Cross-Cluster Replication

Rahul Desirazu

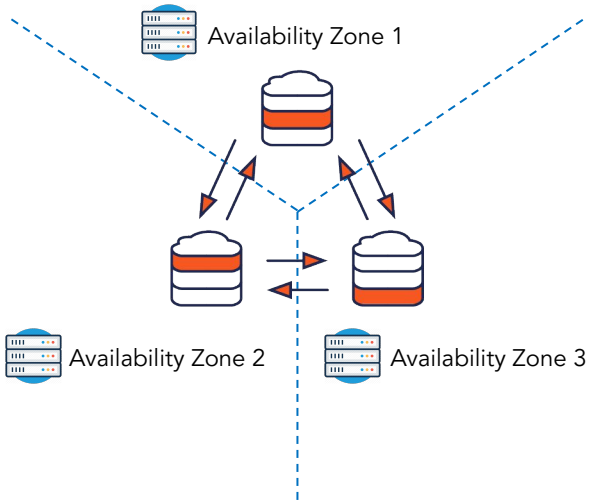
Friday, May/6/2022

YFTT
YUGABYTEDB
FRIDAY
TECH TALKS

yugabyteDB

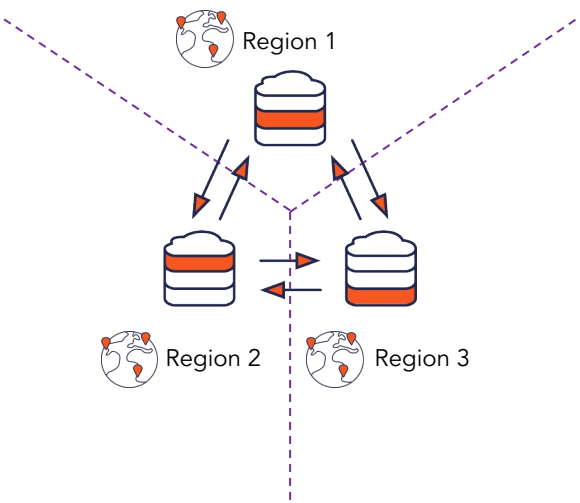
Resilient and strongly consistent across failure domains

1. Single Region, Multi-Zone



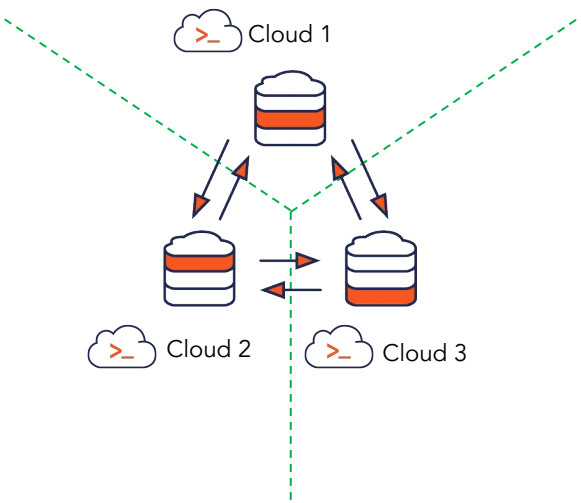
Consistent Across Zones
No WAN Latency But No
Region-Level Failover/Repair

2. Single Cloud, Multi-Region



Consistent Across Regions
with Auto Region-Level
Failover/Repair

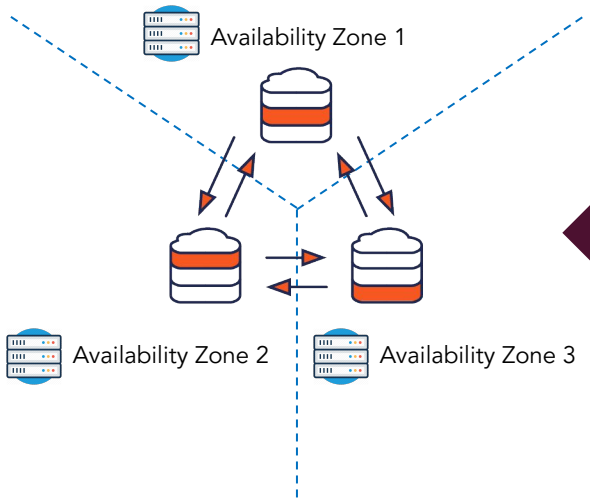
3. Multi-Cloud, Multi-Region



Consistent Across Clouds
with Auto Cloud-Level
Failover/Repair

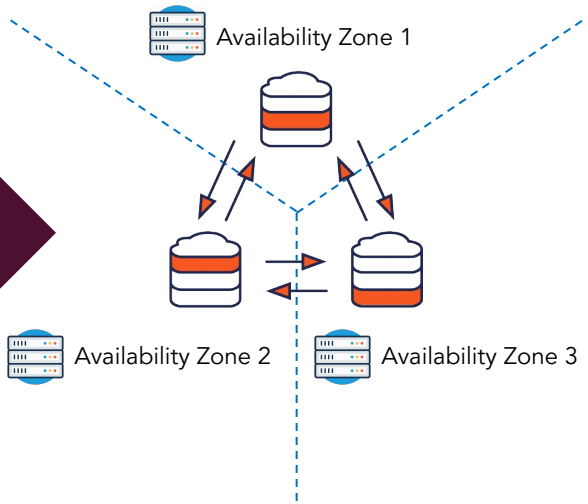
Multi-Master Deployments w/ xCluster Replication

Master Cluster 1 in Region 1



Async Replication

Master Cluster 2 in Region 2



Consistent Across Zones
No Cross-Region Latency for Both Writes & Reads
App Connects to Master Cluster in Region 2 on Failure

Consistent Across Zones
No Cross-Region Latency for Both Writes & Reads
App Connects to Master Cluster in Region 1 on Failure

Additional use cases

Active-active setups where both clusters take writes on the same table and share data with each other.

- Other topologies:
  - 1 : N
  - N : 1
  - Triangle Formations

High-level objectives and functional requirements

- Eventual read consistency on both clusters
- Last-writer wins (i.e. highest hybrid time) in an active-active setup
- Low RPO, in case of DR can recover to most recent time point
- Low RTO, how long it takes to promote target universe

User operations

- Setup, Delete, Alter Replication Groups
- Create a replication stream from existing data
- Pause/Resume replication
- Monitoring replication:
  - Lag metric, computed per table as an aggregate max across all its shards

Architecture

High-level overview

- The GetChanges(tablet, opid) replication polling API is called at a per tablet level
- Each source tablet uses its local (Write- Ahead Log) WAL to serve replication requests
- WALs are retained for 24 hrs (configurable through gflag)

The source side maintains a system cdc_state table to store metadata about streams, usable both internally and externally for observing progress.

Tablet	Stream	Checkpoint OpID	Last Replicated Time

Replication walkthrough

1. User runs DDL on both sides
2. User calls setup_universe_replication on Target Universe for n tables
3. Table Schema Validation
4. Creating the tablet mapping
   a. Target Tablet -> Source Tablet
   b. Leader of Target Tablet polls for Source Tablet
   c. Number of shards doesn’t need to match
5. Source side computes all records since the last poll (using opid) and returns them to the target
6. Target servers apply records from source to local rocksdb

Handling failures and downtime

- Transient Failures on Poll or Apply
- Server/Data-center Outages and Rolling Restarts
- Network Partitions
  - Less than 24hr
  - More than 24hr

Ongoing work

- Automatic DDL Replication
  - Indexes, Create + Alter Table

- Atomic and Ordered Multi-Shard Transactions

- Overall Hardening for Large Setups with Many Tables

Demo

Thank You

Join us on Slack: yugabyte.com/slack (#yftt channel)

Star us on Github: github.com/yugabyte/yugabyte-db

YFTT YUGABYTEDB FRIDAY TECH TALKS

yugabyteDB