redislabs
home of redis

Whitepaper

Six Essential Features for Highly Available Redis

Dave Nielsen
Developer Advocate, Redis Labs

Table of Contents

Executive Summary

As Redis deployments proliferate and fuel high profile, mission-critical business applications, it becomes increasingly important that Redis remains truly highly available. Even a minute of downtime can be prohibitively expensive. With Redis serving thousands (and in some cases hundreds of thousands) of operations per second, the cost of a minute of downtime could easily run into several hundreds of thousands of dollars.

In this report we define six features that are critical to ensuring the high availability of your Redis deployment and detail how they safeguard against every type of failure or outage event. The six features include in-memory replication, multi-rack/zone/datacenter replication, instant auto-failover, AOF data persistence, backup and multi-region/cloud replication. These protect your Redis from process failures, node failures, multi-node failures, rack/zone/datacenter failures, network split events, and complete region or cloud failures.

We also compare the performance of several Redis-as-a-Service providers to demonstrate that only Redis Labs' products provide true high availability, recovering from outages instantaneously and without data loss.

Why You Need a Highly Available Redis

According to Wikipedia, the definition of a "highly available deployment" is one with automatic failover and minimal downtime. But this doesn't specify what an acceptable downtime is. A modest Redis instance in the cloud can easily handle 100,000 operations per second. At this rate, it is possible that one minute of downtime could result in the failure of six million attempted operations and could bring an entire system to its knees.

A recent study by the Ponemon Institute found the average cost of one minute of downtime to be $7,900. Given Redis' critical position as the lynchpin for modern web, mobile and business applications, several minutes of Redis downtime could cost hundreds of thousands of dollars.

Key Requirements for Making Redis Highly Available

To make Redis highly available, all possible failure events that could impact its availability should be addressed:

1. **Process failure:** In order to safeguard against process failure, you need a copy (slave) of your Redis instance, running on a different physical/virtual server, to take over. This is accomplished with in-memory replication. But in-memory replication is not enough; you need a mechanism that instantly detects a failure and fails over seamlessly to the slave Redis instance.
2. **Node failure:** If the server running your Redis fails, you need a copy (slave) of your Redis instance running on a different physical server. As previously mentioned, you also need instant failure detection and a failover mechanism. AOF persistence can be used to recover from a node failure, but this approach is slower due to time required to load data from storage.
3. **Multiple node failures:** In the event of multiple node failures in the same rack or zone, AOF persistence

provides real data durability. In a cloud environment, real data durability can only be achieved using an external storage device like AWS EBS (Elastic Block Store). Otherwise, you will lose all your data during a node failure event or a multi-node failure event, even if it is persisted. Make sure each node in your Redis deployment is connected to an external storage device. Replicating your data to other racks/zones will also help with data recovery (see 'e' and 'f' below for additional technology requirements for these scenarios).

4. **Network split events:** Usually one needs to maintain at least three copies of data so that network split events can be resolved with quorum. If your Redis deployment does not employ any kind of quorum technique to resolve splits, you might be exposed to a split brain scenario in which your database could become inconsistent.

5. **Zone/rack/datacenter failures:** To increase the SLA of your application, you will have to replicate your dataset across different datacenters/zones or racks in the same geo-location/region, while still maintaining a robust instant auto-failover mechanism. To correctly implement that and avoid the split-brain situation mentioned earlier, you should make
   sure your on-premises deployment or your cloud vendor infrastructure supports quorum resolution techniques.

6. **Complete region/cloud failure:** In-order to implement disaster recovery (DR) for your Redis deployment, you should make sure your Redis vendor supports multi-region/cloud replication. This can be a complex task, as WAN links are usually much slower than the rate of 'writes' a typical application needs when it uses Redis, and can only be achieved with technologies like compression, dedup, TCP optimization and gradual replication. Another technology that can be used to make your deployment DR-ready (although with a much slower recovery time) is backup. You should make sure you backup your Redis dataset to a repository deployed across multiple geo-locations, like AWS S3.

To safeguard against all these failures, your Redis service/deployment should support the following MUST HAVE features:

1. **In-memory replication:** Allows a copy of your Redis dataset to be stored on at least one more node, making sure the recovery from a process or node failure event takes as little time as possible.

2. **In-memory multi-rack/multi-zone/datacenter replication:** Allows a copy of your Redis dataset to be stored on at least one more node located in a different rack/zone/datacenter, so your Redis deployment can still be available during a complete data center outage event.

3. **Auto-failover:** This should run on the same physical/virtual nodes as your Redis deployment in order to detect automatically and trigger failover instantaneously in the event of a process, node, multi-node failure or a network split.

4. **AOF data persistence:** Allows configuration of Redis in Append Only File (AOF) mode to guarantee real data durability.

5. **Backup:** Enables automatic backup to external storage like AWS S3. You can use this backup to manually recover from multiple datacenter failure events.

6. **Multi-region/cloud replication:** Allows you to put a replica in a data center in a remote region or on a different cloud, which can serve as a complete DR deployment to overcome region/cloud failure events almost instantly.

Mapping Features to Outage Type

Given the multiple types of outages that could impact your Redis deployment, we mapped which features safeguard against which type of outage. The table below summarizes each HA mechanism that is required for recovery in each of the listed failure events:

Failure Event	In-memory Replication	Multi-DC/ Zone Replication	Auto-failover	AOF Data Persistence	Backup (using snapshots)	Multi-region/ Cloud Replication
Process Failure	🟦		Instant recovery*	slow recovery		
Node Failure	🟦		Instant recovery*	slow recovery		
Multi-node Failure		🟦	Instant recovery*	slow recovery		
Network Split	🟦		Instant recovery*			
Zone/Rack Failure		🟦	Instant recovery*		slow recovery	fast recovery
Complete Region/Cloud Failure					slow recovery	fast recovery

* Auto-failover process should run on same nodes as Redis deployment

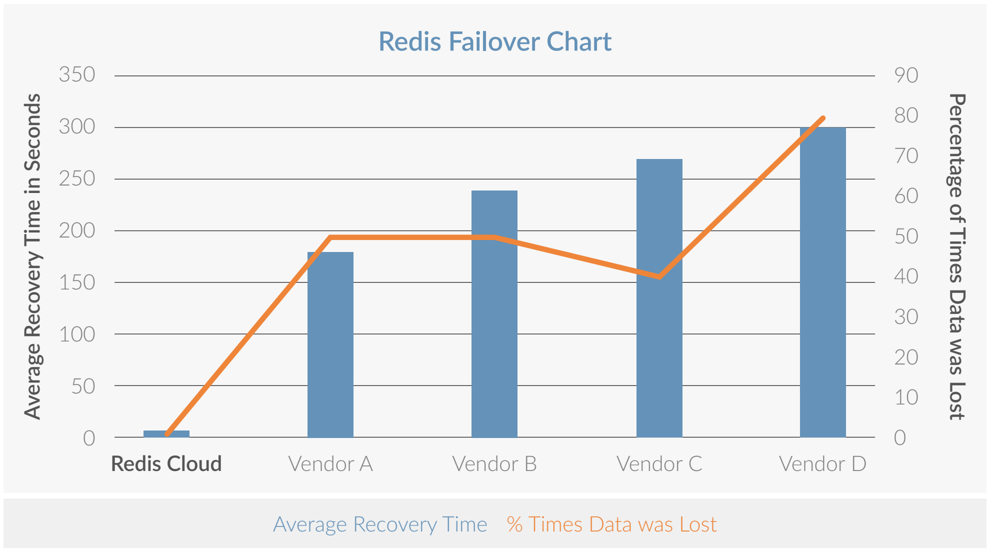
To re-iterate our key takeaways:

- The minimum requirements for most of the failure events are in-memory replication across different racks/ zones/datacenters and an auto-failover mechanism that runs on the same nodes as your Redis deployment.
- AOF data persistence is good, but recovery is slow.
- In order to be really safe against complete failures of an entire region, backing up to an alternate location or cloud is a good option to have. However, a more robust approach is to use a multi-region/cloud replication technique that deals with low throughput high latency WAN links.

Comparing Redis Offerings for Failover Times and Data Loss

Failover time, recovery time and data loss can vary greatly from vendor to vendor across Redis-as-a-Service offerings. We ran an internal benchmark to test failover among the popular Redis services to find out if other Redis service providers have the same commitment to real-time failover that we do.

The following chart shows the average time to recover in seconds, and the percentage of times that the data was lost as well. Compared to all other vendors, Redis Cloud employs the fastest failover mechanism while also safeguarding against data loss.



Notes

1. We decided not to publish other vendors' names with their results. That said, we do think it is important to mention the services we tested (not in the same order as the above graph): Microsoft Azure cache, AWS ElastiCache, Heroku Redis, Compose.io (now IBM).
2. We also decided not to publish the exact mechanism that we used to induce failover because it is a disruptive mechanism.

In the chart above, you can see that Redis Cloud recovers from failure within six seconds, almost immediately after the failure occurs, and doesn't lose data! The other Redis services take much longer – even minutes – and data is often lost. Redis Labs' exemplary performance in this benchmark is due to its very robust failure detection and failover mechanisms.

What Distinguishes Redis Labs' High Availability From the Competition

During the tests we found that although all Redis vendors claim to have a full HA solution, Redis Labs is the only vendor that proves to be truly highly available. This is because Redis Labs' technology includes a full HA suite that was built and designed from the ground up to overcome all the different outage scenarios we reviewed above.

All other vendors provide only limited HA functionality:

- Some use data-persistence without replication, which results in very long node failure recovery times, as the entire dataset must be loaded from disk.
- Some use replication without data-persistence and cannot recover from a multi-node failure event.
- Some use a 3rd party component like HAProxy to manage their Redis deployment, but don't trigger failover when this proxy is down.
- None of them have a robust auto-failover mechanism that can cope with the different types of failure that a distributed database system should handle.
- None of them provide a real-time multi-region/cloud replication across WAN for seamless DR.

Redis Labs' technology supports the six essential HA features both in Redis Cloud as well as its on-premises offering Redis Labs Enterprise Cluster (RLEC).

Critical advantages of the Redis Labs technology include:

- Fully automated setup of in-memory replication that works across multiple physical/virtual servers, racks, zones, datacenters, regions and clouds.
- An automatic failure detection mechanism that runs on the same physical nodes as your Redis dataset and automatically triggers failover, thanks to the advanced clustering architecture developed by Redis Labs.
- AOF data persistence that can be enabled from both the master and/or slave (so you can optimize performance).
- Patent pending technology that only requires an uneven number of nodes in your Redis cluster (not uneven copies of your data), reducing deployment costs.
- A mix of underlying WAN replication technologies such as compression, dedup, TCP optimizations and gradual replication that guarantee minimal gap between the datasets across high-latency low-throughput WAN links, even during high write rate scenarios.

Recreating the Benchmark

You can easily recreate these results (at least with Redis Labs) yourself by downloading a trial version of Redis Labs Enterprise Cluster (RLEC) from redislabs.com/redis-enterprise. RLEC uses the same clustering technology as Redis Cloud and can be used to prove its robustness.

You will need to spin up two nodes of RLEC: one for the master Redis process (shard) and a second for the slave. (Note: in a production environment, you must keep an uneven number of nodes, i.e. at least three, to allow the cluster to reach a consensus in case of a network split event). Once you load up your data, you can terminate the Master node instance and watch the failover process complete seamlessly. Contact expert@redislabs.com if you'd like additional information about how our testing was conducted.

Conclusion

True high availability for Redis requires a solution that handles every possible outage scenario gracefully. We defined six features that are critical for high availability Redis deployments in order to safeguard against every kind of outage: in-memory replication, multi-rack/zone/data center replication, auto- failover, AOF data persistence, backup and multi-region/cloud replication.

When we benchmarked different Redis service providers by time-to-recover from failure events and data availability, we found that only Redis Labs recovers instantaneously and without data loss. While choosing your Redis vendor or deployment mechanism, it's important to keep different outage scenarios in mind and evaluate whether you have all the features and technologies in place to overcome each scenario.