Protecting your data with Patroni and pgBackRest

PGDay Russia 2021

Federico Campoli

Somewhere in the time vortex

Few words about the speaker

KAMEDATA
DEVOPS AND DATA ENGINEERING

- Born in 1972
- Passionate about IT since 1982
- Joined the Oracle DBA secret society in 2004
- In love with PostgreSQL since 2006
- PostgreSQL tattoo on the right shoulder
- Freelance devops and data engineer

Getting in touch

KAMEDATA
DEVOPS AND DATA ENGINEERING

- **Blog:** https://pgdba.org
- **Twitter:** @4thdoctor_scarf
- **Github:** https://github.com/the4thdoctor
- **Linkedin:** https://www.linkedin.com/in/federicocampoli/
- **Youtube:** https://www.youtube.com/c/FedericoCampoli

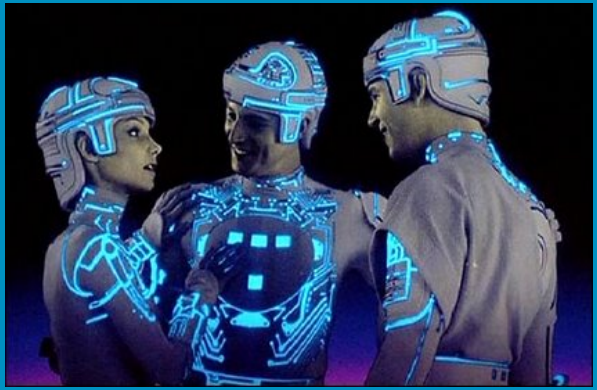
Table of contents

KAMEDATA
DEVOPS AND DATA ENGINEERING

1. Greetings, programs!
2. The grid
3. The light cycle maze
4. End of line
5. I fight for the users

Greetings, programs!

KAMEDATA
DEVOPS AND DATA ENGINEERING



Copyright Walt Disney LTD
Image source
https://liveforfilms.wordpress.com/2009/07/09/greetings-program-tron-2-synopsis/

PostgreSQL

https://www.postgresql.org/

- Enteprise class RDBMS
- ACID compliant
- HA and DR
- With one tiny little catch...
- No built in mechanism for automating backups or failover

Some PostgreSQL auto failover options

KAMEDATA
DEVOPS AND DATA ENGINEERING

- ~~B.Y.O.T, shell script, manually operated, Cthulhu summoning...~~
- Automated with third party tools
  - repmgr
  - pg_auto_failover
  - PostgreSQL Automatic Failover (PAF)
  - Patroni

Patroni

https://github.com/zalando/patroni
Patroni is an auto failover system developed in python by Zalando.
The tool relies on a distributed consensus store (DCS) to maintain the cluster status.
Patroni is available in the pgdg official repository

Patroni

- Developed in Python
- Supports for DCS etcd,consul,zookeper
- Support for python RAFT (requires pysyncobj module)
- Automated boostrap and replica setup
- Automated failover/switchover
- Centralised configuration for PostgreSQL stored in DCS
- Very resilient to split brain
- HAProxy or pgbouncer for connection routed via api check

Caveats

- Everything is managed by Patroni
- Some parts of the documentation is unclear (yes, standby_cluster I'm talking of you)
- Client only in interactive mode, you need to build your api call (e.g. via ansible uri module)

Some PostgreSQL backup options

KAMEDATA
DEVOPS AND DATA ENGINEERING

- ~~logical with pg_dump~~
- physical with tools like
  - pg_basebackup
  - barman
  - WAL-E/WAL-G
  - pgBackRest

pgBackRest

https://pgbackrest.org/
pgBackRest is a simple and reliable solution for automatic the backups.
pgBackRest is a community driven project.

pgBackRest

KAMEDATA
DEVOPS AND DATA ENGINEERING

- Physical backup tool
- Implements HA/DR
- Differential,incremental and full backup
- Parallel jobs configurable
- Can backup from the standby servers
- Async WAL push and pull
- Developed initially in perl now fully migrated to C
- ini style configuration
- Available in deb/yum pgdg repositories
- Cloud backup options GCP/S3/Azure
- Multiple repositories configurable
- Self contained backup repository

Caveats

- Configuring remote backups via SSH may be complex
- Beware of when configuring archive-push-queue-max.
  There is the risk of wal files not being archived if the limit is reached.

The rule of thumb

KAMEDATA
DEVOPS AND DATA ENGINEERING

RTFM

Advanced pgBackRest

KAMEDATA
DEVOPS AND DATA ENGINEERING

For advanced topics on pgBackRest please check Stefan Fercot's Talk
Unleash the Power within pgBackRest
https://pgday.ru/en/2021/papers/297

The grid

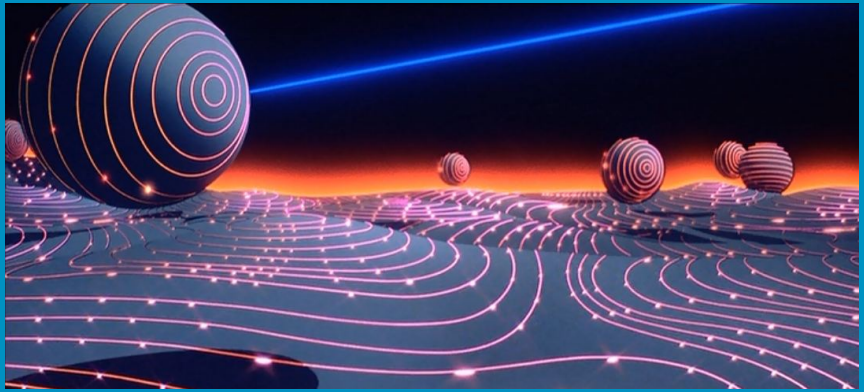
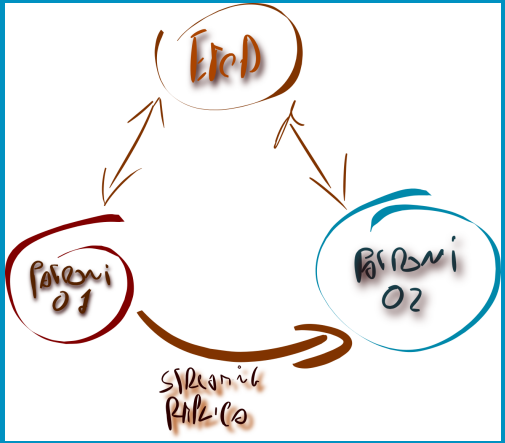


Image source

https://siftingthroughpatterns.wordpress.com/2012/05/12/why-kevin-flynn-is-the-true-villain-behind-tron-legacy/

Patroni super biased configuration

KAMEDATA
DEVOPS AND DATA ENGINEERING



Setup

KAMEDATA
DEVOPS AND DATA ENGINEERING

For our example we'll use CentOS 7 and a GCP bucket

- etcd01
- patroni01
- patroni02

Configure and install the software

KAMEDATA
DEVOPS AND DATA ENGINEERING

- (Optional) Setup /etc/hosts with the names
- Open the required ports on firewalld
- Install etcd on the etcd server
- Configure and start etcd
- Install PostgreSQL 13 and patroni on the two patroni nodes
- Configure patroni
- Start patroni
- profit

Hosts (optional)

If there is no DNS service in place is a good idea to have the /etc/hosts file set for resolving the names

```
127.0.0.1       localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.56.40   patroni01
192.168.56.41   patroni02
192.168.56.43   etcd01

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Firewalld

etcd01

```
sudo firewall-cmd --permanent --zone=public --add-port=2380/tcp
sudo firewall-cmd --permanent --zone=public --add-port=2379/tcp

sudo systemctl reload firewalld
```

patroni01, patroni02

```
sudo firewall-cmd --permanent --zone=public --add-port=5432/tcp
sudo firewall-cmd --permanent --zone=public --add-port=8008/tcp

sudo systemctl reload firewalld
```

etcd01 setup

Install etcd

```
sudo yum install etcd
```

etcd01 setup

Then configure the environment file
/etc/etcd/etcd.conf

```
ETCD_LISTEN_PEER_URLS="http://192.168.56.43:2380"
ETCD_LISTEN_CLIENT_URLS="http://192.168.56.43:2379"
ETCD_NAME="etcd01"
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://192.168.56.43:2380"
ETCD_ADVERTISE_CLIENT_URLS="http://192.168.56.43:2379"
ETCD_INITIAL_CLUSTER="etcd01=http://192.168.56.43:2380"
ETCD_INITIAL_CLUSTER_STATE=new
```

etcd01 setup

Start the service and check the cluster is healty

```
sudo systemctl start etcd

etcdctl --endpoints "http://192.168.56.43:2379" cluster-health
member 6bf749bb3ab16843 is healthy: got healthy result from http
    ://192.168.56.43:2379
cluster is healthy
```

postgresql install

Install the PostgreSQL yum repository and PostgreSQL 13

```
sudo yum install -y \
https://download.postgresql.org/pub/repos/yum/reporpms/EL-7-x86_64/pgdg-redhat-repo-
    latest.noarch.rpm
sudo yum install -y postgresql13 postgresql13-contrib postgresql13-server
```

patroni install

KAMEDATA
DEVOPS AND DATA ENGINEERING

Install the epel-release first then patroni and patroni-etcd

```
sudo yum install -y epel-release
sudo yum install -y patroni patroni-etcd
```

Configure patroni dcs section I

KAMEDATA
DEVOPS AND DATA ENGINEERING

On both patroni machines create the directory /etc/patroni
Then add a new configuration file into the directory named patroni.yml

```
scope: test
namespace: /patroni_test/
name: patroni01 #this values should be unique within the namespace and scope
log:
  dir: /var/log/patroni
restapi:
  listen: 192.168.56.40:8008
  connect_address: 192.168.56.40:8008
etcd:
  hosts: 192.168.56.43:2379
bootstrap:
  dcs:
    ttl: 10
    loop_wait: 10
    retry_timeout: 10
    maximum_lag_on_failover: 1048576
  postgresql:
```

Configure patroni dcs section II


```
    use_pg_rewind: true
    use_slots: false
    parameters:
      wal_level: 'replica'
      archive_mode: 'off'
      unix_socket_directories: '/var/run/postgresql/.'
method: initdb
initdb:  # Note: It needs to be a list (some options need values, others are
    switches)
- encoding: UTF8
- data-checksums
pg_hba:  # Add following lines to pg_hba.conf after running 'initdb'
- host replication replicator 0.0.0.0/0 md5
- host all all 0.0.0.0/0 md5
users:
```

Configure patroni postgresql section I

KAMEDATA
DEVOPS AND DATA ENGINEERING

```
postgresql:
  listen: "*:5432"
  connect_address: patroni01:5432 #this is the local machine name, should be set
      accordingly
  data_dir: /var/lib/pgsql/data/postgresql0
  bin_dir: /usr/pgsql-13/bin/
  pgpass: /tmp/pgpass0
  authentication:
    replication:
      username: replicator
      password: postgres_replica
    superuser:
      username: postgres
      password: postgres_super
    rewind:  # Has no effect on postgres 10 and lower
      username: rewind_user
      password: postgres_rewind
  parameters:
    wal_level: 'replica'
    archive_mode: 'off'
```

Configure patroni postgresql section II

KAMEDATA
DEVOPS AND DATA ENGINEERING

```
    unix_socket_directories: '/var/run/postgresql/.'
tags:
    nofailover: false
    noloadbalance: false
    clonefrom: false
    nosync: false
```

Create the systemd service file I

KAMEDATA
DEVOPS AND DATA ENGINEERING

In /etc/systemd/system create the file patroni.service

```
[Unit]
Description=Runners to orchestrate a high-availability PostgreSQL
After=syslog.target network.target

[Service]
Type=simple

User=postgres
Group=postgres

# Read in configuration file if it exists, otherwise proceed
EnvironmentFile=-/etc/patroni_env.conf

WorkingDirectory=/var/lib/pgsql

# Start the patroni process
ExecStart=/bin/patroni /etc/patroni/patroni.yml
```

Create the systemd service file II

KAMEDATA
DEVOPS AND DATA ENGINEERING

```
# Send HUP to reload from patroni.yml
ExecReload=/bin/kill -s HUP $MAINPID

# only kill the patroni process, not it's children, so it will gracefully stop postgres
KillMode=process

# Give a reasonable amount of time for the server to start up/shut down
TimeoutSec=30

# Do not restart the service if it crashes, we want to manually inspect database on failure
Restart=no

[Install]
WantedBy=multi-user.target
```

Enable and start the patroni service

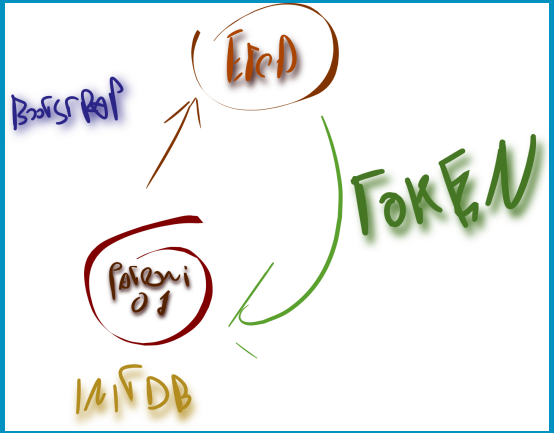
KAMEDATA
DEVOPS AND DATA ENGINEERING

On both patroni machines

```
sudo systemctl daemon-reload
sudo systemctl enable patroni
sudo systemctl start patroni
```

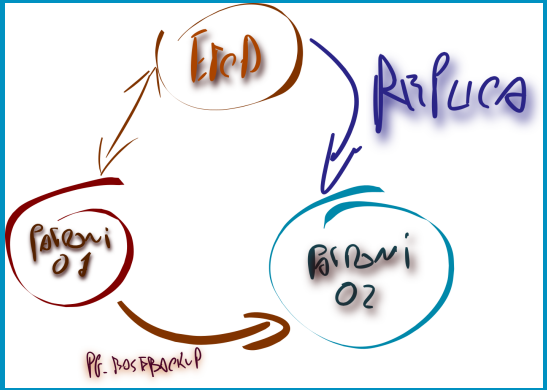
Patroni bootstrap

KAMEDATA
DEVOPS AND DATA ENGINEERING




```
[root@patroni01 patroni]# patronictl -c /etc/patroni/patroni.yml list
+ Cluster: test (6981078877725420504) -----+----+-----------+
| Member     | Host       | Role    | State   | TL | Lag in MB |
+------------+------------+---------+---------+----+-----------+
| patroni01 | patroni01 | Leader | running |  1 |           |
+------------+------------+---------+---------+----+-----------+
```

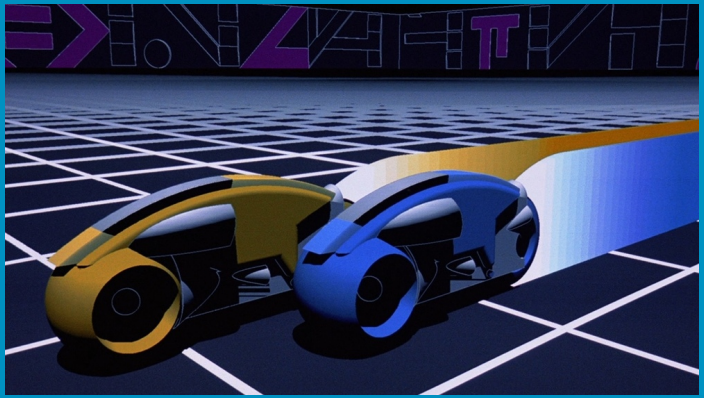
Patroni cloning




```
[root@patroni01 patroni]# patronictl -c /etc/patroni/patroni.yml list
+ Cluster: test (6981078877725420504) ------+----+-----------+
| Member     | Host       | Role    | State   | TL | Lag in MB |
+------------+------------+---------+---------+----+-----------+
| patroni01 | patroni01 | Leader  | running |  1 |           |
| patroni02 | patroni02 | Replica | running |  1 |         0 |
+------------+------------+---------+---------+----+-----------+
```

The light cycle maze

KAMEDATA
DEVOPS AND DATA ENGINEERING

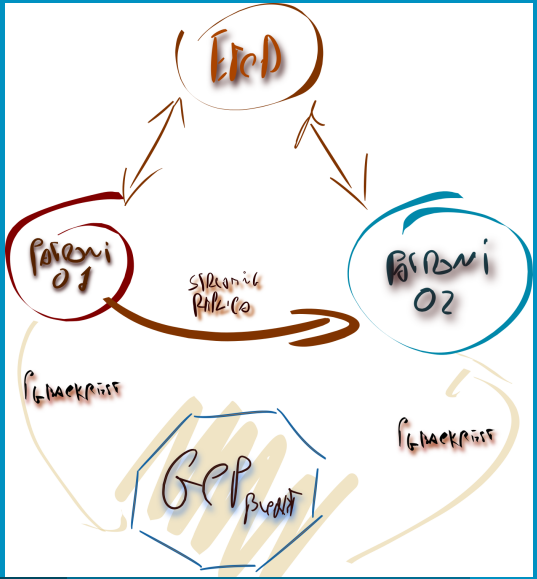


Copyright Walt Disney LTD
Image source

https://geektyrant.com/news/the-history-of-the-tron-lightcycle-infographic

Patroni and pgBackRest

KAMEDATA
DEVOPS AND DATA ENGINEERING



Configure and install the software

KAMEDATA
DEVOPS AND DATA ENGINEERING

- Install pgBackRest on both patroni servers
- Create the GCP bucket
- Create the shared key json file and grant access to the GCP bucket
- Configure pgBackRest for using the GCP bucket
- Configure patroni to use pgBackRest
- Run the pgBackRest first backup
- profit

Configure pgbackrest

KAMEDATA
DEVOPS AND DATA ENGINEERING

On both patroni machines edit the file /etc/pgbackrest.conf

```
[global]
repo1-type=gcs
repo1-path=/repo
repo1-gcs-bucket=patroni-pgbackrest
repo1-gcs-key=/etc/gcp_files/gcp-key.json
repo1-retention-full=1
log-level-console=info
log-level-file=debug
log-path=/var/log/patroni/

[test]
pg1-path=/var/lib/pgsql/data/postgresql0
pg1-port=5432
pg1-user=postgres
```

Optional, configure .pgpass

If the local login is made with password authentication it may be necessary to configure the .pgpass file in the PostgreSQL data directory.

```
patroni01:5432:*:postgres:postgres_super
```

Save the gcp-key.json

KAMEDATA
DEVOPS AND DATA ENGINEERING

Create the directory /etc/gcp_files and save into it the file gcp-key.json generated by the GCP console.
Directoy and file must be accessible by the user running pgbackrest.

```
{
  "type": "service_account",
  "project_id": "XXXXXXXXXXXX",
  "private_key_id": "XXXXXXXXXXXX",
  "private_key": "-----BEGIN PRIVATE KEY-----
XXXXXXXXXXXXXXXXXXX
-----END PRIVATE KEY-----
",
  "client_email": "XXXXXXXXXXXX",
  "client_id": "XXXXXXXXXXXX",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://oauth2.googleapis.com/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/certs",
  "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/x509/XXXXXXXXXXXXXX"
}
```

Configure patroni for using pgbackrest

KAMEDATA
DEVOPS AND DATA ENGINEERING

```
patronictl -c /etc/patroni/patroni.yml edit-config
##add the following lines
postgresql:
  parameters:
    archive_command: pgbackrest --stanza=test archive-push %p
    archive_mode: 'on'
```

After saving the command patronictl -c /etc/patroni/patroni.yml list will show the pending restart status

```
+ Cluster: test (6981352535078471124) ------+----+-----------+-----------------+
| Member    | Host      | Role    | State   | TL | Lag in MB | Pending restart |
+-----------+-----------+---------+---------+----+-----------+-----------------+
| patroni01 | patroni01 | Leader  | running |  1 |           | *               |
| patroni02 | patroni02 | Replica | running |  1 |         0 | *               |
+-----------+-----------+---------+---------+----+-----------+-----------------+
```

Restart patroni for applying the changes

KAMEDATA
DEVOPS AND DATA ENGINEERING

```
patronictl -c /etc/patroni/patroni.yml restart test
+ Cluster: test (6981352535078471124) ------+----+-----------+-----------------+
| Member    | Host      | Role    | State   | TL | Lag in MB | Pending restart |
+-----------+-----------+---------+---------+----+-----------+-----------------+
| patroni01 | patroni01 | Leader  | running |  1 |           | *               |
| patroni02 | patroni02 | Replica | running |  1 |         0 | *               |
+-----------+-----------+---------+---------+----+-----------+-----------------+
When should the restart take place (e.g. 2021-07-05T10:39)  [now]:
Are you sure you want to restart members patroni01, patroni02? [y/N]: y
Restart if the PostgreSQL version is less than provided (e.g. 9.5.2)  []:
Success: restart on member patroni01
Success: restart on member patroni02
```

Run the first backup

KAMEDATA
DEVOPS AND DATA ENGINEERING

Create the stanza and then run the backup

```
pgbackrest --stanza=test stanza-create
2021-07-05 09:41:31.677 P00   INFO: stanza-create command begin 2.34: --exec-id
    =7404-761f890e --log-level-console=info --log-level-file=info --log-path=/var/log
    /patroni --pg1-path=/var/lib/pgsql/data/postgresql0 --pg1-port=5432 --pg1-user=
    postgres --repo1-gcs-bucket=patroni-pgbackrest --repo1-gcs-key=<redacted> --repo1
    -path=/repo --repo1-type=gcs --stanza=test
2021-07-05 09:41:32.284 P00   INFO: stanza-create for stanza 'test' on repo1
2021-07-05 09:41:33.182 P00   INFO: stanza-create command end: completed
    successfully (1506ms)

pgbackrest --stanza=test backup
```

Check the backup status

Check the backup status

```
pgbackrest info
stanza: test
    status: ok
    cipher: none

    db (current)
        wal archive min/max (13): 000000010000000000000004/000000010000000000000004

        full backup: 20210705-145357F
            timestamp start/stop: 2021-07-05 14:53:57 / 2021-07-05 14:54:44
            wal start/stop: 000000010000000000000004 / 000000010000000000000004
            database size: 24.0MB, database backup size: 24.0MB
            repo1: backup set size: 2.9MB, backup size: 2.9MB
```

End of line

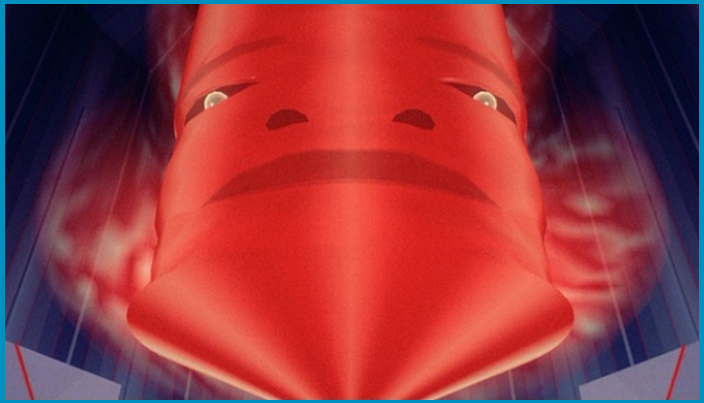


Image source

https://tron.fandom.com/wiki/Master_Control_Program

Recovering from a complete datacenter loss

KAMEDATA
DEVOPS AND DATA ENGINEERING

- Install and configure etcd
- Install and configure pgBackRest
- Install PostgreSQL and patroni
- Configure patroni to bootstrap and clone from pgBackRest
- Start patroni
- profit

KAMEDATA
DEVOPS AND DATA ENGINEERING

Check the pgbackrest can access the repository

```
pgbackrest info
stanza: test
    status: ok
    cipher: none

    db (current)
        wal archive min/max (13): 000000010000000000000004/000000010000000000000004

        full backup: 20210705-145357F
            timestamp start/stop: 2021-07-05 14:53:57 / 2021-07-05 14:54:44
            wal start/stop: 000000010000000000000004 / 000000010000000000000004
            database size: 24.0MB, database backup size: 24.0MB
            repo1: backup set size: 2.9MB, backup size: 2.9MB
```

Configure the custom bootstrap script for patroni

KAMEDATA
DEVOPS AND DATA ENGINEERING

Create the file /etc/patroni/boot_pgbackrest.sh

```
#!/usr/bin/env bash

while getopts ":-:" optchar; do
  [[ "${optchar}" == "-" ]] || continue
  case "${OPTARG}" in
    datadir=* )
        DATA_DIR=${OPTARG#*=}
        ;;
    scope=* )
        SCOPE=${OPTARG#*=}
        ;;
  esac
done

/usr/bin/pgbackrest --stanza=$SCOPE --link-all restore
```

Make the file executable

Configure the patroni boostrap section

KAMEDATA
DEVOPS AND DATA ENGINEERING

```
bootstrap:
  .....
  postgresql:
      use_pg_rewind: true
      use_slots: false
      parameters:
        wal_level: 'replica'
        archive_mode: 'on'
        archive_command: 'pgbackrest --stanza=test archive-push %p'
        unix_socket_directories: '/var/run/postgresql/.'
      recovery_conf:
        recovery_target_timeline: latest
        restore_command: /usr/bin/pgbackrest --stanza=test archive-get %f "%p"
  method: pgbackrest
  pgbackrest:
    command: /etc/patroni/boot_pgbackrest.sh
    keep_existing_recovery_conf: False
    recovery_conf:
      recovery_target_timeline: latest
      restore_command: /usr/bin/pgbackrest --stanza=test archive-get %f "%p"
```

Add pgbackrest for creating replicas

KAMEDATA
DEVOPS AND DATA ENGINEERING

```
postgresql:
  create_replica_methods:
    - pgbackrest
  pgbackrest:
    command: /usr/bin/pgbackrest --stanza=test restore --delta --link-all
    keep_data: True
    no_params: True
  parameters:
    wal_level: 'replica'
    archive_mode: 'on'
    archive_command: 'pgbackrest --stanza=test archive-push %p'
    unix_socket_directories: '/var/run/postgresql/.'
  recovery_conf:
    recovery_target_timeline: latest
    restore_command: /usr/bin/pgbackrest --stanza=test archive-get %f "%p"
  .....
```

Create the empty PGDATA on the patroni machines

KAMEDATA
DEVOPS AND DATA ENGINEERING

```
mkdir -p /var/lib/pgsql/data/postgresql0
chown postgres:postgres /var/lib/pgsql/data/postgresql0
chmod 0700 /var/lib/pgsql/data/postgresql0
```

Start patroni and wait for the bootstrap to complete

KAMEDATA
DEVOPS AND DATA ENGINEERING

```
sudo systemctl start patroni
patronictl -c /etc/patroni/patroni.yml list
+ Cluster: test (initializing) ---+-------------------------------+----+-----------+
| Member     | Host       | Role    | State                         | TL | Lag in MB |
+------------+------------+---------+-------------------------------+----+-----------+
| patroni01 | patroni01 | Replica | running custom bootstrap script |    |   unknown |
+------------+------------+---------+-------------------------------+----+-----------+

patronictl -c /etc/patroni/patroni.yml list
+ Cluster: test (6981439838068958622) -----+----+-----------+
| Member     | Host       | Role   | State   | TL | Lag in MB |
+------------+------------+--------+---------+----+-----------+
| patroni01 | patroni01 | Leader | running |  2 |           |
+------------+------------+--------+---------+----+-----------+
```

Start patroni on the second node

KAMEDATA
DEVOPS AND DATA ENGINEERING

```
sudo systemctl start patroni

patronictl -c /etc/patroni/patroni.yml list
+ Cluster: test (6981439838068958622) ----------------+----+-----------+
| Member     | Host       | Role    | State            | TL | Lag in MB |
+------------+------------+---------+------------------+----+-----------+
| patroni01 | patroni01 | Leader  | running          |  2 |           |
| patroni02 | patroni02 | Replica | creating replica |    |   unknown |
+------------+------------+---------+------------------+----+-----------+

patronictl -c /etc/patroni/patroni.yml list
+ Cluster: test (6981439838068958622) ------+----+-----------+
| Member     | Host       | Role    | State   | TL | Lag in MB |
+------------+------------+---------+---------+----+-----------+
| patroni01 | patroni01 | Leader  | running |  2 |           |
| patroni02 | patroni02 | Replica | running |  2 |         0 |
+------------+------------+---------+---------+----+-----------+
```

I fight for the users

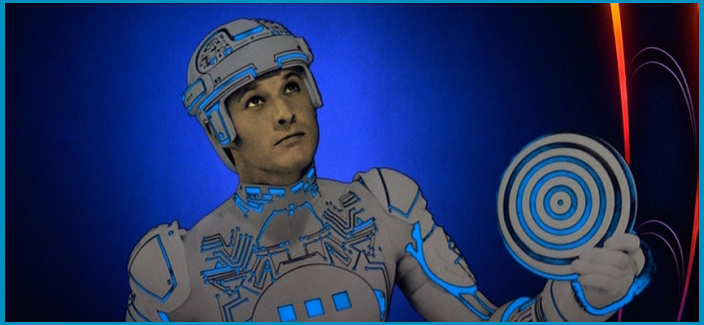


Image source

https://disney.fandom.com/wiki/Tron_(character)

Wrap up

KAMEDATA
DEVOPS AND DATA ENGINEERING

- Patroni and pgBackRest are amazing
- Patroni requires the DBA to change their point of view
- Patroni doesn't implement the DR
- but pgBackRest does it!
- This example is missing a lot of pieces (security, connection routing...)
- Using tools like Puppet or Ansible is a very,very,very,very,very good idea
- Always RTFM!

Thank you for listening!

KAMEDATA
DEVOPS AND DATA ENGINEERING

Any questions?

Protecting your data with Patroni and pgBackRest

PGDay Russia 2021

Federico Campoli

Somewhere in the time vortex