EDB™
POSTGRES

PostgreSQL HA with Pgpool-II and whats been happening in Pgpool world lately...

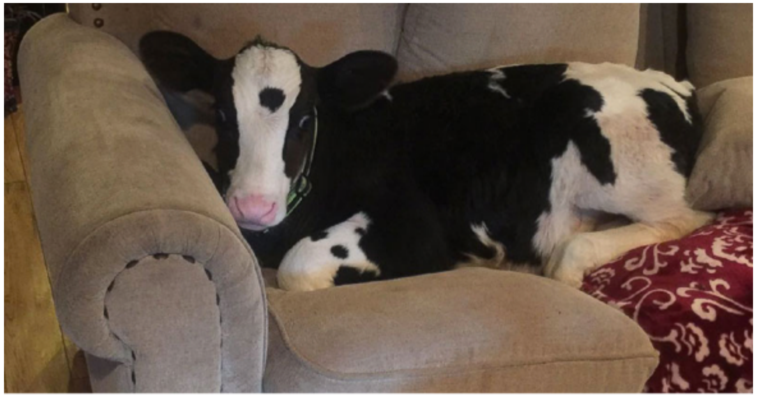
About Me

- Muhammad Usama
- Database Architect with EnterpriseDB
- Pgpool developer and committer
  - Watchdog overhauling
  - New PCP system for Pgpool-II
  - New authentication method support
  - Quorum support for backend failover

muhammad.usama@enterprisedb.com

https://www.linkedin.com/in/muhammadusama

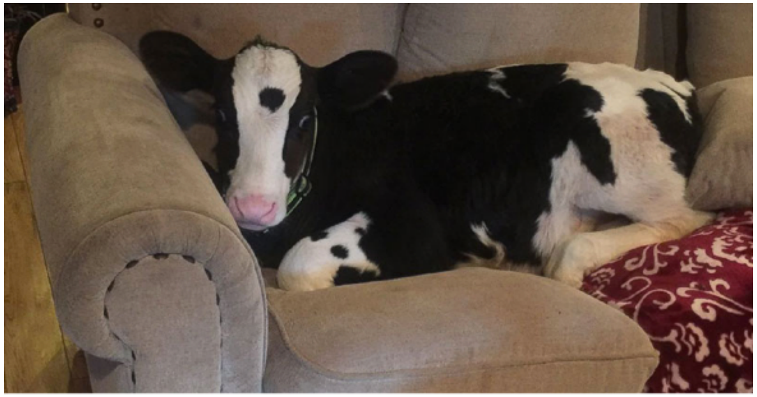
Pets Vs Cattle Herd



- We treat our servers like pets (e.g. Sunshine the cow). If Sunshine gets sick, It feels like the end of the world and everything stops there

- We treat our servers like cattle herd (C01, C02, … C99). If some cattle gets sick, there is system in place to isolate it from the herd and things go on.

Pets Vs Cattle Herd



- Sunshine the cow is unique and indispensable
- Sunshine the cow is hand fed and hand crafted
  - Cannot handle failure

- Herd of similar cattle (C01,C02..)
- No cattle is special and indispensable
  - Can handle failures

Using the herd approach for HA in database

- Herd approach can be used on database servers to achieve performance scalability and high availability.

How?

- Make a homogeneous copies of database servers (replicated databases)
- Have a system to manage the herd and failures.

Challenges in the herd approach

- Herd needs a leader (Primary server).
- A system is required to elect new leader if the current leader becomes unavailable (Primary node failure).
- Needs a mechanism to make herd follow the new leader.
- We need a system to seamlessly retire the sick nodes (Standby node failure).
- A system to add new nodes without effecting the service.
- Require a system to make the whole herd work in collaboration to efficiently utilise the resources (Load balancing)

The Solution

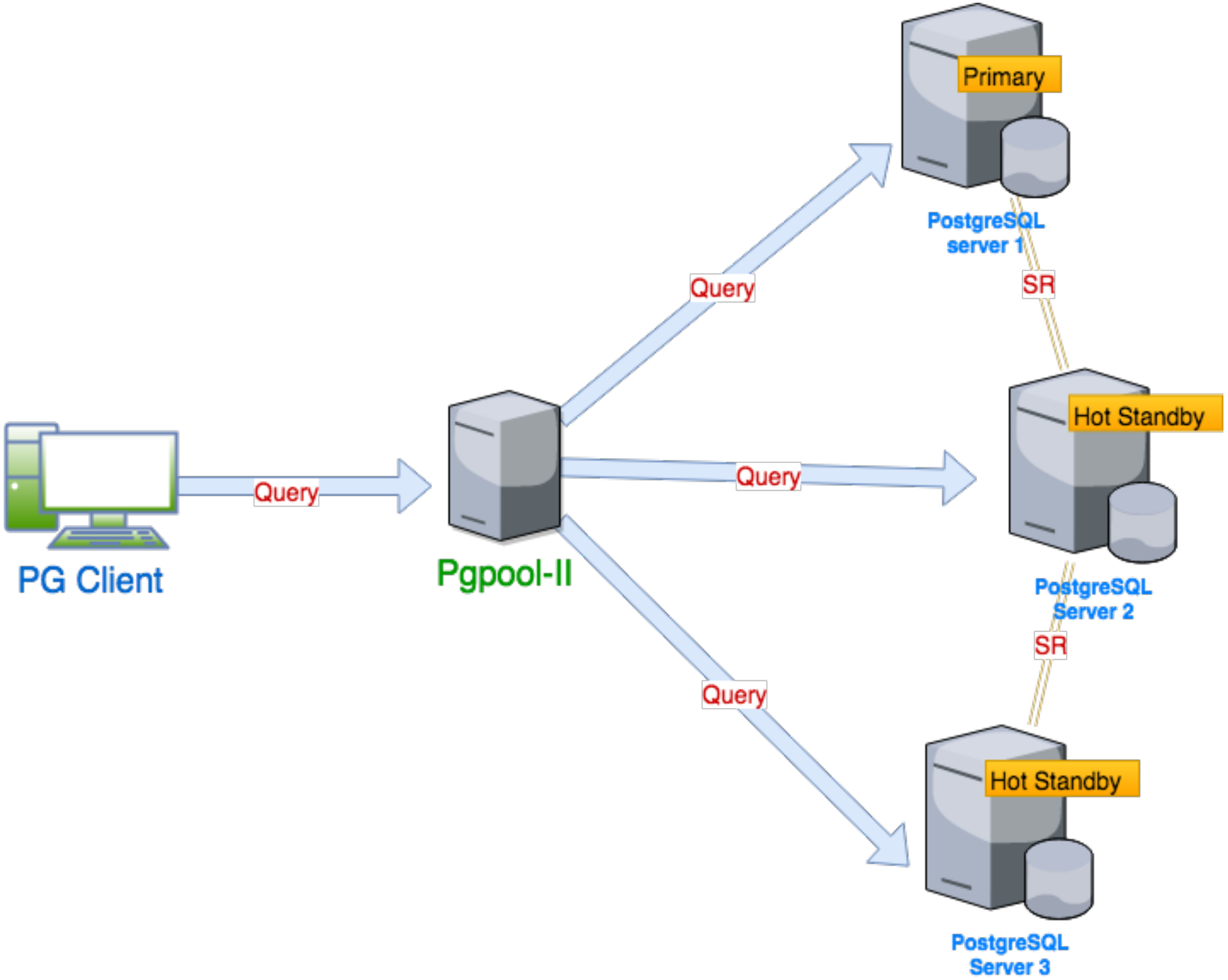
What is Pgpool-II?

- Cluster management tool dedicated for PostgreSQL
- Rich in features
    - Connection pooling
    - Load balancing
    - Automatic failover
    - Query caching
    - Watchdog (High availability)
- OSS project, BSD License

Solving the problem using Pgpool-II

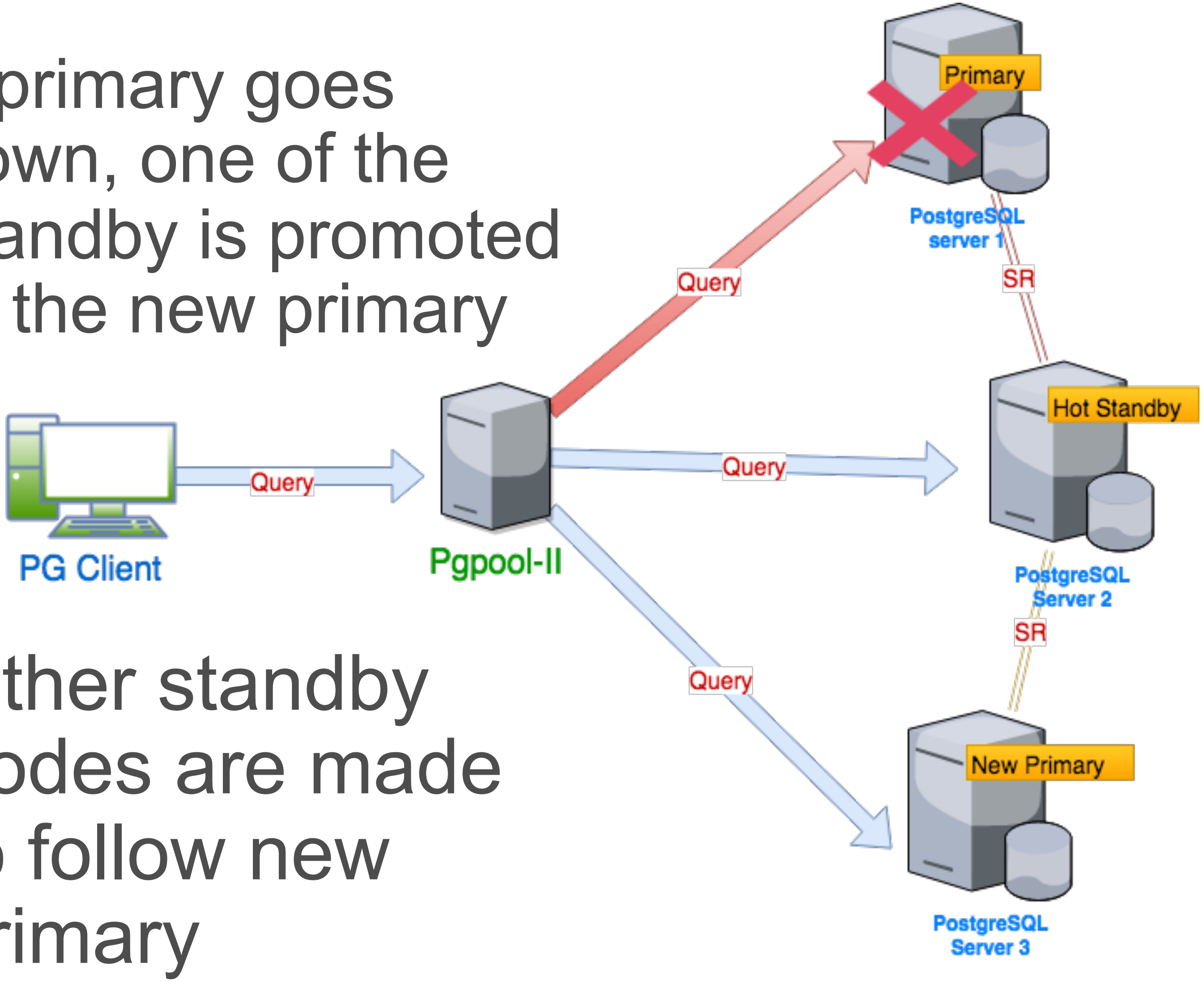
- Pgpool-II make the cluster appears as a single PostgreSQL instance
- All standard PostgreSQL clients work seamlessly
- Automatic failover
- Provides flexibility and control to select the primary node when old primary fails
- Load balancing

Basic idea of Pgpool-II



Primary server fails

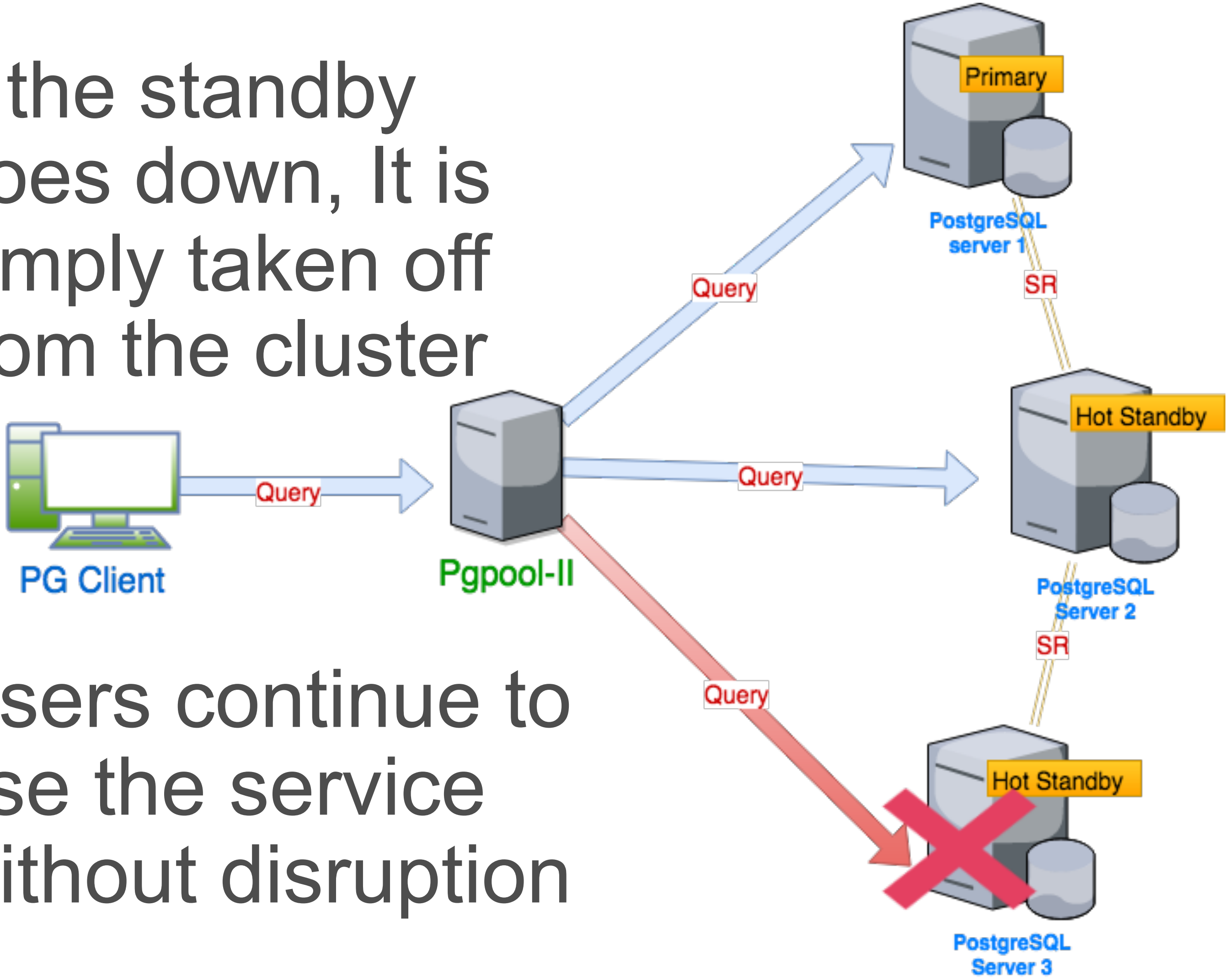
- If primary goes down, one of the standby is promoted to the new primary



- Other standby nodes are made to follow new primary

Standby server fails

- If the standby goes down, It is simply taken off from the cluster



- Users continue to use the service without disruption

Automatic failover

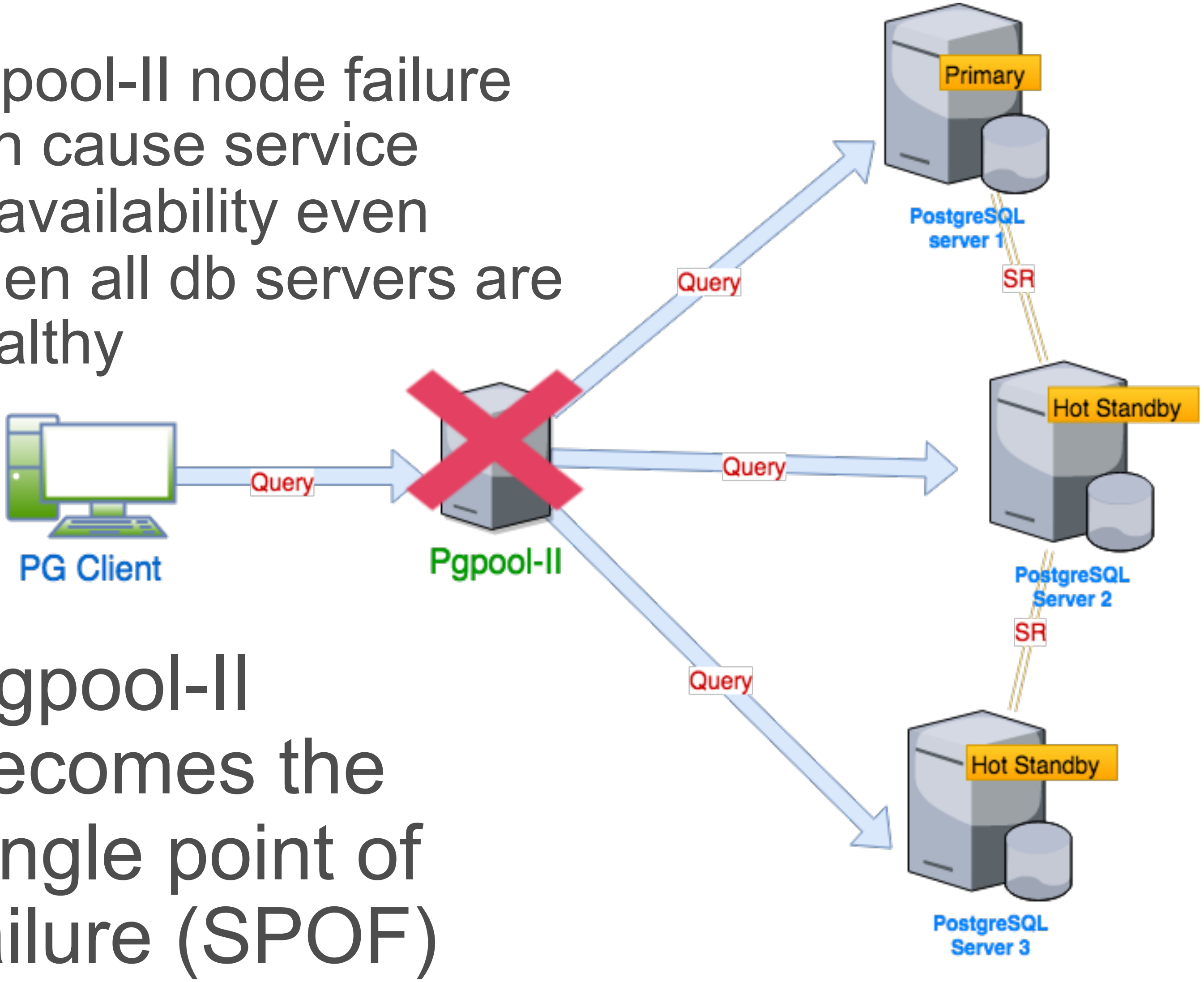
- Mechanism in Pgpool-II to detach problematic nodes
- Reconfigure the standby nodes to follow new primary node
- Automatically triggers
  - When health check monitor the node failure
  - Reading/writing failure to PostgreSQL backend (*failover_on_backend_error* is on)
  - By remote Pgpool-II node (Watchdog)

Does that solves the problem?

- Using Pgpool-II with failover and health check ensures the service availability when PostgreSQL node fails
- But there is still a problem

What if Pgpool-II fails

- Pgpool-II node failure can cause service unavailability even when all db servers are healthy



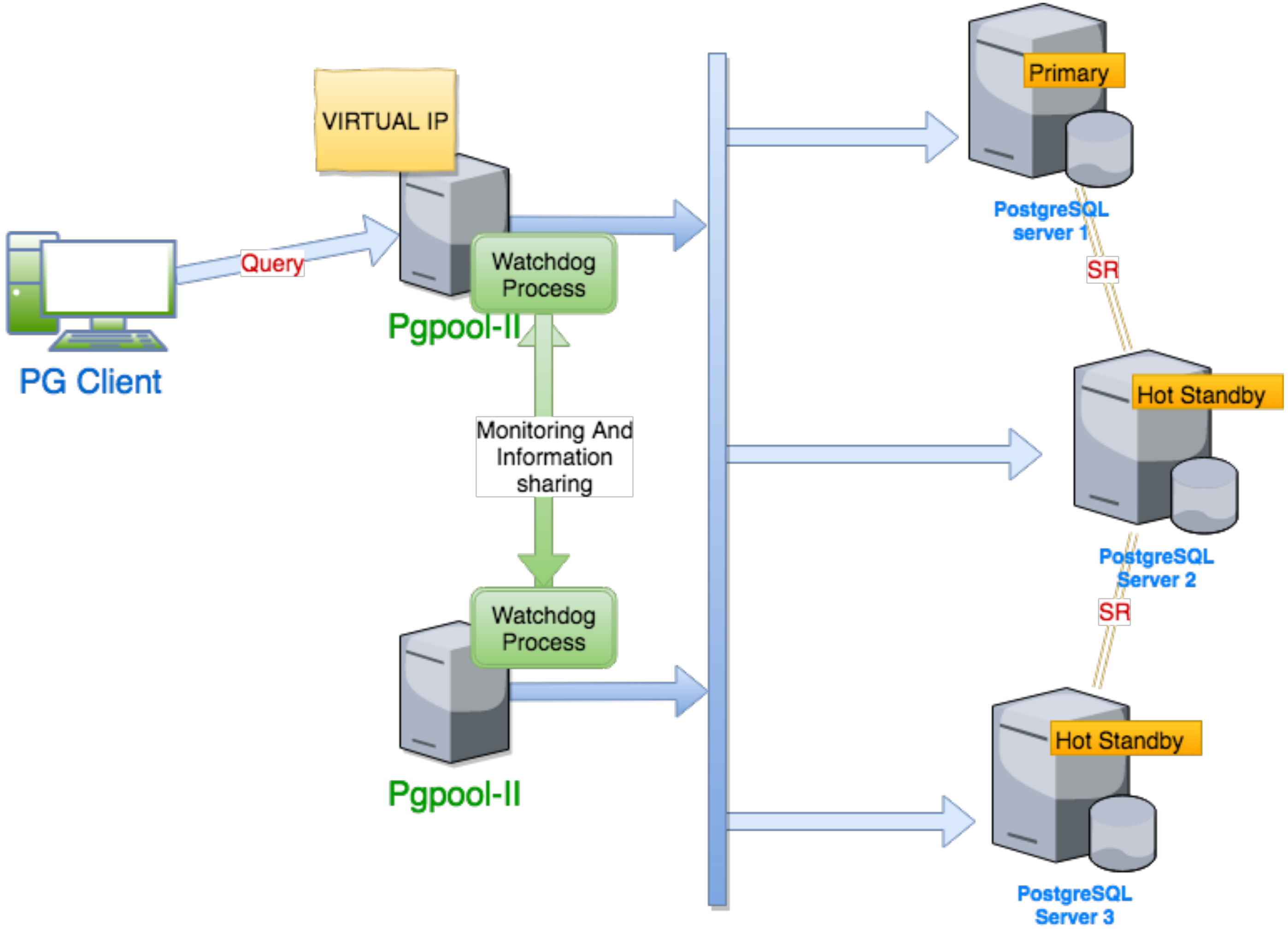
- Pgpool-II becomes the single point of failure (SPOF)

Watchdog

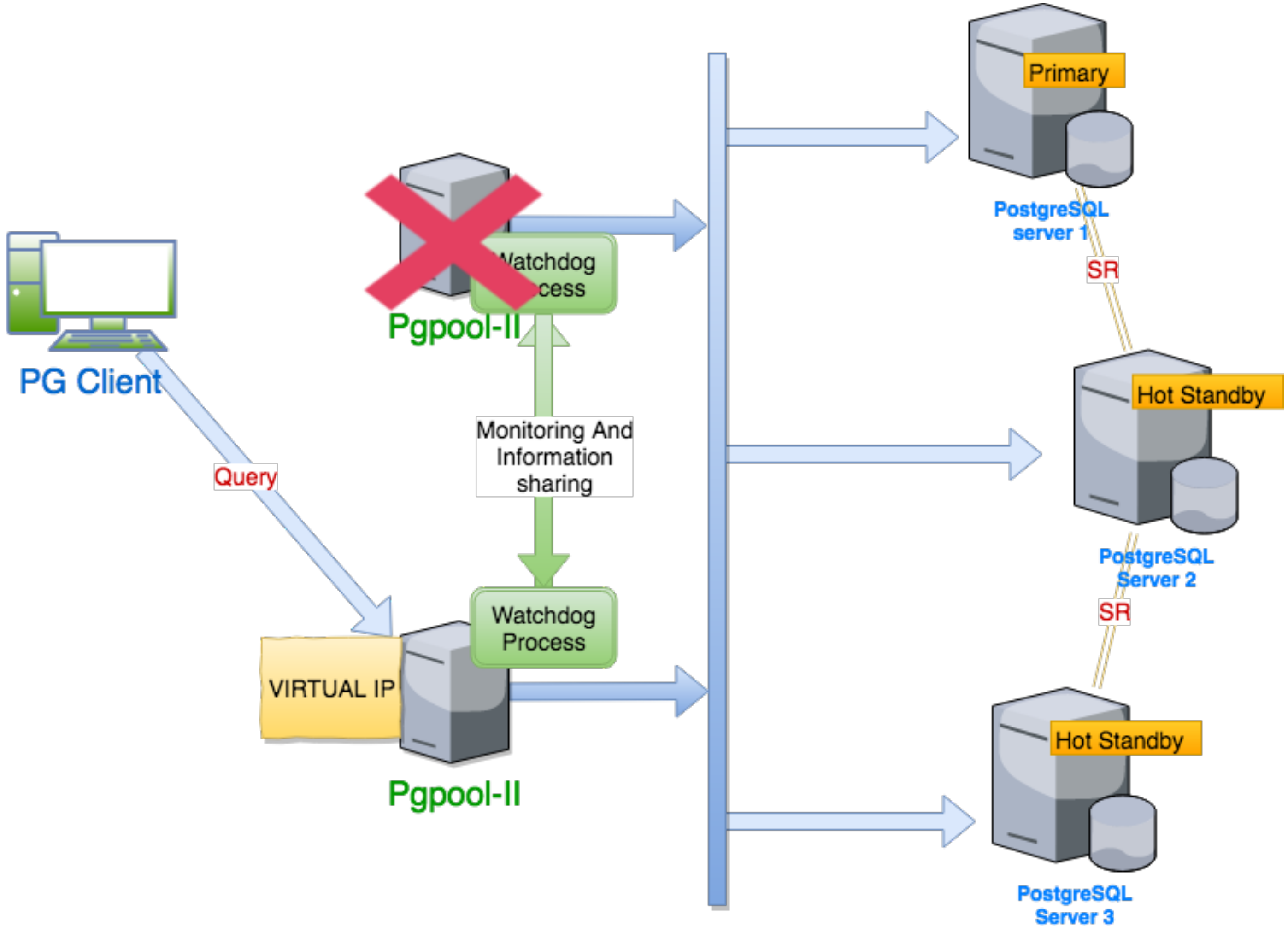
What is Watchdog?

- A sub process of Pgpool-II to handle failures
  - Life checking of Pgpool-II service
  - Mutual monitoring of Pgpool-II nodes in the cluster
  - Leader election to select best master node
  - Virtual-IP control
  - Ensuring same view of PostgreSQL backend states across all Pgpool-II nodes
  - Distributed failover management

Pgpool-II with watchdog



Pgpool-II node failure



Recipe for PostgreSQL HA

Pgpool-II with Watchdog

Whats been happening in Pgpool-II world lately...

Whats been happening in Pgpool-II lately..

- Pgpool-II is becoming more reliable
- Watchdog is getting smarter
- PCP command enhancements
- Failover is getting better
- Performance improvements
- New authentication methods support
- Continuously improving everyday

Watchdog is getting smarter

- Watchdog was very basic until Pgpool-II 3.5
- So many complaints around split-brain syndrome
- Scalability and maintainability were big issues
- Rewritten in Pgpool-II 3.5

No more split-brain syndrome

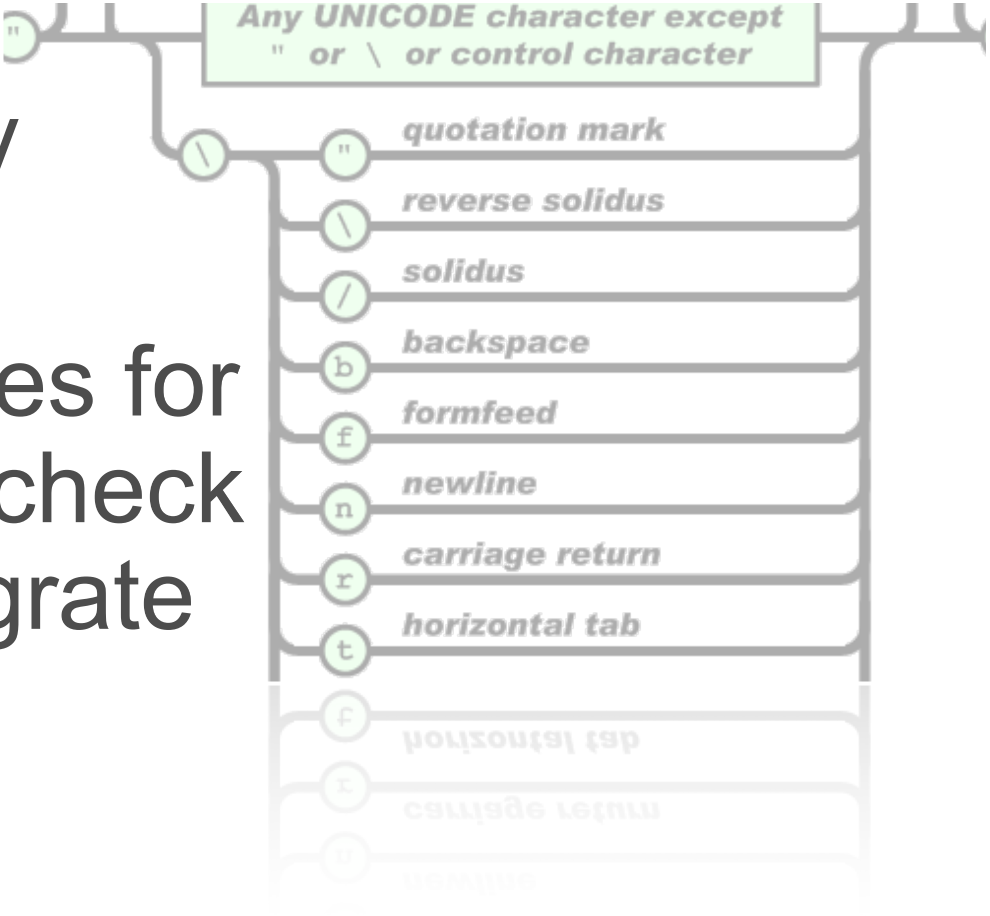
- Ensures the quorum to avoid spilt-brain syndrome during leader election
- Actively look out for split-brain syndrome

Better voting and leader election mechanism

- Watchdog priority (wd_priority config parameter) can be assigned to the Pgpool-II node
- Weighs various node attributes (number of connections, uptime, age, priority)

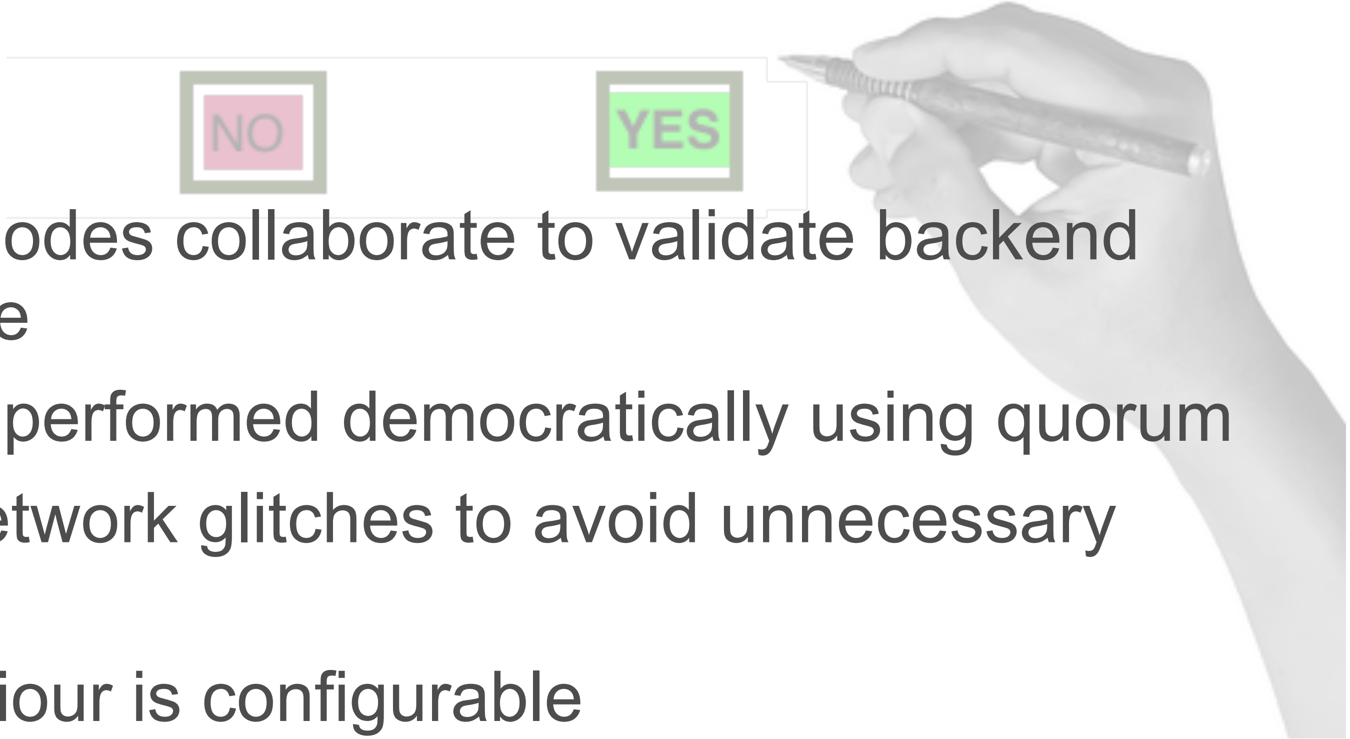
Watchdog now uses sockets and JSON data format for IPC

- Allow third-party integrations
- Expose interfaces for external health check systems to integrate with watchdog
- Extendable and maintainable



© 2018 EDB. All rights reserved.

Watchdog side enhancements in the failover



- Pgpool-II nodes collaborate to validate backend node failure
- Failover is performed democratically using quorum
- Tolerate network glitches to avoid unnecessary failover
- The behaviour is configurable
  - failover_when_quorum_exists
  - failover_require_consensus
  - enable_multiple_failover_requests_from_node

Security and authentication updates in Pgpool-II 4.0

- SCRAM authentication support
- Certificate based authentication support
- Encrypted password file (pool_passwd)
- Encrypted passwords in pgpool.conf

SCRAM authentication

- Supports SCRAM-SHA-256
- Works for both frontend and backend connections
- Works by storing user passwords in pool_passwd file
- Can work with plain text and encrypted passwords in pool_passwd file

Certificate based authentication

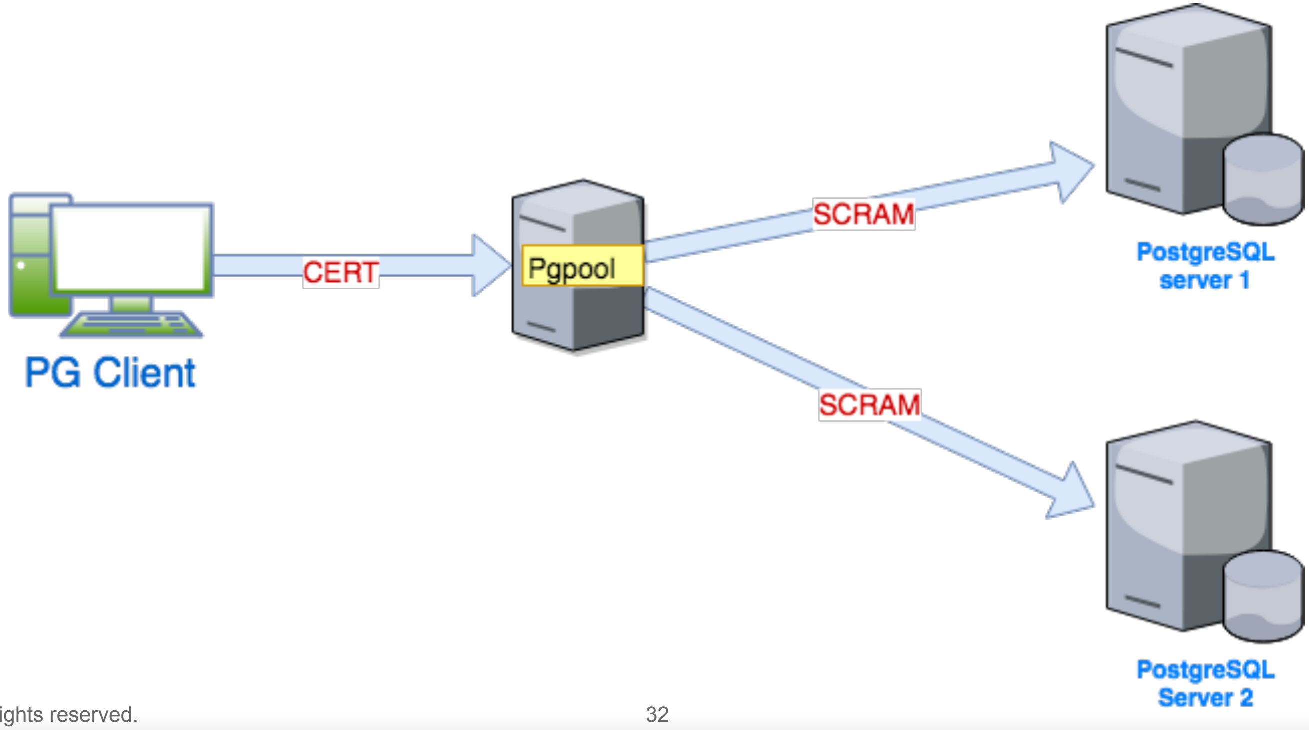
- Available for SSL connections
- Currently works for frontend connections only
- CN (common name) attribute of client certificate compared with database user name. (similar to PostgreSQL)
- Requires same server side certificates used in backends

Secure passwords in Pgpool-II

- pool_passwd and pgpool.conf supports encrypted passwords
- pg_enc utility to create encrypted passwords
- Uses strong AES256 encryption
- Pgpool-II requires pool_key file at startup to decrypt passwords
- Invalid or missing key file makes encrypted password unusable
- One pool_passwd file for all passwords (optional)

More authentication related updates

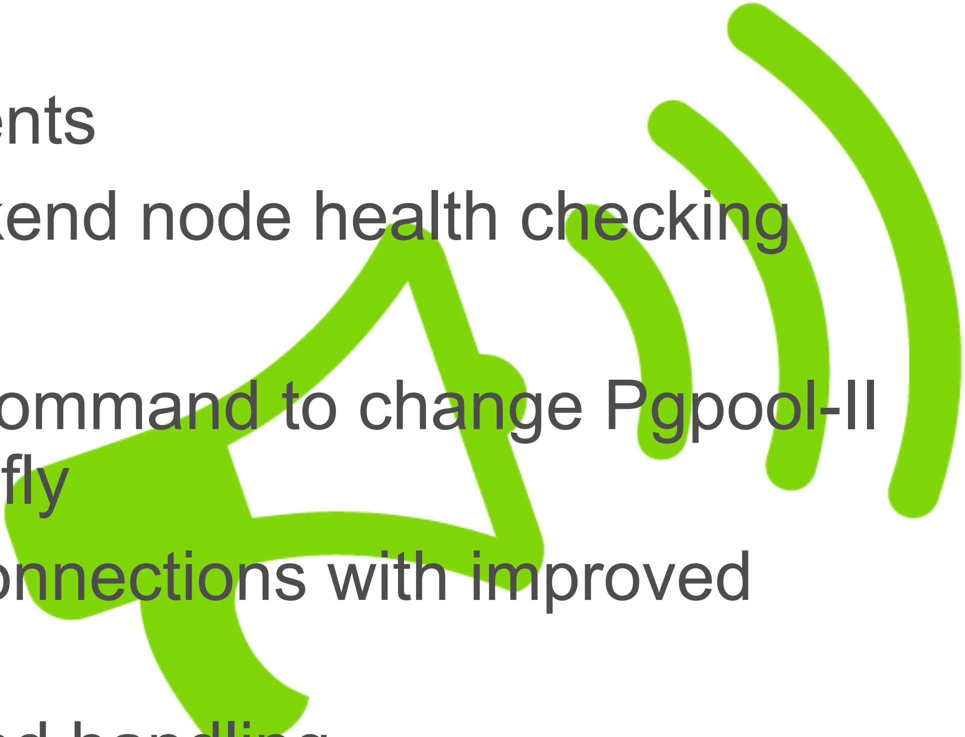
- Backend and frontend connections can now use different authentication methods for same session



Some recent performance updates

- Large SELECT query performance improvement
- Improvements in extended query performance
- Thundering herd problem fix

Some recent notable enhancements in Pgpool-II



- pool_hba enhancements
- Advancement of backend node health checking
- Support AWS Aurora
- New PGPOOL SET command to change Pgpool-II configurations on the fly
- Minimal session disconnections with improved failover mechanism
- Pg_terminate_backend handling
- Allow to specify load balance weights ratio
- Documentation improvement

Thank you for listening

Special thanks to
SRA OSS and Tatsuo Ishii