2ndQuadrant
Professional PostgreSQL

HA with repmgr and pgbouncer

Jaime Casanova

Let me introduce myself

- PostgreSQL contributor
- Reviewer
- ecpug founder (@ecpug, ecpug@postgresql.org)
- Spanish community support (pgsql-es-ayuda@postgresql.org)

- 2ndQuadrant representative in Ecuador
- 2ndQuadrant Principal Consultor
- repmgr developer

The Customer

The “Corporación Nacional de Telecomunicaciones, CNT EP” is the public telecommunications company in Ecuador that offers, in Ecuadorian territory:

- fixed telephony services local, regional and international
- Internet Access (Dial-Up, DSL, mobile Internet)
- satellite television
- mobile telephony

PostgreSQL on CNT

- The “Panic Button”
  - PostGIS enabled database
  - geolocation and georeference
  - track patrols

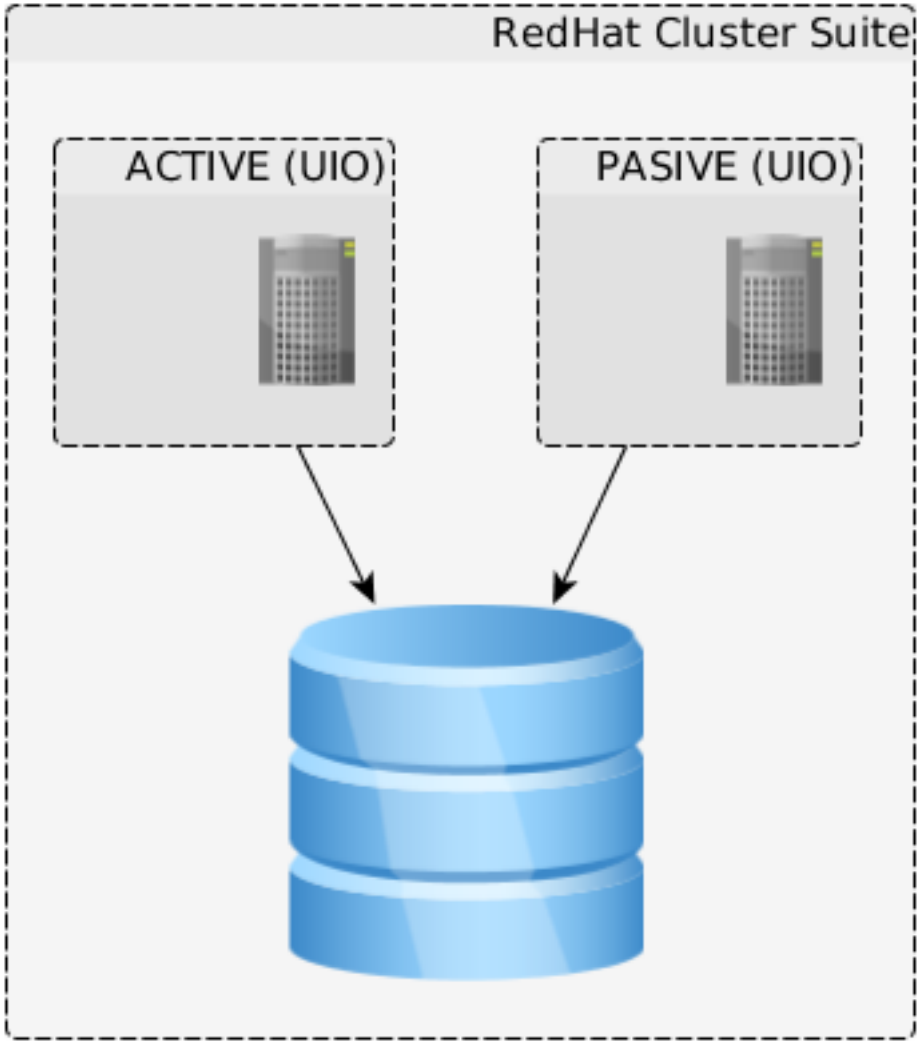
```
A specialized service given to national Police
```

PostgreSQL on CNT

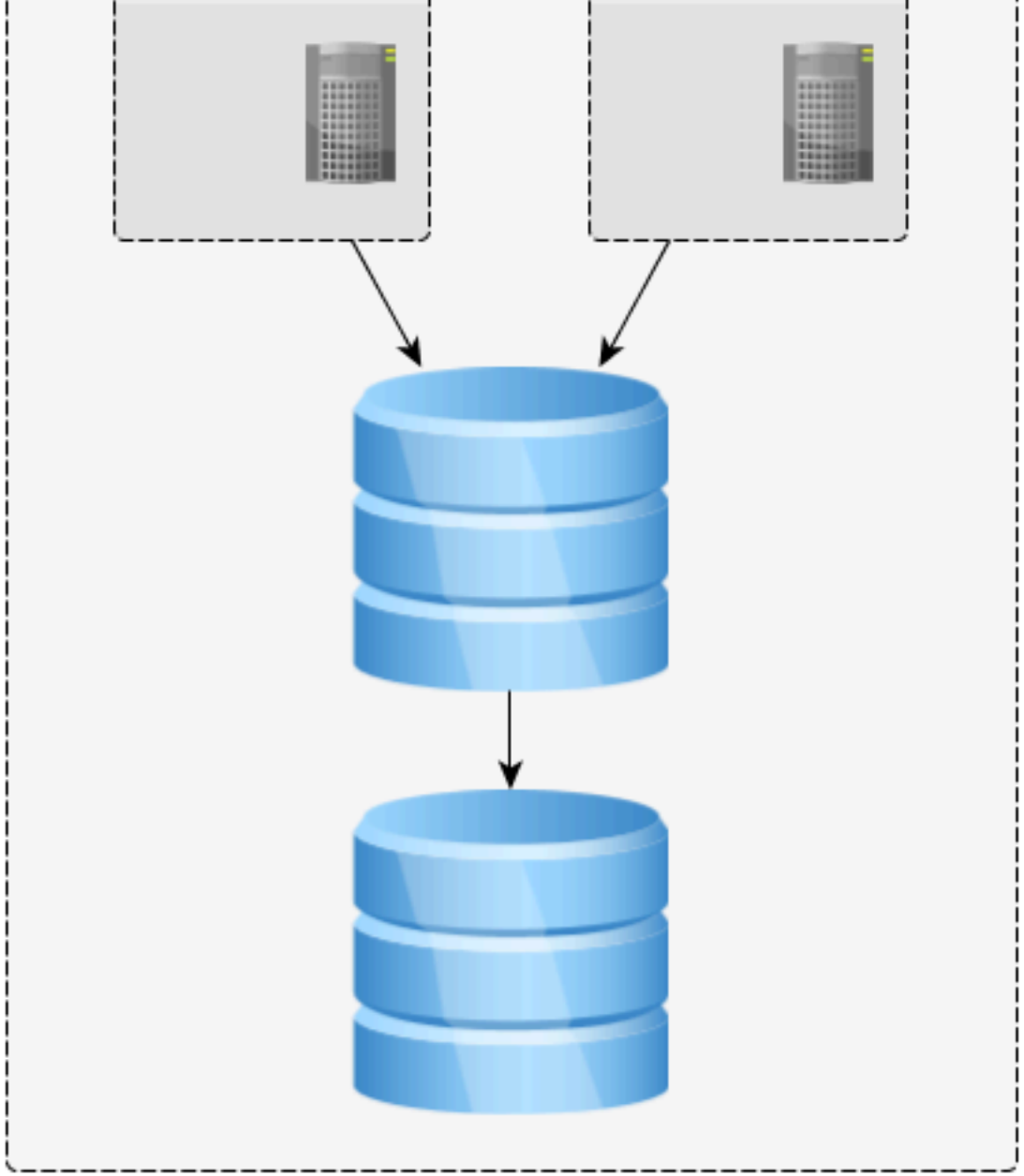
- “Panic Button” for national Police *(CRITICAL)*
- to improve some core business processes

- new services are being implemented on this platform

CNT first approach fo



HA requires redu

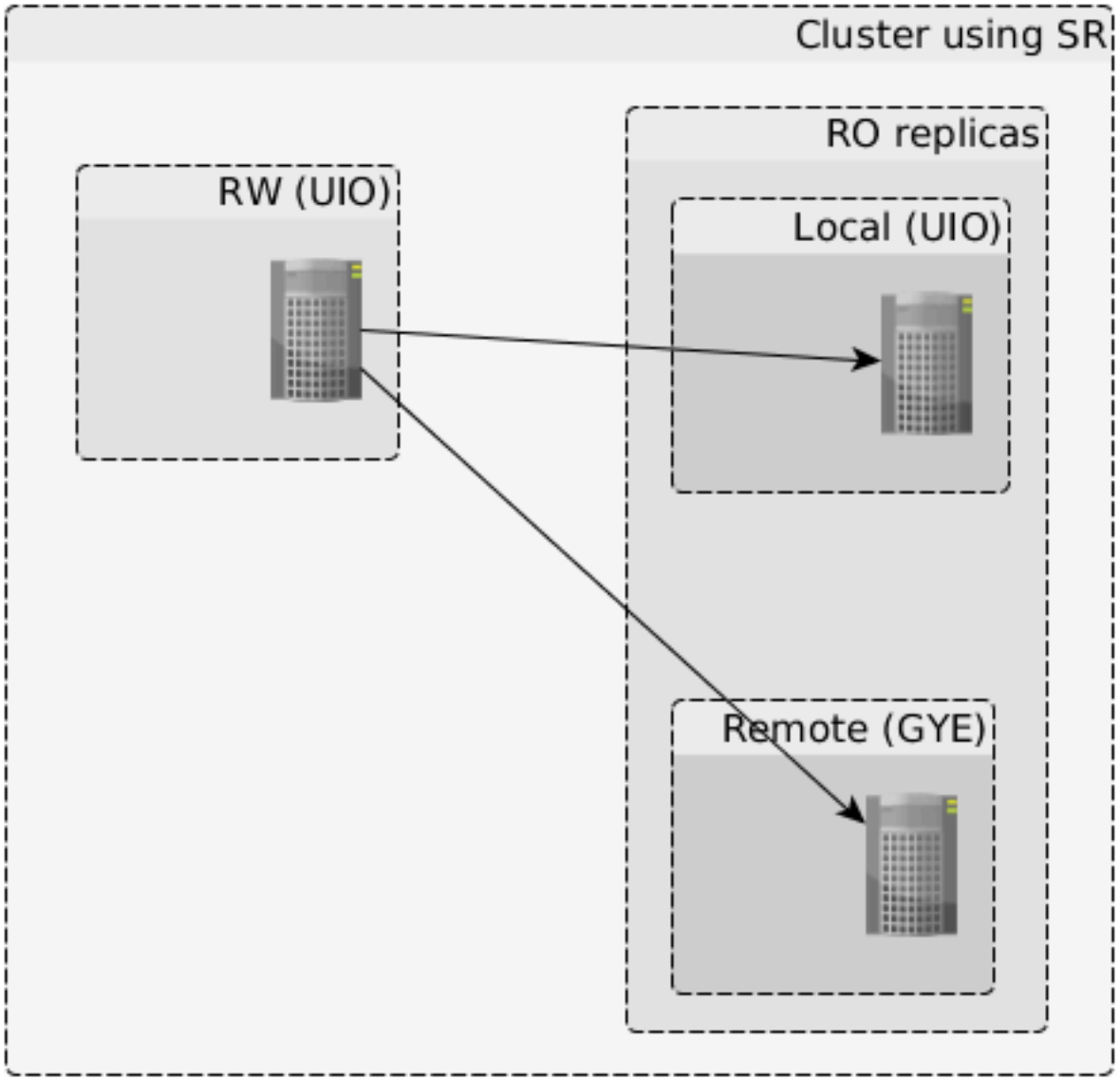


HA requires redundancy

It’s not enough to have redundant data, you need:

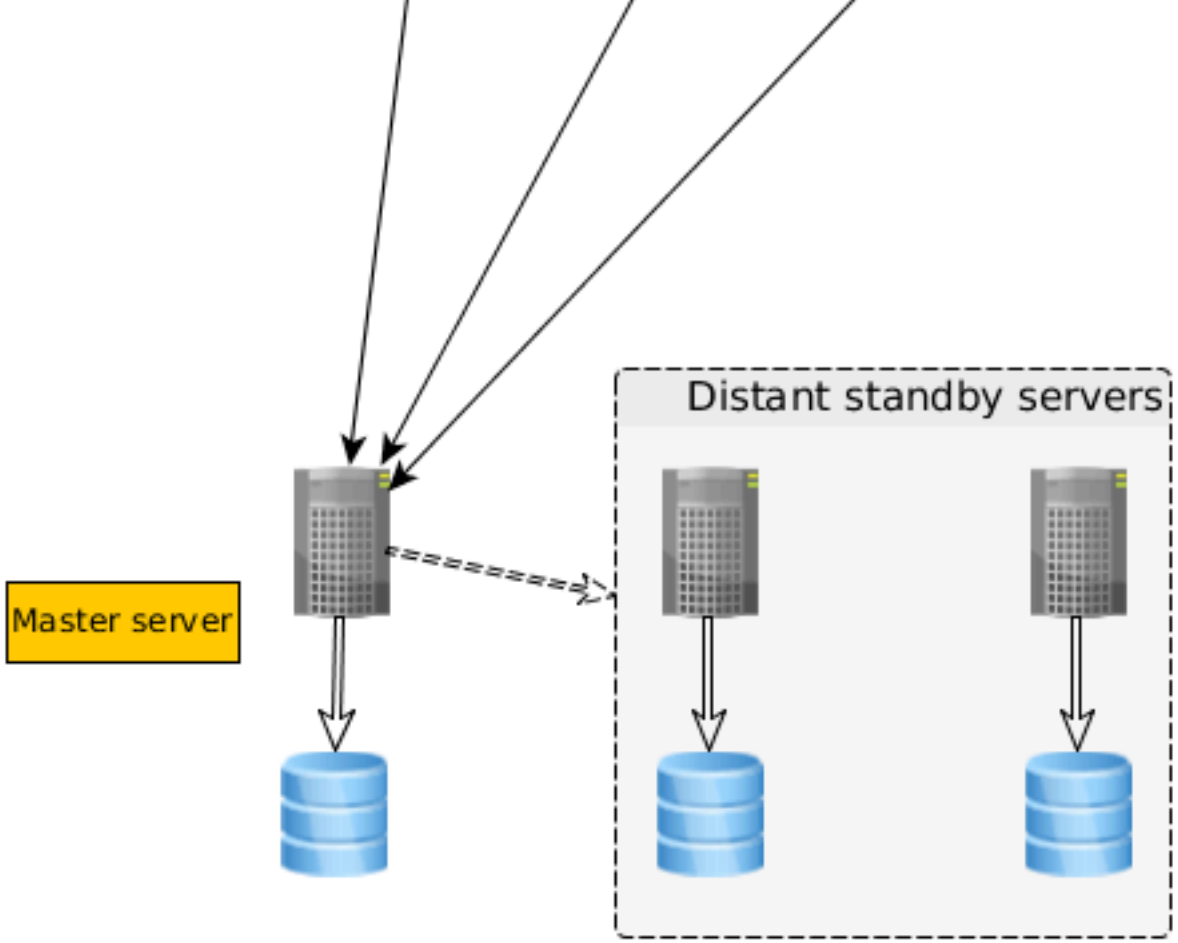
- redundant data centers
- redundant networks
- redundant *everything*

HA requires red



What we need to be highly available?

1. A plan
   a. what do i do if the database server crashes?
   b. what do i do if the server app crashes?
   c. are there other components that can fail?
   d. how much time do i have to restore de service (RPO)?
      i. am i allowed to lose data?
   e. how it affects other components of my system?
2. To detect the problem
3. Redundancy (no SPOF)
   a. Hardware (ie: spare parts)
   b. Data (ie: Backups)
   c. Both (ie: a standby server)



Plan: automatic failover (issues)

- Which standby to promote?
  - Consider the replication lag in **all** servers
- Make remaining standby servers follow the new master
- Inform application servers what happened

Plan: automatic failover (the tools)

repmgr

It enhances PostgreSQL's built-in hot-standby capabilities with tools to set up standby servers, monitor replication, and perform administrative tasks such as failover or manual switchover operations.

pgbouncer

Lightweight connection pooler for PostgreSQL

repmgr

Official website

http://repmgr.org

Download source from

https://github.com/2ndQuadrant/repmgr

Download binaries from

- For Redhat/Centos: yum.postgresql.org
- For Debian/Ubuntu: apt.postgresql.org

Current version
3.1

repmgr - basic commands

repmgr master register
Create the repmgr schema and catalogs in the master server

repmgr standby clone
Create a new standby server by copy master's data_directory and tablespaces (using rsync or pg_basebackup)

repmgr standby register
Insert all information needed to monitor the new standby's replication status and to use it for failover

repmgr cluster show
Shows information about what nodes are in the cluster and what their role are

repmgr - basic commands

repmgr standby promote
Promote a standby to become the new master

repmgr standby follow
Inform a standby that it needs to follow a new master

repmgr - installation (centos 6)

On all database servers

yum install https://download.postgresql.org/pub/repos/yum/9.5/redhat/rhel-6-x86_64/pgdg-centos95-9.5-2.noarch.rpm

yum install postgresql95 postgresql95-server postgresql95-contrib

yum install repmgr95

Bin

/usr/pgsql-9.5/bin

Data

/var/lib/pgsql/9.5/data

repmgr - master configuration

postgresql.conf

```
listen_addresses='*'
shared_preload_libraries='repmgr_funcs'
wal_level=logical
wal_keep_segments=5000
archive_mode=on
archive_command='exit 0'
max_wal_senders=10
max_replication_slots=10
hot_standby=on
```

repmgr - master configuration

pg_hba.conf

```
host postgres postgres 192.168.0.0/24 trust
host replication postgres 192.168.0.0/24 trust
```

Note

- Remember to allow port 5432 in the firewall
- To share ssh keys to allow trusted connections between nodes is also useful

repmgr - adding the standbys

Execute this command in both standby servers

```
/usr/pgsql-9.5/bin/repmgr standby clone -d postgres -U postgres -h 192.168.0.151
```

And you will get a feedback like this one

```
[2015-11-19 23:26:05] [NOTICE] no configuration file provided and default file './repmgr.conf' not found - continuing with default values
[2015-11-19 23:26:05] [NOTICE] starting backup...
NOTICE:  pg_stop_backup completado, todos los segmentos de WAL requeridos han sido archivados
[2015-11-19 23:26:10] [NOTICE] standby clone (using pg_basebackup) complete
[2015-11-19 23:26:10] [NOTICE] HINT: you can now start your PostgreSQL server
[2015-11-19 23:26:10] [NOTICE] for example : /etc/init.d/postgresql start
```

repmgr - sample configuration file

repmgr.conf

cluster=test

node=1

node_name=node1

conninfo='host=192.168.0.151 user=postgres dbname=postgres'

pg_bindir=/usr/pgsql-9.5/bin

repmgr - registering the nodes

On the master

/usr/pgsql-9.5/bin/repmgr -f repmgr.conf --verbose master register

```
[2015-11-19 23:56:18] [NOTICE] opening configuration file: repmgr.conf
[2015-11-19 23:56:18] [INFO] connecting to master database
[2015-11-19 23:56:18] [INFO] connected to master, checking its state
[2015-11-19 23:56:18] [INFO] master register: creating database objects inside the repmgr_test schema
[2015-11-19 23:56:18] [INFO] finding node list for cluster 'test'
[2015-11-19 23:56:18] [NOTICE] master node correctly registered for cluster test with id 1 (conninfo: host=192.168.0.151 user=postgres
 dbname=postgres)
```

repmgr - registering the nodes

On the standbys

/usr/pgsql-9.5/bin/repmgr -f repmgr.conf --verbose standby register

```
[2015-11-20 00:00:20] [NOTICE] opening configuration file: repmgr.conf
[2015-11-20 00:00:20] [INFO] connecting to standby database
[2015-11-20 00:00:21] [INFO] connecting to master database
[2015-11-20 00:00:21] [INFO] finding node list for cluster 'test'
[2015-11-20 00:00:21] [INFO] checking role of cluster node '1'
[2015-11-20 00:00:21] [INFO] registering the standby
[2015-11-20 00:00:21] [INFO] standby registration complete
[2015-11-20 00:00:21] [NOTICE] standby node correctly registered for cluster test with id 2 (conninfo: host=192.168.0.152 user=postgres
 dbname=postgres)
```

repmgr - checking the nodes

On any node

/usr/pgsql-9.5/bin/repmgr -f repmgr.conf --verbose cluster show

```
[2015-11-20 00:02:11] [NOTICE] opening configuration file: repmgr.conf
[2015-11-20 00:02:11] [INFO] connecting to database
Role      | Connection String
* master  | host=192.168.0.151 user=postgres dbname=postgres
  standby | host=192.168.0.152 user=postgres dbname=postgres
  standby | host=192.168.0.153 user=postgres dbname=postgres
```

Note

You can check the same info in the repmgr catalog `repl_nodes` inside the repmgr schema

repmgr - start monitoring the nodes

On all nodes

/usr/pgsql-9.5/bin/repmgrd --monitoring-history --daemonize -f $HOME/repmgr.conf

Note

You can check the status of every standby in the repmgr view `repl_status` inside the repmgr schema

repmgr - monitoring the nodes

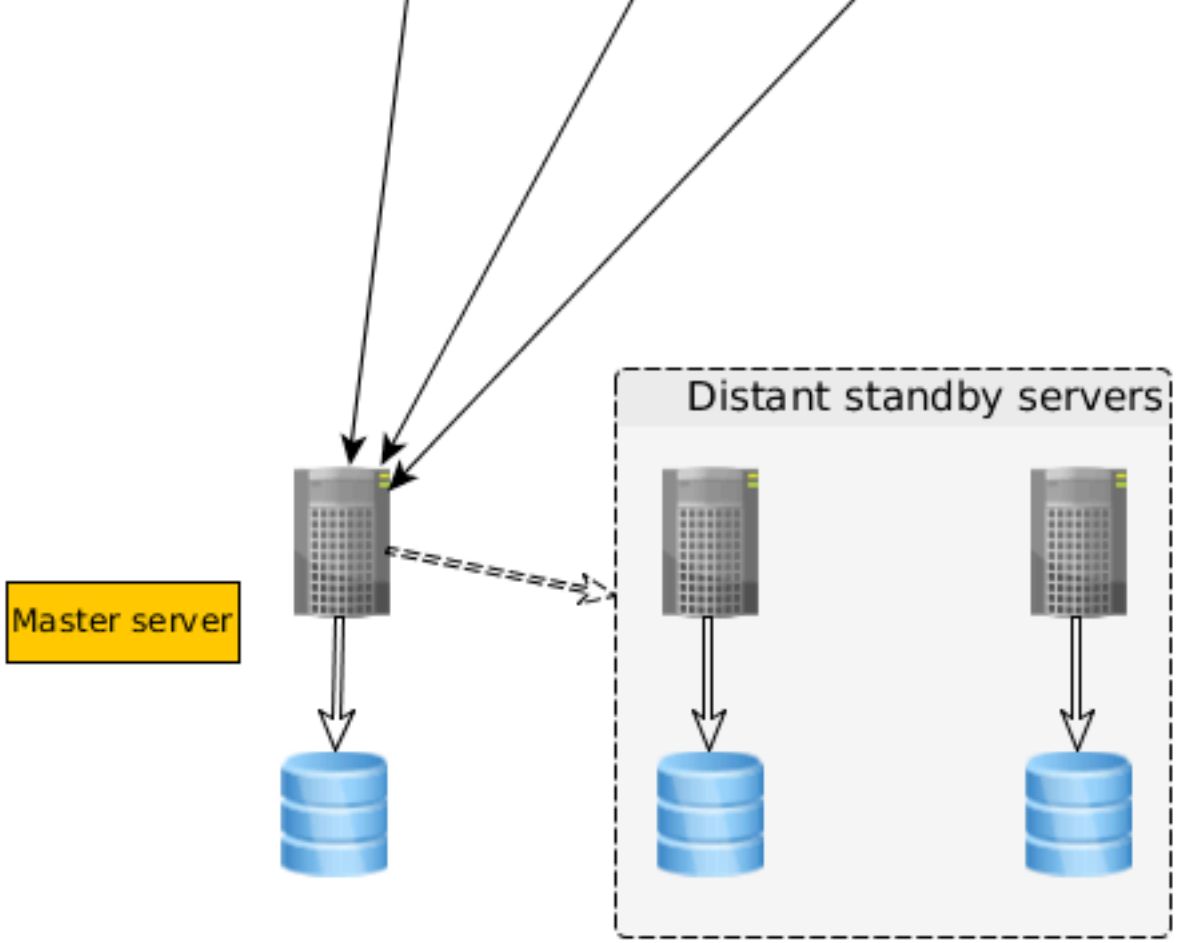
postgres=# select * from repmgr_test.repl_status order by 2;.

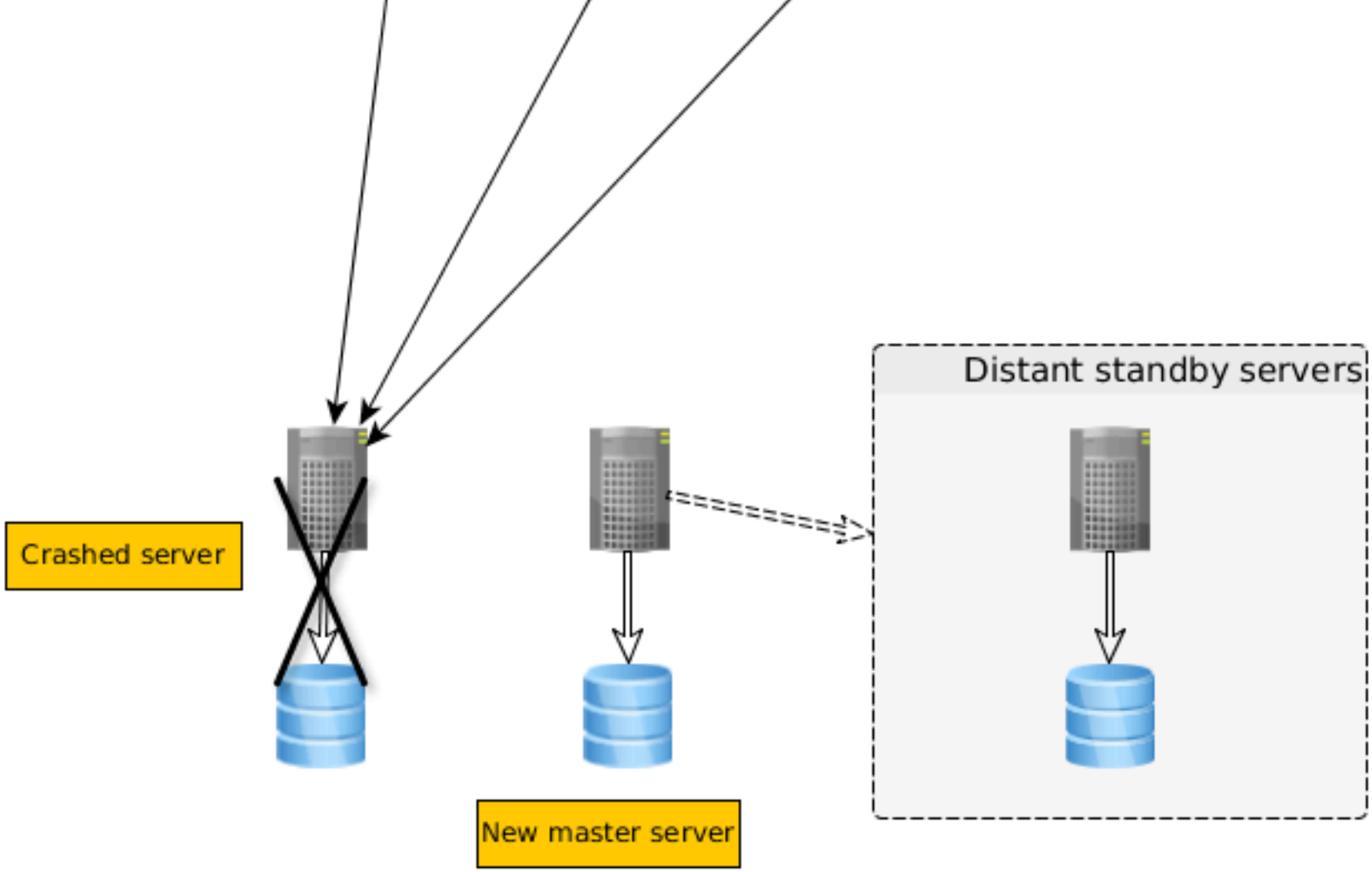
```
primary_node              | 1
standby_node              | 3
standby_name              | node3
node_type                 | standby
active                    | t
last_monitor_time         | 2015-11-20 00:14:40.995391-05
last_wal_primary_location | 0/ADA59F0
last_wal_standby_location | 0/ADA59F0
replication_lag           | 0 bytes
replication_time_lag      | 00:01:02.464177
apply_lag                 | 61 MB
communication_time_lag    | 00:00:00.128505
```

repmgr - autofailover (more configurations)

repmgr.conf

```
master_response_timeout=60
reconnect_attempts=6
reconnect_interval=10
failover=automatic
priority=100
promote_command='/usr/pgsql-9.5/bin/repmgr standby promote -f $HOME/repmgr.conf'
follow_command='/usr/pgsql-9.5/bin/repmgr standby follow -f $HOME/repmgr.conf -W'
retry_promote_interval_secs=300
```





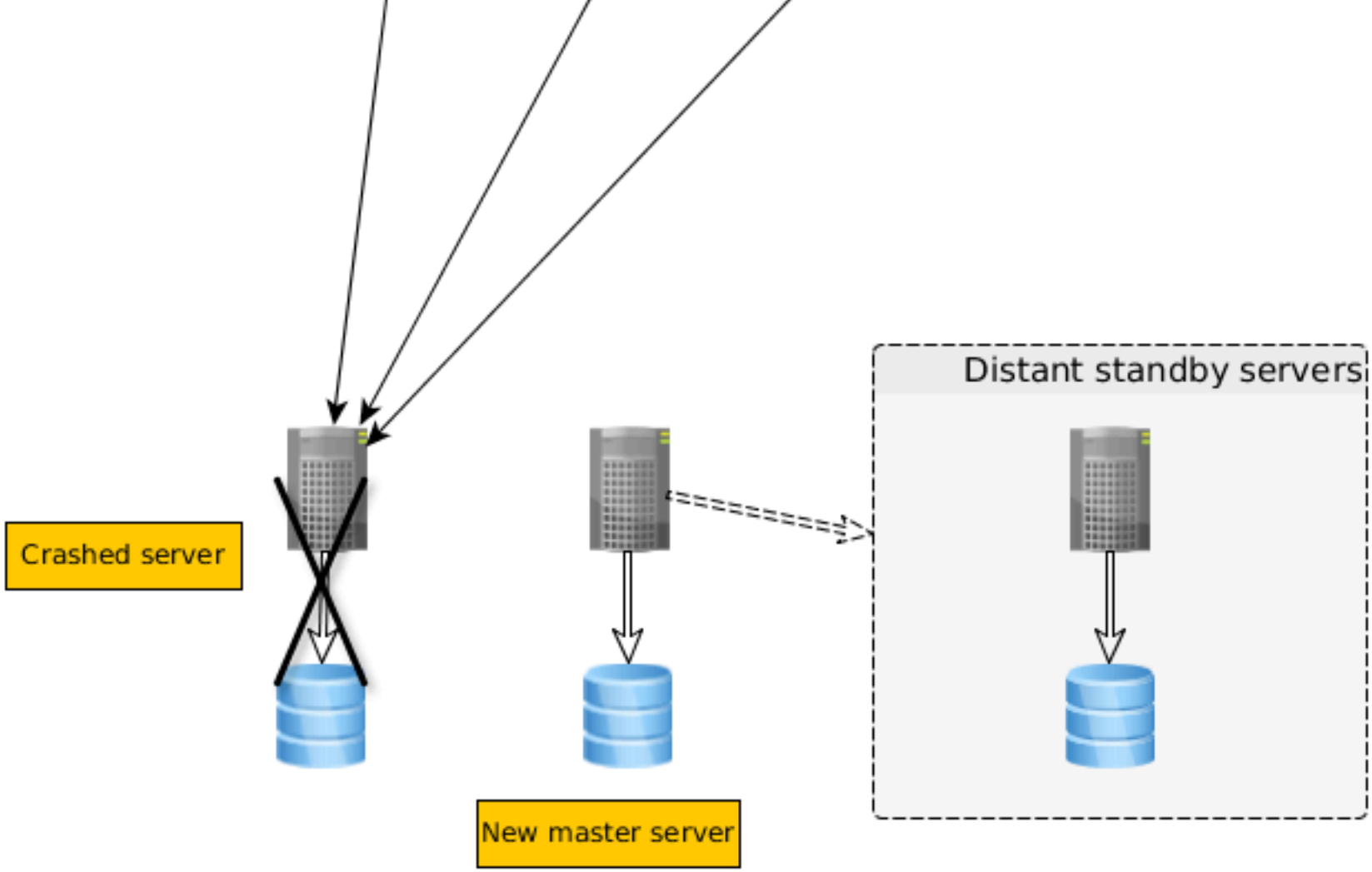
repmgr - autofailover (in action)

```
[2015-11-20 00:42:16] [WARNING] connection to upstream has been lost, trying to recover... 60 seconds before failover decision
[2015-11-20 00:42:26] [WARNING] connection to upstream has been lost, trying to recover... 50 seconds before failover decision
[2015-11-20 00:42:36] [WARNING] connection to upstream has been lost, trying to recover... 40 seconds before failover decision
[2015-11-20 00:42:46] [WARNING] connection to upstream has been lost, trying to recover... 30 seconds before failover decision
[2015-11-20 00:42:56] [WARNING] connection to upstream has been lost, trying to recover... 20 seconds before failover decision
[2015-11-20 00:43:06] [WARNING] connection to upstream has been lost, trying to recover... 10 seconds before failover decision
[2015-11-20 00:43:16] [ERROR] unable to reconnect to upstream (timeout 60 seconds)...
[2015-11-20 00:43:16] [ERROR] connection to database failed: could not connect to server: Connection refused
       Is the server running on host "192.168.0.151" and accepting
       TCP/IP connections on port 5432?

[2015-11-20 00:43:21] [ERROR] connection to database failed: could not connect to server: Connection refused
       Is the server running on host "192.168.0.151" and accepting
       TCP/IP connections on port 5432?

[2015-11-20 00:43:21] [NOTICE] promoting standby
[2015-11-20 00:43:21] [NOTICE] promoting server using '/usr/pgsql-9.5/bin/pg_ctl -D /var/lib/pgsql/9.5/data promote'
[2015-11-20 00:43:23] [NOTICE] STANDBY PROMOTE successful.  You should REINDEX any hash indexes you have.
```

repmgr - autofailover (in action)

postgres=# select id, type, upstream_node_id, name, active from repmgr_test.repl_nodes ;.

```
 id |  type   | upstream_node_id | name  | active
----+---------+------------------+-------+--------
  1 | master  |                  | node1 | f
  2 | master  |                  | node2 | t
  3 | standby |                2 | node3 | t
(3 filas)
```



pgbouncer

Official website

https://pgbouncer.github.io

Download source from

https://pgbouncer.github.io/downloads/

Download binaries from

- For Redhat/Centos: yum.postgresql.org
- For Debian/Ubuntu: apt.postgresql.org

Current version

1.6

pgbouncer - installation (centos 6)

On all application servers

yum install pgbouncer

Bin

/usr/bin

Conf

/etc/pgbouncer/pgbouncer.ini

pgbouncer - configuring (the databases entry)

```
[databases]
```

```
master=host=192.168.0.151 user=postgres dbname=postgres
```

Note

Application servers should connect to the database=master at host=127.0.0.1 using port=6432

pgbouncer - reconfiguring (the databases entry)

postgres=# select type || '=' || conninfo from repmgr_test.repl_nodes where active and type = 'master';.

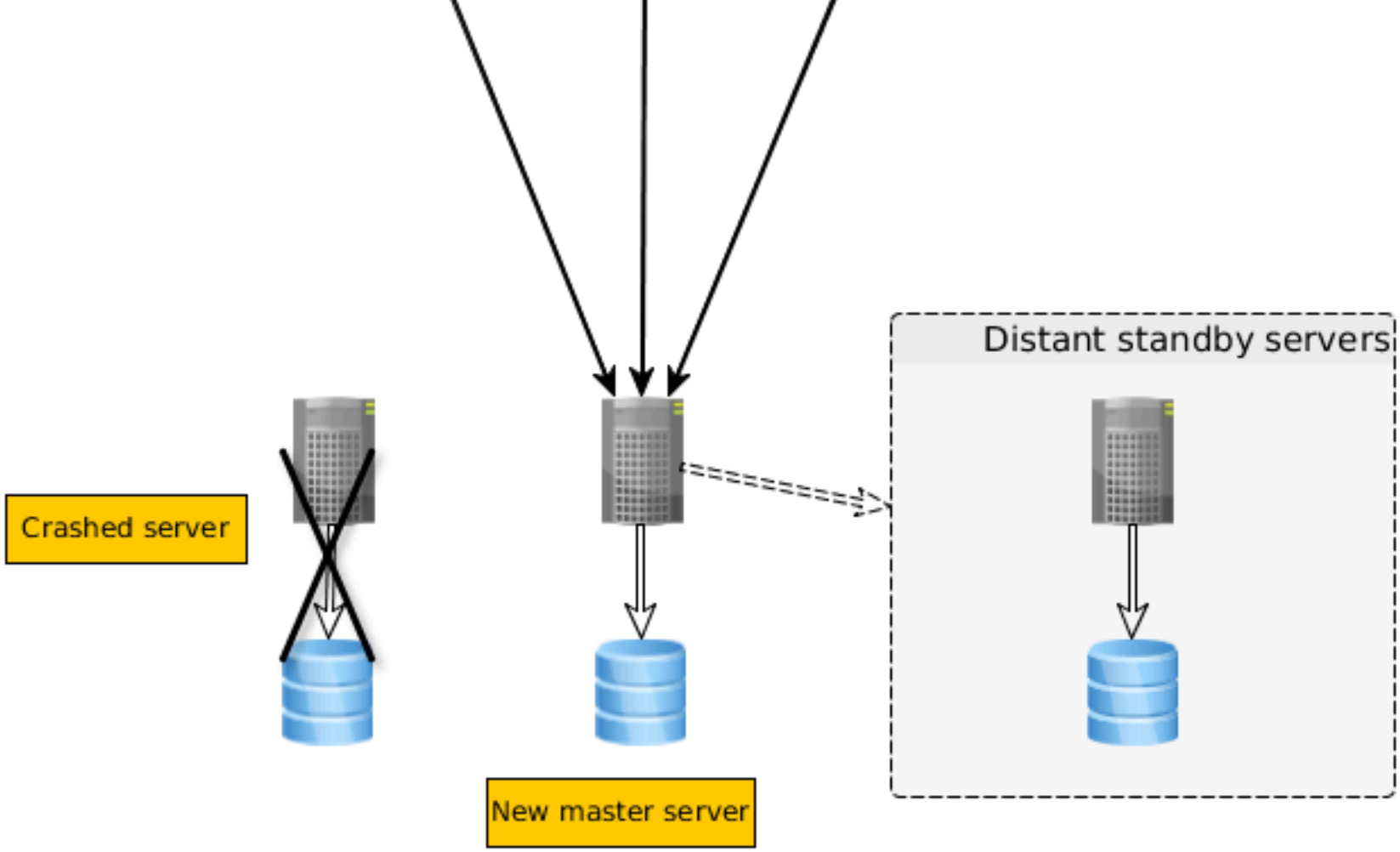
```
                        ?column?
-----------------------------------------------------------
 master=host=192.168.0.152 user=postgres dbname=postgres
(1 fila)
```

pgbouncer - reconfiguring by repmgr

promote_command = '/bin/bash $HOME/execute_failover.sh'.

```
#!/bin/bash

APP_SERVER_LIST="192.168.0.100 192.168.0.101 192.168.0.102"

# Pause all pgbouncer first to minimaze impact on clients
for ip in $APP_SERVER_LIST
do
    psql -tc "pause" -h $ip -p 6432 -U postgres pgbouncer
done

/usr/pgsql-9.5/bin/repmgr standby promote -f $HOME/repmgr.conf

psql -tc "select type || '=' || conninfo from repmgr_test.repl_nodes
          where active and type = 'master'" postgres > pgbouncer.ini
cat pgbouncer.ini.template >> pgbouncer.ini
```

```
for ip in $APP_SERVER_LIST
do
    rsync pgbouncer.ini $ip:/etc/pgbouncer/pgbouncer.ini
    psql -tc "reload" -h $ip -p 6432 -U postgres pgbouncer
    psql -tc "resume" -h $ip -p 6432 -U postgres pgbouncer
done

echo "Reconfigure of pgbouncer complete"
```



Conclusions