It is appreciated that problems/questions regarding this tutorial are asked on answers.ros.org. Don't forget to include in your question the link to this page, versions of your OS & ROS, and also add appropriate tags.

1. Installation of a PostgreSQL server

Description: A step-by-step "cheat-sheet" for installing a PostgreSQL server on a Ubuntu machine. Does not cover any advanced installation procedures, just a very simple installation with no additional options. After installing the server, you can set up multiple databases, restore them from backup files, etc.

Tutorial Level: BEGINNER

Contents

2. Overview

This tutorial is a quick "cheat-sheet" for installing a PostgreSQL server on a Ubuntu machine. It is **not** meant to replace the detailed instructions that are provided along with PostgreSQL. However, we have found that some steps can be unclear in the instructions, and a step-by-step procedure can sometimes help. It also does **not** cover any advanced installation options.

This tutorial works for PostgreSQL 8.4. Here you can find the complete PostgreSQL 8.4 manual.

This tutorial only covers the installation of the database server itself. It does not cover the creation of individual databases once the server is up and running; an example of creating a database and populating it from a backup file can be found in the household objects database installation tutorial.

3. Install the PostgreSQL server

```
sudo apt-get install postgresql
```

- the installation process sets up data and configuration directories by itself, and tells you where they are. You can write them down, but there is another way of finding out where they are (below)
- Postgres server should start automatically after install process
- `ps auxw | grep postgresql` shows server is running as well as data and config directories

In the config directory of PostgreSQL you will find two very important files which we'll be using later:

- `pg_hba.conf`
- `postgresql.conf`

4. Create a PostgreSQL server account

General: PostgreSQL accounts are separate from OS accounts. During the installation process, both a PostgreSQL and an OS account will be created, both called **postgres**. The postgres account is a superuser, meaning it can add other PostgreSQL accounts.

What we need to do next is to login to PostgreSQL using the `postgres` account in order to create a new account for us to use.

4.1 Log in to the PostgreSQL server using root access

Become root, then su into the `postgres` OS account. That's it; if you are logged in to OS as the `postgres` account you can also log into PostgreSQL as the same account without a password.

Start the `psql` front end, then skip to the 'Add a new database user' section:

```
psql
```

4.2 Log in to the PostgreSQL server using sudo access

If you have sudo but not root access you should still be able to switch the the postgres user using sudo su and then open psql.

```
sudo su - postgres
psql
```

If that doesn't work see the next section, if it does then jump to the 'Add a new database user' section

4.3 Log in to the PostgreSQL server without root access

If you have sudo privileges, but not root access, and can not become the `postgres` OS account, you must edit `pg_hba.conf` to allow username `posgres` to be logged in by anybody without password

Find the lines:

```
# Database administrative login by UNIX sockets
local   all         postgres                          ident sameuser
```

Change `ident sameuser` to `trust`. **Remember to change back when done with this tutorial!!!**

After modifying `pg_hba.conf`, have the Postgresql server reload it by issuing a HUP signal:

```
ps auxw | grep postgresql
sudo kill -HUP xxxx
```

You should now be able to start the Postgresql front-end (`psql`) as the superuser postgres:

```
psql --username postgres
```

4.4 Add a new database user

Once you are logged into `psql` as `postgres`, you can add a user. It is recommended to start by adding a user WITHOUT superuser privileges, but with CREATEROLE and CREATEDB privilages. This is achieved by the following command:

```
CREATE ROLE willow LOGIN CREATEDB CREATEROLE PASSWORD 'willow';
```

You can then quit psql (\q).

If you have modified `pg_hba.conf` to let the `postgres` user connect without a password, be sure to revert the change.

5. Allow connections through TCP/IP

We will now enable TCP/IP connections to the PostgreSQL server.

Edit `pg_hba.conf`. At the end, where the list of authentication methods is, add the following lines. This allows anybody to connect through TCP/IP and access all databases, as long as they supply a right username and password. The password is md5 encrypted

```
# Anybody through TCP/IP with password
host    all         all         0.0.0.0   0.0.0.0     md5
```

Now you must let the server know it's supposed to accept TCP/IP connections from anybody. Edit the `postgresql.conf` file and set

```
listen_addresses = '*'
```

You will then need to restart the server altogether. The simplest way of doing this is to restart the machine.

5.1 Localhost connections

If you do not want to enable TCP/IP connections, one alternative is to allow your brand new user to connect from the local machine, WITH a password. Note that this means only code running on the same machine as the server will be able to connect.

```
# willow user can connect locally with password
local   all         willow                            md5
```

Remember that after modifying `pg_hba.conf` you must have the Postgresql server reload it by issuing a HUP signal.

```
ps auxw | grep postgresql
sudo kill -HUP xxxx
```

To check that it is working, try logging into `psql` on localhost as the new user:

```
psql --username willow --password --dbname postgres
```

6. PGAdmin3

PGAdmin is a graphical front-end to a database server. To install it use:

```
sudo apt-get install pgadmin3
```

The run it using:

```
pgadmin3
```

If you have set your server to accept TCP/IP connections, you can run pgadmin3 on any machine that can talk to your server machine over the network. If your server only accepts localhost connections, you'll have to run pgadmin3 on the same machine on which you just installed the server.

In `pgadmin3`, go through the following steps:

- `File` -> `Add Server...`
- type in the address of the machine that you installed the server on as well as the new postgres username you just created.
- connect to the server. If all goes well, you should see a list of databases running on the server (by default, there will be a single database available called `postgres`).

That's it, the server is up and running! You can now create your own databases on the server. Here is a tutorial for creating and restoring the household_objects database on your server, using a Willow Garage database backup file.

Except
where Wiki: sql_database/Tutorials/Installing a PostgreSQL Server (last edited 2012-11-20 16:17:51 by AnthonySoroka)