Understanding High Availability options for PostgreSQL

Madan Kumar K
Member of Technical Staff, ScaleGrid.io

@ImMadanK

ScaleGrid

High Availability 101

★ Redundancy is the key

❖ Standalone vs. Master-Standby

➔ Master Server

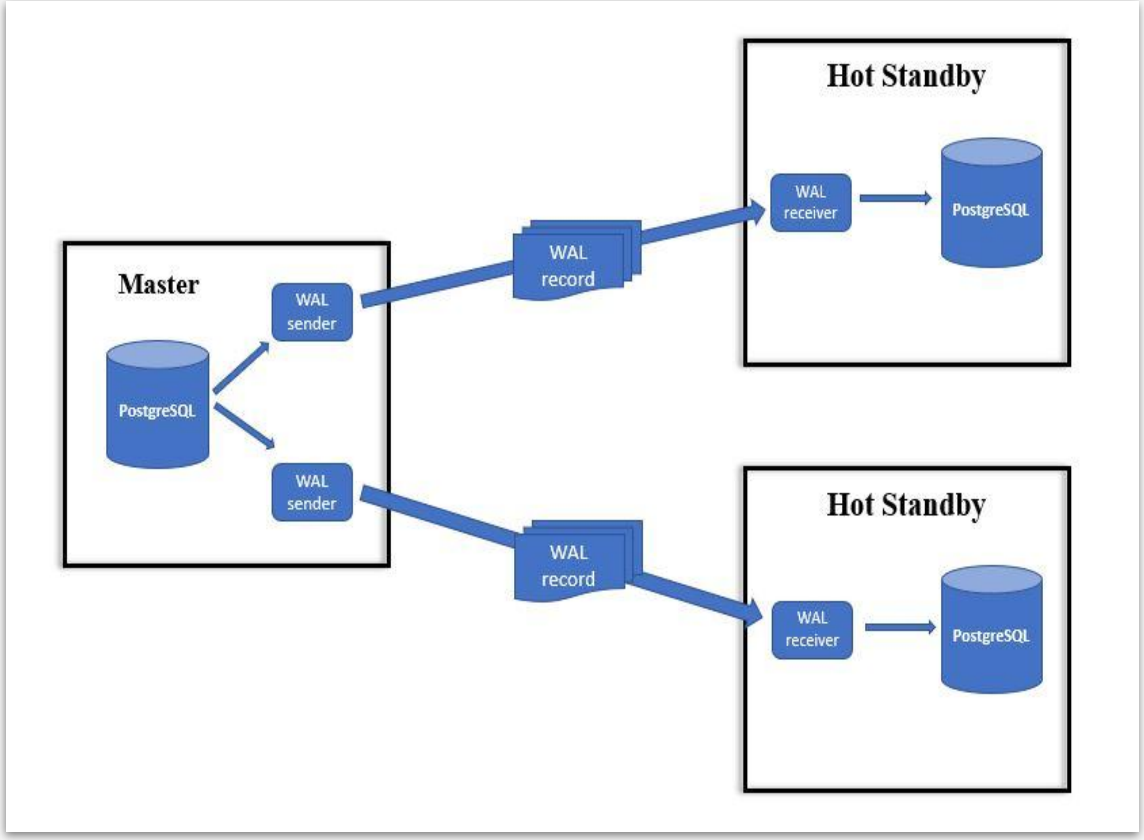
➔ Standby Server

◆ Warm Standby

◆ Hot Standby

Streaming Replication

- Uses Write-Ahead Logging (WAL)
- Built-in feature
- Types
  - Async
  - Sync



Managing High Availability

Framework Requirements

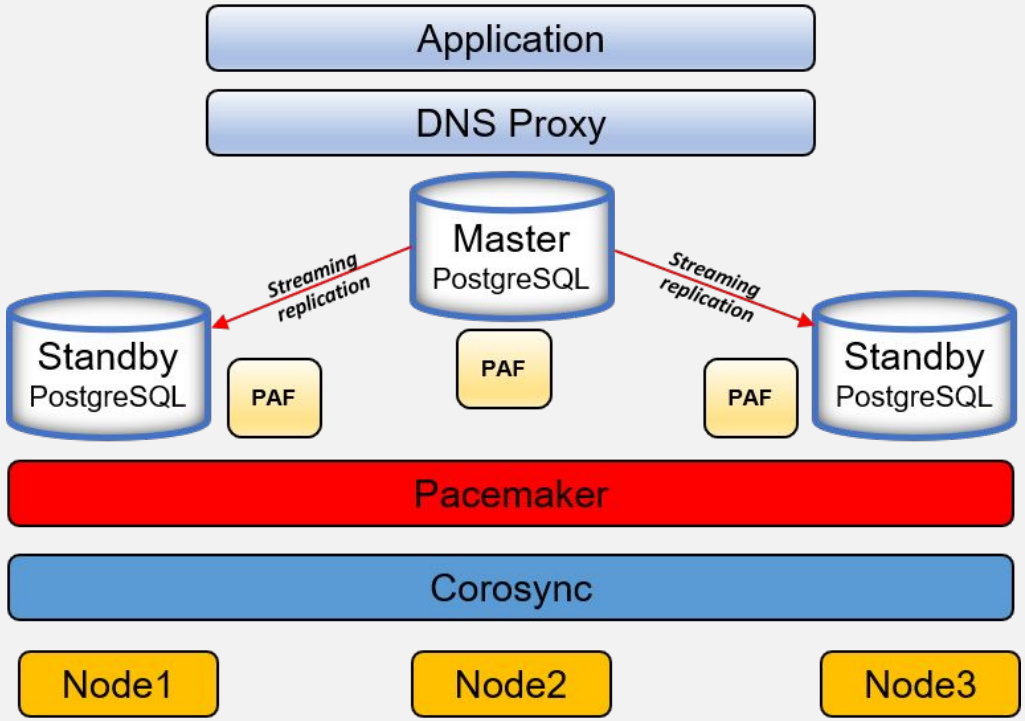
- ➔ Failure detection
- ➔ Failure recovery
- ➔ Automatic failover
- ➔ Consensus

Well-Known Frameworks

- ❑ PostgreSQL Automatic Failover (PAF)
- ❖ Replication Manager
- ★ Patroni

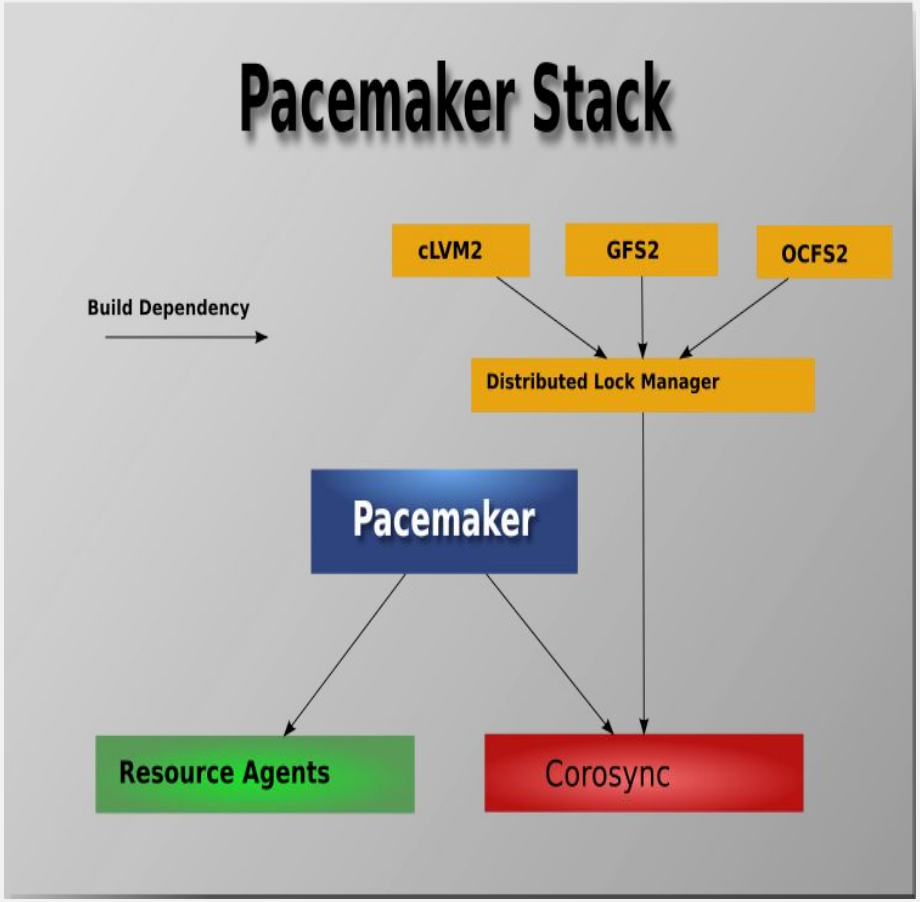
PostgreSQL Automatic Failover

- HA management Solution by Cluster Labs
- Pacemaker + Corosync stack
- Pacemaker Resource Agent
- Written in Perl
- Open Cluster Framework Compliant



How PAF works?

- ➔ Pacemaker interaction
- ➔ Monitors status of each node to detect failure
- ➔ Recover the Failure
- ➔ Irrecoverable failure on master? Failover
- ➔ Elects the best available Standby



PAF: Setup Requirements

- ❖ PostgreSQL 9.3 & above
- ❖ Streaming replication
- ❖ Hot Standby
- ★ Recovery template
  - ◆ standby_mode
  - ◆ recovery_target_timeline
  - ◆ primary_conninfo (must include application_name)
- ★ Custom Parameters
  - ➔ bindir
  - ➔ pgdata
  - ➔ pgport
  - ➔ recovery_template
  - ➔ start_ops
  - ➔ maxlag

Replication Manager

- Replication & failover management tool suite by 2ndQuadrant®
- PostgreSQL Extension
- Written in C language

repmgr

★ Command Line Tool

- ➔ Setup standby
- ➔ Promote standby
- ➔ Switchover

repmgrd

★ Daemon that actively monitors servers

- ➔ Failure detection
- ➔ Automatic failover
- ➔ Event notification

How repmgr works?

- Utilities to setup replication
- Primary and secondary nodes registration
- repmgr schema
  - Tables & Views
- Promote, Follow & Switchover
- repmgrd (shared preload lib)
- Automatic Failover
- Rejoin cluster
- Event notification

repmgr: Setup Requirements

- PostgreSQL 9.3 & above
- Passwordless ssh connectivity between all servers
  - Switchover
  - Cluster crosscheck
  - Copy config files

★ repmgr conf

- node_id
- node_name
- conninfo
- data_directory
- service specific commands
- use_replication_slots

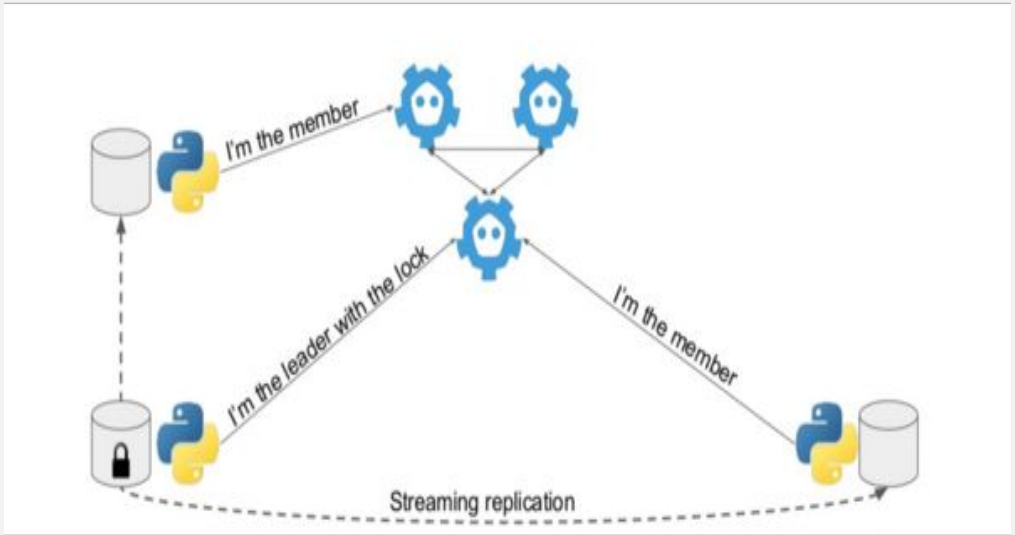
Patroni

- HA solution template by zalando
- Supports many Distributed Configuration Store (DCS)

- ★ Customizable standby creation methods
- ★ REST API
- ★ Dynamic configuration
- ★ Callbacks

How Patroni works?

- ➔ Initialize the cluster from single node
- ➔ Replica creation
- ➔ Patroni agent
- ➔ Leader lock using DCS
- ➔ Automatic failover
- ➔ Rejoin using pg_rewind
- ➔ Callbacks
- ➔ Watchdog



Patroni: Setup Requirements

- One of the DCS must be installed.
- Python module specific to that DCS
- Environment Config settings
  - To override values in yaml config

★ YAML Configuration

- ➔ Global/Universal
- ➔ Bootstrap
- ➔ DCS specific
- ➔ PostgreSQL
- ➔ REST API

Framework Comparison

- Interfaces & Utilities
- How they work in Distributed System? Consensus, Network Split etc..
- Failure detection & recovery.
- Features supported
  - Ports
  - Event notification
  - Public IP based deployments

Master Failure

- PAF restarts the master service in case of process stop/kill.
  - Irrecoverable failure leads to election
- repmgr doesn't restart the master service in case of process kill/stop.
  - Instead wait for fixed interval before triggering election
  - Manual intervention is required.
- Patroni restarts the master service in case of process stop/kill.
  - If master doesn't recover within ***master_start_timeout***, election is triggered.

Master Failure

- ❑ PAF uses IP address failover to ensure Standby follows the new master.
- ❖ repmgr restarts the PostgreSQL service on Standby to follow new master.
- ★ Patroni restarts the PostgreSQL service on Standby to follow new master.

Standby Failure

- ❑ PAF restarts the standby service in case of process stop/kill.
- ❖ repmgr doesn't restart the standby service in case of process kill/stop.
  - ➢ Manual intervention is required.
- ★ Patroni restarts the standby service in case of process stop/kill.

pg_rewind support

- ❑ PAF doesn't support pg_rewind
- ❖ repmgr supports pg_rewind as part of node rejoin command.
- ★ Patroni support pg_rewind.
  - ○ Automatically detects if rewind is required.

Consensus Algorithm

- PAF uses Pacemaker + Corosync.
  - Totem Single-Ring Ordering and Membership Protocol
- repmgr doesn't have consensus algorithm.
- Patroni supports various DCS and consensus algorithm will be specific to that DCS.
  - Etcd and Consul uses RAFT
  - Zookeeper uses Zab

Network Partitioning

- PAF stops the service on the node which is isolated from majority based on Quorum policy.
- repmgr provides ***location*** parameter to address the concern.
  - In case of Split, Promotes the standby which has same *location value* as of previous primary.
  - if nothing is specified, “default” is the value for location. Can lead to multi-master scenario.
- Patroni demotes the PostgreSQL on the node which is isolated from majority.

Handling Lagging Standby

- ❏ PAF exposes parameter **maxlag**, above which standby will be set a negative master score.
- ❖ repmgr doesn't handle lagging standby separately.
- ★ Patroni has **maximum_lag_on_failover** parameter which will ensure standby lagging behind that value will not be considered for master election.

Maintenance mode

- PAF supports putting resources in maintenance mode.
  - Can be individual resource/single node/complete cluster
- repmgr doesn't have maintenance mode.
- Patroni provides pause/resume to support maintenance mode for resources.
  - Supports only for entire cluster

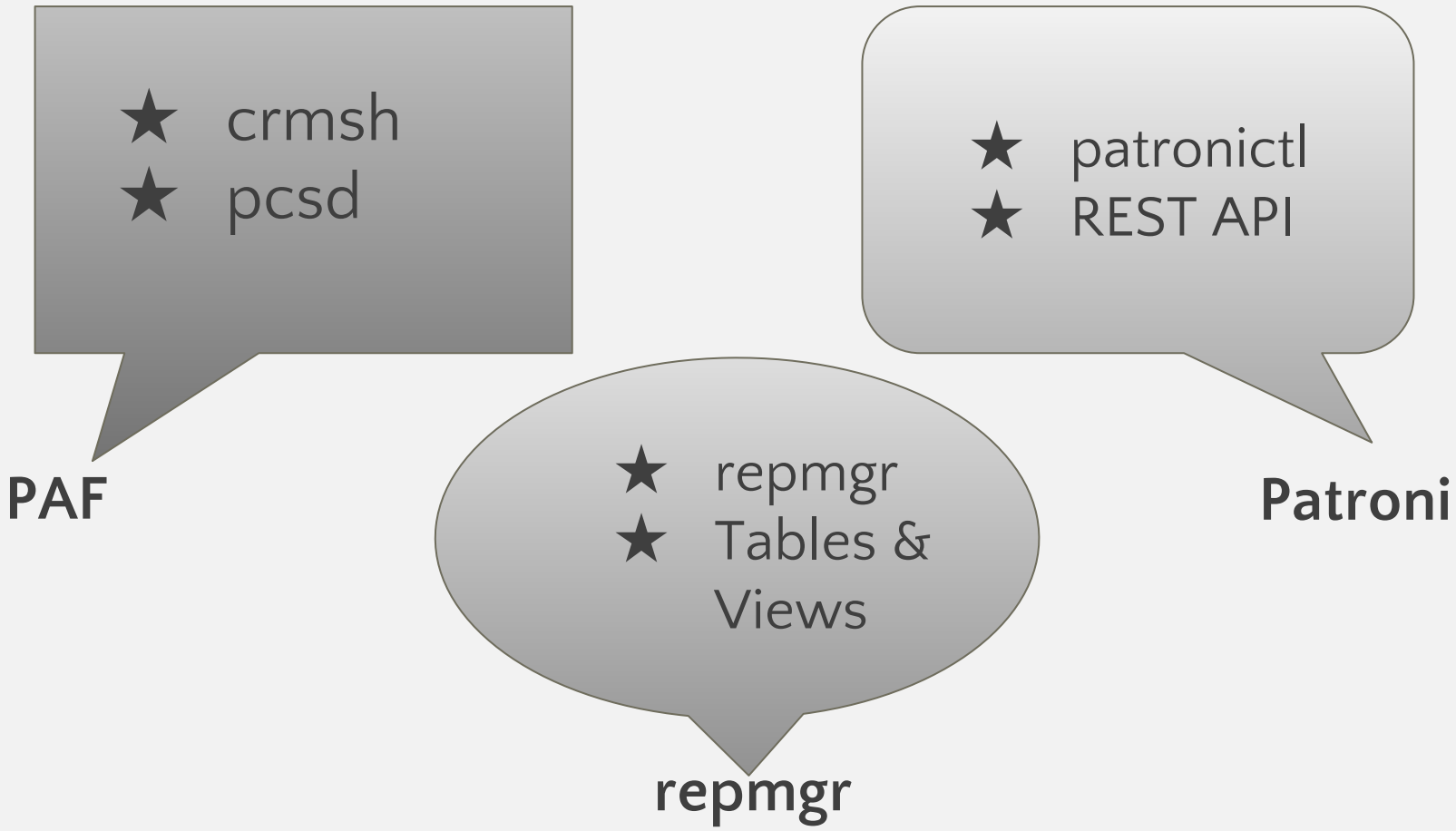
Ports usage

- ❏ PAF uses an extra UDP port (default: 5405) for corosync communication.
- ❖ repmgr doesn't need any extra ports.
- ★ Patroni uses minimum three extra ports.
  - One port for REST API
  - Minimal two for DCS.
  - Based on DCS being used number of ports can vary.

NAT Support

- ❏ PAF uses corosync, hence it doesn't support NAT/Public IP with load balancer.
- ❖ repmgr has no restrictions.
- ★ Patroni has no restrictions.

Interfaces and Utilities



Event Notification

- ❑ PAF supports event notification using Alert agents.
- ❖ repmgr supports even notification by allowing single script and passing arguments to it.
- ★ Patroni provides parameters to specify multiple scripts based on event type.

Multi-Databases Support

- PAF uses Pacemaker & Corosync Stack
- Database specific Resource Agents
- Easy maintenance

Questions?

You can reach me at

- @ImMadanK
- madan.kumar@scalegrid.io

ScaleGrid

Agenda

- ★ High Availability
- ★ PostgreSQL Streaming Replication
- ★ High Availability Management Frameworks
- ★ Frameworks Comparison

High Availability

- ★ Ensures Operational continuity for higher than normal period.
- ★ Percentage of time the services are up in a given time period.
- ★ Represented as series of 9's.
- ★ Defined based on business requirements

Availability %	Downtime per year
90% ("**one nine**")	36.53 days
99% ("**two nines**")	3.65 days
99.9% ("**three nines**")	8.77 hours
99.99% ("**four nines**")	52.60 minutes
99.999% ("**five nines**")	5.26 minutes
99.9999% ("**six nines**")	31.56 seconds

Framework Agent Failure

- ❑ PAF uses pacemaker whose failure/process kill will disable resource management for that node.
- ❖ repmgrd failure/stop will disable that node from participating in election.
- ★ Patroni supports watchdog.
  - agent crash/not run due to high load
  - slow shutdown of PostgreSQL
  - Split brain protection