ZABBIX

Put it together

RHEIN-MAIN SOLUTIONS

Thomas Oftring
(Zabbix Certified Professional & Database Architect)

ZABBIX

Preset Goals

RHEIN-MAIN SOLUTIONS

Goals that are preset by the project:

- System should be high available
- Possibility to choose between manual and automatic failover
- Archive server with configurable time delay
- Short meantime to recover (MTR) on database
- Point in time recovery (PITR) is a must have
- Cluster management with toolchain
- Backup of database with toolchain
- SSL encryption for the Zabbix web frontend

ZABBIX

RHEIN-MAIN SOLUTIONS

Facts given by Zabbix

Zabbix server facts:

- Zabbix server itself is a stateless server
- Zabbix server config file (zabbix_server.conf)
- Only one running Zabbix server instance is allowed

Zabbix web frontend facts:

- Zabbix web frontend itself is stateless
- Zabbix web frontend config file
- Web server configuration (Apache, Nginx, …)

ZABBIX

RHEIN-MAIN SOLUTIONS

Facts given by PostgreSQL

PostgreSQL facts:

- Streaming replication is build in
- HotStandby server is possbile
- Archive server is possible
- POITR is build in by WAL archiving
- Short MTR by POITR
- Every Server in streaming replication can be promoted to a primary server
- PostgreSQL Cluster should be monitored by Zabbix

ZABBIX

Facts given by Apache

RHEIN-MAIN SOLUTIONS

Apache facts:

- Web server itself is stateless
- SSL encryption is available

ZABBIX

RHEIN-MAIN SOLUTIONS

Used tools

repmgr:

- Open-source tool suite for managing replication and failover in a cluster of PostgreSQL servers
- Base configuratione in config file

Barman:

- Open-source tool for backup and recovery of PostgreSQL database servers
- Base configuration in config file

ZABBIX

How to put this together

RHEIN-MAIN SOLUTIONS

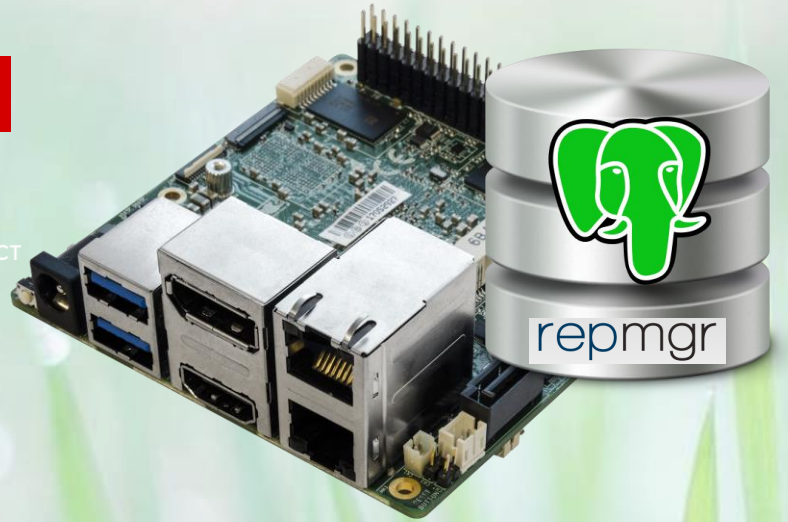
Solution:

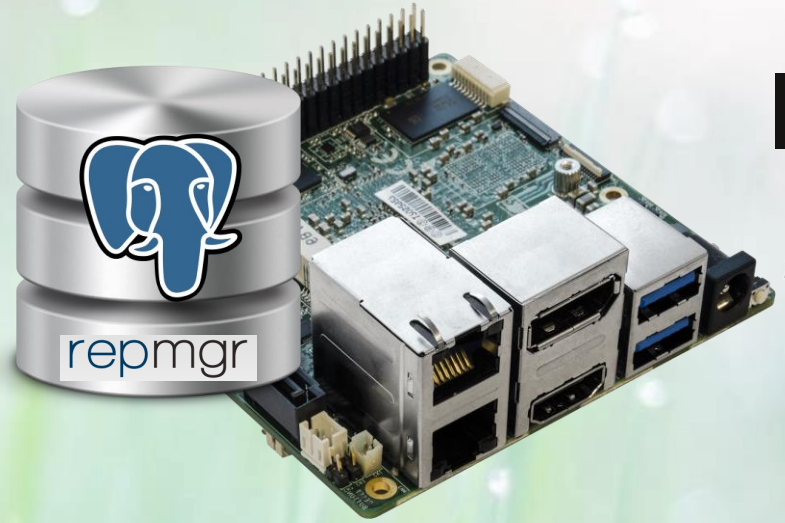
- Zabbix Server on primary and hot standbay server connect to localhost db
- Three PostgreSQL database servers
  - Primary server
  - Hot standby (read only)
  - Archive server (with time gap)
- Zabbix web frontend running on Apache web server
- repmgr for cluster management
- barman for backup and recovery management

RHEIN-MAIN SOLUTIONS

ZABBIX

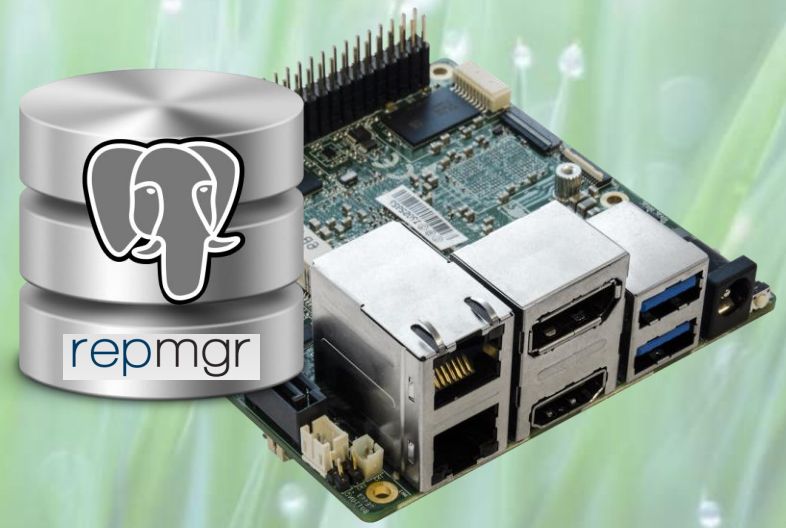
APACHE
HTTP SERVER PROJECT





ZABBIX

APACHE
HTTP SERVER PROJECT



ZABBIX

RHEIN-MAIN SOLUTIONS

Live Demo

ZABBIX

Live Demo

abstract Zabbix server

RHEIN-MAIN SOLUTIONS

2. rmspg01

```
[root@rmspg01 log]# cat /etc/zabbix/zabbix_server.conf |grep DB |grep "^[^#;]"
DBHost=localhost
DBName=zabbix
DBUser=zabbix
DBPassword=SuperSecurePassword!?1234
DBPort=5432
```

2. rmspg01

```
[root@rmspg01 log]# systemctl list-unit-files --type service |egrep "zabbix|postgresql|httpd"
httpd.service                                 enabled
postgresql-10.service                         enabled
zabbix-agent.service                          enabled
zabbix-proxy.service                          enabled
zabbix-server.service                         enabled
```

ZABBIX

RHEIN-MAIN SOLUTIONS

Live Demo

abstract PostgreSQL

PostgreSQL configuration files:

- postgresql.conf
  contains the base configuration of the database
- recovery.conf
  manage the parameters for the recovery of a standby server
- pg_hba.conf
  manage the access rights to the database by network access

ZABBIX

RHEIN-MAIN SOLUTIONS

Live Demo

abstract repmgr

Control of automatic/manual failover by running repmgr daemon

Setup repmgr for PostgreSQL Clusters

Attention: All repmgr commands have to be executed as user postgres (su postgres)

1. Register the primary server (execute on the system that should be the primary)
   ```
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf primary register
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf cluster show
   ```

2. Clone the standby server (execute on the system that should be the standby)
   ```
   /usr/pgsql-10/bin/repmgr -h your.servername1.example -U repmgr -d repmgr -f /etc/repmgr/10/repmgr.conf standby clone --dry-run
   /usr/pgsql-10/bin/repmgr -h your.servername1.example -U repmgr -d repmgr -f /etc/repmgr/10/repmgr.conf standby clone -c
   systemctl start postgresql-10
   ```

3. Register the new standby server (execute on the system that should be the standby)
   ```
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf standby register
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf cluster show
   ```

4. Clone a standby archive server (Use the standby server as streaming source)
   ```
   /usr/pgsql-10/bin/repmgr -h your.servername2.example -U repmgr -d repmgr -f /etc/repmgr/10/repmgr.conf standby clone --upstream-node-id=2 --dry-run
   /usr/pgsql-10/bin/repmgr -h your.servername2.example -U repmgr -d repmgr -f /etc/repmgr/10/repmgr.conf standby clone --upstream-node-id=2 -c
   systemctl start postgresql-10
   ```

5. Register the standby archive server with time gap
   ```
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf standby register --upstream-node-id=2
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf cluster show
   ```

PostgreSQL

RHEIN-MAIN SOLUTIONS

Administration repmgr for PostgreSQL Clusters

Attention: All repmgr commands have to executed as user postgres (su postgres)

1. Check cluster state (Never use on the archive server with time gap. You will see the cluster state in the past)
   ```
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf cluster show
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf cluster matrix
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf cluster crosscheck
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf cluster event
   ```

2. Check the state of the actual node (where the command is executed)
   ```
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf node status
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf node check
   ```

3. Promote the standby server to a primary server (do this only if the primary is not longer available, for example on a crash)
   ```
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf standby promote
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf cluster show
   ```

4. Standby server manual failover (Standby will be promoted to primary and primary will become the standby)
   ```
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf standby switchover --dry-run
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf standby switchover
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf cluster show
   ```

5. Connect a standby server to the new primary server as streaming replication client
   ```
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf standby follow --dry-run
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf standby follow
   /usr/pgsql-10/bin/repmgr -f /etc/repmgr/10/repmgr.conf cluster show
   ```

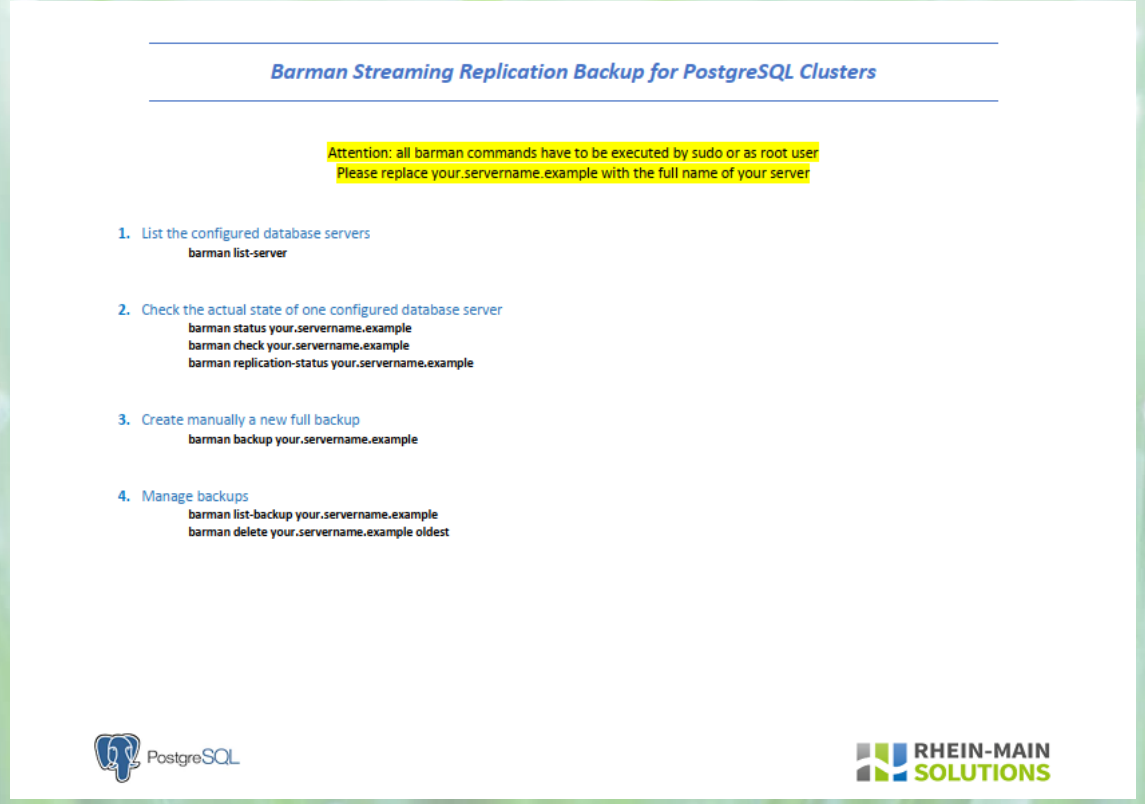
PostgreSQL

RHEIN-MAIN SOLUTIONS

ZABBIX

Live Demo

abstract barman



Barman Streaming Replication Backup for PostgreSQL Clusters

Attention: all barman commands have to be executed by sudo or as root user
Please replace your.servername.example with the full name of your server

1. List the configured database servers
   barman list-server

2. Check the actual state of one configured database server
   barman status your.servername.example
   barman check your.servername.example
   barman replication-status your.servername.example

3. Create manually a new full backup
   barman backup your.servername.example

4. Manage backups
   barman list-backup your.servername.example
   barman delete your.servername.example oldest

PostgreSQL

RHEIN-MAIN SOLUTIONS

RHEIN-MAIN SOLUTIONS

ZABBIX

RHEIN-MAIN SOLUTIONS

Caveats and pitfalls

Zabbix:

- Zabbix server db connection only to localhost
- Zabbix web frontend db connection only to localhost
- Syncronize the config files (Server and web frontend)

PostgreSQL:

- Do not forget to monitor the replication state

barman:

- Have enough space for the backups
- Do not forget to monitor the backup process

ZABBIX

RHEIN-MAIN SOLUTIONS

Questions!

ZABBIX

The Ultimate Enterprise – class Monitoring Platform

Thomas Oftring
(Zabbix Certified Professional & Database Architect)