Сервер-Гексагон

Установка postgresql:

apt-get install -y postgresql16 postgresql16-server postgresql16-contrib

Развертываем БД:

/etc/init.d/postgresql initdb

Автозагрузка:

systemctl enable --now postgresql

Включаем прослушивание всех адресов:

nano /var/lib/pgsql/data/postgresql.conf

изображение

Перезагружаем:

systemctl restart postgresql

Проверяем:

ss -tlpn | grep postgres

Заходим в БД под рутом:

psql -U postgres

Создаем базы данных "prod","test","dev":

CREATE DATABASE prod;

CREATE DATABASE test;

CREATE DATABASE dev;

Создаем пользователей "produser","testuser","devuser":

CREATE USER produser WITH PASSWORD 'P@ssw0rd';

CREATE USER testuser WITH PASSWORD 'P@ssw0rd';

CREATE USER devuser WITH PASSWORD 'P@ssw0rd';

Назначаем на БД владельца:

GRANT ALL PRIVILEGES ON DATABASE prod to produser;

GRANT ALL PRIVILEGES ON DATABASE test to testuser;

GRANT ALL PRIVILEGES ON DATABASE dev to devuser;

Заполняем БД с помощью pgbench:

pgqbench -U postgres -i prod

pgqbench -U postgres -i test

pgqbench -U postgres -i dev

Проверяем

изображение

Настройка аутентификации для удаленного доступа:

nano /var/lib/pgsql/data/pg\_hba.conf

изображение

Перезапускаем:

systemctl restart postgresql

Сервер-Барагаон

Установка:

apt-get install -y postgresql16 postgresql16-server postgresql16-contrib

Репликация

Сервер-Гексагон

Конфиг

nano /var/lib/pgsql/data/postgresql.conf

wal\_level = replica

max\_wal\_senders = 2

max\_replication\_slots = 2

hot\_standby = on

hot\_standby\_feedback = on

wal\_level указывает, сколько информации записывается в WAL (журнал операций, который используется для репликации); max\_wal\_senders — количество планируемых слейвов; max\_replication\_slots — максимальное число слотов репликации; hot\_standby — определяет, можно ли подключаться к postgresql для выполнения запросов в процессе восстановления; hot\_standby\_feedback — определяет, будет ли сервер slave сообщать мастеру о запросах, которые он выполняет.

systemctl restart postgresql

Сервер-Барагаон

Останавливаем:

systemctl stop postgresql

Удаляем каталог:

rm -rf /var/lib/pgsql/data/\*

Запускаем репликацию (может длиться несколько десятков минут):

pg\_basebackup -h 10.0.10.2 -U postgres -D /var/lib/pgsql/data --wal-method=stream --write-recovery-conf

image

Назначаем владельца:

chown -R postgres:postgres /var/lib/pgsql/data/

Запускаем:

systemctl start postgresql

Сервер-Гексагон

Создаем тест:

psql -U postgres

CREATE DATABASE testik;

Сервер-Барагаон

HAPROXY

Сыч-Гексагон (test, not prod)

Установка haproxy:

apt-get install haproxy

Автозагрузка:

systemctl enable --now haproxy

Конфиг /etc/haproxy/haproxy.cfg:

listen stats

bind 0.0.0.0:8989

mode http

stats enable

stats uri /haproxy\_stats

stats realm HAProxy\ Statistics

stats auth admin:toor

stats admin if TRUE

frontend postgre

bind 0.0.0.0:5432

default\_backend my-web

backend postgre

balance first

server srv-hq 10.0.10.2:5432 check

server srv-br 10.0.20.2:5432 check