WHAT'S NEW IN PATRONI 4

ANTS AASMA

www.cybertec-postgresql.com

Overview of Patroni v4

- No more master, use primary/leader instead

Overview of Patroni v4

- No more master, use primary/leader instead
- PostgreSQL 17 support

Overview of Patroni v4

- No more master, use primary/leader instead
- PostgreSQL 17 support
- bootstrap.users has been removed

Overview of Patroni v4

- No more master, use primary/leader instead
- PostgreSQL 17 support
- bootstrap.users has been removed
- Quorum mode!!!

A walk through durability

- Default Patroni replication provides “best effort” durability guarantee
  - “best effort” here is the telco definition
  - Try not to lose too much data
- `patronictl switchover` tries to wait for replication
- `maximum_lag_on_failover` refuses to promote a replica if it’s too far behind
- Refuse to promote if leader is still alive
- Refuse to promote if another replica capable of promoting is ahead
- But if someone did promote, use pg_rewind to nuke any data that was not replicated

Synchronous mode when it’s convenient

- Customer: “We want synchronous replication in our 2 node cluster.”

Synchronous mode when it’s convenient

- Customer: “We want synchronous replication in our 2 node cluster.”
- Me: “So when the replica fails all commits will hang?”

Synchronous mode when it’s convenient

- Customer: “We want synchronous replication in our 2 node cluster.”
- Me: “So when the replica fails all commits will hang?”
- Customer: “No, that’s not cool at all.”

Synchronous mode when it’s convenient

- Customer: “We want synchronous replication in our 2 node cluster.”
- Me: “So when the replica fails all commits will hang?”
- Customer: “No, that’s not cool at all.”
- Me: “So add a third node?”

Synchronous mode when it’s convenient

- Customer: “We want synchronous replication in our 2 node cluster.”
- Me: “So when the replica fails all commits will hang?”
- Customer: “No, that’s not cool at all.”
- Me: “So add a third node?”
- Customer: “Too expensive.”

Synchronous mode when it’s convenient

- Customer: “We want synchronous replication in our 2 node cluster.”
- Me: “So when the replica fails all commits will hang?”
- Customer: “No, that’s not cool at all.”
- Me: “So add a third node?”
- Customer: “Too expensive.”
- Customer: “Can you make it synchronous when everything is working but async when something fails?”

Synchronous mode when it’s convenient

- Customer: “We want synchronous replication in our 2 node cluster.”
- Me: “So when the replica fails all commits will hang?”
- Customer: “No, that’s not cool at all.”
- Me: “So add a third node?”
- Customer: “Too expensive.”
- Customer: “Can you make it synchronous when everything is working but async when something fails?”
- Me: “. . . ”

Patroni synchronous mode

- Needed to figure out some useful but rigorous formulation of this requirement.

Patroni synchronous mode

- Needed to figure out some useful but rigorous formulation of this requirement.
- Synchronous mode guarantee:

  “A commit that has been confirmed will never be lost without sysadmin intervention.”

Patroni synchronous mode

- Needed to figure out some useful but rigorous formulation of this requirement.
- Synchronous mode guarantee:

  “A commit that has been confirmed will never be lost without sysadmin intervention.”
- Simpler version: *“No silent data loss”*

How synchronous mode operates

- When there is a good enough replica it gets assigned synchronous status.

```
synchronous_standby_names = "replica"
```


How synchronous mode operates

- When there is a good enough replica it gets assigned synchronous status.

  ```
  synchronous_standby_names = "replica"
  ```

- We store in etcd that leader and synchronous replica are guaranteed to have the latest state.

How synchronous mode operates

- When there is a good enough replica it gets assigned synchronous status.

  ```
  synchronous_standby_names = "replica"
  ```

- We store in etcd that leader and synchronous replica are guaranteed to have the latest state.
- When there is no leader in the cluster we refuse to promote anyone except the last leader and synchronous replica.

How synchronous mode operates

- When there is a good enough replica it gets assigned synchronous status.

  `synchronous_standby_names = "replica"`

- We store in etcd that leader and synchronous replica are guaranteed to have the latest state.
- When there is no leader in the cluster we refuse to promote anyone except the last leader and synchronous replica.
- If the replica goes down, remove it from etcd and reset `synchronous_standby_names`.

Ordering is important

- Need to set `synchronous_standby_names` first and then store in etcd.
- When removing synchronous standby then this needs to happen in reverse.
- If we do it in the wrong order and there is a failure we could lose data.

Latency on failure

- When the synchronous replica fails we have to wait until Patroni notices before any commits can succeed.
- Even when we have more than one replica.
- Otherwise we need to check every synchronous replica who has the latest state.

Quorum commit

- Enables use of PostgreSQL quorum commit

```
synchronous_standby_names = "ANY 2 (n2, n3, n4, n5)"
```

- Store this information in etcd.

```
{
    "leader": "n1",
    "sync_standby": "n2,n3,n4,n5",
    "quorum": 2
}
```

- When the leader fails, ask 2 other nodes for their WAL position.
- If we are up to date, try to promote.

Based on set overlaps

- Because consensus is handled by etcd have less constraints.

Based on set overlaps

- Because consensus is handled by etcd have less constraints.
- Can have an even number of nodes.

Based on set overlaps

- Because consensus is handled by etcd have less constraints.
- Can have an even number of nodes.
- Can replicate to more or less than half of nodes.

Based on set overlaps

- Because consensus is handled by etcd have less constraints.
- Can have an even number of nodes.
- Can replicate to more or less than half of nodes.
- The set of nodes that are reachable must overlap the set of nodes that acknowledged commit.

Based on set overlaps

- Because consensus is handled by etcd have less constraints.
- Can have an even number of nodes.
- Can replicate to more or less than half of nodes.
- The set of nodes that are reachable must overlap the set of nodes that acknowledged commit.
-

$$|Sync| + |Reachable| > |Nodes| \Rightarrow |Sync \cap Reachable| \geq 1$$

Different configurations

With a 5 node cluster:

- 2 copies of data (`s_s_n = ANY 1 (...)`) means need 4 nodes up

Different configurations

With a 5 node cluster:

- 2 copies of data (`s_s_n = ANY 1 (...)`) means need 4 nodes up
- 3 copies of data (`s_s_n = ANY 2 (...)`) means need 3 nodes up

Different configurations

With a 5 node cluster:

- 2 copies of data (`s_s_n = ANY 1 (...)`) means need 4 nodes up
- 3 copies of data (`s_s_n = ANY 2 (...)`) means need 3 nodes up
- 4 copies of data (`s_s_n = ANY 3 (...)`) means need 2 nodes up

Different configurations

With a 5 node cluster:

- 2 copies of data (`s_s_n = ANY 1 (...)`) means need 4 nodes up
- 3 copies of data (`s_s_n = ANY 2 (...)`) means need 3 nodes up
- 4 copies of data (`s_s_n = ANY 3 (...)`) means need 2 nodes up
- 5 copies of data (`s_s_n = ANY 4 (...)`) means need 1 node up

A long time coming

- The year was 2018...

A long time coming

- The year was 2018. . .
- PostgreSQL 10 with quorum commit had just been released last fall

A long time coming

- The year was 2018...
- PostgreSQL 10 with quorum commit had just been released last fall
- ```
  commit 0ae0cb0602a82b85a6f51b45d64f2097f6d11c72
  Author: Ants Aasma <ants.aasma@eesti.ee>
  Date:   Mon Apr 30 11:32:20 2018 +0300

      Quorum commit feature
  ...
  8 files changed, 585 insertions(+), 141 deletions(-)
  ```



  ```

What took so long

- It’s complicated.

What took so long

- It’s complicated.
- We started adding node 6, but got interrupted after changing `s_s_n`.

What took so long

- It’s complicated.
- We started adding node 6, but got interrupted after changing `s_s_n`.
- Meanwhile node 2 failed.

What took so long

- It’s complicated.
- We started adding node 6, but got interrupted after changing `s_s_n`.
- Meanwhile node 2 failed.
- What changes need to be made to configuration and in which order to not lose guarantees?

Using the feature

- Step 1: turn it on

```
patronictl edit-config --set synchronous_mode=quorum
```


Using the feature

- Step 1: turn it on

```
patronictl edit-config --set synchronous_mode=quorum
```

- Step 2: wait 10 seconds

Using the feature

- Step 1: turn it on

```
patronictl edit-config --set synchronous_mode=quorum
```

- Step 2: wait 10 seconds
- Step 3: done!

That’s all

Thanks!

Questions?