Cassandra exports as a trivially parallelizable problem

Emilio Del Tessandoro
Spotify

	Agenda
1	The problem
2	Introducing Cassandra-Hezo
3	Wrap up

About Emilio

- From Lucca, Italy
- Studied (Theoretical) Computer Science
- Software Engineer at Spotify
- Started 6 months ago!

CASSANDRA SUMMIT 2016

What Emilio does at Spotify?

- Part of the bases team
- Making sure that data is reliably stored, backed up and restore tested
- Advising and creating tools for operating Cassandra

Like:

- Cassandra Reaper (last year talk)
- Hecuba (other talk)
- Cassandra-Hezo (this talk)

The problem

...of exporting terabytes of data from a distributed database

The problem

We want to export all data from a distributed database.

A lot of open problems in this area, but we are eventually consistent... :)

Export data like:

- Playlists
- Financially relevant information
- Various kinds of user generated content

To be able to quickly analyze it.

So, what's out there?

Not much for batch processing (although there are streaming solutions).

- SELECT * is not enough
- COPY is not enough
- Bunch of small github projects

And at Spotify?

cass2hdfs

- Not too bad, but very fragile
- Involves shipping SSTables to Hadoop
- Custom parsing and Avro conversion in MapReduce jobs
- Runtime is dependent on the SSTable size

How we would like to solve it

- No impact on the source cluster
- Cassandra version agnostic
- Point in time snapshot
- Horizontally Scalable
- Composable (easy to understand and test)
- Possibly incremental

Introducing Cassandra-Hezo

Let’s start with this...

- We don’t want to impact the source cluster

➔ So we need to have data off the source cluster quickly

- But we also want want to be horizontally scalable

➔ So we actually need to be able to get the data to multiple machines quickly

Also...

- We want to avoid custom parsing code

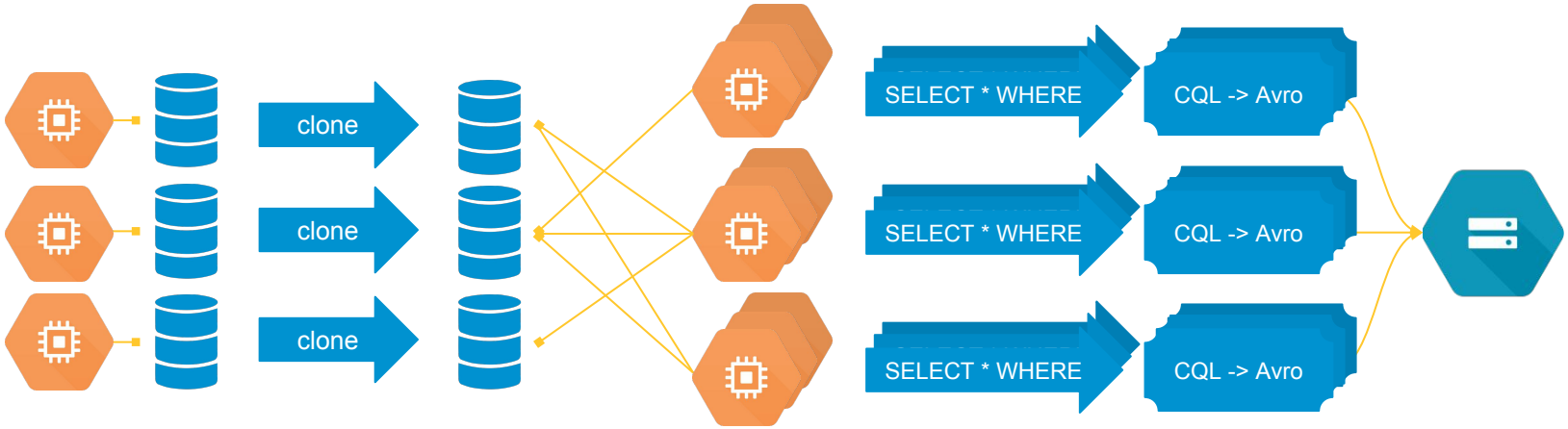
➔ So we need to use Cassandra read path, on those machines

- But SELECT * is too expensive

➔ So we need to make data more local

➔ SELECT * WHERE token(pk) < X AND token(pk) > Y

Cassandra-Hezo architecture



In case you didn’t know

Spotify is now using GCP (Google Cloud Platform).

news.spotify.com/us/2016/02/23/announcing-spotify-infrastructures-googley-future

How to clone storage in GCP

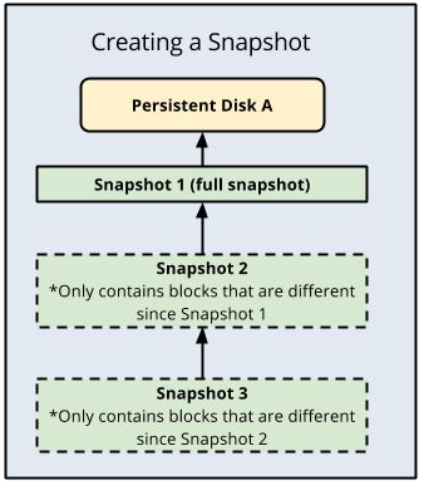
With Persistent disks (PDs)!

Interesting features like:

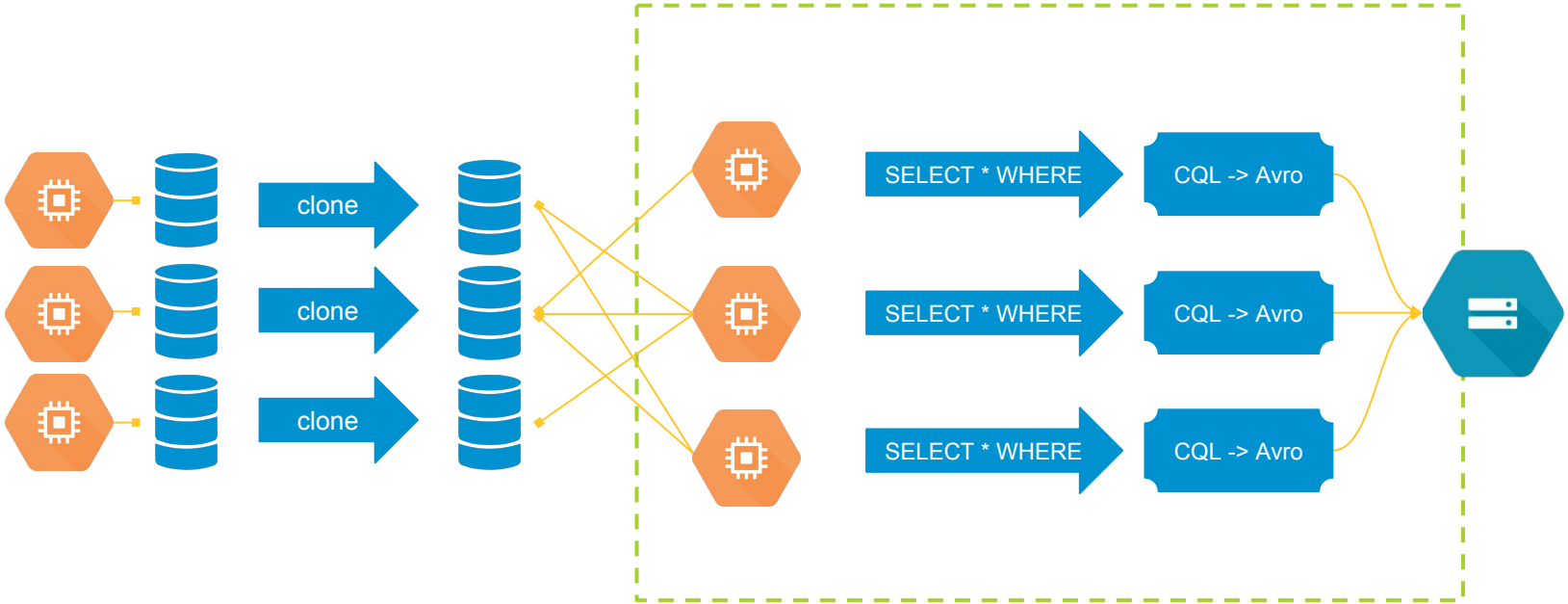
- PD snapshotting
- PD creating (from snapshot)
- PD attaching

cloning!

PD Snapshots are incremental!



Why one node clusters



Why one node clusters

- No need for internode communications
- Easier setup
- No need to attach everything to everything
- Perfect setup for even further read-tuning
- The perfect distributed application!

Implementation

- An orchestrator written in Python.
- “Just” a state machine with a bunch of external binaries.
- Super fine grain parallelization (file descriptors and I/O events).

Less than 2000 lines of code.

Including everything, from start to end.

Looking back at cass2hdfs

- ✓ We now use Cassandra read path
- ✓ Robust to topology changes
- ✓ We can easily dump single tables and exclude columns
- ✓ No need for a worker to see all the data
- ✓ Much less Cassandra specific code
- ✓ Automatic CQL -> Avro conversion

And back to our requirements

- ✓ No impact on the source cluster
- ✓ Cassandra version agnostic
- ✓ Point in time snapshot
- ✓ Horizontally Scalable
- ✓ Composable (easy to understand and test)
- ✗ Partially incremental

Performance

	Small	Medium	Large
Cassandra size	415GiB	530GiB	12.8TiB
Output size	290GiB	58GiB	2.7TiB
Avg row size	57B	124B	730B
Export time	~40min	~70min	~80min
Workers	16	24	32
Total processes	128	192	256
Export cost	$18	$30	~$75

- Around 10x faster than our previous solution.
- Without any tuning.
- Without even fully utilizing the dump machines!

Wrapping up

Wrapping up

- We can now dump our biggest cluster in less than 1 hour.
- A synergy of Cassandra and GCP snapshots.
- Developed in ~2 months by 4 people.
- Working on deployment and deprecation of the old tool.

Cassandra specific, but maybe possible for other databases.

Thanks!

Questions?