Patroni Administration

The Ultimate Cheat Sheet

bootvar.com

TABLE OF CONTENTS

Chapter One

Patroni & Patronictl

Overview

Patroni & Patronictl

In the world of database management, ensuring high availability is paramount, especially for critical systems like PostgreSQL. Patroni, a robust open-source solution, stands as a key player in orchestrating high-availability PostgreSQL clusters. At the heart of Patroni lies PatroniCTL, a powerful command-line tool designed to simplify and streamline the management of these clusters.

- Patroni is open source application which provides Postgres High Availability and Disaster recovery out of the box. It allows production grade systems to avoid single point of failures.
- It automates the failover process, seamlessly transitioning from a failed primary node to a standby node, minimizing downtime and ensuring continuous operations.
- Patroni's features for high availability and disaster recovery are readily available without the need for extensive configuration or customization.
- Tailored for deployment in production environments, Patroni is equipped to handle the demands and complexities of real-world, mission-critical systems.
- The application is designed to scale horizontally, accommodating the growth of workloads and supporting the expansion of PostgreSQL clusters.

Configuration

Patroni & Patronictl

When Patroni is installed on any node - patronictl command will also get installed with the same package. You can follow below instructions to install Patroni and configure patornictl command:

- Install Patroni and patronictl using the following pip command:

  `pip install patroni`
- The patronictl binary is installed at `/usr/local/bin/patronictl`. For more details on installation and configuration you can visit bootvar.
- Verify the installation by checking the version or help:

  `patronictl version`
- After configuring PostgreSQL and Patroni, locate the Patroni configuration file (e.g. `/etc/patroni/patroni.yml`).
- Use patronictl to check the status of the cluster and all nodes:

  `patronictl -c /etc/patroni/patroni.yml list`
- The output of the patronictl list command should display the status of the entire cluster, including all nodes.
- Ensure that the patronictl command is in the system's PATH. If not found, add the location to the PATH variable.
- Now for each patronictl command you need to give configuration file with -c parameter - instead you can add alias for this command as below

  `alias patronictl="patronictl -c /etc/patroni/patroni.yml"`
- For persistence of this command you can add this in your `.bashrc` or `.zshrc` or `.profile` file located at home location

Chapter Two

Informative Commands

patronictl version

Informative Commands

Version command can be used to get version information about patronictl binary, patroni binary and Postgres installations.

Usage

To get version of all components included in the cluster like Patroni, Postgres, patronictl.

When to Use

- When you want to check patronictl binary version
- When you want to get patroni version installed on each node
- When you want to get Postgres version installed in all nodes

Example usage:

Below command will give you patronictl binary version

```
patronictl -c /etc/patroni/patroni.yml version
```

To get versions of all components you can provide CLUSTER_NAME in above command

```
patronictl -c /etc/patroni/patroni.yml version pg_cluster
```

Output will look like below:

```
patronictl version 3.0.1
pgdb1: Patroni 3.0.1 PostgreSQL 14.7
pgdb2: Patroni 3.0.1 PostgreSQL 14.7
```

patronictl list

Informative Commands

The informative commands in patronictl are employed to gather details regarding the cluster, individual nodes, their statuses, and historical events within the cluster. Node, cluster status can be specifically examined using the list command.

Usage

The list command provides a comprehensive list of all nodes in the cluster, showcasing their respective statuses, along with specific details such as IP address, role, state, timeline (TL), and replication lag measured in megabytes.

When to Use

- When you want to assess the cluster's status.
- When you want to retrieve node information, distinguishing between leaders and replicas.
- When you want to examine replication lag in replicas.
- The output will also indicate a "Restart Required" column, signaling nodes that necessitate a restart.

Example usage:

```
patronictl -c /etc/patroni/patroni.yml list
```

CLUSTER_NAME can be passed to the command if you want to see details for particular cluster.

To get extended status `--extended` flag can be passed, similarly to watch the status continuously `--watch` can be passed.

patronictl topology

Informative Commands

Topology command gives cluster and node details in tree structure, similar to list command.

Usage

The topology command provides a comprehensive list of all nodes in the cluster, showcasing their respective statuses in tree structure format.

When to Use

When you want to check node status and their hierarchy

Example usage:

```
patronictl -c /etc/patroni/patroni.yml topology
```

To get extended status `--extended` flag can be passed, similarly to watch the status continuously `--watch` can be passed.

patronictl show-config

Informative Commands

Show-config command will give you configuration information which is used for kicking of the Postgres cluster including all the parameters used.

Usage

The `show-config` command will display comprehensive configuration details for the cluster, encompassing PostgreSQL parameters conveyed through the cluster configuration.

When to Use

When you need to inspect the configuration provided to the cluster via the configuration file, including PostgreSQL parameters specified within the config file.

Example usage:

```
patronictl -c /etc/patroni/patroni.yml show-config
```

patronictl history

Informative Commands

Patroni cluster goes through various transitions like switching roles, doing failovers/switchovers. This record is maintained in the cluster state.

Usage

The history command is used to check history of events occurred in the patroni cluster.

When to Use

- When you want to check history of events like failover/switchover
- When you want to check LSNs and their replication timestamps

Example usage:

```
patronictl -c /etc/patroni/patroni.yml history
```

Chapter Three

Cluster Management Commands

patronictl edit-config

Management Commands

To edit postgres configuration parameters you can use edit-config command. It will open configuration file in editor, make the required changes and Patroni will validate all parameters before saving configuration file.

Usage

The edit-config command opens the editor, enabling modifications to the configuration. Once saved, the changes will be applied.

When to Use

If you want to change some postgres parameter, add/remove pg_hba entries - you can use `patronictl edit-config` command.

Example usage:

```
patronictl -c /etc/patroni/patroni.yml edit-config
```

patronictl reload

Management Commands

This command will reload parameters from configuration file and takes required action like restart on cluster nodes.

Usage

To reload the configuration on disk we can use reload command to sync this with paroni cluster.

When to Use

If you have changed parameters in configuration file using `edit-config` you can use reload command for parameters to take effect

Example usage:

We need to pass CLUSTER_NAME to reload parameters for that particular cluster, in below example patroni_cluster is the CLUSTER_NAME.

```
patronictl -c /etc/patroni/patroni.yml reload patroni_cluster
```

patronictl pause

Management Commands

Patroni will stop managing postgres cluster and will turn on the maintenance mode. If you want to do some manual activities for maintenance you need to stop patroni from auto managing cluster.

Usage

To detach postgres instance from patroni utility we can use pause command.

When to Use

If you want to put cluster in maintenance mode and manage Postgres database manually for some time, you can use pause command so that Patroni will stop managing the cluster

Example usage:

Below command will pause the cluster handling using patroni utility

```
patronictl -c /etc/patroni/patroni.yml pause
```

patronictl resume

Management Commands

If PostgreSQL is not under Patroni management due to the execution of the pause command, the management can be reinstated by utilizing the resume command.

Usage

To attach postgres instance to patroni utility we can use resume command.

When to Use

If you want to turn off maintenance mode, you can use resume command and patroni will start managing the cluster

Example usage:

Below command will resume the cluster handling using patroni utility

```
patronictl -c /etc/patroni/patroni.yml resume
```

patronictl switchover

Management Commands

It involves designating a chosen replica as the master node, effectively diverting all traffic to the newly selected node. A scheduled switchover can also be executed at a specified time.

Usage

To make a switch between replica and master node.

When to Use

If you want to promote any replica to master node - you can use switchover command.

Example usage:

Below command will ask for replica node and time if we want to have scheduled switchover..

```
patronictl -c /etc/patroni/patroni.yml switchover
```

patronictl restart

Management Commands

It initiates the restart of either a single node within the PostgreSQL cluster or all nodes (the entire cluster). Patroni orchestrates a rolling restart for PostgreSQL across all nodes.

Usage

To restart the Patroni cluster and associated postgres on nodes.

When to Use

Sometimes you need to restart all nodes in the cluster without downtime, you can use this command for rolling restart.

Example usage:

Below command will restart whole cluster without any downtime:

```
patronictl -c /etc/patroni/patroni.yml restart <CLUSTER_NAME>
```

Below command will restart single node in cluster:

```
patronictl -c /etc/patroni/patroni.yml restart <CLUSTER_NAME> <NODE_NAME>
```

patronictl reinit

Management Commands

It will reinitialize node in the cluster. If you want to reinitialize particular replica or slave node you can reinitialize node using reinit command.

Usage

To reinitialize node from scratch reinit command can be used, this command will delete all the data on node and start with the fresh base.

When to Use

Sometimes you need to reinitialize single node in the cluster due to corruption, you can reinitialize node using reinit.

Example usage:

Below command will reinit replica node:

```
patronictl -c /etc/patroni/patroni.yml reinit <CLUSTER_NAME> <NODE_NAME>
```

Stay Informed from bootvar.com

Unlock exclusive content, updates, and insights. Join our community and receive the latest news directly to your inbox. Don't miss out – subscribe now!

SIGN UP FOR FREE