Real world tales of repair
About The Last Pickle

We help people deliver and improve Apache Cassandra based solutions.

With staff in 5 countries:
New Zealand, Australia, France, Spain, USA
What and why?

Full repair
Incremental repair
How to make it work
What is repair?

A maintenance operation that (briefly) restores strong consistency throughout the cluster.
Why do we need repair?

- Eventual consistency
- Downtime / failure recovery
- Safe deletes

www.thelastpickle.com
Tombstones need repair too

Missing tombstones can lead to zombie data
(repair within gc_grace_seconds)
Tombstones need repair too
Tombstones need repair too
Tombstones need repair too
Tombstones need repair too

www.thelastpickle.com
Tombstones need repair too
Tombstones need repair too
What and why?

Full repair
Incremental repair
How to make it work
How does anti-entropy repair work?

Reads all data
How does anti-entropy repair works?

**Reads all data**

**Calculates hashes**
How does anti-entropy repair works?

- Reads all data
- Calculates hashes
- Compares hashes
How does anti-entropy repair works?

- Reads all data
- Calculates hashes
- Compares hashes
- Streams mismatching partitions
How does anti-entropy repair works?
Merkle tree is requested to all replicas
Validation compaction

READS ALL SSTABLES AND CALCULATES HASH

SSTable  SSTable  SSTable  ······  SSTable

OUTPUTS A MERKLE TREE

www.thelastpickle.com
Merkle tree comparison
Streaming

Leaves not matching are streamed
How do we run repair?

nodetool repair

www.thelastpickle.com
Improving repair
Improving repair
Improving repair
Improving repair

repairing each range once is enough
Improving repair

`nodetool repair -pr`

www.thelastpickle.com
Improving repair

`nodetool repair -pr`

not suitable for node recovery
Sequential or parallel?

Sequential: takes a snapshot on all replicas and computes merkle trees one replica at a time (on the snapshots)
Sequential or parallel?

Parallel: No snapshot, all replicas compute merkle trees at the same time
Repair too slow?

Sequential repair is the default since C* 2.0
Repair too slow?

```
nodetool repair -par
```
The problem with dense nodes

Overstreaming

Leaves of the Merkle tree contain several partitions.

32k leaves at most.

www.thelastpickle.com
The solutions with dense nodes

cassandra_range_repair  
(Matt Stump & Brian Gallew)  
Breaks the repair sessions in n steps

Cassandra reaper  
(Spotify)  
Full orchestration tool for repairs + sub range repair support

www.thelastpickle.com
The solutions with dense nodes

vnodes : one repair session per vnode

Drawback : if you have many vnodes, repair takes longer
Repair in...

www.thelastpickle.com
The early days of your cluster

Node density is low, repair works just fine however you run it.

www.thelastpickle.com
The early days of your cluster

So maybe like I did, you run « nodetool repair » on all nodes... at the same time

www.thelastpickle.com
The (not so) early days of your cluster

As nodes gets higher in density, repair takes longer... and longer...

www.thelastpickle.com
The (not so) early days of your cluster

... and latencies rise as repair is a CPU and I/O intensive operation
Your cluster is a grown up now

... until it breaks your cluster

www.thelastpickle.com
How can it break?

Load gets too high

www.thelastpickle.com
How can it break?

Load gets too high
You don’t meet your latency SLA anymore
How can it break?

Load gets too high

www.thelastpickle.com
How can it break?

Load gets too high
Streams get stuck
How can it break?

Load gets too high
Streams get stuck
and out of nowhere, all nodes start to eat
all your CPU doing nothing

www.thelastpickle.com
The fun part?

You need to run repair to recover from the repair outage!
The cluster keeps growing
And you realize orchestration is needed to stop blowing up your cluster
Orchestrating repair

Repair must not run on all nodes at the same time
Tools to orchestrate repairs

OpsCenter repair service (DSE users)
Cassandra reaper

www.thelastpickle.com
Cassandra reaper

https://github.com/spotify/cassandra-reaper
https://github.com/thelastpickle/cassandra-reaper

www.thelastpickle.com
Cassandra reaper

Performs subrange repair

www.thelastpickle.com
Cassandra reaper

Performs subrange repair
Limits repair pressure
Retries failed sessions
Cassandra reaper

Performs subrange repair
Limits repair pressure
Retries failed sessions
(auto-)Schedules cyclic repairs
Cassandra reaper

Performs subrange repair
Limits repair pressure
Retries failed sessions
(auto-)Schedules cyclic repairs
Optimizes cluster load

www.thelastpickle.com
Cassandra reaper - with UI (thx Stefan Podkowinski)

### Repair

<table>
<thead>
<tr>
<th>ID</th>
<th>State</th>
<th>Cluster</th>
<th>Keyspace</th>
<th>CFs</th>
<th>Incremental</th>
<th>Repaired</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NOT_STARTED</td>
<td>twcs308</td>
<td>test</td>
<td>true</td>
<td>false</td>
<td>0/01</td>
</tr>
</tbody>
</table>

**Repair**

- **Cluster**: twcs308
- **Keyspace**: test
- **Tables**: table1, table2, table3
- **Owner**: alex
- **Segment count**: amount of segments to create for repair
- **Parallism**: Parallel
- **Repair intensity**: repair intensity
- **Cause**: reason repair was started
- **Incremental**: false

[www.thelastpickle.com](http://www.thelastpickle.com)
What and why?
Full repair
Incremental repair
How to make it work
Automated repairs
What if we stopped repairing repaired data?
Here comes the savior!

C* 2.1 introduces incremental repair

Default repair mode since C* 2.2
How does incremental repair work?

First repair:
- Reads all unrepaired SSTables and calculates hash
- SSTable 1: repairedAt: 0
- SSTable 2: repairedAt: 0

Second repair:
- SSTable 1: repairedAt: 1472114760
- SSTable 2: repairedAt: 1472114760
- Reads all unrepaired SSTables and calculates hash
- SSTable 3: repairedAt: 0
- SSTable 4: repairedAt: 0

www.thelastpickle.com
Anticompaction

SSTable 1
repaiAt: 0
A B C

Repair session for vnode A

SSTable 2
repaiAt: 1472115235
A

Anticompaction

SSTable 3
repaiAt: 0
B C

Repair session for vnode B

SSTable 4
repaiAt: 1472116285
B

Anticompaction

SSTable 5
repaiAt: 0
C

Repair session for vnode C

Anticompaction

SSTable 5
repaiAt: 1472125401
C

www.thelastpickle.com
Anticompaction (repair on all ranges on local node)

SSTable 1
repairedAt: 0
A B C

Repair session for vnode A

Repair session for vnode B

Repair session for vnode C

SSTable 1
repairedAt: 1472125401
A B C

Anticompaction
Incremental repair looks awesome…

…but has flaws and drawbacks
Incremental repair caveats

Carefully prepare your switch to incremental repair

www.thelastpickle.com
Incremental repair caveats

Carefully prepare your switch to incremental repair

i.e. do not run « nodetool repair -inc » straight away…

www.thelastpickle.com
Incremental repair caveats

It doesn’t handle missing/corrupted data that was already repaired

Obviously.

www.thelastpickle.com
Incremental repair caveats

It splits SSTables in 2 sets that cannot be compacted together (think tombstone purge)

www.thelastpickle.com
Incremental repair caveats

It is incompatible with subrange repair (anticompaction)
Incremental repair caveats

It doesn’t like concurrency very much

www.thelastpickle.com
Incremental repair caveats

Validator.java:261 -

Failed creating a merkle tree for [repair #e4c782d0-11fc-11e6-b616-51a3849870bb on table_v2/table_attributes,
[(8835460833482333317,8838777311566358575],
(-7300486781514672850,-7298192396576668423],
(-959298474675167225,-959177964106074209]]], /10.10.10.33
(see log for details)

www.thelastpickle.com
Incremental repair caveats

CompactionManager.java:1320 - Cannot start multiple repair sessions over the same sstables
Incremental repair caveats

CASSANDRA-8316

A running anticompation prevents validation compaction

www.thelastpickle.com
Incremental repair caveats

Do not use -pr with incremental repair
Incremental repair caveats

Do not use -pr with incremental repair

Useless: data is repaired once only
Incremental repair caveats

Do not use -pr with incremental repair

Useless: data is repaired once only anyway
Misleading: anticompaction partially disabled
Incremental repair bugs

CASSANDRA-11696
Fixed in 2.1.15, 2.2.7, 3.0.8, 3.8

Incremental repairs can mark too many ranges as repaired

www.thelastpickle.com
Incremental repair bugs

CASSANDRA-13153
Fixed in 2.2.10, 3.0.13, 3.11.0, 4.0

Reappearing Data when Mixing Incremental and Full Repairs

www.thelastpickle.com
Incremental repair bugs

CASSANDRA-9143
Fix planned for 4.0

SSTables marked as repaired on some nodes only

Because: node can fail during anti compaction
or: SSTables can get compacted during repair

www.thelastpickle.com
Incremental repair bugs

CASSANDRA-10446
Fix planned for 4.0

Spotted by Paulo Motta in the comments: SSTables are streamed with a repairedAt value.

www.thelastpickle.com
Incremental repair will not…

Fix a poor repair strategy
Incremental repair will not...

Prevent you from having to run full repair

www.thelastpickle.com
Reaper does support incremental repair

github.com/thelastpickle

www.thelastpickle.com
Reaper and incremental repair

No subrange repair
Reaper and incremental repair

No subrange repair

Single repair thread => no concurrency
What and why?
Full repair
Incremental repair
How to make it work

www.thelastpickle.com
Put your repair strategy in place on day 1
Repair best practices

Use appropriate tooling or build your own

www.thelastpickle.com
Spread repair over a `gc_grace_seconds` cycle
Repair best practices

Adjust repair pressure on your cluster (Reaper does that)

www.thelastpickle.com
Repair best practices

Don’t repair everything!

Pick tables with deletes and those with critical data

www.thelastpickle.com
If every data is critical, then none is ;)

Hmm Interesting...
Repair best practices

Be tight on your schedule with inc repair

Tombstones and anticompaction

www.thelastpickle.com
Repair best practices

Avoid concurrency with inc repair
One node at a time

www.thelastpickle.com
Repair best practices

Wait for 4.0.x before moving to incremental repair…?
Thanks!

@alexanderdeja