PostgreSQL

Table of Contents

Deploying PostgreSQL

Introduction

PostgreSQL

Deploy Postgres

PostgreSQL is a powerful, open-source object-relational database system that has earned a strong reputation for reliability, feature robustness, and performance.

In many cases, databases are external entities. Many cloud providers now provide PostgreSQL-as-a-service or project may opt for standing up a dedicated instance or instances in a highly available configuration. It may be therefore preferable to host PostgresSQL in a virtual machine locally to replicate such environments.

These steps detail deploying PostgreSQL using CentOS Stream 8.

Installation

Install PostgreSQL 12 from the module stream:

```
$ sudo dnf install @postgresql:12 -y
```

Database configuration

First step is to initialise PostgrSQL:

```
$ sudo /usr/bin/postgresql-setup --initdb
```

Enable and start service

```
$ sudo systemctl enable postgresql.service --now
```

Open access

If `firewalld` is being used, add the service:

```
$ sudo firewall-cmd --add-service=postgresql --permanent
$ sudo firewall-cmd --reload
```

Access PostgreSQL:

```
$ sudo su - postgres
$ psql
```

Create a user and database:

```
CREATE USER dbuser WITH PASSWORD 'changeme';
ALTER ROLE dbuser SET client_encoding TO 'utf8';
ALTER ROLE dbuser SET default_transaction_isolation TO 'read committed';
ALTER ROLE dbuser SET timezone TO 'UTC';
CREATE DATABASE exampleforyou_db;
GRANT ALL PRIVILEGES ON DATABASE exampleforyou_db TO dbuser;
```

List and quit:

```
$ \l
$ \q
$ exit
```

Ensure it is configured to listen on the IP Address of the host.

Edit `postgresql.conf`:

```
$ sudo vi /var/lib/pgsql/data/postgresql.conf
```

```
listen_addresses = '192.168.0.70'
```

And update the configuration to allow any host on the same subnet to access the database.

Edit `pg_hba.conf`:

```
$ sudo vi /var/lib/pgsql/data/pg_hba.conf
```

Original:

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD
local   all             all                                     peer
host    all             all             127.0.0.1/32            ident
host    all             all             ::1/128                 ident
local   replication     all                                     peer
host    replication     all             127.0.0.1/32            ident
host    replication     all             ::1/128                 ident
```

Change methods to `md5` for `localhost` and if needed the subnet of an IP range, in this case for running PostgreSQL in a Virtual Machine on `192.168.0.111`, adding `/24` for that subnet:

```
# TYPE  DATABASE        USER            ADDRESS                 METHOD
local   all             all                                     peer
host    all             all             127.0.0.1/32            md5
host    all             all             192.168.0.1/24          md5
host    all             all             192.168.122.1/24        md5
host    all             all             ::1/128                 md5
local   replication     all                                     peer
host    replication     all             127.0.0.1/32            ident
host    replication     all             ::1/128                 ident
```

Restart PostgreSQL service:

```
$ systemctl restart postgresql
```

pgAdmin

pgAdmin is a browser based management console for PostgreSQL.

Ensure EPEL is available:

```
# dnf install epel-release
```

Add the pgadmin repository:

```
# dnf install -y https://ftp.postgresql.org/pub/pgadmin/pgadmin4/yum/pgadmin4-redhat-repo-1-1.noarch.rpm
```

Install `pgadmin4`:

```
# dnf install pgadmin4 -y
```

It uses Apache, make sure `mod_ssl` is installed so SSL redirect works:

```
# dnf install mod_ssl
```

Run the initial setup:

```
# /usr/pgadmin4/bin/setup-web.sh
```

Access pgAdmin using a browser, for example: https://localhost/pgadmin4