PERCONA

# Pgpool-II What, Why and Where?

Muhammad Usama

Software Engineer @ Percona

Contributor & Committer @ Pgpool-II

# About Me

- Senior Software Engineer in PostgreSQL engineering at Percona
- Hacking PostgreSQL since 2007
- Major contributor & Committer of Pgpool-II
  - Contributed many features to Pgpool-II including
    - Watchdog
    - Dynamic Spare Process Management
    - Consensus and Quorum based backend failover

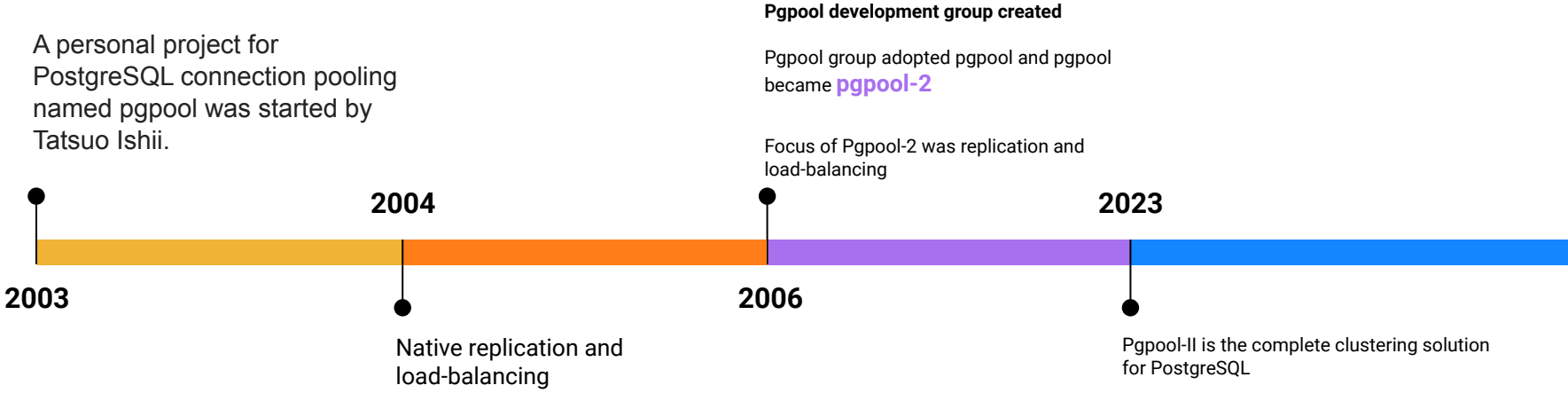
# Agenda

What is Pgpool?
- Overview, Features and Architecture

Why use it?
- Advantages and when it's not the best choice

Where is the community heading?
- Some recent enhancements and possible areas of improvement

# What's with the name Pgpool-2?






PERCONA

# Pgpool-II: Today

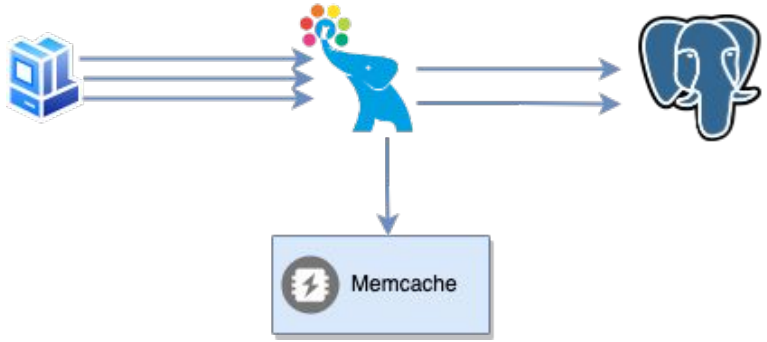
- Not just a pooler. A complete cluster management tool for PostgreSQL
- Rich featureset
- BSD License
- Actively maintained. The community has released a major version every year since its inception.
  - Multiple point releases in between
- Works with almost all PostgreSQL flavors

# Features & Capabilities

# Load balancing

- One of the most used and powerful features of Pgpool-II
- Pgpool-II automatically identifies READ-ONLY and WRITING statements and routes them to the appropriate PostgreSQL server.
- No change is required on the application side, It can work with any PostgreSQL client or application that is designed without any consideration of load balancing.
- Intelligently handles transaction states and replication lag
  - Keeps track of replication lag between primary and standby servers and dynamically adjusts the read-write routing.
- Provides a fine grain control for routing of statements.

# Query caching




- Pgpool-II has a built-in capability to cache the results of SELECT queries.
  - Subsequent queries that have results cached get automatically served by Pgpool-II without involving the PostgreSQL server.
- Provides two options for caching.
  - Shared memory caching
  - Memcache
- Supports automatic and time based cache invalidation
- Provides the configurable option to select/deselect database tables for caching
- ** Creates the cache entry at transaction commit*
  - *Breaks row visibility rules for repeatable read isolation*

# Connection pooling

- Only session level connection pooling is supported
- Each child process maintains its own pool of connections
- Configurable to manage pooled connections lifecycle
- *Big update is in the pipeline.*

# Managing Replication and failover

- Provides a statement based replication solution and complements the PostgreSQL's built-in replication capabilities.
- Extensive set of cluster management features
  - Failover
  - Failback
  - Attach, Detach
  - Online recovery
- This makes it easier to manage, control, scale up and scale down PostgreSQL cluster.

# Managing Replication and failover ..

- Configurable health check and automatic failover provides the HA for PostgreSQL nodes.
  - Backend failure detection coordinates with all Pgpool-II nodes in cluster to validate failure
- No disrupting the service when the standby nodes fails
- Provides out of the box options for streaming replication to detect and detach false-primary nodes.

# Watchdog (HA of Pgpool nodes)

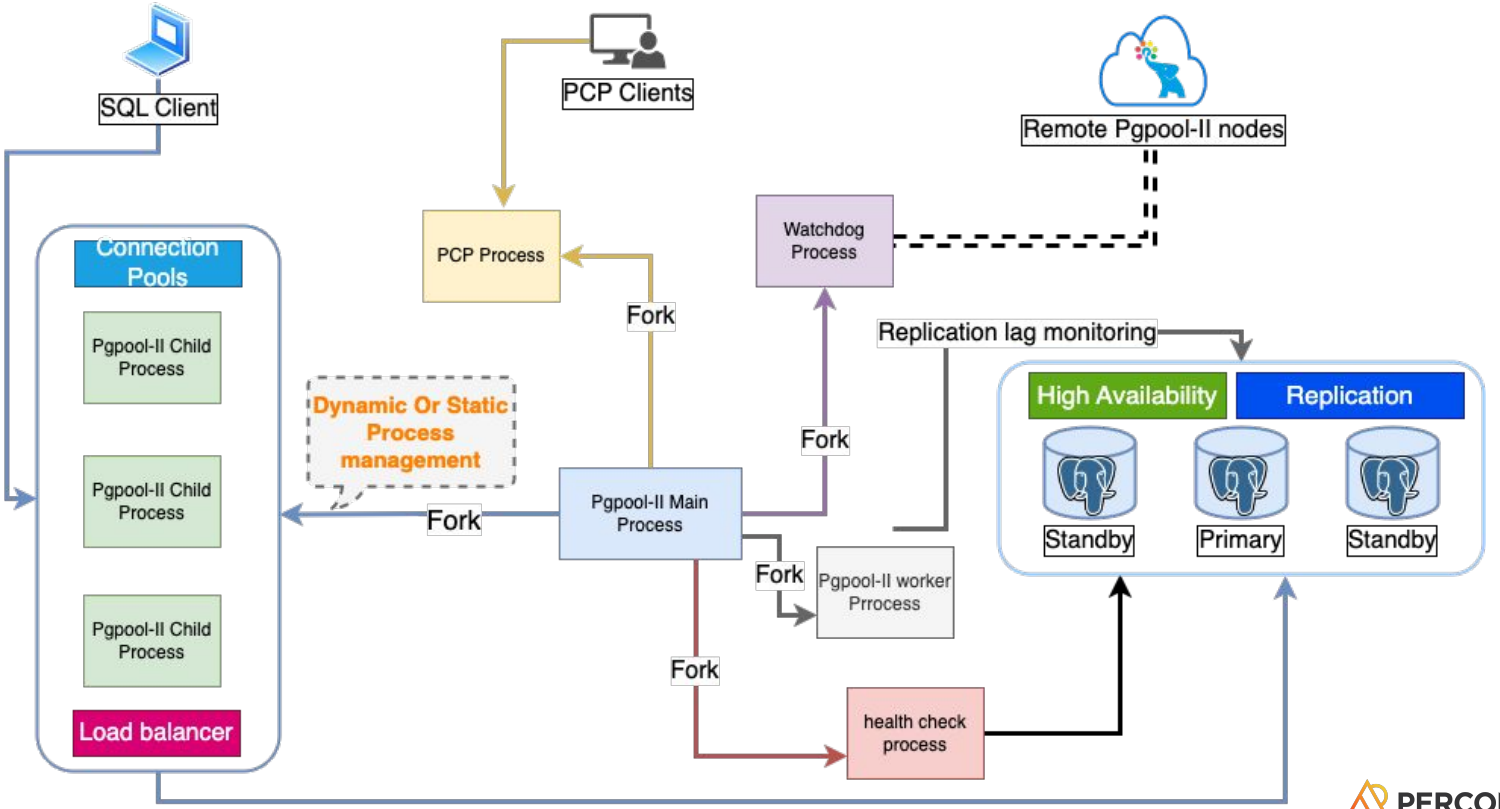
- Pgpool-II provides complete end to end HA solution for PostgreSQL
  - Watchdog solves SPOF of Pgpool-II node.
    - Virtual IP controller keeps the single point of contact for client applications
    - Keeps the consistent view of cluster
    - Leader election ensures best node selection and avoids split-brain
  - Backend health checking along with automatic failover ensures the PostgreSQL server availability
    - Consensus and quorum aware failure detection
- Pgpool-II can provide failover and redundancy to help minimize downtime and ensure availability.

# Extensive management interface

- PCP admin interface allows to remotely manage and monitor Pgpool-II instances.
  - 13 builtin PCP utilities
- PGPOOl SHOW and PGPOOL SET commands on SQL interface allows to get-set the pgpool configuration parameter values

# Architecture

# Pgpool-II Architecture






PERCONA

# Key architecture points

- Works as a proxy between PG cluster and client applications
- Pgpool inspects all statements
- Performs automatic read/write split on single client connection
- Transparent to client applications.
  - No change required on application side
- Multiprocess architecture
  - Unlike PostgreSQL, the process does not goes away when the client disconnects
- Two stage user authentication

# Why use Pgpool-II ?

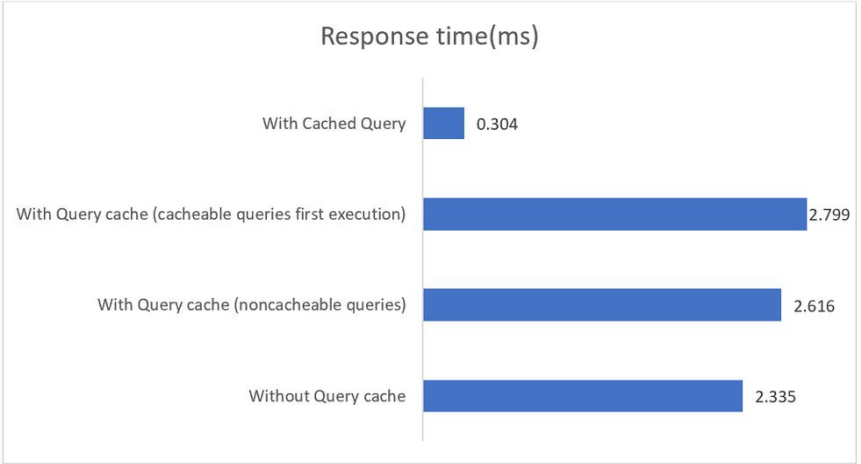
# Improving read performance

- One of the most used use cases of Pgpool-II.
  - Using Load balancing and/or query caching
- Load balancing
  - Improves overall performance for high-traffic applications.
  - Balance the workload by spreading queries across multiple servers.
  - Helps prevent any single server from becoming overwhelmed and causing a bottleneck
  - Supports session level and statement level read-node selection
- Query caching
  - Applications that frequently execute the same SELECT queries can benefit from pgpool-II's query caching feature.
  - Caching the results of frequently executed queries can reduce the load on the PostgreSQL server and improve the performance of the application.

# Query cache benchmark

- Benchmark done by someone at Microsoft[1]
- Repeating SELECT shows **86%** improvement and around 20% degradation in first time selection

*Latest version has even more performance improvements with query caching*




[1]: https://techcommunity.microsoft.com/t5/azure-database-for-postgresql/improve-performance-of-read-intensive-workloads-on-azure-db-for/ba-p/743860

# High Availability

- When your HA strategy consists of multiple standby PostgreSQL servers Pgpool-II can be the best choice.
- Offloads the read load to the standby PostgreSQL nodes that would have been sitting idle otherwise

# Scalability

- **Pgpool-II can add or remove PostgreSQL servers dynamically**.
  - Online recovery allows to setup and synchronize database nodes and attach a node without stopping the service
  - Newly attached database nodes automatically gets used for load balancing.
  - pcp_detach_node allows to remove the database node from the cluster.
- **Pgpool-II can be particularly useful in situations where you expect a sudden surge in traffic or workload**.

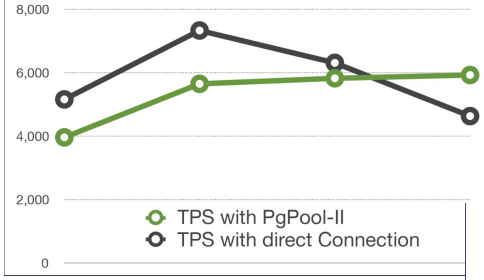
# When Pgpool-II is not the best choice?

# Small database deployments

- Pgpool is a middleware and has its own processing and communication overheads
- Load balancing yields performance benefits when resources on a single server hit the ceiling.
- For smaller workloads, a direct connection to PostgreSQL performs better.

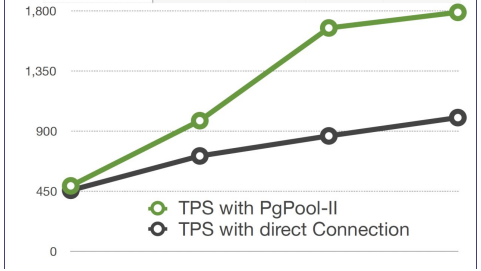
pgbench with 100 scale factor

| NO OF CLIENTS | 5 | 15 | 25 | 45 |
|---|---|---|---|---|
| TPS with PgPool-II | 3,947 | 5,635 | 5,810 | 5,913 |
| TPS with direct Connection | 5,144 | 7,317 | 6,296 | 4,618 |




pgbench with 500 scale factor

| NO OF CLIENTS | 5 | 15 | 25 | 45 |
|---|---|---|---|---|
| TPS with PgPool-II | 487 | 975 | 1,669 | 1,786 |
| TPS with direct Connection | 454 | 711 | 861 | 998 |






PERCONA

# Limited resources or high latency-network

- When Pgpool-II shares the host with database server that has a limited hardware resources.
- Introducing Pgpool-II in solution where network has high-latency can degrade the overall performance.

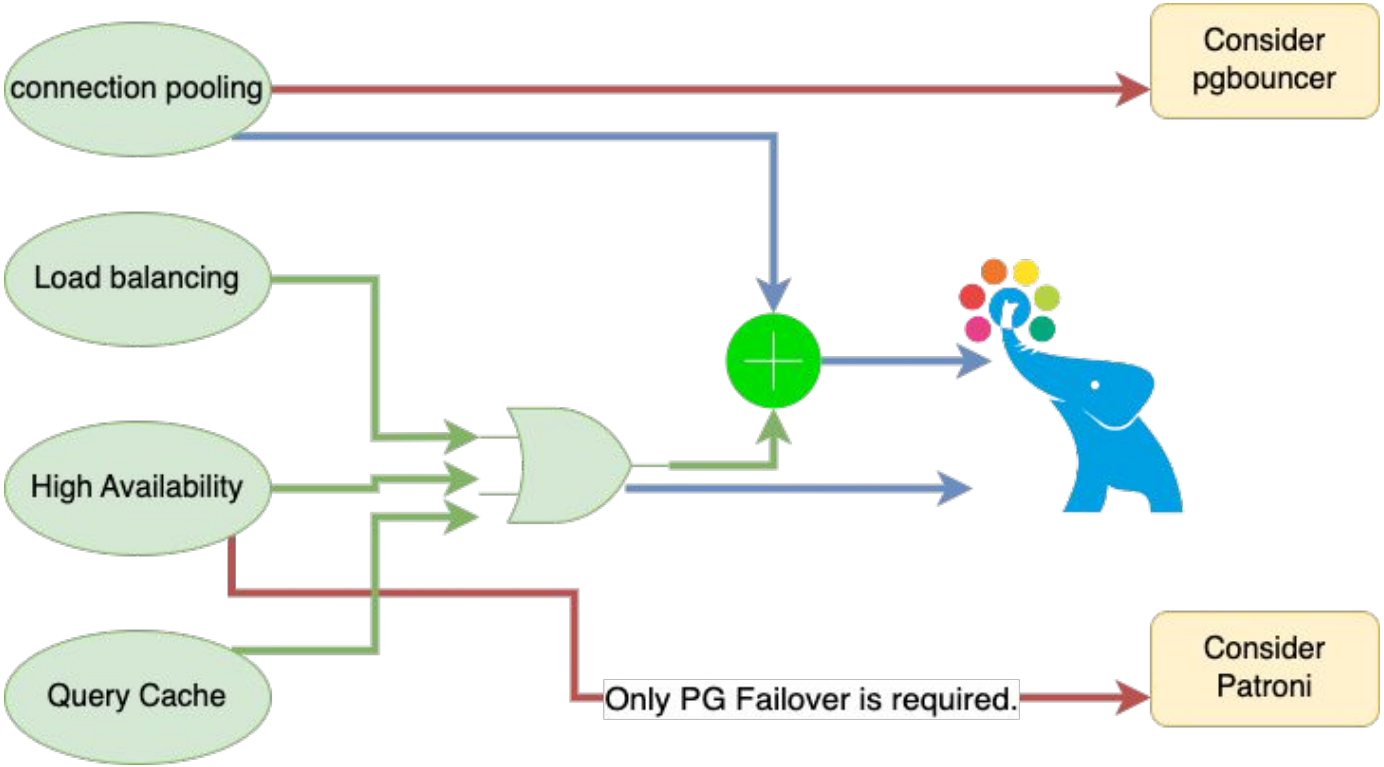
# Only requirement is connection pooling

- Pgpool-II only supports session level connection pooling.
  - Transaction and Statement level pooling is not available.
- When you want to minimize the number of open connections to the database server

# Key takeaways

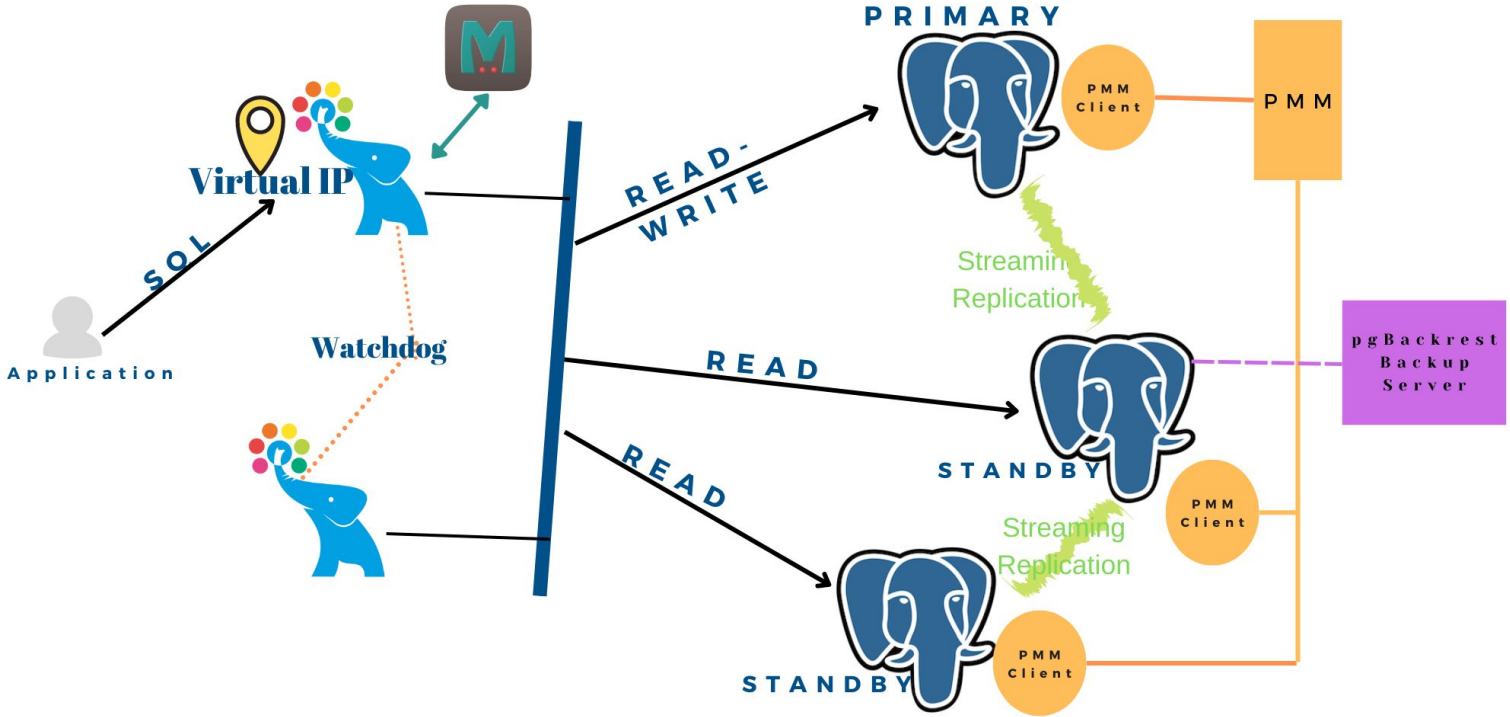
- When the only requirement is connection pooling. There are better choices than Pgpool.
- If your workload requires query result cache and/or read scalability and/or high availability, then Pgpool-II can be a good choice.
  - Connection pooling of pgpool can complement these use cases.
- If you are planning for HA only. You can evaluate Pgpool-II and other HA solutions (like: patroni) based on the requirements
  - Pgpool-II provides a very reliable HA solution that is easier to manage and set up.
  - On the other hand, it adds a hop between the server and the client.

# Considering Pgpool-II?

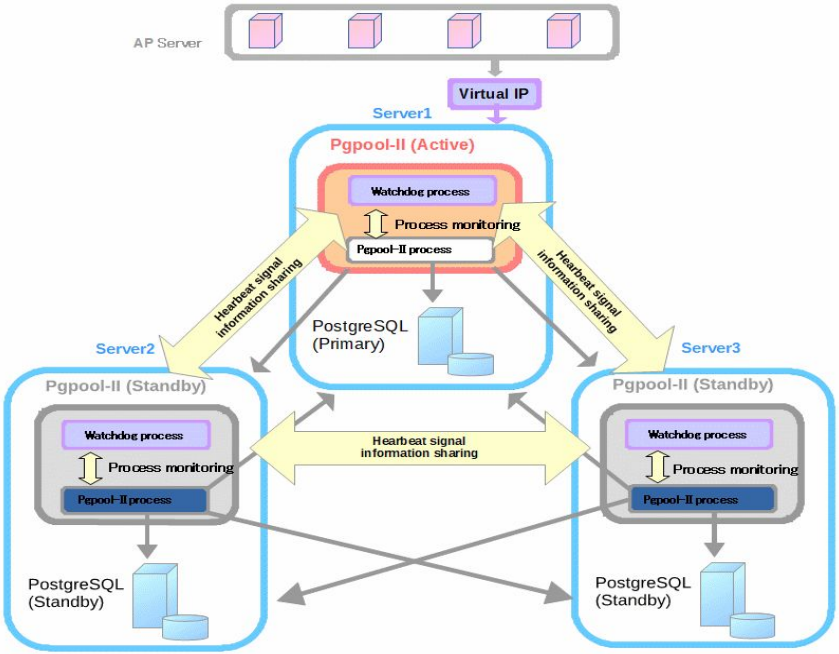


# Typical Deployments

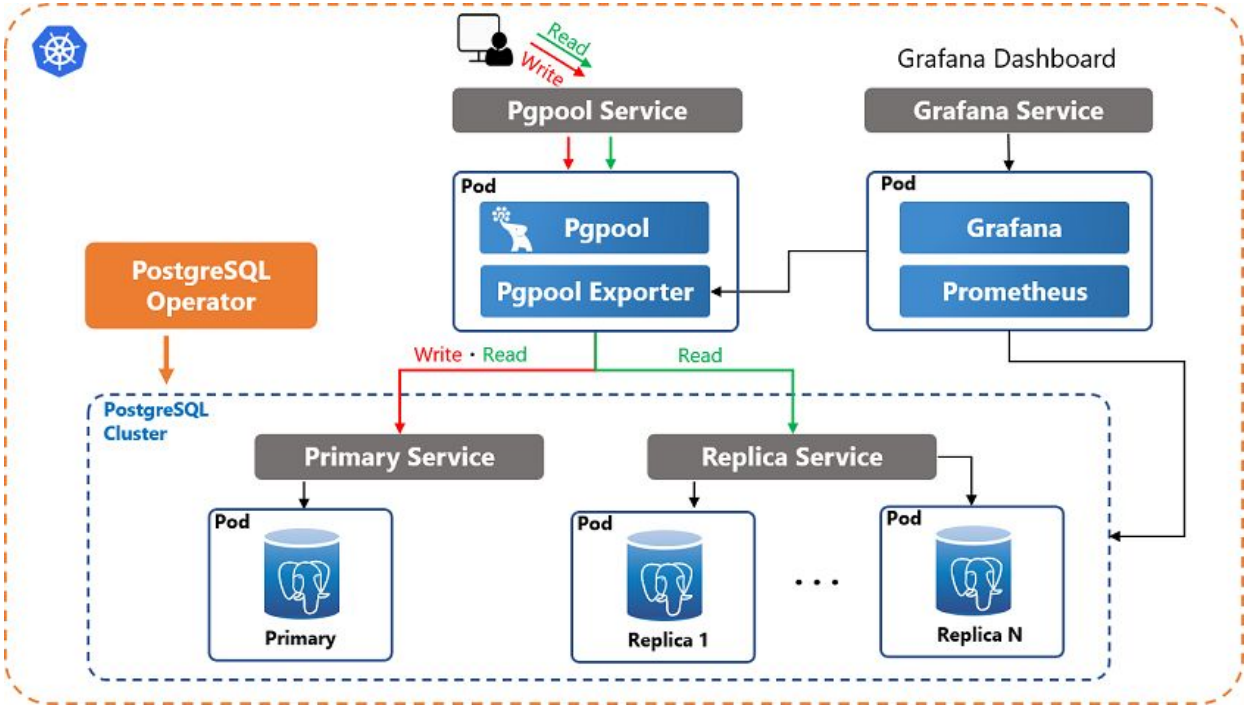
# Deploying Pgpool-II for HA, Load balancing & caching



# PG & Pgpool-II sharing same hosts



# Pgpool-II on Kubernetes



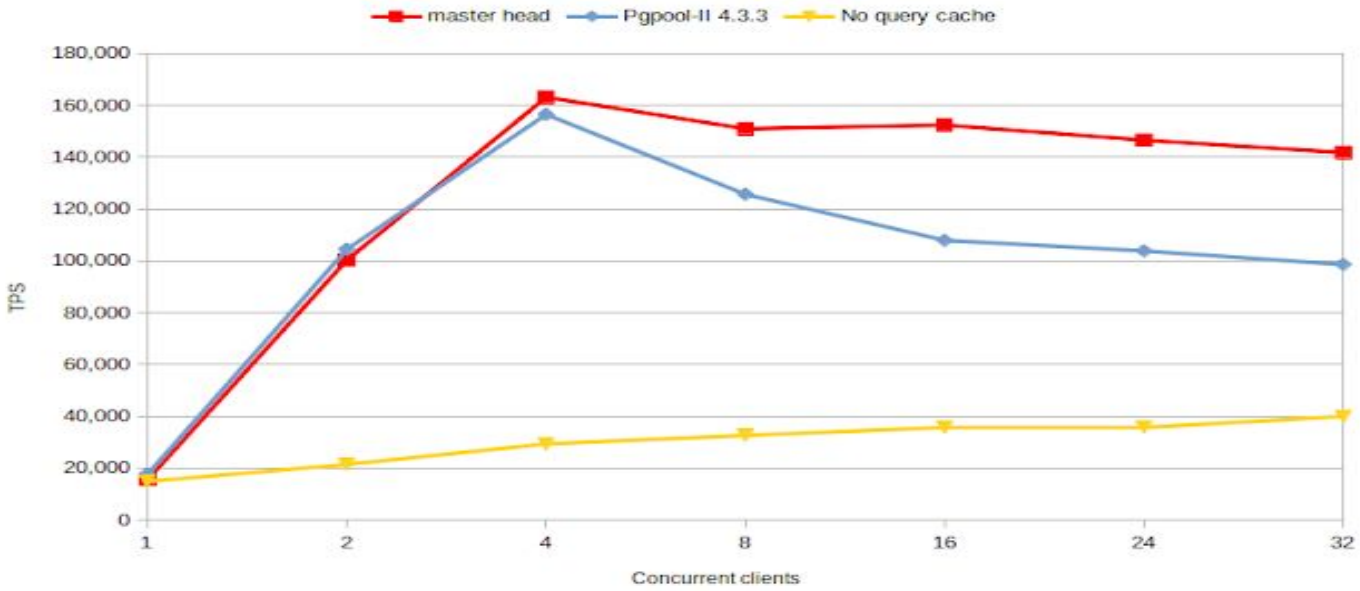

Ref: https://www.pgpool.net/docs/pgpool-II-4.2.8/en/html/example-kubernetes.html

# What's been happening in the Pgpool-II world?

# Latest updates in Pgpool-II

- New dynamic spare process management feature.
- Allow to specify replication delay by time in streaming replication mode.
- Speed up query cache by reducing lock contention.

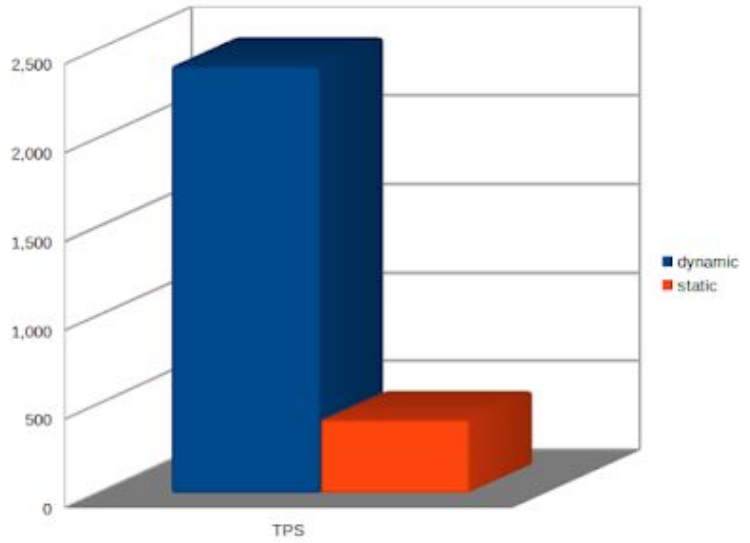
# 140% performance improvement from 4.3




Ref: https://pgsqlpgpool.blogspot.com/2022/09/query-cache-improvement-in-pgpool-ii-44.html

# Dynamic process management mode

- Traditionally Pgpool-II spawns all child processes at startup
- Dynamically adjust the spare child process based on the load
- Very useful for workloads that have spikes in number of concurrent connections
- Initial benchmarks show **5X** performance improvements in certain use cases




Ref: https://pgsqlpgpool.blogspot.com/2022/11/dynamic-spare-process-management-in.html

# Some more noteworthy updates

## Watchdog (HA) advancements

- Consensus based backend failure detection.
- Dynamic membership of Pgpool-II nodes in watchdog cluster

## Ease of use

- Support for include directive in configuration file
- Unified configuration for watchdog
- More relevant default configuration values

## Load balancing improvements

- prefer_lower _delay_standby consider replication delay while choosing load balancing node
- Automatic identification of writing function
- Statement level load balancing mode

# Some more noteworthy updates …

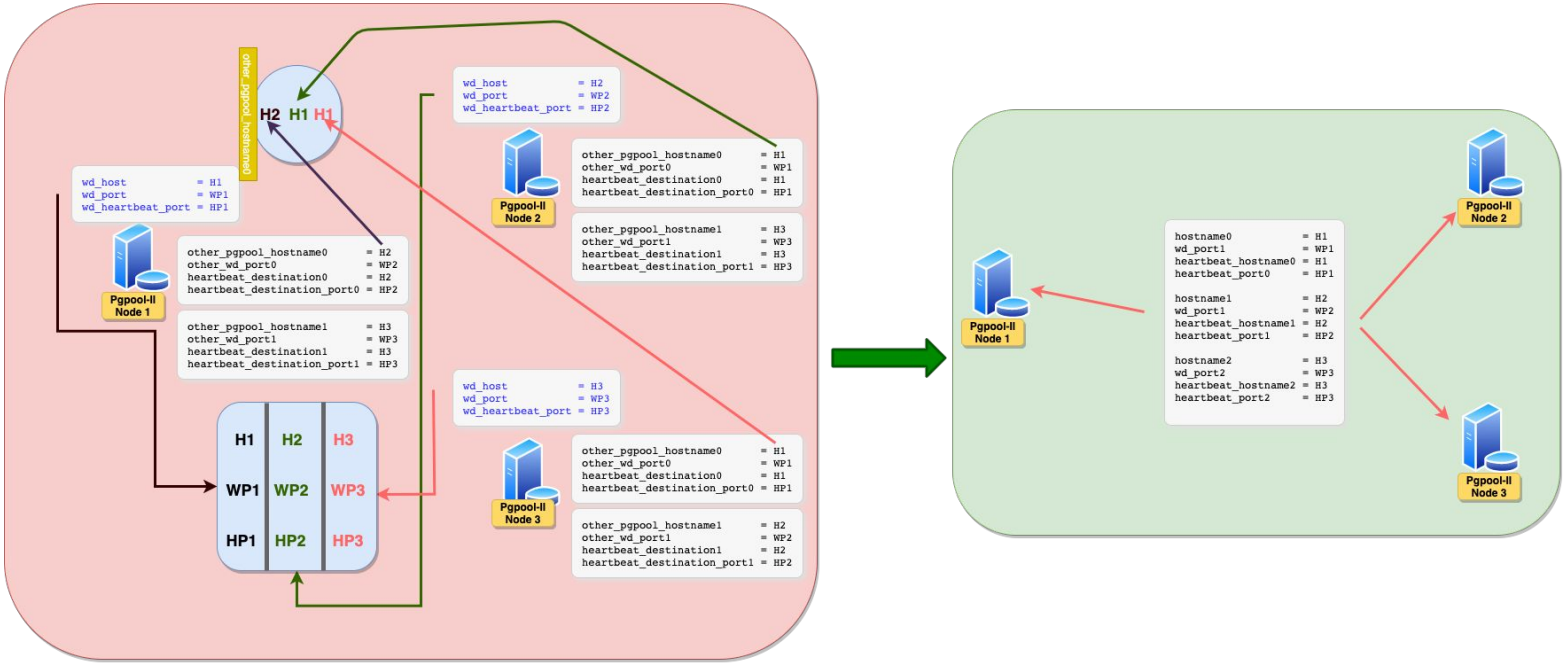
## Performance

- Multiple parsers to eliminate unnecessary parsing
- Extended query protocol enhancements

## Miscellaneous updates

- Switchover mode for pcp_promote_node
- Logging collector
- Support for LDAP authentication
- False primary detection
- Support for AWS Aurora and logical replication

# Watchdog configuring is getting easier





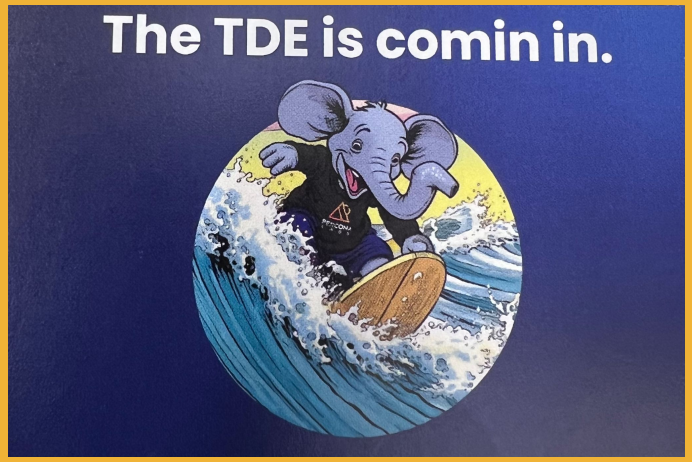

PERCONA

# Possible improvement avenues

- Documentation
- Connection pooling
- Pgpool administration tools
- Snapshot isolation clustering mode can be enhanced to create read-write scale out solution
- Change to multithreaded architecture

# Enhancing connection pooling for next version

- Currently each child process maintains it's own connection pool.
  - It is not an efficient technique
  - Requires lot of backend connections
  - Only session level connection pooling is possible
- Moving towards global connection pool
  - Possible to pre-warm the connection pool
  - Provide transaction and statement level pooling
  - Reduce the number of backend connections
- Want to try/contribute?
  - WIP for global connection pool implementation https://github.com/codeforall/pgpool2




# Percona & PostgreSQL Better Together

**Percona stands for Evolution**

**Percona stands for Ease-Of-Use**

**Percona stands for Freedom**


- **Email:**
  - muhammad.usama@percona.com
- **LinkedIn:**
  - https://www.linkedin.com/in/muhammadusama/
- **GitHub:**
  - https://github.com/codeforall


# THANK YOU!

percona.com