CEP-13: Denylisting partitions

Status

Current state: *Done*

Discussion thread: https://lists.apache.org/thread.html/r1547c5f2fb8548e2f7dcbe1a26da8c2a95ebec81adeeb2ea0545924d%40%3Cdev.cassandra.apache.org%3E

JIRA: ***CASSANDRA-12106*** *- Getting issue details...* ***STATUS***

Released: TBD

Please keep the discussion on the mailing list rather than commenting on the wiki (wiki discussions get unwieldy fast).

Motivation

When operating Cassandra in production, situations arise where reading certain partition keys can cause undesirable effects on the entire node or within the cluster.. Reasons could be

- Reading such partition is expensive due to the nature of the partition (like many live cells or tombstones) that may start to impact the performance of the node as a whole, thereby impacting unrelated queries
- Partitions with heavy write load spread across many sstables that have not yet been compacted
- A potential DDOS attempt using one or more partition keys

Ideally, Cassandra would handle these cases through other optimizations but in the interim a tool for operators to reduce the impact of these partitions is useful.

Audience

Cassandra developers and operators.

Goals

Allows users to be able to specify one or more partition keys to be disallowed from being read from or written to C*.

Non-Goals

This feature will not include any auto detection of such partition keys, and instead relies on the operator to specify the partition keys to be disallowed.

Proposed Changes

This design document explains the proposed changes in detail, along with past community collaboration in the form of comments in the document.

New or Changed Public Interfaces

StorageServiceMBean interface

- New method: public void refreshDenylistedPartitionsCache();

Nodetool

- New command: refreshdenylistedpartitionscache

Sidecar change

- New API to aid with adding partitions to deny list
- [Alternate solution] Nodetool command and jmx to add partition to deny list (along the lines of public boolean denyListKey(String keyspace, String table, String keyAsString))
  - Pro: More friendly and easier for operator
  - Con: Although being a nodetool command, it would impact the entire cluster by informing the cluster of the new partition to be denylisted (and not just the node the command is being run on)

Compatibility, Deprecation, and Migration Plan

As this is an addition only, there is no impact on compatibility / deprecation and there is no migration plan that is needed.
Note: The feature of denylisting will work as expected once every node in the cluster is upgraded to the C* version that has this feature

Test Plan

In addition to the planned unit tests to be provided with the patch, measure performance in terms of latencies when denylist contains many partitions and partitions being large.