How to Maximize the Availablity of Apache Cassandra®

Overview

This white paper provides a range of strategies for ensuring that your Apache Cassandra® deployment is highly tuned to maximize application uptime and continued high availability.

This paper covers the basics of Cassandra high-availability and then provides a defense in-depth approach by detailing a range of strategies that can be employed at the architecture, infrastructure, and application layers.

Basics of Cassandra High-Availability

We'll begin with what veteran Cassandra users will recognize as already-ingrained strategies, but these tips bear mentioning because too often we see that devs aren't following them.

These also apply to any production Cassandra cluster, and can prevent single-instance hardware failures or process crashes on an individual machine from leading to widespread outages.

Tip 1: Your cluster should include three nodes (two at absolute minimum)

Hardware failures and interruptions in Cassandra service on a single node can (and will) happen. When they do, a redundant secondary node is necessary to ensure that service will continue seamlessly during these events. Three nodes are required to enable writes with strong consistency, making this the optimal strategy.

Tip 2: Have the right replication factor and strategy in place

Three nodes can't effectively safeguard uptime if the cluster's replication factor is set to one (meaning the cluster only includes a single replica). A replication factor of three is correct in most cases. It's also smart to use NetworkTopologyStrategy for the replication strategy. This strategy allows for easy expansion to other racks and data centers as needed. While not every organization will see an immediate need for those capabilities, NetworkTopologyStrategy offers a great deal of flexibility going forward – with zero downside to being prepared from the beginning.

Tip 3: Diversify the physical locations of data with Cassandra rack configuration

Use Cassandra rack configuration to make sure that a disaster at one of the locations where your data replicas are stored will have absolutely no effect on the others. AWS assists with this by allowing users to map racks to geographically distinct availability zones (while other cloud providers offer similar solutions). If managing your own data center racks, make sure to physically separate replicas by practicing this don't-put-all-your-eggs-in-one-basket approach.

Tip 4: Understand and choose a consistency level that tolerates faults

If the consistency level is unrealistic—say if it's set to ALL—a node failure can derail availability even if healthy replicas are on hand. Consistency level is most often set to QUORUM or LOCAL_QUORUM, allowing operation with strong consistency despite a node failure. However, it's important to understand the relationship between QUORUM and replication factor, for example, QUORUM with a replication factor of two will require two functioning nodes. Make sure this is set up logically so that the loss of a single node won't result in costly downtime.

Architectural Strategies for Cassandra High-Availability

Now that we've covered the basics, it's important to take a look at your Cassandra deployment at the architectural level, and further maximize availability by preventing failures in the physical infrastructure.

Utilize Multi-Data Center Support

If a complete cloud provider region or on-premise physical data center experiences a failure, basic preparations can no longer safeguard availability. To address this, use Cassandra's native multi-datacenter support to maintain a remote hot standby of the cluster in case the need arises.

Span Multiple Cloud Providers

As we well know from recent events, cloud providers themselves can suffer from availability issues. To prepare for this reality, a Cassandra cluster can be designed to span both regions and providers, or to have nodes both on-premise and in the cloud, maintaining availability even when providers cannot.

Use Replication Factor 5

While replication factor 3 means you can survive the loss of a single replica, it then means being vulnerable anytime a single replica is down. Unfortunately, there are common situations where routine maintenance could require a node to be down for hours or days, leaving the cluster vulnerable. Building architecture capable of running replication factor 5, and using QUORUM as the consistency level, maintains availability even if periods of maintenance and single node failures coincide.

Infrastructure Strategies for Cassandra High-Availability

Even with a strong architecture in place, infrastructural issues can hamper a Cassandra cluster's availability. Namely, the cluster can be overloaded by spikes in client load, or by events such as an increase in tombstones or a repair operation gone haywire. Making the right infrastructural preparations can prevent these events from causing downtime.

Keep Some Capacity in Reserve

By keeping disk usage under 70% during periods of normal load, as well as CPU utilization at around 60-70%, sudden load increases shouldn't lead to instability. Going above these thresholds is possible, but probably only advisable if you have an expert understanding of your cluster and trust that you can safely push your luck.

Use More, Smaller Nodes

Smaller nodes offer advantages in maintaining availability. Maintenance operations performed on smaller nodes will finish faster, reducing the risks present while they are occupied. At the same time, the total processing capacity of the cluster takes less of a hit when a single small node fails as opposed to a larger one. Compare running a cluster of 6x m4.xl AWS instances versus a cluster of 3 x m4.2xls—the cluster of smaller nodes offers the same or greater processing power for the same value, while achieving an infrastructure where a single node failure has only half the impact.

Be careful, however: there is such a thing as going too small, causing the base OS and similar overhead to become too limiting, while increasing the need to rely on automated provisioning and management.

Be Vigilant in Monitoring the Cluster

The truth is that catastrophic issues rarely show up out of the blue. Most often there's a history of danger signs before the failure: possibly an increase in pending compactions, warnings of tombstones and large partitions, spikes in latency, or mutations getting dropped. Observe these issues and address them as they come; a small fix in time will save you from a large failure.

Application-Level Strategies for Cassandra High-Availability

Lastly, taking some precautions at the application level can be effective in reducing Cassandra issues.

Optimize Your Cassandra Driver Configuration

Cassandra drivers have features to make your cluster much more resilient, if you take the time to configure them for your requirements. A few examples: drivers can route queries away from slow nodes, or implement automatic fall back to lower consistency levels as needed to avert issues.

Set up Retry Strategies That Know When to Quit

Persistence is a virtue, and it's often beneficial for applications to retry failed queries. However, without a strategy to terminate queries as necessary, their never-ending efforts will drag the cluster toward failure. As queries timeout from a build up of tombstones or other issues, retries continue adding load to the cluster. When that load builds to where several queries are failing, application retries multiply that load, until catastrophic failure occurs.

Use Multiple Clusters

Following every best practice mentioned here will significantly reduce the risk of downtime to your Cassandra cluster, but some threats will always remain. The solution to dealing with this remaining risk is to utilize multiple clusters. Building redundant clusters across separate data centers means that a single action cannot take down the entire global Cassandra deployment. It might also be an option to split clusters by application function; for example, using one cluster solely to take in data and another to store it and perform analytics. In this use case, a failure of one cluster will still leave a partially functioning application, making a complete Cassandra failure and application outage extremely unlikely.

About Instaclustr

Instaclustr helps organizations deliver applications at scale through its managed platform for open source technologies such as **Apache Cassandra®**, **Apache Kafka®**, **Apache Spark™**, **Redis™**, **OpenSearch®**, **PostgreSQL®**, and **Cadence**.

Instaclustr combines a complete data infrastructure environment with hands-on technology expertise to ensure ongoing performance and optimization. By removing the infrastructure complexity, we enable companies to focus internal development and operational resources on building cutting edge customer-facing applications at lower cost. Instaclustr customers include some of the largest and most innovative Fortune 500 companies.