynchronous replication, with asynchronous slave-to-master acknowledges of the amount of data processed.

> A master can have multiple slaves.

> Slaves are able to accept connections from other slaves.

> Redis replication is non-blocking on the master side.

> Replication is also largely non-blocking on the slave side.

> Replication can be used both for scalability, in order to have multiple slaves for read-only queries (for example, slow O(N) operations can be offloaded to slaves), or simply for improving data safety and high availability.

> It is possible to use replication to avoid the cost of having the master writing the full dataset to disk.

> In setups where Redis replication is used, it is strongly advised to have

Persistence turned on in the master and in the slaves.