Two nodes PostgreSQL replication with high availability using Patroni

Intro

Setting up PostgreSQL replication with high availability using Patroni involves several components working together: Patroni as the orchestration tool, etcd/Consul/Zookeeper as the distributed consensus store, and HAProxy/pgbouncer for load balancing. Below are the detailed steps to set up Patroni for PostgreSQL replication.

Prerequisites

- ✓ **Servers** - At least three servers for etcd/consul and at least two servers for PostgreSQL
- ✓ **Dependencies -** Install required packages on each PostgreSQL server
  - *python3*
  - *python3-pip*
  - *PostgreSQL*
  - *Patroni*

Steps involved to complete the installation and configuration-

1. Install Dependencies
   a. Install PostgreSQL and related tools
      i. sudo apt-get update
      ii. sudo apt-get install -y postgresql postgresql-contrib
      iii. sudo apt-get install -y python3 python3-pip
   b. Install Patroni
      i. sudo pip3 install patroni[etcd]
   c. Install etcd (or Consul/Zookeeper) - For etcd, download and install binaries
      i. wget https://github.com/etcd-io/etcd/releases/download/v3.4.34/etcd-v3.4.34-linux-amd64.tar.gz
      ii. tar xvf etcd-v3.4.34-linux-amd64.tar.gz
      iii. sudo mv etcd-v3.4.34-linux-amd64/etcd* /usr/local/bin/
2. Configure etcd - On each etcd server
   a. Create an etcd configuration file (e.g., /etc/etcd/etcd.conf)

```
name: 'etcd1'
data-dir: '/var/lib/etcd'
initial-advertise-peer-urls: 'http://192.168.1.101:2380'
listen-peer-urls: 'http://192.168.1.101:2380'
listen-client-urls: 'http://192.168.1.101:2379'
advertise-client-urls: 'http://192.168.1.101:2379'
initial-cluster:
'etcd1=http://192.168.1.101:2380,etcd2=http://192.168.1.102:2380,etcd3=http://192.168.1.103:2380'
initial-cluster-state: 'new'
initial-cluster-token: 'etcd-cluster'
```

   b. Start etcd - *sudo etcd --config-file /etc/etcd/etcd.conf*

Two nodes PostgreSQL replication with high availability using Patroni

3. Configure Patroni - On each PostgreSQL server, create a Patroni configuration file (e.g., /etc/patroni.yml)

```
scope: postgres-ha
namespace: /service/
name: postgresql0
restapi:
  listen: 0.0.0.0:8008
  connect_address: 192.168.1.101:8008

etcd:
  host: 192.168.1.101:2379,192.168.1.102:2379,192.168.1.103:2379

bootstrap:
  dcs:
    ttl: 30
    loop_wait: 10
    retry_timeout: 10
    maximum_lag_on_failover: 1048576
    postgresql:
      use_pg_rewind: true
      parameters:
        max_connections: 100
        wal_level: replica
        archive_mode: "on"
        archive_command: 'cp %p /var/lib/postgresql/data/archive/%f'
        max_wal_senders: 5
        wal_keep_size: 64
        hot_standby: "on"

  initdb:
  - encoding: UTF8
  - data-checksums

  users:
    admin:
      password: admin_password
      options:
        - createrole
        - createdb

postgresql:
  listen: 0.0.0.0:5432
  connect_address: 192.168.1.101:5432
  data_dir: /var/lib/postgresql/data
  bin_dir: /usr/lib/postgresql/<version>/bin
  authentication:
    replication:
      username: replicator
      password: rep_password
    superuser:
      username: postgres
      password: postgres_password
    rewind:
      username: rewind_user
      password: rewind_password

tags:
  nofailover: false
  noloadbalance: false
  clonefrom: false
  nosync: false
```

Two nodes PostgreSQL replication with high availability using Patroni

4. Start Patroni - Start Patroni on each PostgreSQL server
    a. sudo patroni /etc/patroni.yml
5. Verify the Setup
    a. Check Patroni API
        i. Visit **http://192.168.1.101:8008/patroni** to see the status of the Patroni node
    b. Check Cluster Status
        i. patronictl -c /etc/patroni.yml list
6. Configure HAProxy for Load Balancing
    a. Install HAProxy
        i. sudo apt-get install -y haproxy
    b. Edit HAProxy Configuration (/etc/haproxy/haproxy.cfg)

```
frontend pgsql_frontend
  bind *:5000
  mode tcp
  default_backend pgsql_backend

backend pgsql_backend
  mode tcp
  option tcp-check
  default-server inter 3s fall 3 rise 2 on-marked-down shutdown-sessions
  server postgresql0 192.168.1.101:5432 maxconn 100 check port 8008
  server postgresql1 192.168.1.102:5432 maxconn 100 check port 8008
```

    c. Restart HAProxy - *sudo systemctl restart haproxy*