Проекты VK WorkSpace

Кластер PostgreSQL

Портал с документацией: https://biz.mail.ru/docs/on-premises/

Оглавление

Шаг 1. Подготовка нод (серверов) 3

Шаг 2. Подготовка кластера etcd 3

Шаг 3. Установка PostgreSQL 7

Шаг 4. Настройка PostgreSQL 7

Действия для узла postgres-server1.your_domain 7

Действия для узла `postgres-server2.your_domain` и `postgres-server3.your_domain`: 8

Шаг 5. Установка Patroni 8

Шаг 7. Подготовка кластера PostgreSQL+Patroni 14

Шаг 8. Подготовка PGBouncer (опционально) 15

Для корректной работы системы требуется PostgreSQL версии 11–15. В статье описана установка PostgreSQL 13 для ОС Ubuntu Linux 20.04 и 22.04. Также вы можете ознакомиться с руководством в официальной документации PostgreSQL.

Установка состоит из 10 этапов:

1. Подготовка нод (серверов)
2. Подготовка кластера etcd
3. Установка PostgreSQL
4. Настройка PostgreSQL
5. Установка Patroni
6. Настройка Patroni
7. Подготовка кластера PostgreSQL+Patroni
8. Подготовка PGBouncer (опционально)

Шаг 1. Подготовка нод (серверов)

Создайте три ноды (сервера) с последовательно пронумерованными именами хостов.

Минимальное количество серверов для организации кластера — три.

В этом примере используется три узла со следующими hostname и IP-адресами:

- postgres-server1.your_domain, 192.168.1.1
- postgres-server2.your_domain, 192.168.1.2
- postgres-server3.your_domain, 192.168.1.3

Создайте необходимые сопоставления имён хостов в DNS. Если такой возможности нет, внесите нужные записи в `/etc/hosts`.

Шаг 2. Подготовка кластера etcd

1. Установите `etcd` на все узлы:

```
sudo apt-get install etcd -y
```

2. Остановите `etcd` на всех узлах:

```
sudo systemctl stop etcd
```

3. Удалите каталог данных:

```
sudo rm -rf /var/lib/etcd/*
```

4. Переместите конфигурационный файл по умолчанию:

```
sudo mv /etc/default/etcd{,.original}
```

5. Создайте и откройте для редактирования новый конфигурационный файл:

```
sudo nano /etc/default/etcd
```

6. Добавьте пример конфигураций в файл для узла `postgres-server1.your_domain`:

```
ETCD_NAME="postgres-server1"
ETCD_DATA_DIR="/var/lib/etcd/default"
ETCD_HEARTBEAT_INTERVAL="1000"
ETCD_ELECTION_TIMEOUT="5000"
ETCD_LISTEN_PEER_URLS="http://192.168.1.1:2380"
ETCD_LISTEN_CLIENT_URLS="http://192.168.1.1:2379,http://localhost:2379"
```

```
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://192.168.1.1:2380"
ETCD_INITIAL_CLUSTER="postgres-server1=http://192.168.1.1:2380,postgres-server2=http://
192.168.1.2:2380,postgres-server3=http://192.168.1.3:2380"
ETCD_INITIAL_CLUSTER_STATE="new"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-postgres-cluster"
ETCD_ADVERTISE_CLIENT_URLS="http://192.168.1.1:2379"
ETCD_ENABLE_V2="true"
ETCD_INITIAL_ELECTION_TICK_ADVANCE="false"
```

Пример конфигураций с включением TLS/SSL для узла `postgres-server1.your_domain`

```
ETCD_NAME="postgres-server1"
ETCD_DATA_DIR="/var/lib/etcd/default"
ETCD_HEARTBEAT_INTERVAL="1000"
ETCD_ELECTION_TIMEOUT="5000"
ETCD_LISTEN_PEER_URLS="https://192.168.1.1:2380"
ETCD_LISTEN_CLIENT_URLS="https://192.168.1.1:2379,https://localhost:2379"
ETCD_INITIAL_ADVERTISE_PEER_URLS="https://postgres-server1.your_domain:2380"
ETCD_INITIAL_CLUSTER="postgres-server1=https://postgres-server1.your_domain:2380,postgres-
server2=https://postgres-server2.your_domain:2380,postgres-server3=https://postgres-
server3.your_domain:2380"
ETCD_INITIAL_CLUSTER_STATE="new"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-postgres-cluster"
ETCD_ADVERTISE_CLIENT_URLS="https://postgres-server1.your_domain:2379"
ETCD_ENABLE_V2="true"
ETCD_CERT_FILE="/path/to/public.crt"
ETCD_KEY_FILE="/path/to/private.key"
ETCD_CLIENT_CERT_AUTH="true"
ETCD_TRUSTED_CA_FILE="/path/to/certCA.pem"
ETCD_PEER_CERT_FILE="/path/to/public.crt"
ETCD_PEER_KEY_FILE="/path/to/private.key"
ETCD_PEER_CLIENT_CERT_AUTH="true"
ETCD_PEER_TRUSTED_CA_FILE="/path/to/certCA.pem"
ETCD_INITIAL_ELECTION_TICK_ADVANCE="false"
```

7. Добавьте пример конфигураций в файл для узла `postgres-server2.your_domain`:

```
ETCD_NAME="postgres-server2"
ETCD_DATA_DIR="/var/lib/etcd/default"
ETCD_HEARTBEAT_INTERVAL="1000"
ETCD_ELECTION_TIMEOUT="5000"
ETCD_LISTEN_PEER_URLS="http://192.168.1.2:2380"
ETCD_LISTEN_CLIENT_URLS="http://192.168.1.2:2379,http://127.0.0.1:2379"
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://192.168.1.2:2380"
ETCD_INITIAL_CLUSTER="postgres-server1=http://192.168.1.1:2380,postgres-server2=http://
192.168.1.2:2380,postgres-server3=http://192.168.1.3:2380"
ETCD_INITIAL_CLUSTER_STATE="new"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-postgres-cluster"
ETCD_ADVERTISE_CLIENT_URLS="http://192.168.1.2:2379"
ETCD_ENABLE_V2="true"
ETCD_INITIAL_ELECTION_TICK_ADVANCE="false"
```

Пример конфигураций с включением TLS/SSL для узла `postgres-server2.your_domain`:

```
ETCD_NAME="postgres-server2"
ETCD_DATA_DIR="/var/lib/etcd/default"
ETCD_HEARTBEAT_INTERVAL="1000"
ETCD_ELECTION_TIMEOUT="5000"
ETCD_LISTEN_PEER_URLS="https://192.168.1.2:2380"
ETCD_LISTEN_CLIENT_URLS="https://192.168.1.2:2379,https://localhost:2379"
```

```
ETCD_INITIAL_ADVERTISE_PEER_URLS="https://postgres-server2.your_domain:2380"
ETCD_INITIAL_CLUSTER="postgres-server1=https://postgres-server1.your_domain:2380,postgres-server2=https://postgres-server2.your_domain:2380,postgres-server3=https://postgres-server3.your_domain:2380"
ETCD_INITIAL_CLUSTER_STATE="new"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-postgres-cluster"
ETCD_ADVERTISE_CLIENT_URLS="https://postgres-server2.your_domain:2379"
ETCD_ENABLE_V2="true"
ETCD_CERT_FILE="/path/to/public.crt"
ETCD_KEY_FILE="/path/to/private.key"
ETCD_CLIENT_CERT_AUTH="true"
ETCD_TRUSTED_CA_FILE="/path/to/certCA.pem"
ETCD_PEER_CERT_FILE="/path/to/public.crt"
ETCD_PEER_KEY_FILE="/path/to/private.key"
ETCD_PEER_CLIENT_CERT_AUTH="true"
ETCD_PEER_TRUSTED_CA_FILE="/path/to/certCA.pem"
ETCD_INITIAL_ELECTION_TICK_ADVANCE="false"
```

8. Добавьте пример конфигураций в файл для узла `postgres-server3.your_domain`:

```
ETCD_NAME="postgres-server3"
ETCD_DATA_DIR="/var/lib/etcd/default"
ETCD_HEARTBEAT_INTERVAL="1000"
ETCD_ELECTION_TIMEOUT="5000"
ETCD_LISTEN_PEER_URLS="http://192.168.1.3:2380"
ETCD_LISTEN_CLIENT_URLS="http://192.168.1.3:2379,http://localhost:2379"
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://192.168.1.3:2380"
ETCD_INITIAL_CLUSTER="postgres-server1=http://192.168.1.1:2380,postgres-server2=http://192.168.1.2:2380,postgres-server3=http://192.168.1.3:2380"
ETCD_INITIAL_CLUSTER_STATE="new"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-postgres-cluster"
ETCD_ADVERTISE_CLIENT_URLS="http://192.168.1.3:2379"
ETCD_ENABLE_V2="true"
ETCD_INITIAL_ELECTION_TICK_ADVANCE="false"
```

Пример конфигураций с включением TLS/SSL для узла `postgres-server3.your_domain`

```
ETCD_NAME="postgres-server3"
ETCD_DATA_DIR="/var/lib/etcd/default"
ETCD_HEARTBEAT_INTERVAL="1000"
ETCD_ELECTION_TIMEOUT="5000"
ETCD_LISTEN_PEER_URLS="https://192.168.1.3:2380"
ETCD_LISTEN_CLIENT_URLS="https://192.168.1.3:2379,https://localhost:2379"
ETCD_INITIAL_ADVERTISE_PEER_URLS="https://postgres-server3.your_domain:2380"
ETCD_INITIAL_CLUSTER="postgres-server1=https://postgres-server1.your_domain:2380,postgres-server2=https://postgres-server2.your_domain:2380,postgres-server3=https://postgres-server3.your_domain:2380"
ETCD_INITIAL_CLUSTER_STATE="new"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-postgres-cluster"
ETCD_ADVERTISE_CLIENT_URLS="https://postgres-server3.your_domain:2379"
ETCD_ENABLE_V2="true"
ETCD_CERT_FILE="/path/to/public.crt"
ETCD_KEY_FILE="/path/to/private.key"
ETCD_CLIENT_CERT_AUTH="true"
ETCD_TRUSTED_CA_FILE="/path/to/certCA.pem"
ETCD_PEER_CERT_FILE="/path/to/public.crt"
ETCD_PEER_KEY_FILE="/path/to/private.key"
ETCD_PEER_CLIENT_CERT_AUTH="true"
ETCD_PEER_TRUSTED_CA_FILE="/path/to/certCA.pem"
ETCD_INITIAL_ELECTION_TICK_ADVANCE="false"
```

Рассмотрим введённые параметры:

- ETCD_NAME — имя этого узла кластера. Должно быть уникально в кластере;
- ETCD_LISTEN_CLIENT_URLS — точка подключения для клиентов кластера
- ETCD_ADVERTISE_CLIENT_URLS — список URL-адресов, по которым его могут найти остальные узлы кластера;
- ETCD_LISTEN_PEER_URLS — точка подключения для остальных узлов кластера
- ETCD_INITIAL_ADVERTISE_PEER_URLS — начальный список URL-адресов, по которым его могут найти остальные узлы кластера
- ETCD_INITIAL_CLUSTER_TOKEN — токен кластера, должен совпадать на всех узлах кластера;
- ETCD_INITIAL_CLUSTER — список узлов кластера на момент запуска;
- ETCD_INITIAL_CLUSTER_STATE — может принимать два значения: new и existing;
- ETCD_DATA_DIR — расположение каталога данных кластера;
- ETCD_ELECTION_TIMEOUT — время в миллисекундах, которое проходит между последним принятым оповещением от лидера кластера, до попытки захватить роль лидера на ведомом узле;
- ETCD_HEARTBEAT_INTERVAL — время в миллисекундах, между рассылками лидером оповещений о том, что он всё ещё лидер
- ETCD_CERT_FILE — путь до файла сертификата сервера;
- ETCD_KEY_FILE — путь до файла закрытого ключа;
- ETCD_TRUSTED_CA_FILE — путь до файла корневого CA;
- ETCD_CLIENT_CERT_AUTH — может принимать два значения: true и false;
- ETCD_PEER_CERT_FILE — путь до файла сертификата сервера;
- ETCD_PEER_TRUSTED_CA_FILE — путь до файла корневого CA;
- ETCD_PEER_CLIENT_CERT_AUTH — может принимать два значения: true и false.

9. Перезапустите `etcd` на всех узлах:

```
sudo systemctl restart etcd
```

10. Проверьте состояние кластера

Для кластера без TLS:

```
sudo etcdctl cluster-health
```

Для кластера с TLS:

```
sudo etcdctl -C https://postgres-server1.your_domain:2379 --key-file /path/to/
private.key --cert-file /path/to/public.crt --ca-file /path/to/certCA.pem cluster-health
```

Шаг 3. Установка PostgreSQL

1. Для установки PostgreSQL добавьте официальный репозиторий `postgresql`:

```
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-
pgdg main" > /etc/apt/sources.list.d/pgdg.list'
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
```

2. Обновите кеш пакетов, выполнив команду:

```
sudo apt update
```

3. Установите PostgreSQL на все узлы:

```
sudo apt install postgresql-13 -y
```

Шаг 4. Настройка PostgreSQL

Для пароля разрешается применять следующие символы:

- заглавные латинские буквы: от A до Z
- строчные латинские буквы: от a до z
- цифры от 0 до 9
- символы: -_.

Зарезервированные (недопустимые) символы: ! * ' ( ) ; : @ & = + $, / ? % # [ ]

Действия для узла postgres-server1.your_domain

1. Создайте новую роль `{{ shell_product_name }}` с паролем `SecretPassword`:

```
sudo -u postgres psql -c \
"CREATE ROLE {{ shell_product_name }} WITH login password 'SecretPassword';"
```

2. Создайте необходимые базы данных, например `{{ shell_product_name }}db` с владельцем `{{ shell_product_name }}`:

```
sudo -u postgres psql -c \
"CREATE DATABASE {{ shell_product_name }}v;"
```

3. Добавьте необходимые расширения для базы данных `{{ shell_product_name }}`:

```
sudo -u postgres psql -d {{ shell_product_name }}db -c \
"CREATE EXTENSION \"uuid-ossp\"; CREATE EXTENSION pg_trgm;"
```

4. Создайте новую роль `replicator` с паролем `ReplicatorPassword` для работы с репликами. Должно совпадать с настройками `Patroni` из блока `postgresql - authentication - replication` и списком разрешённых хостов postgresql в файле `pg_hba.conf`:

```
sudo -u postgres psql -c \
"CREATE USER replicator WITH REPLICATION ENCRYPTED PASSWORD 'ReplicatorPassword';"
```

5. Установите пароль для пользователя postgres:

```
sudo -u postgres psql -c "ALTER USER postgres PASSWORD 'PostgresPassword';"
```

6. Остановите PostgreSQL:

```
systemctl stop postgresql
```

Действия для узла `postgres-server2.your_domain` и `postgres-server3.your_domain`:

1. Остановите PostgreSQL:

```
systemctl stop postgresql
```

2. Удалите каталог данных на нодах `postgres-server2.your_domain` и `postgres-server3.your_domain`:

```
rm -rf /var/lib/postgresql/13/main
```

Шаг 5. Установка Patroni

1. Создайте файл настроек:

```
sudo nano /etc/patroni/config.yml
```

2. В созданный файл `/etc/patroni/config.yml` нужно поместить пример начальной конфигурации, изменив ip-адреса на свои на каждом узле кластера. Обратите внимание на комментарии в данном файле.

   Пример начальной конфигурации:

```
scope: postgres-cluster # одинаковое значение на всех узлах
name: postgresql-server1 # разное значение на всех узлах
```

```
namespace: /service/ # одинаковое значение на всех узлах

restapi:
listen: postgres-server1.your_domain:8008 # адрес узла, на котором находится этот файл
connect_address: postgres-server1.your_domain:8008 # адрес узла, на котором находится этот
файл

etcd:
hosts: postgres-server1.your_domain:2379,postgres-server2.your_domain:2379,postgres-
server3.your_domain:2379 # список всех узлов, на которых установлен etcd

bootstrap:
method: initdb
dcs:
    ttl: 30
    loop_wait: 10
    retry_timeout: 10
    maximum_lag_on_failover: 1048576
    master_start_timeout: 300
    synchronous_mode: false
    synchronous_mode_strict: false
    synchronous_node_count: 1
    postgresql:
    use_pg_rewind: true
    use_slots: true
    parameters:
        max_connections: 2000
        superuser_reserved_connections: 5
        max_locks_per_transaction: 64
        max_prepared_transactions: 0
        huge_pages: try
        shared_buffers: 512MB
        work_mem: 128MB
        maintenance_work_mem: 256MB
        effective_cache_size: 4GB
        checkpoint_timeout: 15min
        checkpoint_completion_target: 0.9
        wal_compression: on
        min_wal_size: 2GB
        max_wal_size: 4GB
        wal_buffers: 32MB
        default_statistics_target: 1000
        seq_page_cost: 1
        random_page_cost: 4
        effective_io_concurrency: 2
        synchronous_commit: on
        autovacuum: on
        autovacuum_max_workers: 5
        autovacuum_vacuum_scale_factor: 0.01
        autovacuum_analyze_scale_factor: 0.02
        autovacuum_vacuum_cost_limit: 200
        autovacuum_vacuum_cost_delay: 20
        autovacuum_naptime: 1s
        max_files_per_process: 4096
        archive_mode: on
        archive_timeout: 1800s
        archive_command: cd .
        wal_level: replica
        wal_keep_segments: 130
        max_wal_senders: 10
        max_replication_slots: 10
        hot_standby: on
        hot_standby_feedback: True
```

```
        wal_log_hints: on
        shared_preload_libraries: pg_stat_statements,auto_explain
        pg_stat_statements.max: 10000
        pg_stat_statements.track: all
        pg_stat_statements.save: off
        auto_explain.log_min_duration: 10s
        auto_explain.log_analyze: true
        auto_explain.log_buffers: true
        auto_explain.log_timing: false
        auto_explain.log_triggers: true
        auto_explain.log_verbose: true
        auto_explain.log_nested_statements: true
        standard_conforming_strings: true
        track_io_timing: on
        log_lock_waits: on
        log_temp_files: 3
        track_activities: on
        track_counts: on
        track_functions: all
        log_checkpoints: on
        logging_collector: on
        log_truncate_on_rotation: on
        log_rotation_age: 1d
        log_rotation_size: 0
        log_line_prefix: '%t [%p-%l] %r %q%u@%d '
        log_filename: 'postgresql-%a.log'
        log_directory: /var/log/postgresql

initdb: # List options to be passed on to initdb
    - encoding: UTF8
    - locale: en_US.UTF-8
    - data-checksums

pg_hba: # должен содержать адреса ВСЕХ машин, используемых в кластере
    - local all postgres peer
    - local all all peer
    - host all all 0.0.0.0/0 md5
    - host replication replicator localhost trust
    - host replication replicator 192.168.1.1/32 md5
    - host replication replicator 192.168.1.2/32 md5
    - host replication replicator 192.168.1.3/32 md5

postgresql:
listen: 192.168.1.1,127.0.0.1:5432 # адрес узла, на котором находится этот файл
connect_address: 192.168.1.1:5432 # адрес узла, на котором находится этот файл
use_unix_socket: true
data_dir: /var/lib/postgresql/13/main # каталог данных
bin_dir: /usr/lib/postgresql/13/bin
config_dir: /etc/postgresql/13/main
pgpass: /var/lib/postgresql/.pgpass_patroni
authentication:
    replication:
    username: replicator
    password: ReplicatorPassword
    superuser:
    username: postgres
    password: PostgresPassword
parameters:
    unix_socket_directories: /var/run/postgresql
pg_hba: # должен содержать адреса ВСЕХ машин, используемых в кластере
    - local all postgres peer
    - local all all peer
    - host all all 0.0.0.0/0 md5
```

```
    - host replication replicator localhost trust
    - host replication replicator 192.168.1.1/32 md5
    - host replication replicator 192.168.1.2/32 md5
    - host replication replicator 192.168.1.3/32 md5

remove_data_directory_on_rewind_failure: false
remove_data_directory_on_diverged_timelines: false

create_replica_methods:
    - basebackup
basebackup:
    max-rate: '100M'
    checkpoint: 'fast'

watchdog:
mode: off # Allowed values: off, automatic, required
device: /dev/watchdog
safety_margin: 5

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

Пример начальной конфигурации для включения поддержки TLS/SSL в Patroni.

```
scope: postgres-cluster # одинаковое значение на всех узлах
name: postgresql-server1 # разное значение на всех узлах
namespace: /service/ # одинаковое значение на всех узлах

restapi:
listen: postgres-server1.your_domain:8008 # адрес узла, на котором находится этот файл
connect_address: postgres-server1.your_domain:8008 # адрес узла, на котором находится этот
файл
cafile: /path/to/pgCA.pem
certfile: /path/to/pg.crt # путь до файла сертификата сервера
keyfile: /path/to/pg.key # путь до файла закрытого ключа
verify_client: required # путь до файла корневого CA

etcd:
protocol: https
cert: /path/to/public.crt # путь до файла сертификата сервера
key: /path/to/private.key # путь до файла закрытого ключа
cacert: /path/to/certCA.pem # путь до файла корневого CA
hosts: postgres-server1.your_domain:2379,postgres-server2.your_domain:2379,postgres-
server3.your_domain:2379 # список всех узлов, на которых установлен etcd

ctl:
insecure: false # Allow connections to SSL sites without certs
certfile: /path/to/pg.crt # путь до файла сертификата сервера
keyfile: /path/to/pg.key # путь до файла закрытого ключа
cacert: /path/to/pgCA.pem # путь до файла корневого CA

bootstrap:
method: initdb
dcs:
    ttl: 30
    loop_wait: 10
    retry_timeout: 10
    maximum_lag_on_failover: 1048576
    master_start_timeout: 300
```

```
    synchronous_mode: false
    synchronous_mode_strict: false
    synchronous_node_count: 1
    postgresql:
    use_pg_rewind: true
    use_slots: true
    parameters:
        max_connections: 2000
        superuser_reserved_connections: 5
        max_locks_per_transaction: 64
        max_prepared_transactions: 0
        huge_pages: try
        shared_buffers: 512MB
        work_mem: 128MB
        maintenance_work_mem: 256MB
        effective_cache_size: 4GB
        checkpoint_timeout: 15min
        checkpoint_completion_target: 0.9
        wal_compression: on
        min_wal_size: 2GB
        max_wal_size: 4GB
        wal_buffers: 32MB
        default_statistics_target: 1000
        seq_page_cost: 1
        random_page_cost: 4
        effective_io_concurrency: 2
        synchronous_commit: on
        autovacuum: on
        autovacuum_max_workers: 5
        autovacuum_vacuum_scale_factor: 0.01
        autovacuum_analyze_scale_factor: 0.02
        autovacuum_vacuum_cost_limit: 200
        autovacuum_vacuum_cost_delay: 20
        autovacuum_naptime: 1s
        max_files_per_process: 4096
        archive_mode: on
        archive_timeout: 1800s
        archive_command: cd .
        wal_level: replica
        wal_keep_segments: 130
        max_wal_senders: 10
        max_replication_slots: 10
        hot_standby: on
        hot_standby_feedback: True
        wal_log_hints: on
        shared_preload_libraries: pg_stat_statements,auto_explain
        pg_stat_statements.max: 10000
        pg_stat_statements.track: all
        pg_stat_statements.save: off
        auto_explain.log_min_duration: 10s
        auto_explain.log_analyze: true
        auto_explain.log_buffers: true
        auto_explain.log_timing: false
        auto_explain.log_triggers: true
        auto_explain.log_verbose: true
        auto_explain.log_nested_statements: true
        standard_conforming_strings: true
        track_io_timing: on
        log_lock_waits: on
        log_temp_files: 3
        track_activities: on
        track_counts: on
        track_functions: all
```

```
        log_checkpoints: on
        logging_collector: on
        log_truncate_on_rotation: on
        log_rotation_age: 1d
        log_rotation_size: 0
        log_line_prefix: '%t [%p-%l] %r %q%u@%d '
        log_filename: 'postgresql-%a.log'
        log_directory: /var/log/postgresql
        ssl: on
        ssl_ca_file: '/path/to/pgCA.pem'
        ssl_cert_file: '/path/to/pg.crt'
        ssl_key_file: '/path/to/pg.key'

initdb: # List options to be passed on to initdb
    - encoding: UTF8
    - locale: en_US.UTF-8
    - data-checksums

pg_hba: # должен содержать адреса ВСЕХ машин, используемых в кластере
    - local all postgres peer
    - local all all peer
    - hostssl all all 0.0.0.0/0 md5
    - hostssl replication replicator localhost trust
    - hostssl replication replicator 192.168.1.1/32 md5
    - hostssl replication replicator 192.168.1.2/32 md5
    - hostssl replication replicator 192.168.1.3/32 md5

postgresql:
listen: 192.168.1.1,127.0.0.1:5432 # адрес узла, на котором находится этот файл
connect_address: 192.168.1.1:5432 # адрес узла, на котором находится этот файл
use_unix_socket: true
data_dir: /var/lib/postgresql/13/main # каталог данных
bin_dir: /usr/lib/postgresql/13/bin
config_dir: /etc/postgresql/13/main
pgpass: /var/lib/postgresql/.pgpass_patroni
authentication:
    replication:
    username: replicator
    password: ReplicatorPassword
    sslcert: /path/to/pg.crt # путь до файла сертификата сервера
    sslkey: /path/to/pg.key # путь до файла закрытого ключа
    sslrootcert: /path/to/pgCA.pem # путь до файла корневого CA
    superuser:
    username: postgres
    password: PostgresPassword
    sslcert: /path/to/pg.crt # путь до файла сертификата сервера
    sslkey: /path/to/pg.key # путь до файла закрытого ключа
    sslrootcert: /path/to/pgCA.pem # путь до файла корневого CA
parameters:
    unix_socket_directories: /var/run/postgresql
pg_hba: # должен содержать адреса ВСЕХ машин, используемых в кластере
    - local all postgres peer
    - local all all peer
    - hostssl all all 0.0.0.0/0 md5
    - hostssl replication replicator localhost trust
    - hostssl replication replicator 192.168.1.1/32 md5
    - hostssl replication replicator 192.168.1.2/32 md5
    - hostssl replication replicator 192.168.1.3/32 md5

remove_data_directory_on_rewind_failure: false
remove_data_directory_on_diverged_timelines: false

create_replica_methods:
```

```
    - basebackup
basebackup:
    max-rate: '100M'
    checkpoint: 'fast'

watchdog:
mode: off # Allowed values: off, automatic, required
device: /dev/watchdog
safety_margin: 5

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

Сделайте пользователя postgres владельцем файла закрытого ключа `pg.key`:

Файл закрытого ключа `pg.key` указанный в секции `postgresql` должен иметь разрешения u=rw (0600) или меньше, если он принадлежит пользователю базы данных postgres, или разрешения u=rw, g=r (0640) или меньше, если он принадлежит пользователю root.

```
sudo chown postgres:postgres -R /path/to/pg.key
sudo chmod 600 /path/to/pg.key
```

Подробнее о настройке TLS/SSL в Patroni читайте в официальной документации Patroni.

3. Сделайте пользователя postgres владельцем каталога настроек:

```
sudo chown postgres:postgres -R /etc/patroni
sudo chmod 700 /etc/patroni
```

Шаг 7. Подготовка кластера PostgreSQL+Patroni

1. Запустите службу Patroni на узле `postgres-server1.your_domain`, а затем на узлах `postgres-server2.your_domain` и `postgres-server3.your_domain`:

```
sudo systemctl enable --now patroni.service
```

2. Проверьте состояние кластера:

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

Шаг 8. Подготовка PGBouncer (опционально)

Следуйте инструкции по подготовке PGBouncer

Технический писатель: Белова Ирина

6 мая 2026 г.