IRRD4

**Cambiar versión de Python a 3.6**

sudo add-apt-repository ppa:jonathonf/python-3.6

sudo apt-get update

sudo apt-get install python3.6

sudo update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.5 1

sudo update-alternatives --install /usr/bin/python3 python3 /usr/bin/python3.6 2

sudo update-alternatives --config python3

python3 --version

**IRRd requires:**

* Linux or MacOS. Other platforms are untested, but may work.
* Python 3.6, with *pip* and *virtualenv* installed. At this point, 3.7 has had limited testing.
* A recent version of PostgreSQL. Versions 9.6 and 10.5 have been extensively tested.
* At least 8GB RAM (mostly for initial large imports).

A number of other Python packages are required. However, those are automatically installed in the installation process.

**PostgreSQL configuration**

Descargar postgressql 10.10

<https://www.enterprisedb.com/downloads/postgres-postgresql-downloads>

CREATE DATABASE irrd;

CREATE ROLE irrd WITH LOGIN ENCRYPTED PASSWORD 'irrd';

GRANT ALL PRIVILEGES ON DATABASE irrd TO irrd;

\c irrd

#### Entrar a postgres desde la consola:

#### **sudo -u postgres psql**

Crear una -bd:

#### **createdb irrd -h localhost -U irrd**

Comprobar que el usuario irrd esté creado

***select \* from "pg\_user";***

Acceso a la BD:

#### **psql -d irrd -h localhost -U irrd**

Eliminar base de datos:

DROP DATABASE nombre\_db

Conectar con Bases de Datos

***\c irrd;***

Crear la extension pgcrypto:

CREATE EXTENSION IF NOT EXISTS pgcrypto;

List databases:

**\l**

List tables in database:

**\d**

Quit

**\q**

cd /etc/postgresql/9.5/main/

vim postgresql.conf

**A few PostgreSQL settings need to be changed from their default:**

* **random\_page\_cost** should be set to 1.0. Otherwise, PostgreSQL is too reluctant to use the efficient indexes in the IRRd database, and will opt for slow full table scans.
* **work\_mem** should be set to 50MB (52428800)to allow PostgreSQL to do in-memory sorting and merging.
* **shared\_buffers** should be set to around 1/8th - 1/4th of the system memory to benefit from caching, with a max of a few GB.
* **max\_connections** may need to be increased from 100. IRRd also has an internal maximum number of connections, which is set toserver.whois.max\_connections plus the number of configured sources, times 3. Generally, each open whois connection will result in one PostgreSQL connection, as will each running process for a change submitted by a user, each running mirror import, and each running mirror export.
* **log\_min\_duration\_statement** can be useful to set to 0 initially, to log all SQL queries to aid in debugging any issues. Note that initial imports of data produce significant logs if all queres are logged - importing 1GB of data can result in 2-3GB of logs.

The database will be in the order of three times as large as the size of the RPSL text imported.

**Important:**

The PostgreSQL database is the only source of IRRd’s data. This means backups of the database should be run regularly. It is also possible to restore data from recent exports, but changes made since the most recent export will be lost.

sudo apt install python3-apt

cd /usr/lib/python3/dist-packages

sudo cp apt\_pkg.cpython-35-x86\_64-linux-gnu.so apt\_pkg.so

**Installing IRRd**

Create the virtualenv with a command like this:

virtualenv -p python3 /home/irrd/irrd-venv

To run commands inside the virtualenv, use either of:

/home/irrd/irrd-venv/bin/<command>

*# or:*

*# Persists. Leave the venv with `deactivate`*

source /home/irrd/irrd-venv/bin/activate

<command>

sudo apt-get install build-essential python3 python-dev python3-dev

sudo apt-get install python3.5-gdbm

pip install --no-cache-dir ujson

To install the latest version of IRRd inside the virtualenv, use pip3:

/home/irrd/irrd-venv/bin/pip3 install irrd

Creating a configuration file

/etc/irrd.yaml

Adding a new empty source

sources:

NEW-SOURCE:

authoritative: true

keep\_journal: true

This sample shows most configuration options:

postgresql://irrd:irrd@localhost:5432/irrd

**irrd**:

**database\_url**: 'postgresql://localhost:5432/irrd'

**access\_lists**:

**http\_database\_status**:

- '::/32'

- '127.0.0.1'

**generic\_nrtm\_access**:

- '192.0.2.0/24'

**server**:

**http**:

**access\_list**: http\_database\_status

**interface**: '::0'

**port**: 8080

**whois**:

**interface**: '::0'

**max\_connections**: 50

**port**: 8043

**auth**:

**gnupg\_keyring**: /home/irrd/gnupg-keyring/

**override\_password**: {hash}

**email**:

**footer**: 'email footer'

**from**: example@example.com

**smtp**: localhost

**notification\_header**: |

This is to notify you of changes in the {sources\_str} database or object authorisation failures.

You may receive this message because you are listed in

the notify attribute on the changed object(s), or because

you are listed in the mnt-nfy or upd-to attribute on a maintainer

of the object(s).

**log**:

**logfile\_path**: /var/log/irrd/irrd.log

**level**: DEBUG

**sources\_default**:

- AUTHDATABASE

- MIRROR-SECOND

- MIRROR-FIRST

**sources**:

**AUTHDATABASE**:

*# Authoritative database, allows local changes, full export every 2h*

**authoritative**: true

**keep\_journal**: true

**export\_destination**: /var/ftp/

**export\_timer**: 7200

**nrtm\_access\_list**: generic\_nrtm\_access

**MIRROR-FIRST**:

*# Run a full import at first, then periodic NRTM updates.*

**authoritative**: false

**keep\_journal**: true

**import\_serial\_source**: 'ftp://ftp.example.net/MIRROR-FIRST.CURRENTSERIAL'

**import\_source**: 'ftp://ftp.example.net/mirror-first.db.gz'

**nrtm\_host**: rr.ntt.net

**nrtm\_port**: 43

**object\_class\_filter**:

- as-set

- aut-num

- filter-set

- inet-rtr

- key-cert

- mntner

- peering-set

- route

- route6

- route-set

- rtr-set

**MIRROR-SECOND**:

*# Every hour, a new full import will be done.*

**authoritative**: false

**import\_source**:

- 'ftp://ftp.example.net/mirror-second.db.as-set.gz'

- 'ftp://ftp.example.net/mirror-second.db.aut-num.gz'

- 'ftp://ftp.example.net/mirror-second.db.filter-set.gz'

- 'ftp://ftp.example.net/mirror-second.db.route-set.gz'

- 'ftp://ftp.example.net/mirror-second.db.route.gz'

- 'ftp://ftp.example.net/mirror-second.db.route6.gz'

- 'ftp://ftp.example.net/mirror-second.db.route-set.gz'

**import\_timer**: 3600

**Configuration options:**

<https://irrd4.readthedocs.io/en/latest/admins/configuration.html>

**The general plan for switching over to a new IRRd v4 instance would be:**

* Block update emails.
* Ensure an NRTM update has run so that the instances are in sync (it may be worthwhile to lower import\_timer)
* Remove the mirror configuration from the new IRRd 4 instance for any authoritative sources.
* Set the authoritative sources to authoritative: true in the config.
* Redirect queries to the new instance.
* Redirect update emails to the new instance.
* Ensure published exports are now taken from the new instance

**Note:**

During an initial import of many large sources at the same time, IRRd’s memory use may reach 3-4GB. During this import, query performance may be reduced. This may take around 30-45 minutes.

**Creating tables**

/home/irrd/irrd-venv/bin/irrd\_database\_upgrade

**Running as a non-privileged user**

It is recommended to run IRRd as a non-privileged user. This user needs read access to:

* the virtualenv
* the configuration file
* sources.{{name}}.import\_source (if this is a local file)
* sources.{{name}}.import\_serial\_source (if this is a local file)

The user also needs write access to access to:

* auth.gnupg\_keyring
* sources.{name}.export\_destination
* log.logfile\_path. As IRRd creates log.logfile\_path itself, it needs write access to the directory this file is in

**Starting IRRd**

IRRd runs as a Twisted process, and can be started with:

/home/irrd/irrd-venv/bin/twistd --uid=irrd --pidfile=/var/irrd.pid irrd

Useful options (to be placed before irrd):

* --uid=<user> makes the process run as a non-privileged user, after binding to TCP ports.
* -n makes the process run in the foreground. If log.logfile\_path is not set, this also shows all log output in the terminal.
* --pidfile=<path> has Twisted store the pidfile in a specific path. By default, the path is twistd.pid in the current directory.

Logrotate configuration

The following logrotate configuration can be used for IRRd:

/home/irrd/server.log {

missingok

daily

compress

delaycompress

dateext

rotate 35

olddir /home/irrd/logs

postrotate

systemctl reload irrd.service > /dev/null 2>&1 || true

endscript

}

This assumes the log.logfile\_path setting is set to /home/irrd/server.log. This file should be created in the path /etc/logrotate.d/irrd with permissions 0644.

**Systemd configuration**

The following configuration can be used to run IRRd under systemd, using setcap, to be created in /lib/systemd/system/irrd.service:

[Unit]

Description=IRRD4 Service

Wants=basic.target

After=basic.target network.target

[Service]

Type=simple

WorkingDirectory=/home/irrd

User=irrd

Group=irrd

PIDFile=/home/irrd/irrd.pid

ExecStart=/home/irrd/irrd-venv/bin/twistd -n --pidfile=/home/irrd/irrd.pid irrd

Restart=on-failure

ExecReload=/bin/kill -HUP $MAINPID

[Install]

WantedBy=multi-user.target

Then, IRRd can be started under systemd with:

systemctl daemon-reload

systemctl enable irrd

systemctl start irrd

**Processing email changes**

To process incoming requested changes by email, configure a mailserver to deliver the email to the irrd\_submit\_email command.

When using the virtualenv as set up above, the full path is:

/home/irrd/irrd-venv/bin/irrd\_submit\_email --irrd\_pidfile /home/irrd/irrd.pid

A --config parameter can be passed to set a different configuration file path. Results of the request are sent to the sender of the request, and [any relevant notifications are also sent](https://irrd4.readthedocs.io/en/latest/users/database-changes.html).

The --irrd\_pidfile parameter is required and should be set to the pidfile of the running IRRd instance. This is needed to signal the IRRd instance to update preloaded data. If the file does not exist, or the instance is not running, e.g. during a quick restart, no signal is sent, as the preloaded data will be updated when IRRd starts.

**Note:**

As a separate script, *irrd\_submit\_email* **always acts on the current configuration file** - not on the configuration that IRRd started with.