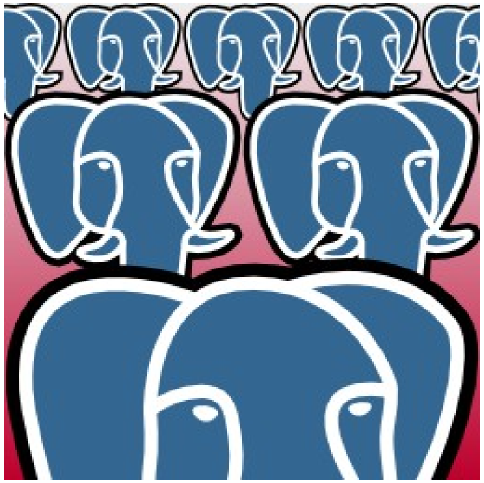


How to manage a herd of elephants:

Introducing new features of pgpool-II

SRA OSS, Inc. Japan
Tatsuo Ishii
石井　達夫

About me

- Came from Tokyo, Japan
- PostgreSQL committer
- Original author of pgpool-II
- Working for SRA OSS for 10 years
- Visiting China 7 times (including this time)
- Love Chinese food!

Copyright(c) 2015 SRA OSS, Inc. Japan

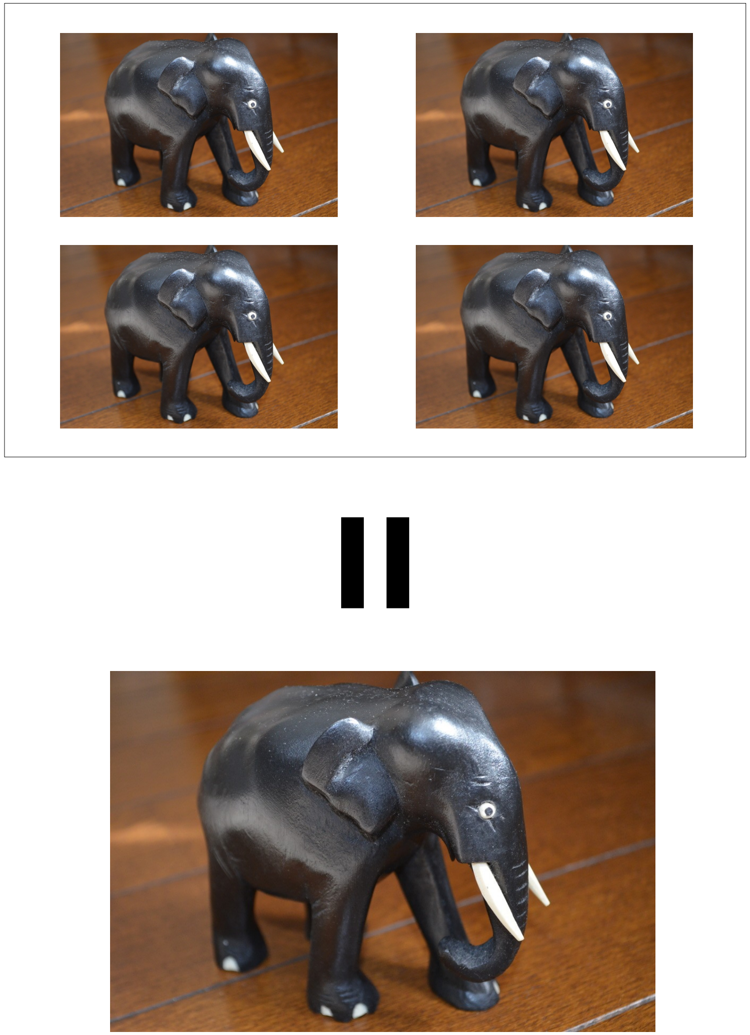
About SRA OSS, Inc. Japan

- One of the oldest PostgreSQL companies in the world
- Doing PostgreSQL and other OSS related business for 10 years
- Main business is support. Has over 600 support contracts
- Contributions to PostgreSQL include:
  - Multi byte support
  - Recursive queries (WITH RECURSIVE)
  - 64 bit large objects

Copyright(c) 2015 SRA OSS, Inc. Japan

A herd of PostgreSQL

- An elephant is powerful
- However, a herd of elephants is even more powerful!
- Problem is, how to manage the herd of elephants?



Copyright(c) 2015 SRA OSS, Inc. Japan

Problems of the herd of elephants include:

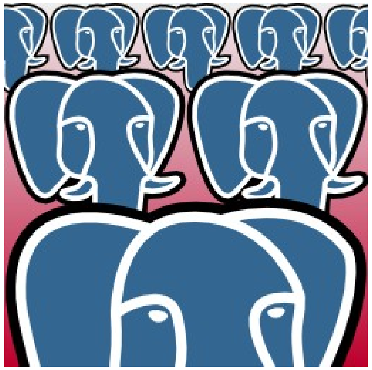
- It needs a leader (a primary server)
- If the leader retires, a new leader must take over its role (fail over)
- Other elephants must follow the new leader
- If a non leader elephant cannot continue to work, it must retire and leave the herd (standby fail over)
- If a new elephant wants to join the herd, it should be accepted without disturbing the herd
- Elephants should help each other to perform a task in an efficient way (load balancing)
- There are tasks which can only be performed by the leader (write queries – needs query dispatching)

Copyright(c) 2015 SRA OSS, Inc. Japan

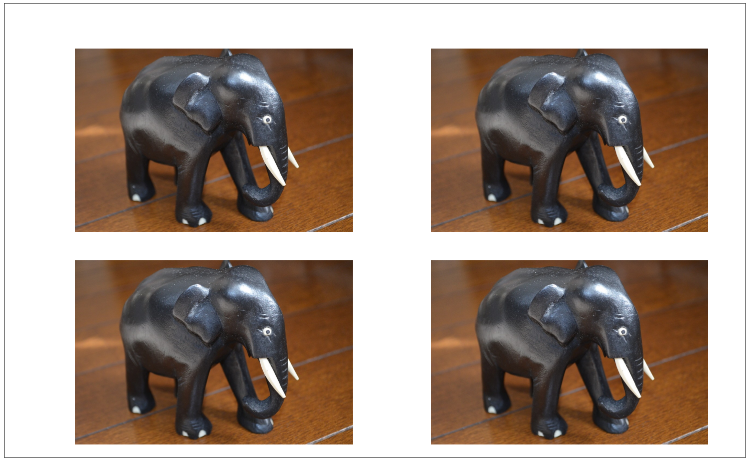
Solving the problems by using pgpool-II

- By using pgpool-II, a streaming replication PostgreSQL cluster almost looks like a single PostgreSQL server
- User supplied “fail over script”could define which standby server should take over when the primary server goes down
- User supplied “follow master command” allow other standbys follow the new primary server
- If a standby server goes down, it is removed from the cluster definition and clients can continue to use the cluster
- pgpool-II examines each query. If the query is a read only one or not. If not, it is forwarded to one of servers (load balancing).
- If a query is a write query, it will be forwarded to the primary server

pgpool-II

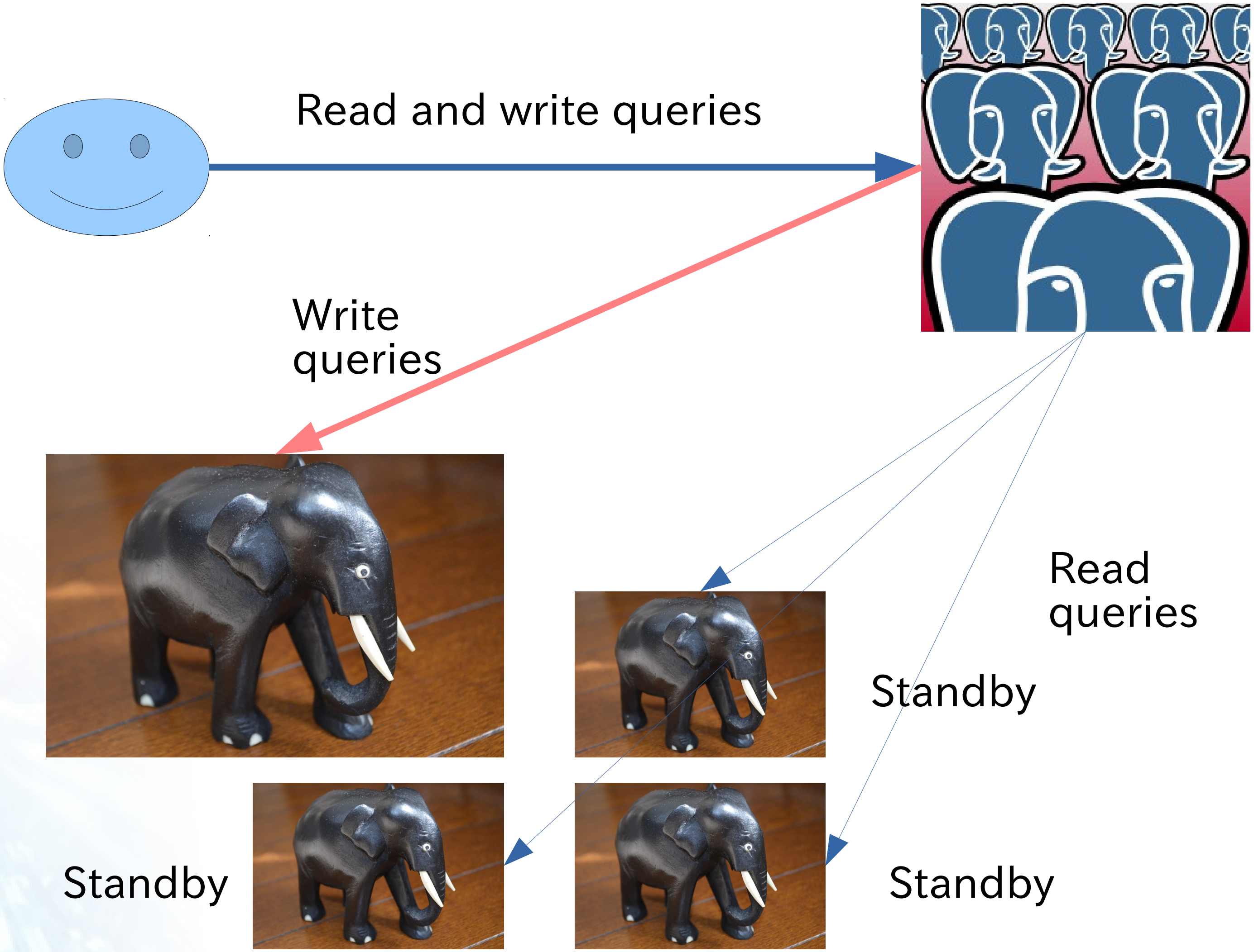


A herd of elephants looks like single big elephant



Copyright(c) 2015 SRA OSS, Inc. Japan

Query dispatching



Read and write queries

Write queries

Read queries

Primary

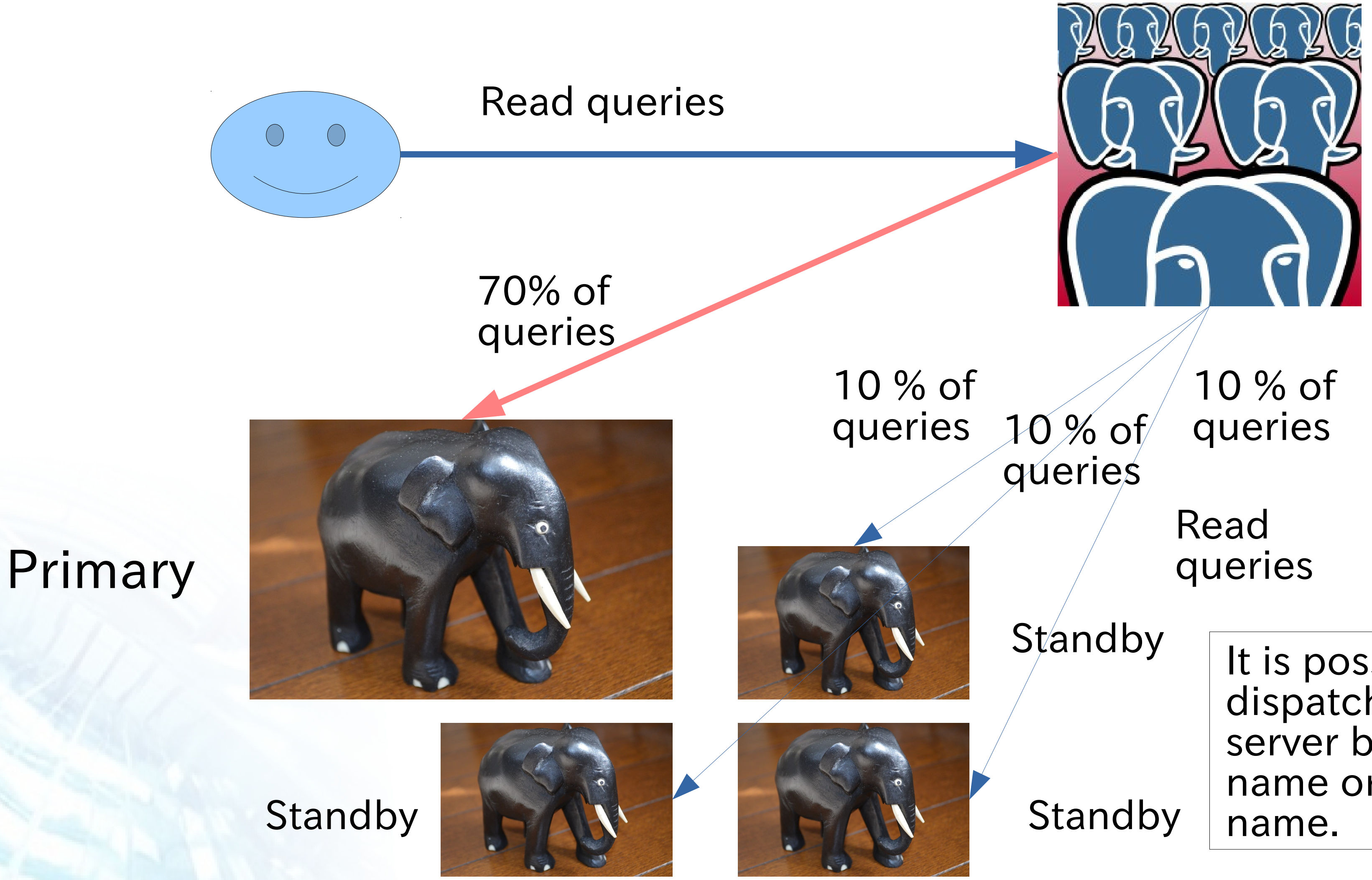
Standby

Standby

Standby

Copyright(c) 2015 SRA OSS, Inc. Japan

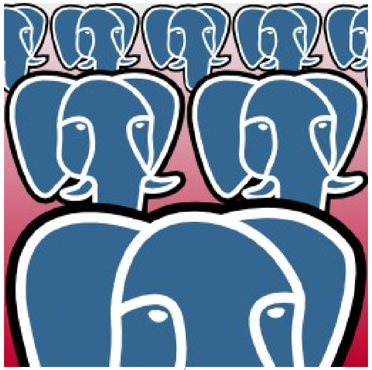
Load balancing

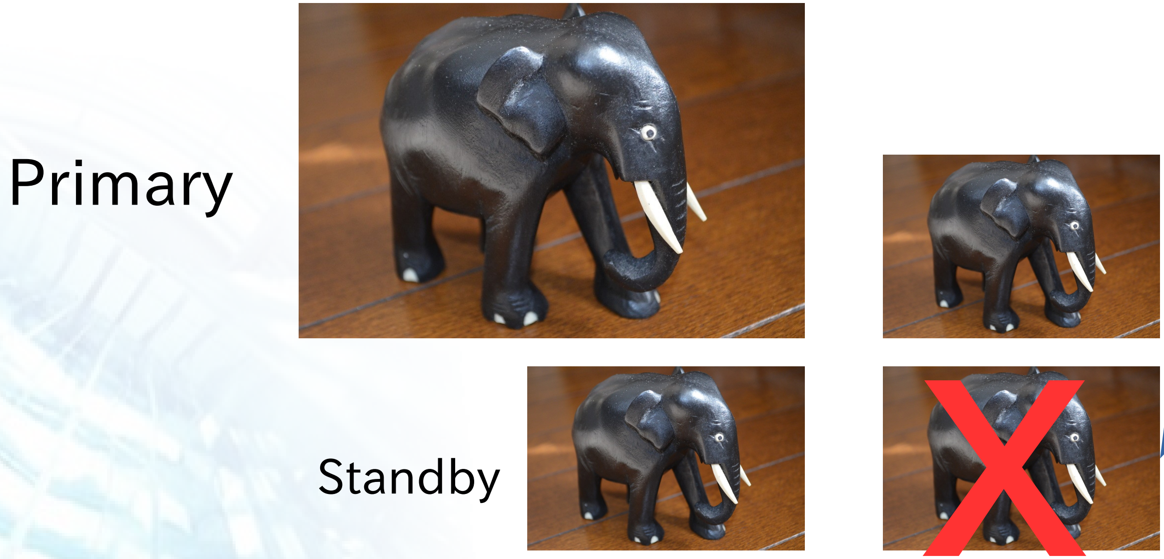


It is possible to dispatch to particluar server by application name or database name.

Copyright(c) 2015 SRA OSS, Inc. Japan

Standby server fails

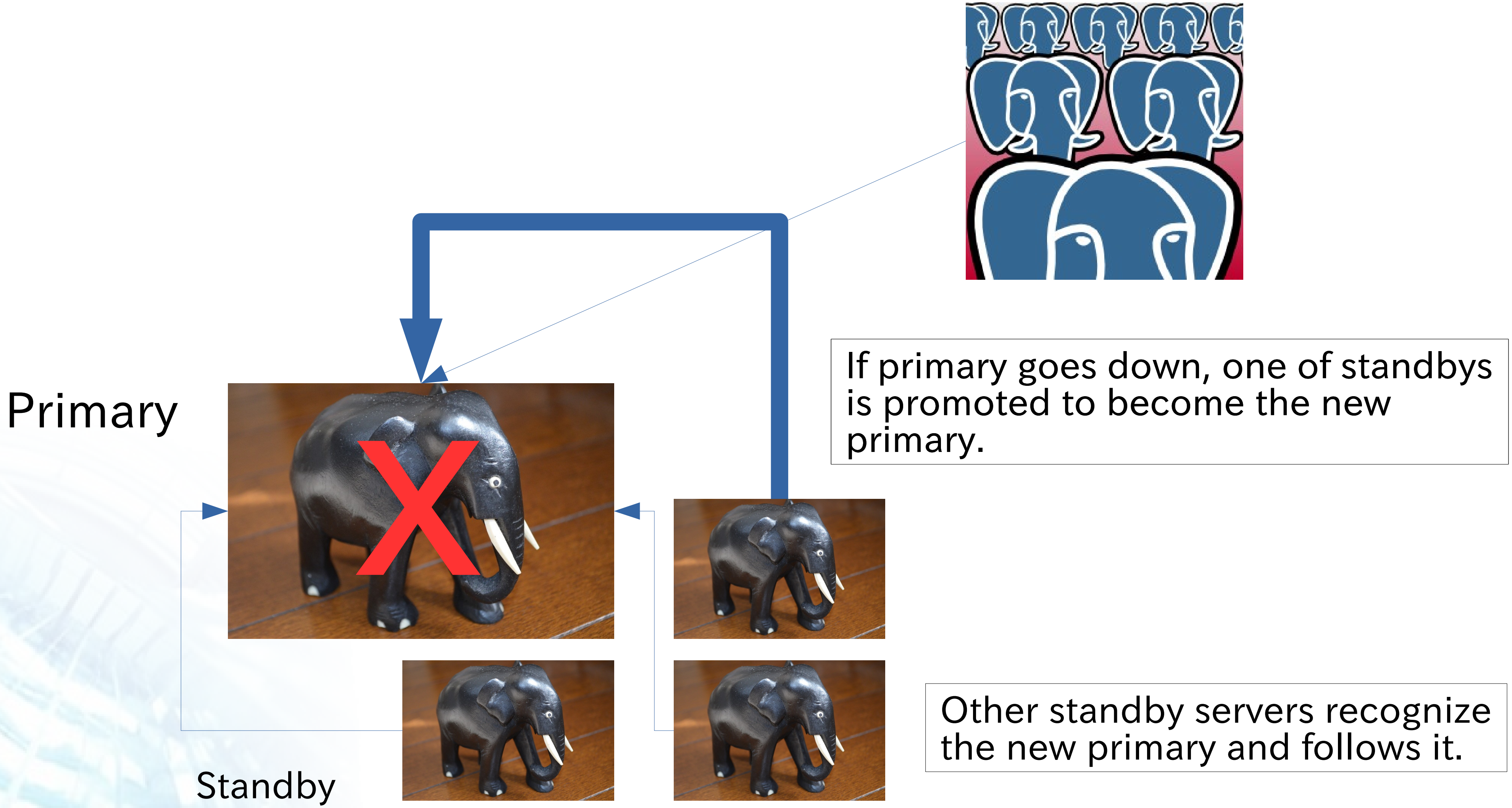




If a standby goes down, it is simply take off from the cluster and user can continue to use the service

Copyright(c) 2015 SRA OSS, Inc. Japan

Primary server fails

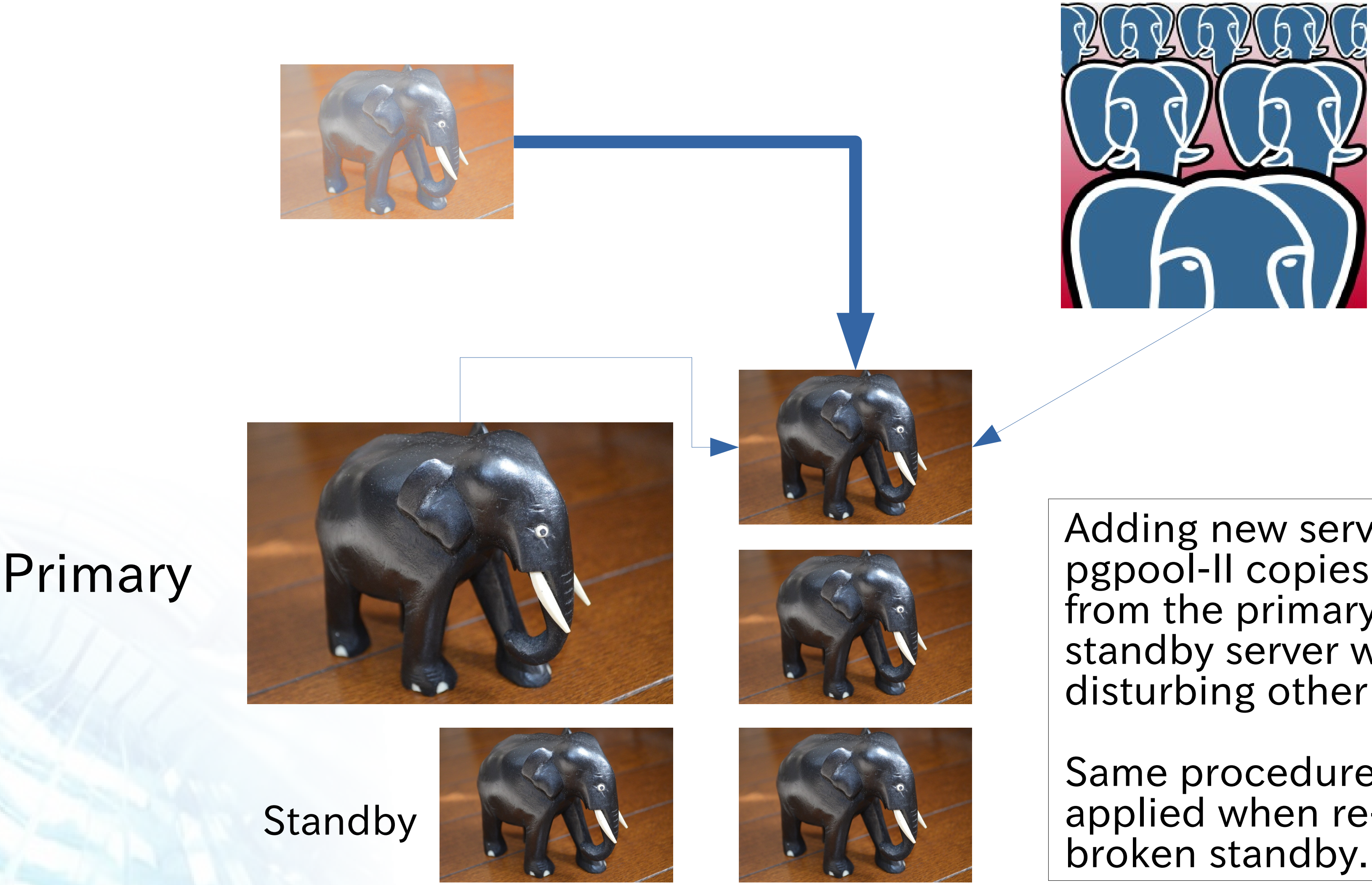


If primary goes down, one of standbys is promoted to become the new primary.

Other standby servers recognize the new primary and follows it.

Copyright(c) 2015 SRA OSS, Inc. Japan

Adding new server



Adding new server is easy. pgpool-II copies the database from the primary to the new standby server without disturbing other servers.

Same procedure can be applied when re-sync broken standby.

Copyright(c) 2015 SRA OSS, Inc. Japan

Upcoming pgpool-II 3.5

- Improving performance
- Improving watchdog
- Importing PostgreSQL 9.5's SQL parser
- Others
- Expected to release on December 15th

Copyright(c) 2015 SRA OSS, Inc. Japan

Improving performance

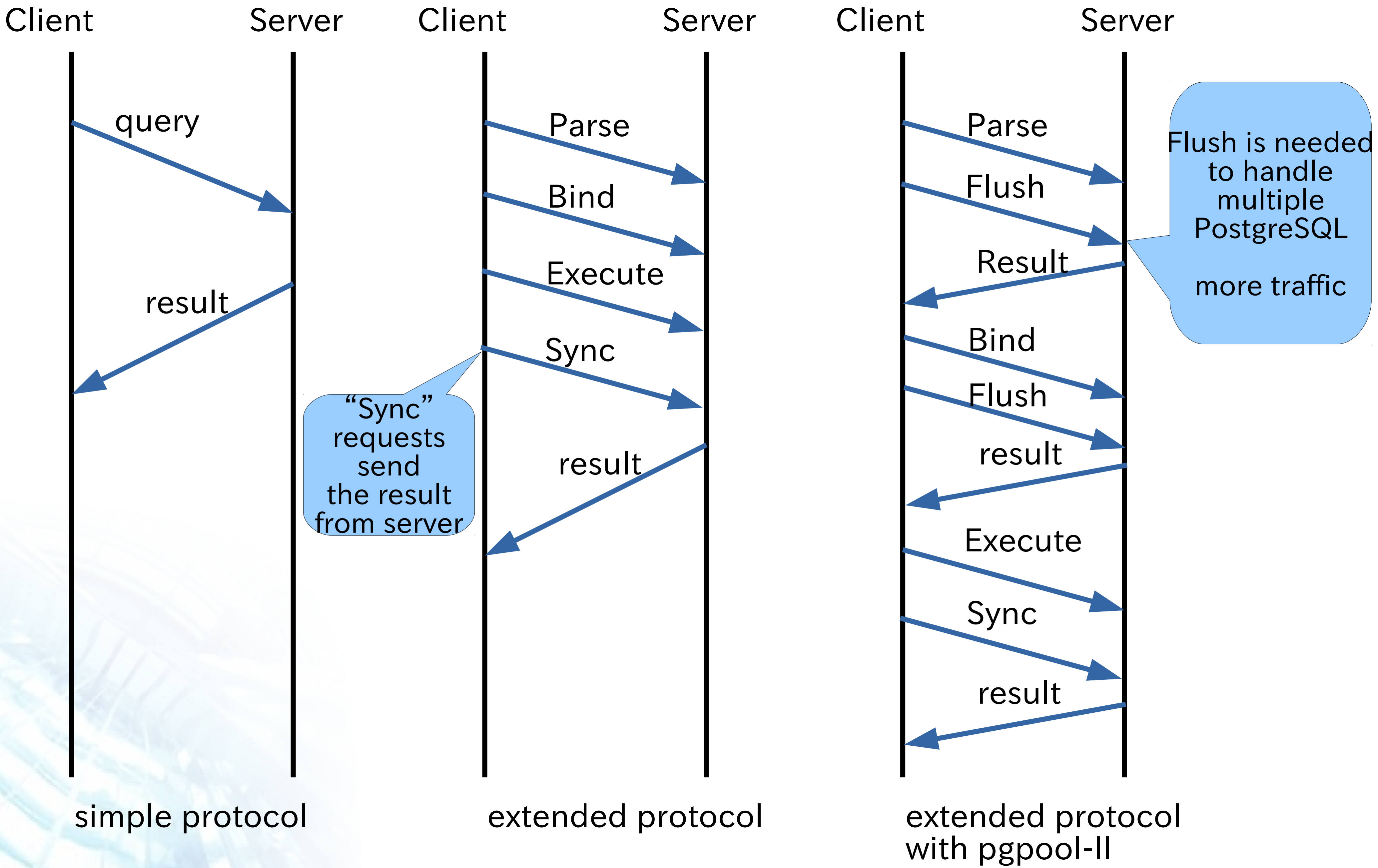
Copyright(c) 2015 SRA OSS, Inc. Japan

Improving extended query perforamance

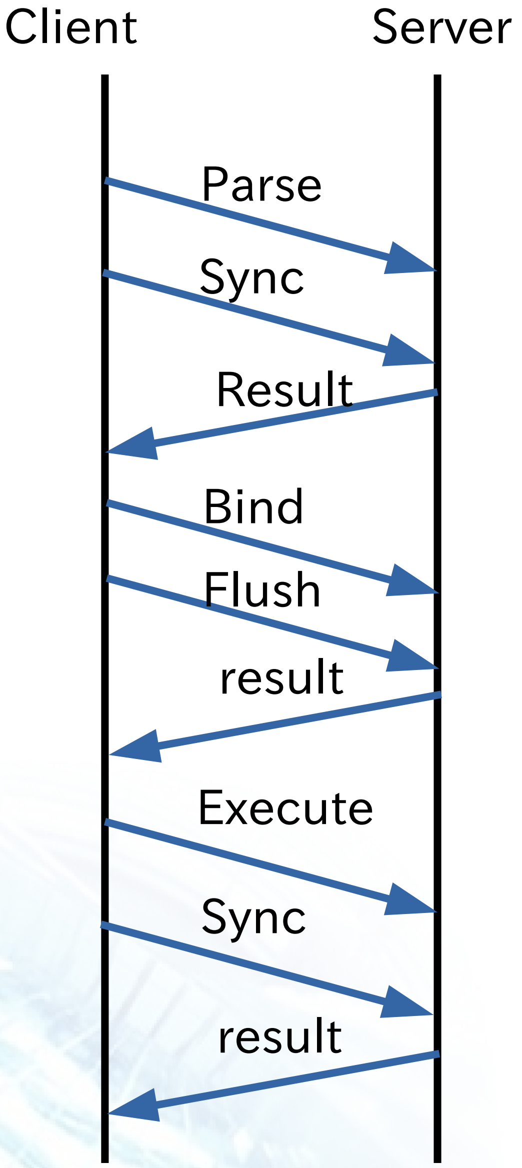
- Using extended protocol (typically used in Java) in pgpool-II is slow (as slow as half of simple protocol)
- Current implementation of pgpool-II for extended protocol is not so great: it requires additional flush messages, and this is the source of poor performance
- First, need to understand: how extended protocol is handled?

Copyright(c) 2015 SRA OSS, Inc. Japan

Some details are omitted



Copyright(c) 2015 SRA OSS, Inc. Japan

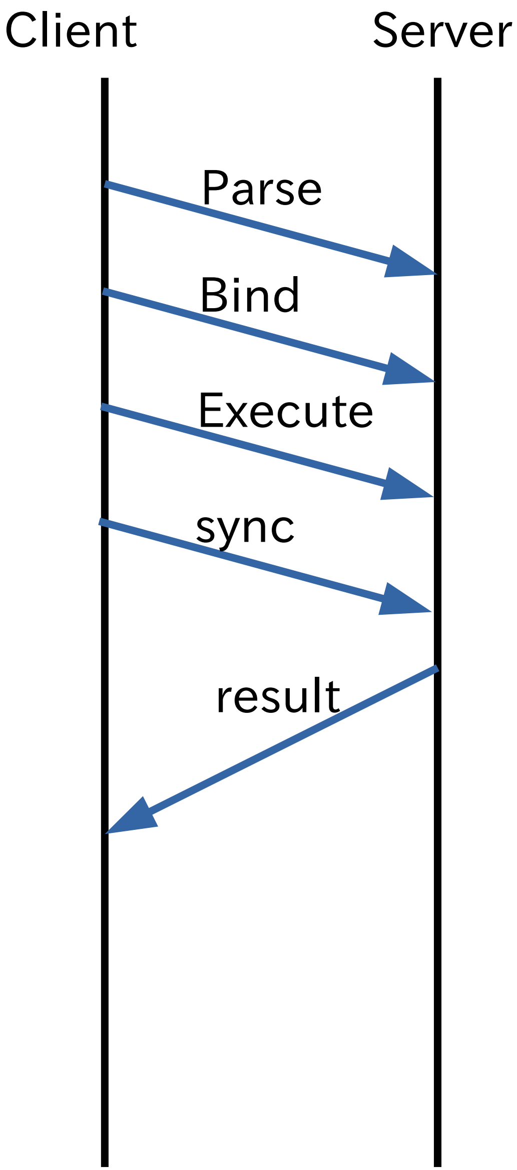


extended protocol
with pgpool-II

Too many Flush

In streaming replication
we could omit some of
Flush messages

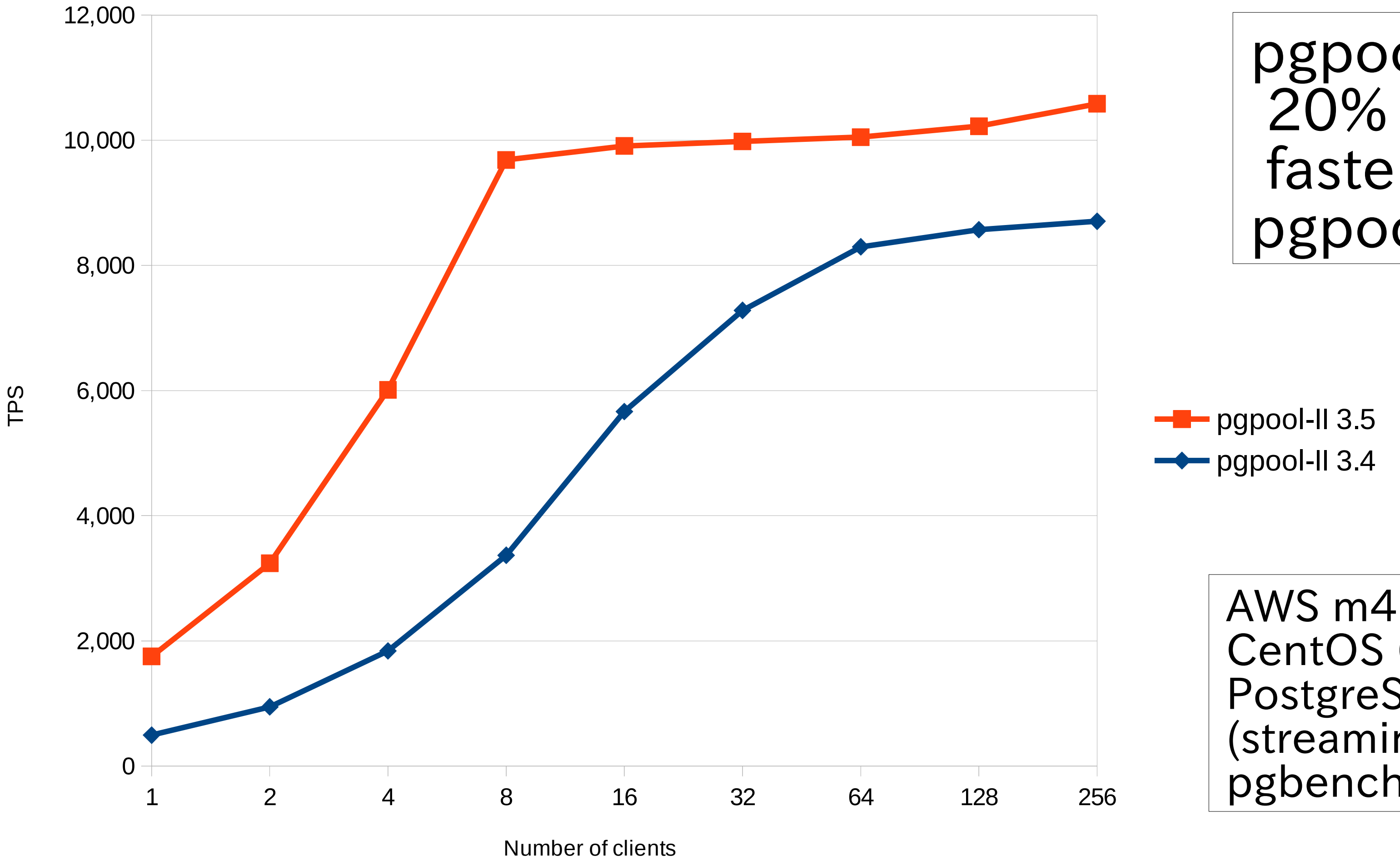
enhanced!



extended protocol
with pgpool-II in 3.5

Copyright(c) 2015 SRA OSS, Inc. Japan

Benchmarking Extended protocol query performance



pgpool-II 3.5 is 20% to 250% faster than pgpool-II 3.4!

AWS m4.large instance
CentOS 6
PostgreSQL 9.4 x2
(streaming replication)
pgbench -S

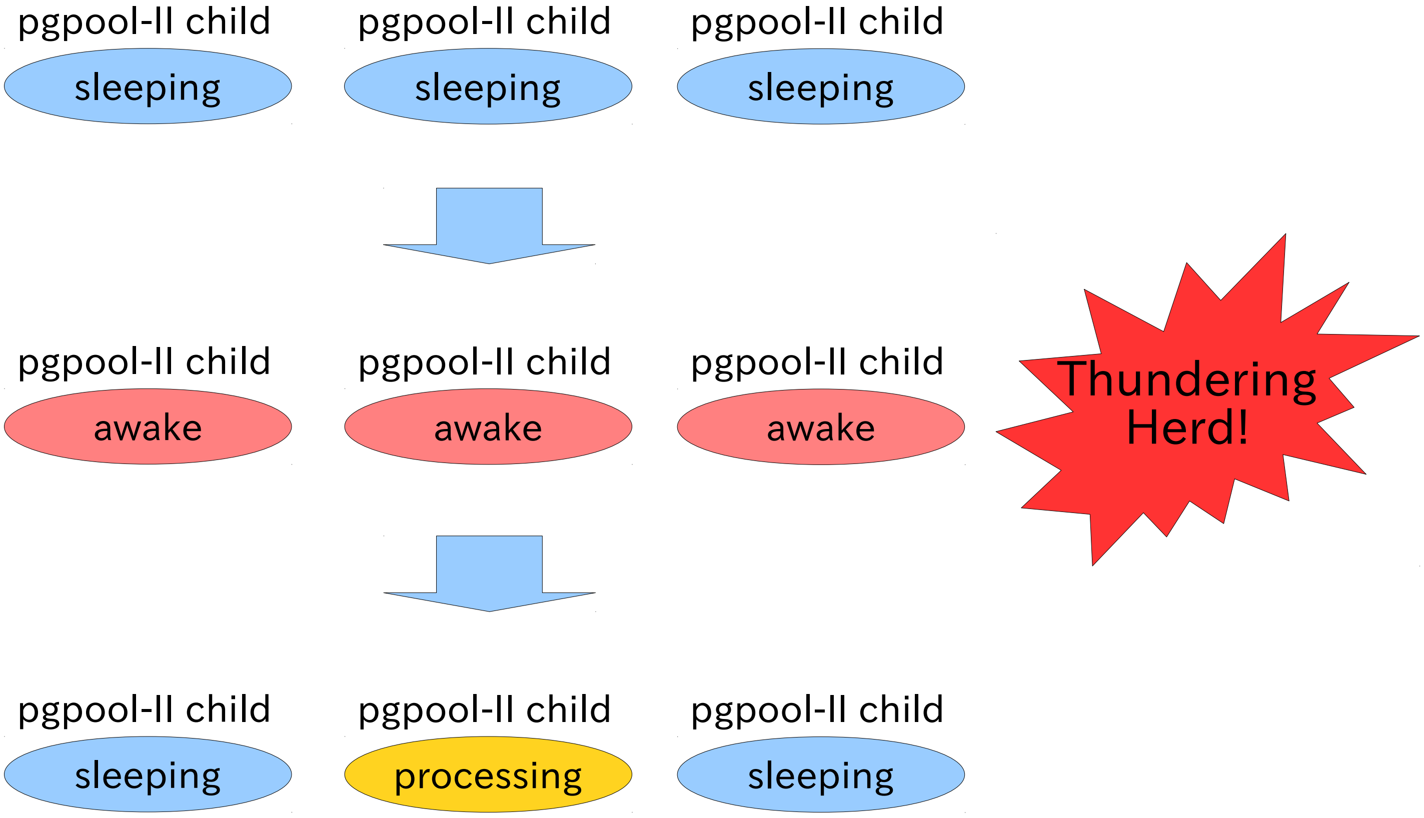
Copyright(c) 2015 SRA OSS, Inc. Japan

Overcoming Thundering herd problem

- What is “the thundering herd problem”?
  - > “The thundering herd problem occurs when a large number of processes waiting for an event are awoken when that event occurs, but only one process is able to proceed at a time. After the processes wake up, they all demand the resource and a decision must be made as to which process can continue. After the decision is made, the remaining processes are put back to sleep, only to all wake up again to request access to the resource.”
    >
    > From Wikipedia
- pgpool-II forks off many child process and they are waiting for connection requests from clients
- If a connection request arrives, all of the child process are awoken but only one of them is allowed to accept the request
- Other child process start sleeping again, to wait for next connection request
- This leads to an excessive context switching and results in poor performance

Overcoming Thundering herd problem

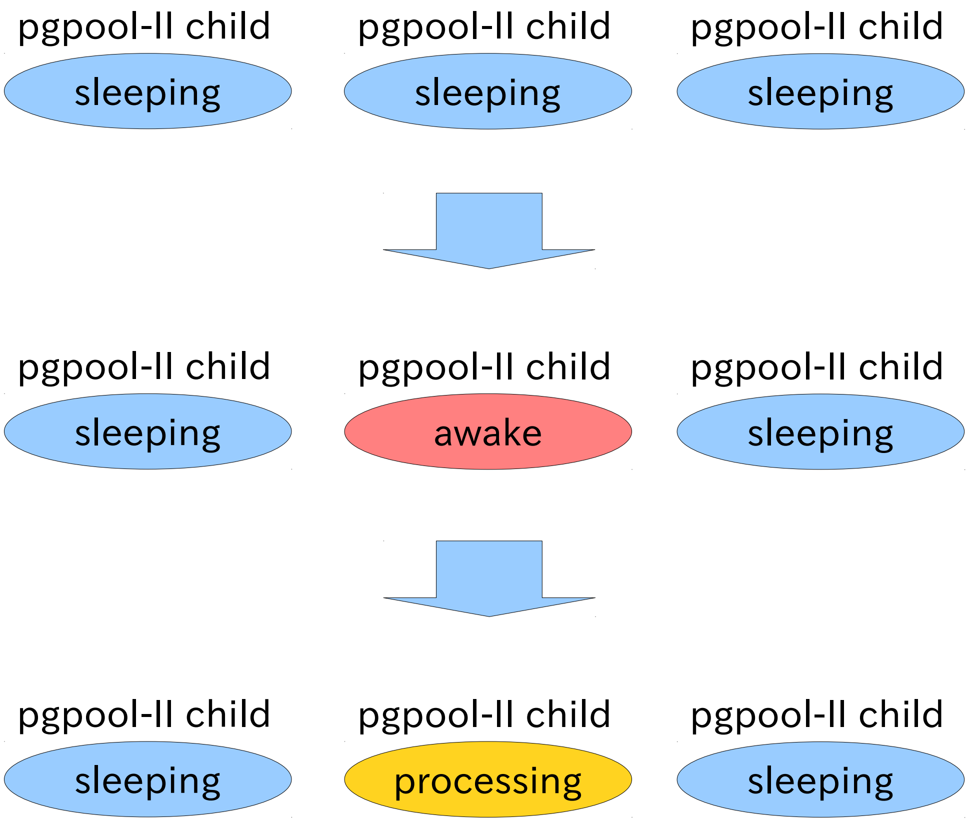
pgpool-II 3.4

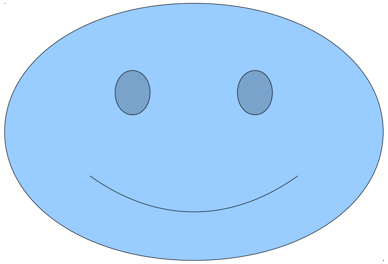


Copyright(c) 2015 SRA OSS, Inc. Japan

Overcoming Thundering herd problem

pgpool-II 3.5

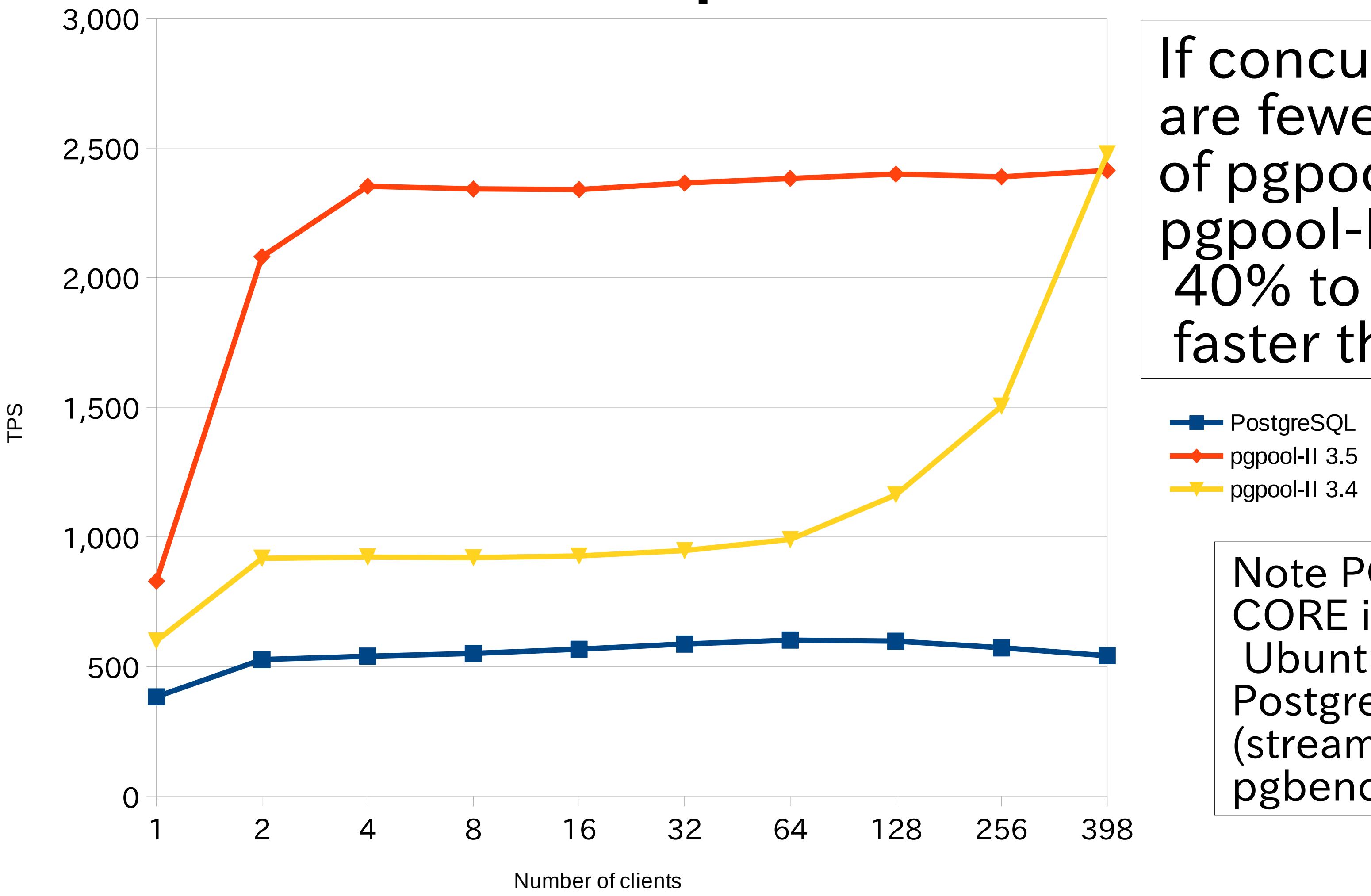




No thundering
Herd problem

Copyright(c) 2015 SRA OSS, Inc. Japan

Overcoming Thundering herd problem



If concurrent clients are fewer than number of pgpool child process, pgpool-II 3.5 is 40% to 150% faster than pgpool-II 3.4

Note PC with 16GB Mem, CORE i7 x2, 512GB SSD
Ubuntu 14.04
PostgreSQL 9.4 x2
(streaming replication)
pgbench -S -C -T 300

Copyright(c) 2015 SRA OSS, Inc. Japan

Settings to avoid the thundering herd problem in pgpool-II 3.5

- set “serialize_accept” to on
- set “child_life_time” to 0
- If concurrent connections are roughly equal to num_init_children, this function does not do the best (see previous slide)

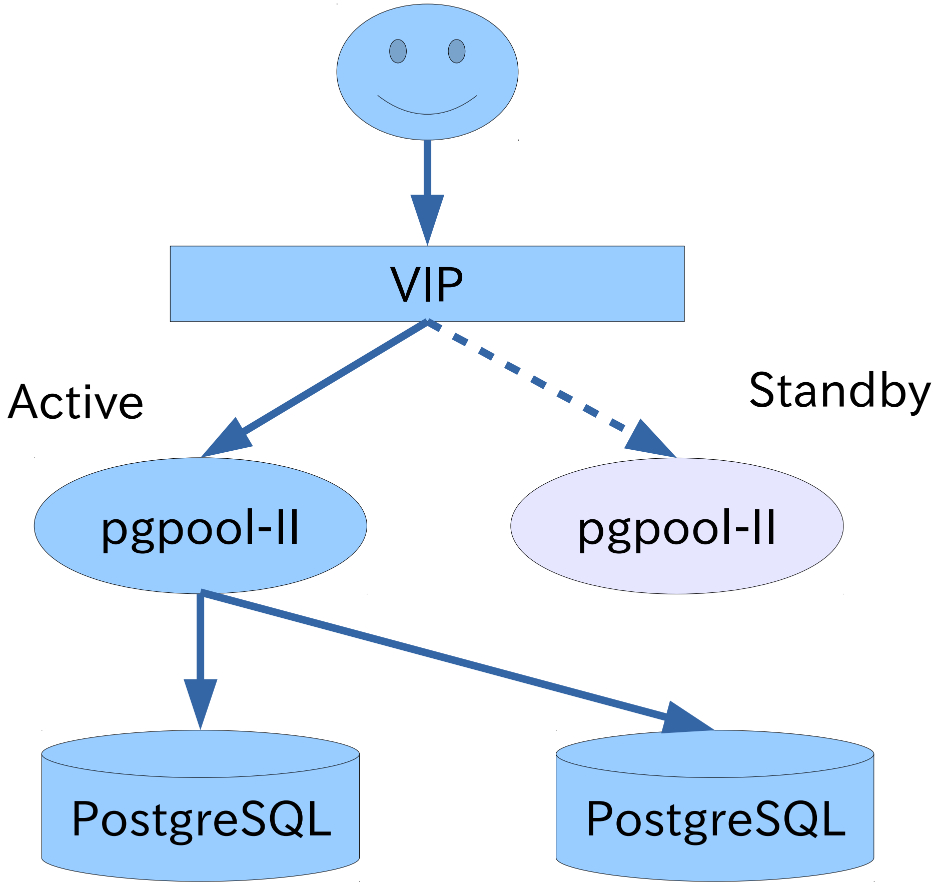
Copyright(c) 2015 SRA OSS, Inc. Japan

Improving watchdog

Copyright(c) 2015 SRA OSS, Inc. Japan

What is “Watchdog”?

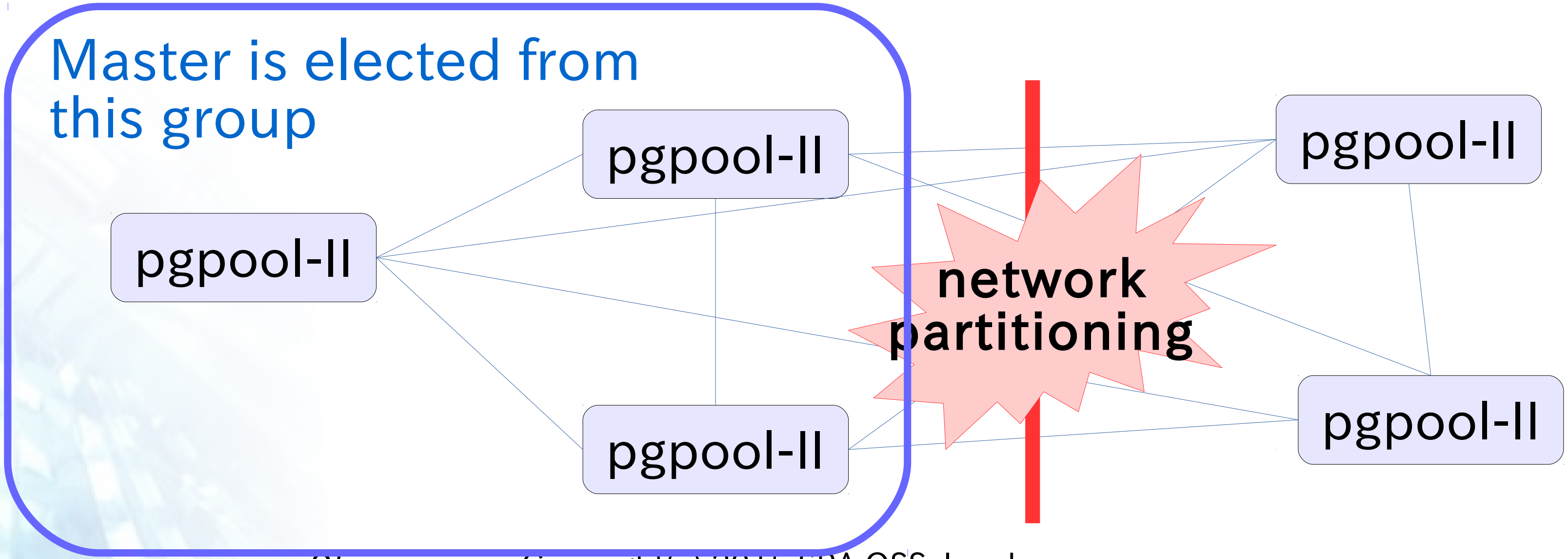
- Because pgpool-II works as a proxy, pgpool-II could be Single point of failure (SPOF)
- “Watchdog” is a built-in High Availability (HA) feature of pgpool-II
- Two or more pgpool-II instances monitors each other. If “Active” pgpool-II goes down, “Standby” pgpool-II takes over and becomes new active pgpool-II
- Active pgpool-II holds Virtual IP (VIP). Client connects to the VIP and are not worried about which is pgpool-II is alive



Copyright(c) 2015 SRA OSS, Inc. Japan

New Watchdog Enhances Robustness against Split-brain

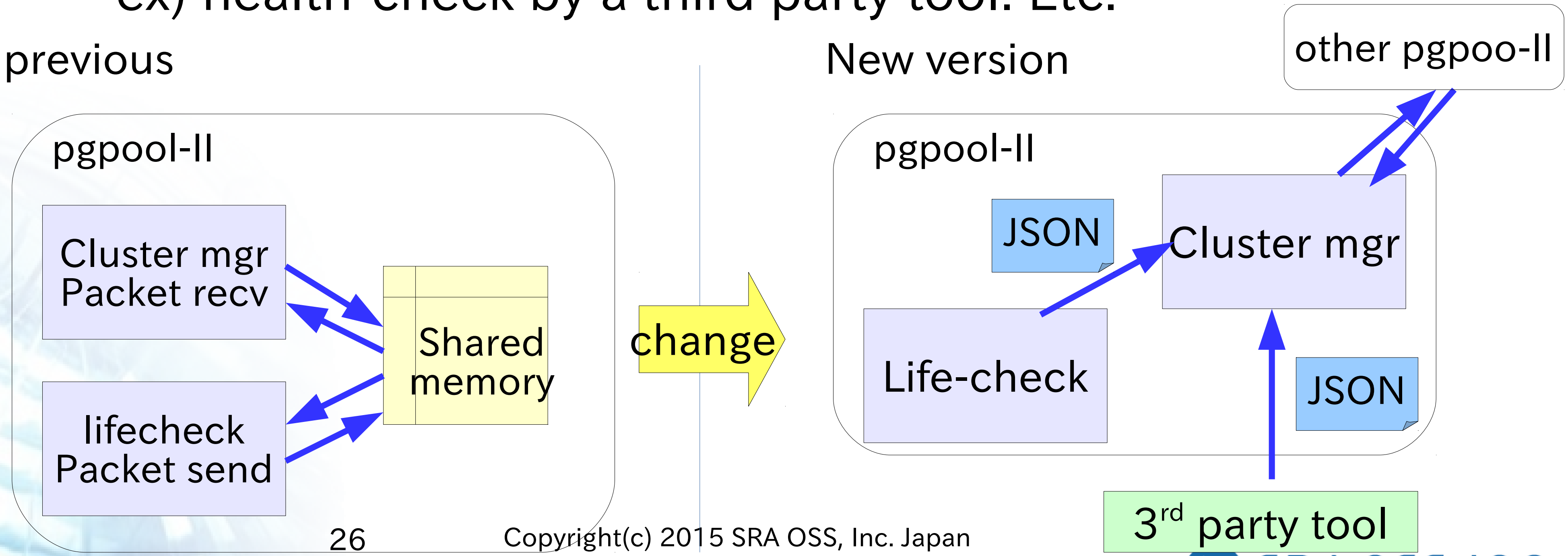
- Split-brain syndrome
  - It can't be decided which pgpool-II should be elected as the master when the network is participated
  - Quorum support
    - Check if more than half of nodes are belong to the group which local pgpool-II is belonging to
    - Number of pgpool-II nodes must be odd to make quorum working (in other case you can use “trusted_servers”)



Copyright(c) 2015 SRA OSS, Inc. Japan

New Watchdog enhances inter-process communication

- Change in inter-process communication method with in watchdog
  - UNIX domain socket & JSON format data
- This allows pgpool-II to work with third-party tools
  - ex) health-check by a third party tool. Etc.



Copyright(c) 2015 SRA OSS, Inc. Japan

Watchdog enhancement others

- Verifies the consistency of important configuration parameters among all nodes
  - make sure they are consistent among all nodes
  - help to eliminate problems caused by ll-configured pgpool-II nodes

- Watchdog nodes can have priorities
  - Watchdog nodes can be assigned with different priorities
  - Nodes with higher watchdog will get a preference when the cluster is electing its new leader node.

Copyright(c) 2015 SRA OSS, Inc. Japan

Importing PostgreSQL 9.5's parser

Copyright(c) 2015 SRA OSS, Inc. Japan

Importing PostgreSQL 9.5 parser

- Parser of PostgreSQL 9.5 is ported into pgpool-II 3.5
  - Previous parser was imported from PostgreSQL 9.4
- Load-balancing and query-cache supports the new select syntax.
  - GROUPING SET, CUBE, ROLLUP,
  - TABLESAMPLE
- Query-rewriting in the native replication mode supports the new insert/update syntax.
  - INSERT … ON CONFLICT
  - UPDATE tab SET (col1,col2,...) = (SELECT ...), ...

Copyright(c) 2015 SRA OSS, Inc. Japan

Improvements in pgpool-II 3.5: others

Copyright(c) 2015 SRA OSS, Inc. Japan

Health check and replication delay checking target database

- In some systems (for example, Heroku) do not allow to connect to “postgres” or “template1” database
- pgpool-II issues query to those databases for health check and streaming replication delay
- pgpool-II allows to use particular database instead of the databases
  - health_check_database
  - sr_check_database

Copyright(c) 2015 SRA OSS, Inc. Japan

Showing SELECT count

- “show pool_nodes” command now shows the number of SELECTs issued to each DB node

New!

```
test=# show pool_nodes;
 node_id | hostname | port  | status | lb_weight |  role   | select_cnt
---------+----------+-------+--------+-----------+---------+------------
 0       | /tmp     | 11002 | 2      | 0.500000  | primary | 338230
 1       | /tmp     | 11003 | 2      | 0.500000  | standby | 163939
(2 rows)
```

Copyright(c) 2015 SRA OSS, Inc. Japan

Availability of pgpool-II 3.5

- We expect to release pgpool-II 3.5 on December 15th
- pgpool-II 3.5 alpha1 was released this week
- Stay tuned!

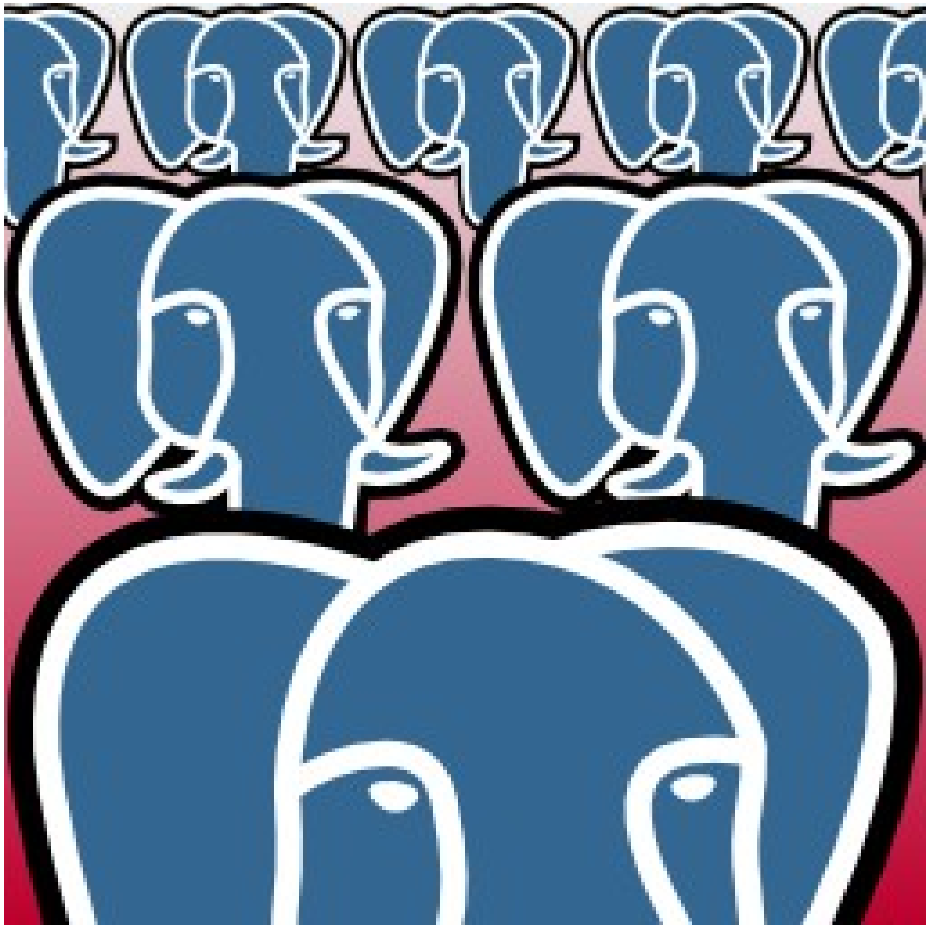
Coming soon!

URLs

- pgpool-II official site
  - http://www.pgpool.net
- SRA OSS
  - http://www.sraoss.co.jp

Copyright(c) 2015 SRA OSS, Inc. Japan

Thank you!



Copyright(c) 2015 SRA OSS, Inc. Japan