Guía de publicación del fondo de mapa OpenStreetMap, a través de Web Map Service

Elaborado por:
Davis Mendoza Paco
davis.men.pa@gmail.com
Sylvain Lesage
severo@rednegra.net
Reynaldo Condori
reymundito@gmail.com

Fecha de creación: *06/02/2014*
Descripción: *En esta guía se explica cómo instalar y configurar un servidor para la publicación del fondo de mapa de OpenStreetMap a través de Web Map Service, sobre una plataforma* ***Gnu/Linux-CentOS****, utilizando tecnologías de software libre como: apache, mapserver, postgres, postgis, imposm3.*
Fuente: *GeoBolivia - Infraestructura de Datos Espaciales del Estado Plurinacional de Bolivia.*
Idioma: *Español*
Cobertura: *La Paz - Bolivia*
Palabras claves: *Publicación, OpenStreetMap, Servicio de mapas, WMS, OGC, mapserver, postgres, postgis, imposm3.*

Control de versiones

Fecha	Autor/Modificado por	Versión	Descripción
07/02/2014	Davis Mendoza	3.0	Cambios realizados: Sistema Operativo utilizado **CentOS 6.4**.
06/02/2014	Davis Mendoza	2.0	Cambios realizados: Sistema Operativo utilizado **Debian-7 Wheezy**. Instalación y configuración de la base de datos PostgreSQL 9.1, imposm-3, lenguaje Go, basemaps anterior (mapserver-utils).
01/04/2012	Sylvain Lesage Reynaldo Condori	1.0	Primera versión del documento. Sistema Operativo utilizado **Debian-6 Squeeze**. Instalación y configuración de la base de datos PostgreSQL 8.4, imposm-2, lenguaje Python, mapserver-utils.

Índice de contenidos

1 Introducción

La presente guía describe el proceso de instalación de un servidor dedicado a la generación de un fondo de mapas utilizando los datos de OpenStreetMap http://www.openstreetmap.org, y su publicación mediante el servicio web WMS. Al fin del proceso de instalación, tendremos:

- Un servidor de base de datos PostGreSQL/PostGIS, conteniendo una base de datos geográfica con los datos de OpenStreetMap
- Un servidor de mapas MapServer, que provee, vía el servidor Apache, un servicio WMS (http://direcciónde IP/cgi-bin/mapserv), para acceder de manera remota al mapa de OpenStreetMap

El fondo de mapa generado puede ser utilizado de manera remota (vía Internet, o en una red local) utilizando cualquier tipo de cliente WMS desktop: ArcGis, gvSig, QuantumGis, etc. web: OpenLayers, GeoNetwork, MapFishApp, etc. En caso de publicación en la web, es muy importante definir la región que será disponible en el flujo WMS. En efecto, hay pocos proveedores libres de flujos WMS en el mundo, y si se publica el fondo WMS para todo el planeta, existe un gran riesgo que el servidor sea saturado (a nivel del ancho de banda y a nivel del trabajo del procesador) por las numerosas conexiones. La solución, en el caso de Bolivia, es de proveer un fondo solo para Bolivia y sus alrededores. Para lograr este objetivo, los pasos son los siguientes:

- Importación de los datos de OpenStreetMap
  - Descarga del archivo de datos de OpenStreetMap desde un repositorio en la Internet (los datos de OpenStreetMap son disponibles de libre acceso para la descarga).
  - Instalación de PostGreSQL, el servidor de base de datos.
  - Instalación de imposm, la herramienta de importación. creación de una base de datos, que llamaremos imposm, con el soporte geográfico (PostGIS)
  - Importación del archivo de datos de OpenStreetMap dentro de la base de datos.
- Generación del fondo de mapa
  - Instalación de MapServer, el servidor de mapas
  - Publicación del archivo osm-default.map mediante el servicio Apache

El proceso de publicación del fondo de mapa esta descrito en el esquema siguiente: Todas las herramientas a utilizar son software libre, y solo se hace referencia a la instalación sobre una plataforma Gnu/Linux. Las instrucciones son similares para otras versiones de CentOS. La instalación de paquetes y las direcciones de los archivos pueden ser diferentes para otras distribuciones. No se describe la instalación sobre una plataforma Windows.

Nota La guía hace referencia a la versión alfa de imposm3, la cual es una versión inestable, y no es muy seguro utilizarlo en ambientes de producción. Todavía se aguarda a que se eliminen los errores o a la puesta en práctica completa de toda su funcionalidad, pero satisface la mayoría de las funcionalidades de imposm2.

2 Instalación y configuración de PostgreSql

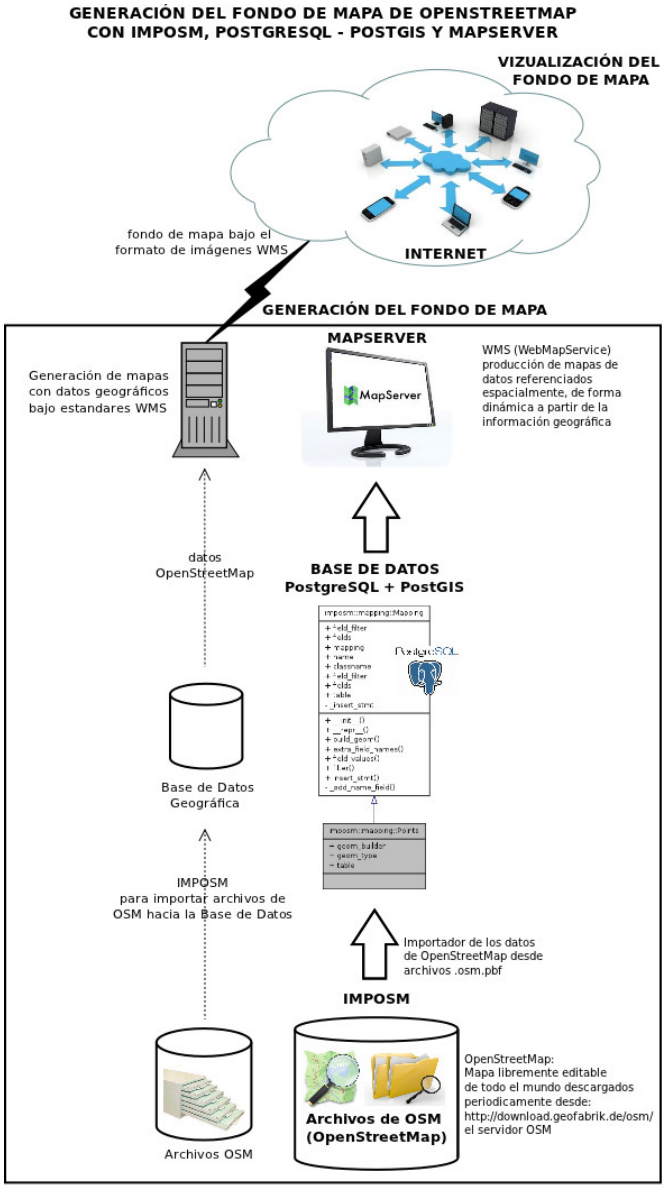
2.1 Instalación de PostgreSql y Postgis

Para la instalación debemos tomar en cuenta la versión 1.5 de PostGIS y la versión 9.1 de PostGreSQL. Es necesario excluir los paquetes propios de postgreSQL, esto es importante para conseguir el repositorio funcionando.

```
nano /etc/yum.repos.d/CentOS-Base.repo
```

En este caso exclude=postgres* y exclude=geos* en Base y Updates

```
[base]
name=CentOS-$releasever - Base
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=os
#baseurl=http://mirror.centos.org/centos/$releasever/os/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
exclude=postgres*
exclude=geos*
```




```
#released updates
[updates]
name=CentOS-$releasever - Updates
mirrorlist=http://mirrorlist.centos.org/?release=$releasever&arch=$basearch&repo=updates
#baseurl=http://mirror.centos.org/centos/$releasever/updates/$basearch/
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
exclude=postgres*
exclude=geos*
```

Ahora descargar el paquete "pgdg" desde los repositorios de redhat, e instalar.

```
mkdir /opt/library && cd /opt/library
wget http://yum.pgrpms.org/9.1/redhat/rhel-6-x86_64/pgdg-centos91-9.1-4.noarch.rpm
rpm -ivh pgdg-centos91-9.1-4.noarch.rpm
```

Instalar postgres y postgis.

```
yum install postgresql91-server postgresql91-contrib postgresql91 postgresql91-devel postgresql91-docs postgis91
```

Iniciando los servicios de postgres

```
service postgresql-9.1 initdb
```

```
service postgresql-9.1 start
```

Iniciando el servicio de postgres automáticamente, luego de reiniciar el servidor:

```
service chkconfig --level 235 postgresql-9.1 on
```

2.2 Configuración de PostgreSQL

Creamos un usuario con derechos de administración.

```
su postgres -
createuser -P -s -e nombreusuario
```

La ejecución de este comando debe retornar un mensaje como el que aparece a continuación.

```
Ingrese la contrasena para el nuevo rol:  .....(colocamos una contrasena)
Ingrésela nuevamente: .....(repetimos la contrasena)
----> nos debería dar lo siguiente:
  CREATE ROLE nombreusuario PASSWORD 'md51512c7191c2fe77ffad84afcfad7ef2f' SUPERUSER CREATEDB CREATEROLE INHERIT LOGIN;
```

2.3 Acceso al servidor PostGreSql

2.3.1 IP y puerto

Asumiendo que la dirección IP del servidor es '192.168.0.1', añadir a la lista de direcciones junto con 'localhost' para poder acceder a la base de datos local o remotamente.

```
nano /var/lib/pgsql/9.1/data/postgresql.conf
```

En este archivo buscar la sección 'listen_addresses', la configuración debe ser parecido

```
listen_addresses = 'localhost, 192.168.0.1'
port = 5432                    # (change requires restart)
```

2.3.2 Derechos de acceso

Cambiamos los derechos de acceso al servidor PostGreSQL para permitir el acceso del usuario nombreusuario desde otra maquina, por ejemplo '192.168.0.2', y también desde el propio servidor '127.0.0.1'.

```
nano /var/lib/pgsql/9.1/data/pg_hba.conf
```

La configuración debe ser parecido

```
# Database administrative login by UNIX sockets
  local    all     postgres              ident
# TYPE  DATABASE    USER       CIDR-ADDRESS        METHOD
# "local" is for Unix domain socket connections only
  local    all     all                   md5
# IPv4 local connections:
  host      all     all        127.0.0.1/32    md5
  host      all     nombreusuario   127.0.0.1/32    md5
  host      all     nombreusuario   192.168.0.2/32   md5
# IPv6 local connections:
  host    all     all       ::1/128       md5
```

Para que se tomen en cuenta los cambios, recargamos la configuración en el servidor postgresql:

```
service postgresql reload
```

Detalle: según el archivo de configuración, existen dos formas de tomar en cuenta las modificaciones: cuando se configura el archivo "postgresql.conf", se tiene que reiniciar el servidor

```
service postgresql restart
```

cuando se configura el archivo "pg_hba.conf", solo hay que recargar la configuración

```
service postgresql reload
```

Para conectarse al servidor PostGreSQL desde el propio servidor (192.168.0.1):

```
psql -U nombreusuario -W -h 127.0.0.1 -p5432 postgres
```

Para conectarse al servidor PostGreSQL desde la maquina del usuario (192.168.0.2):

```
psql -U nombreusuario -W -h 192.168.0.1 -p5432 postgres
```

Para salir del postgresql

```
\q
```

OJO: en el caso de que el servidor no permita el ingreso a la base de datos con el usuario creado verificar las reglas de un eventual firewall.

3 Instalación de Imposm-3

Imposm, permite importar los archivos .osm, o .osm.pbf, dentro de una base de datos PostGreSQL. La versión 3 de Imposm fue reescrito completamente utilizando el lenguaje Go, y es liberado como código abierto bajo la licencia Apache 2.0.

3.1 Dependencias y Librerías y Adicionales

3.1.1 Dependencias CentOS

Instalando el sistema de control de versiones git, mercurial y librerías adicionales.

```
yum install git mercurial snappy geos boost-devel gcc gcc-c++ flex
```

3.1.2 Base de Datos LevelDB

LevelDB es una base de datos de código abierto basada en pares clave-valor con almacenamiento en disco.LevelDB no es una base de datos SQL al igual que otros almacenamientos NoSQL y Dbm, no posee un modelo de datos relacional, no soporta sentencias SQL y no soporta índices. Para poder instalar, descargar el código fuente desde el repositorio de Google Code

```
cd /opt/library
wget https://leveldb.googlecode.com/files/leveldb-1.15.0.tar.gz
```

Descomprimir e ingresar a la carpeta leveldb-x.x.x

```
tar -xvf leveldb-1.15.0.tar.gz && cd leveldb-1.15.0/
```

Compilando el programa y copiando los binarios.

```
make
cp -r include/leveldb /usr/include/
cp libleveldb.* /usr/lib/
ldconfig
```

3.1.3 Lenguaje Go v1.2

Descargar y descomprimir la versión 1.2 del lenguaje para 64 bits, desde el repositorio de Google Code

```
cd /opt
wget https://go.googlecode.com/files/go1.2.linux-amd64.tar.gz
tar -xvf go1.2.linux-amd64.tar.gz
```

Instalar el lenguaje, para ello ingresar a la carpeta “go/src” y ejecutar el script “all.bash”

```
cd /opt/go/src
./all.bash
```

La ejecución de este comando debe retornar un mensaje como el que aparece a continuación.

```
ALL TESTS PASSED
---
Installed Go for linux/amd64 in /opt/go
Installed commands in /opt/go/bin
*** You need to add /opt/go/bin to your PATH.
```

Adicionar la carpeta “/opt/go/bin” a las variables de entorno de CentOS, para ello crear el archivo "go.sh" dentro de "/etc/profile.d".

```
nano /etc/profile.d/go.sh
chmod 755 /etc/profile.d/go.sh
```

El contenido del archivo es:

```
#!/bin/bash
GOROOT=/opt/go
GOPATH=/root
PATH=$GOROOT/bin:$PATH

export PATH GOPATH GOROOT
export CLASSPATH=.
```

Ejecutar el archivo creado:

```
source /etc/profile.d/go.sh
```

3.1.4 Instalación de Levigo

levigo es un contenedor para LevelDB. Configuración de las variables de entorno para levigo "CGO_CFLAGS, CGO_LDFLAGS" que apunten al directorio donde se encuentra instalado levelDB.

```
CGO_CFLAGS="-I/opt/leveldb-1.15.0/include" CGO_LDFLAGS="-L/opt/leveldb-1.15.0"
go get github.com/jmhodges/levigo
```

3.1.5 Instalación de geos

GEOS (Geometry Engine - Open Source) es un port a C++ de JTS Topology Suite (JTS). Incluye las OpenGIS Simple Features para funciones de predicado espacial SQL y operadores espaciales, así como funciones topológicas específicas de JTS.

```
cd /opt/library
wget http://download.osgeo.org/geos/geos-3.4.2.tar.bz2
tar -xvf geos-3.4.2.tar.bz2 && cd geos-3.4.2
./configure && make && make install
```

3.1.6 Instalación go-sqlite3

Controlador sqlite3 ajustado a la interfaz de la base de datos.

```
go get github.com/mattn/go-sqlite3
```

3.1.7 Instalación Protocol Buffers

Buffers Protocolo es una forma de codificar datos estructurados en un formato extensible.

```
cd /opt/library
wget https://protobuf.googlecode.com/files/protobuf-2.5.0.tar.gz
tar -xvf protobuf-2.5.0.tar.gz && cd protobuf-2.5.0/
./configure
make
make install
```

Este paquete proporciona soporte Go, en la forma de una biblioteca y un protocolo de plugin de compilador, para los búferes de protocolo de Google. (RPC no es compatible.)

```
go get code.google.com/p/goprotobuf/{proto,protoc-gen-go}
```

3.1.8 Instalación pq

Controlador postgres para el lenguaje Go.

```
go get github.com/lib/pq
```

3.2 Compilación e Instalación

Creando un espacio de trabajo, y actualizar la variable GOPATH en base a la carpeta actual, utilizando para ello el comando pwd.

```
mkdir /opt/imposm && cd /opt/imposm
export GOPATH=$PWD
```

Obteniendo el código fuente de Imposm 3 desde el repositorio de Google Code

```
git␣clone␣https://github.com/omniscale/imposm3␣src/imposm3
go␣get␣imposm3
```

Instalando imposm3

```
go install imposm3
```

Después de la instalación de imposm3, se genera un archivo binario en la ruta "$GOROOT/bin". Si por alguna razón no se genero el binario en la ruta $GOROOT, verificar la ruta $GOPATH

```
echo $GOPATH
```

El resultado del comando es

```
/opt/imposm
```

Copiar el contenido de GOPATH a GOROOT

```
cp -r /opt/imposm/* /opt/go/
```

Una vez copiado los archivos, ya se puede utilizar imposm3.

```
imposm3 version
```

4 Creación del Usuario y la Base de Datos

Creamos el usuario imposm, dedicado a la futura base de datos que contendra la información de OpenStreetMap. Este usuario será dueño de está base, y no tendrá acceso a las otras bases.

Para crear el usuario, desde el servidor:

```
createuser -h 127.0.0.1 -U nombreusuario --no-superuser --no-createrole --createdb -P imposm
```

```
Ingrese la contrasena para el nuevo rol: ..... (agregamos una contrasena)
Ingrésela nuevamente: ..... (repetimos la contrasena)
```

Modificamos el pg_hba.conf para dar acceso al usuario "imposm" los derechos de conexión en local

```
nano /etc/postgresql/9.1/main/pg_hba.conf
```

La configuración debería ser de la siguiente manera:

```
# IPv4 local connections:
host    all     imposm    127.0.0.1/32        md5
hostssl   all     imposm    127.0.0.1/32        md5
```

Recargamos la configuración

```
service postgresql reload
```

Creamos la base de datos imposm, con el dueño imposm

```
createdb -h 127.0.0.1 -U nombreusuario -E UTF8 -O nombreusuario -T template0 imposm
```

Importamos el soporte geográfico, Postgis, en la base imposm, conectándonos con el nuevo usuario imposm.

OJO: Solo para estas acciones colocar al usuario “IMPOSM”, con privilegios de superusuario una vez ejecutadas colocarlo como estaba. Actualizando los privilegios del usuario imposm como superusuario

```
su
su postgres -
psql
ALTER USER imposm WITH SUPERUSER;
```

Quitando los privilegios de superusuario.

```
ALTER USER imposm WITH NOSUPERUSER;
```

Importando los scripts SQL de imposm en la base de datos creada.

```
createlang -h 127.0.0.1 -U imposm plpgsql imposm\
psql -h 127.0.0.1 -U imposm -d imposm -f /usr/pgsql-9.1/share/contrib/postgis-1.5/postgis.sql
psql -h 127.0.0.1 -U imposm -d imposm -f /usr/pgsql-9.1/share/contrib/postgis-1.5/spatial_ref_sys.sql
```

5 Importación de los datos de OpenStreetMap

Imposm funciona en dos fases:

- Análisis del archivo de datos, con exportación en una carpeta (opción –read)
- Importación de los datos analizados en la base de datos, con un prefijo sobre las tablas (opción –write)

5.1 Descarga del archivo de datos de OpenStreetMap

El sitio http://geofabrik.de publica un espejo de los datos de OpenStreetMap para su descarga. Se pueden bajar los datos a nivel mundial, regional o nacional. Por ejemplo, para Bolivia:

- bolivia.osm.pbf
- south-america-latest.osm.pbf

Podríamos utilizar el archivo de datos a nivel de Bolivia, pero es más interesante utilizar el archivo de datos a nivel de America del Sur, lo que permite tener los datos alrededor de las fronteras, para que Bolivia no parezca flotando en el aire, en el mapa producido. Descargar el archivo south-america-latest.osm.pbf, con los datos de toda América del Sur, en el directorio /opt/OSM/. El formato .osm corresponde a los datos OpenStreetMap, y el formato .osm.pbf es su versión comprimida.

```
mkdir /opt/OSM
cd /opt/OSM
wget -c http://download.geofabrik.de/south-america-latest.osm.pbf
```

La ejecución de este comando debe retornar un mensaje como el que aparece a continuación.

```
Resolving download.geofabrik.de (download.geofabrik.de)... 46.4.99.10
Connecting to download.geofabrik.de (download.geofabrik.de)|46.4.99.10|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 359700738 (343M) [application/octet-stream]
Saving to: 'south-america-latest.osm.pbf'
100%[++++++++++++=====================>] 359.700.738  267K/s   in 33m 15s
2013-01-15 16:20:57 (176 KB/s) - 'south-america-latest.osm.pbf' saved [359700738/359700738]
```

5.2 Configuración del archivo para importar

Creamos la carpeta donde se almacenará el análisis del archivo south-america-latest.osm.pbf

```
mkdir -p /opt/OSM/imposm
cd /opt/OSM/imposm
```

Es necesario contar con un archivo .JSON que contenga la cartografiá de la base de datos, para ello copiamos el template de ejemplo.

```
cp /opt/imposm/src/imposm3/example-mapping.json mapping.json
```

Es necesario crear la tabla "railways" en el archivo "mapping.json".

```
nano /opt/OSM/imposm/mapping.json
```

La estructura de la tabla

```
"railways" : {
      "fields": [
            {
                  "type": "id",
                  "name": "osm_id",
                  "key": null
            },
            {
                  "type": "geometry",
                  "name": "geometry",
                  "key": null
            },
            {
                  "type": "string",
                  "name": "name",
                  "key": "name"
            },
            {
                  "type": "mapping_value",
                  "name": "type",
                  "key": null
            },
            {
                  "type": "boolint",
                  "name": "tunnel",
                  "key": null
            },
            {
                  "type": "boolint",
                  "name": "bridge",
                  "key": null
            },
            {
                  "type": "string",
                  "name": "ref",
                  "key": null
            }
      ],
      "type": "linestring",
      "mapping": {
            "railway": [
                  "rail",
                  "abandoned",
                  "tram",
                  "light_rail",
                  "subway",
                  "narrow_gauge",
                  "preserved",
                  "funicular",
                  "monorail"
            ]
      }
}
```

Creando el archivo de configuración

```
nano config-imposm.json
```

Contenido del archivo, donde se especifica la carpeta donde se generan los archivos de caché, el template y los parámetros de conexión a la base de datos.

```
{
   "cachedir": "/opt/OSM/cache",
   "connection": "postgis://user:password@localhost:port/database",
      "-dbschema-production": "public",
   "mapping": "mapping.json"
}
```

OJO: Tener en cuenta que la memoria RAM del servidor tiene que ser superior o igual a 1024MB, para que imposm pueda ejecutarse sin error.

5.3 Importación a la base de datos

Importamos los datos de OpenStreetMap:

```
imposm3 import -config config-imposm.json -appendcache -deployproduction -read south-america-latest.osm.pbf -write
```

La respuesta es la siguiente:

```
[Jan 28 15:18:21] [INFO] [reader] reading south-america-latest.osm.pbf with data till 2013-03-29 16:00:03 -0400 BOT
[Jan 28 15:18:21] [ 0] C:     0/s     0/s (200000) N:  0/s  0/s (218) W:
[Jan 28 15:18:22] [1s] C:  967000/s  410000/s (408000) N: 400/s  0/s (220) W:
[Jan 28 15:18:22] [1s] C:  646000/s  390000/s (600000) N: 200/s  0/s (230) W:
[Jan 28 15:18:23] [2s] C:  540000/s  335000/s (768000) N: 100/s  0/s (248) W:
[Jan 28 15:18:23] [2s] C:  474000/s  289000/s (912000) N: 100/s  0/s (251) W:
[Jan 28 18:20:59] [INFO] [PostGIS] Clustering on geometry took: 746.219874ms
[Jan 28 18:20:59] [INFO] [PostGIS] Rotating osm_waterways_gen0 from import -> public -> backup
[Jan 28 18:20:59] [INFO] [PostGIS] Rotating osm_waterways_gen1 from import -> public -> backup
[Jan 28 18:20:59] [INFO] [PostGIS] Rotating osm_landusages_gen1 from import -> public -> backup
[Jan 28 18:20:59] [INFO] [PostGIS] Rotating osm_landusages_gen0 from import -> public -> backup
[Jan 28 18:20:59] [INFO] [PostGIS] Rotating tables took: 113.826451ms
[Jan 28 18:20:59] [INFO] Imposm took: 31.190873979s
```

Para mas información sobre las distintas opciones que cuenta imposm, utilizar el siguiente comando.

```
imposm3 import -help
```

6 Generación del fondo de mapa

Para generar el fondo de mapa a partir de los datos de OpenStreetMap, instalamos MapServer (el servidor de mapas) y descargamos un archivo de estilo dedicado a los datos OSM. Finalmente configuramos el servidor web Apache para que llame al script de MapServer para generar el fondo de mapa según las consultas web.

6.1 Instalación de MapServer 6.2.2

Se necesita instalar la versión 6.2.2 de MapServer para poder generar el fondo de mapa con el estilo que descargaremos. Necesitamos entonces compilar MapServer desde su código fuente. Para preparar la compilación, instalamos las utilidades y librerías necesarias

```
yum install libpng-devel freetype-devel gd-devel zlib-devel
yum install unixODBC-devel giflib-devel netcdf-devel
```

```
cd /opt/library
wget http://dl.fedoraproject.org/pub/epel/6/x86_64/epel-release-6-8.noarch.rpm
rpm -Uvh epel-release-6-8.noarch.rpm
yum install libgeotiff libgeotiff-devel
```

Instalamos las librerías recomendadas

```
yum install proj-devel curl-devel gdal-devel agg-devel
```

Instalamos las librerías opcionales

```
yum install libtiff-devel libjpeg-devel libxml2-devel geos-devel ming-devel
```

Crear la carpeta mapserver en "/opt"

```
mkdir -p /opt/mapserver
cd /opt/mapserver
```

Descargamos el código fuente de MapServer, luego descomprimir e ingresar a la carpeta que se genero.

```
wget http://download.osgeo.org/mapserver/mapserver-6.2.2.tar.gz
tar -xvzf mapserver-6.2.2.tar.gz
cd /opt/mapserver/mapserver-6.2.2
```

Antes de lanzar la compilación, lanzamos la configuración, que verifica, entre otros, si el sistema tiene todos los requisitos para la compilación, tomar en cuenta que el carácter "\" al final de la linea sirve para concatenar.

```
./configure --prefix=/usr --with-wfs --with-wcs --with-sos --with-wmsclient \
--with-wfsclient --with-proj --with-gdal --with-ogr --with-geos --with-gd \
--with-tiff --with-jpeg --with-png=/usr --with-agg --with-eppl \
 --with-postgis=/usr/pgsql-9.1/bin/pg_config --with-sde --with-freetype --with-threads --with-experimental-png
```

La respuesta a la orden

```
MapServer is now configured for x86_64-unknown-linux-gnu
------------ Compiler Info ------------
 C compiler:          gcc -O2  -Wall -Wdeclaration-after-statement -DNDEBUG
 C++ compiler:         g++ -O2  -Wall -DNDEBUG
 Debug:
 Generic NINT:
------------ Renderer Settings --------
 OpenGL support:
 zlib support:
 png support:          -DUSE_PNG
 gif support:          -DUSE_GIF
 jpeg support:         -DUSE_JPEG
 freetype support:       -DUSE_FREETYPE
 iconv support:        -DUSE_ICONV
 AGG support:          internal
 GD support:          -DUSE_GD  -DUSE_GD_PNG -DUSE_GD_JPEG -DUSE_GD_GIF
 Cairo (SVG,PDF) support:
 Cairo SVG symbol support:
 KML support:
------------ Support Libraries --------
 Proj.4 support:        -DUSE_PROJ
 Proj Fastpaths:
 Libxml2 support:       -DUSE_LIBXML2
 FriBidi support:
 Curl support:         -DUSE_CURL -DUSE_CURLOPT_PROXYAUTH
 FastCGI support:
 Exempi support:
 Threading support:      -DUSE_THREAD
 GEOS support:         -DUSE_GEOS
 XML Mapfile support:
 XSLT support:
 EXSLT support:
------------ Data Format Drivers -------
 PostGIS support:
 ArcSDE support:
 OGR support:          -DUSE_OGR
 GDAL support:         -DUSE_GDAL
 Oracle Spatial support:
------------ OGC Services -------------
 WMS Server:           -DUSE_WMS_SVR
 WMS Client:           -DUSE_WMS_LYR
 WFS Server:           -DUSE_WFS_SVR
 WFS Client:           -DUSE_WFS_LYR
 WCS Server:           -DUSE_WCS_SVR
 SOS Server:           -DUSE_SOS_SVR
------------ MapScript ----------------
 PHP MapScript:         no
 Python MapScript:       no
```

Ahora lanzamos la compilación

```
make && make install
```

El resultado de la instalación es:

```
libtool: install: /usr/bin/install -c .libs/mapserv /usr/bin/mapserv
***** MapServer CGI / FastCGI Installation *****
To install MapServer, copy or symlink the "/usr/bin/mapserv" file
to your web server's cgi-bin directory.
If you use MapScript then see the documentation for your specific MapScript
version for installation instructions.
test -z "" || (cd mapscript/php; make install DESTDIR=;)
test -z "" || (cd mapscript/python; make install DESTDIR=;)
```

Verificamos que la compilación funcionó correctamente, mapserv es el único archivo creado en el directorio "/opt/mapserver/mapserver-6.2.2", ejecutar el archivo

```
./mapserv
```

La respuesta normal tiene que ser la siguiente si funcionó la compilación.

```
This script can only be used to decode form results and
should be initiated as a CGI process via a httpd server.
```

6.2 Instalación del archivo de estilo

Para instalar el archivo de estilo para la región que queremos (Bolivia y sus alrededores), seguimos los siguientes pasos: descarga de basemaps, que contiene un estilo base y un entorno de compilación (Makefile) que permite generar un archivo de estilo adaptado configuración de la compilación según los parámetros de Bolivia generación del archivo de estilo por compilación

6.2.1 Instalación de basemaps

La página https://github.com/mapserver/basemaps explica como utilizar basemaps. Recuperamos el código fuente mediante la herramienta "git".

```
cd /opt/OSM
git clone https://github.com/mapserver/basemaps.git
```

lo que devolverá la lista de los archivos descargados

```
Initialized empty Git repository in /opt/OSM/basemaps/.git/
remote: Reusing existing pack: 686, done.
remote: Total 686 (delta 0), reused 0 (delta 0)
Receiving objects: 100% (686/686), 22.68 MiB | 385 KiB/s, done.
Resolving deltas: 100% (481/481), done.
```

6.2.2 Configuración de la compilación

En el archivo

```
nano /opt/OSM/basemaps/imposm-mapping.py
```

Configuramos el acceso a la base de datos imposm

```
db_conf = Options(
  db='osm',
  host='localhost',
  port=5432,
  user='USER',
  password='PASSWORD',
  sslmode='allow',
  prefix='osm_',
  proj='epsg:900913',
)
```

Modificamos también el archivo Makefile

```
nano /opt/OSM/basemaps/Makefile
```

Para configurar el prefijo de las tablas, y el estilo a aplicar

```
OSM_PREFIX=osm_
OSM_SRID=900913
OSM_UNITS=meters
OSM_DB_CONNECTION=host=localhost dbname=DBNAME user=USERNAME password=PASSWORD port=5432
STYLE=default
```

Modificar el Makefile de la carpeta data/, porque algunos archivos cambiaron de nombre

```
nano /opt/OSM/basemaps/data/Makefile
```

El contenido del archivo es:

```
SHPTREE=shptree

all: TM_WORLD_BORDERS-0.3 ne_10m_admin_0_boundary_lines_land processed_p shoreline_300

# Implicit rules to map 'all' targets to their .shp and .qix dependencies
%: %.shp %.qix ;

.PRECIOUS: %.qix %.shp ;

%.qix: %.shp
  $(SHPTREE) $< 8

processed_p.shp: processed_p.tar.bz2
  tar xmjf processed_p.tar.bz2

shoreline_300.shp: shoreline_300.tar.bz2
  tar xmjf shoreline_300.tar.bz2

TM_WORLD_BORDERS-0.3.shp: TM_WORLD_BORDERS-0.3.zip
  unzip TM_WORLD_BORDERS-0.3.zip
  touch TM_WORLD_BORDERS-0.3.shp
```

```
ne_10m_admin_0_boundary_lines_land.shp: ne_10m_admin_0_boundary_lines_land.zip
  unzip ne_10m_admin_0_boundary_lines_land.zip
  touch ne_10m_admin_0_boundary_lines_land.shp

processed_p.tar.bz2:
  wget http://tile.openstreetmap.org/processed_p.tar.bz2

ne_10m_admin_0_boundary_lines_land.zip:
  wget
      http://www.naturalearthdata.com/http//www.naturalearthdata.com/download/10m/cultural/ne_10m_admin_0_boundary_lines_land.zip

shoreline_300.tar.bz2:
  wget http://tile.openstreetmap.org/shoreline_300.tar.bz2

TM_WORLD_BORDERS-0.3.zip:
  wget http://thematicmapping.org/downloads/TM_WORLD_BORDERS-0.3.zip
```

6.2.3 Compilación del archivo de estilo

Lanzamos la creación del archivo de estilo .map (OjO: muy largo la primera vez, porque se tienen que descargar algunos archivos SHP)

```
cd /opt/OSM/basemaps/
make
```

La Respuesta será

```
python generate_style.py -s default -g > generated/defaultstyle.msinc
python generate_style.py  -s default -l 0 > generated/defaultlevel0.msinc
python generate_style.py  -s default -l 1 > generated/defaultlevel1.msinc
python generate_style.py  -s default -l 2 > generated/defaultlevel2.msinc
python generate_style.py  -s default -l 3 > generated/defaultlevel3.msinc
python generate_style.py  -s default -l 4 > generated/defaultlevel4.msinc
python generate_style.py  -s default -l 5 > generated/defaultlevel5.msinc
python generate_style.py  -s default -l 6 > generated/defaultlevel6.msinc
python generate_style.py  -s default -l 7 > generated/defaultlevel7.msinc
python generate_style.py  -s default -l 8 > generated/defaultlevel8.msinc
python generate_style.py  -s default -l 9 > generated/defaultlevel9.msinc
python generate_style.py  -s default -l 10 > generated/defaultlevel10.msinc
python generate_style.py  -s default -l 11 > generated/defaultlevel11.msinc
python generate_style.py  -s default -l 12 > generated/defaultlevel12.msinc
python generate_style.py  -s default -l 13 > generated/defaultlevel13.msinc
python generate_style.py  -s default -l 14 > generated/defaultlevel14.msinc
python generate_style.py  -s default -l 15 > generated/defaultlevel15.msinc
python generate_style.py  -s default -l 16 > generated/defaultlevel16.msinc
python generate_style.py  -s default -l 17 > generated/defaultlevel17.msinc
python generate_style.py  -s default -l 18 > generated/defaultlevel18.msinc
gcc -E -x c -D_debug=1 -D_layerdebug=1  -DOSM_PREFIX=osm_ -DOSM_SRID=900913 -P -o osm-default.map osmbase.map -DTHEME=default -
      D_proj_lib=\"`pwd`\" -Igenerated
sed -i 's/##.*$//g' osm-default.map
sed -i '/^ *$/d' osm-default.map
sed -i -e 's/OSM_PREFIX_/osm_/g' osm-default.map
sed -i -e 's/OSM_SRID/900913/g' osm-default.map
sed -i -e 's/OSM_UNITS/meters/g' osm-default.map
sed -i -e 's/OSM_EXTENT/-20000000 -20000000 20000000 20000000/g' osm-default.map
sed -i -e 's/OSM_WMS_SRS/EPSG:900913 EPSG:4326 EPSG:3857 EPSG:2154 EPSG:310642901 EPSG:4171 EPSG:310024802 EPSG:310915814
      EPSG:310486805 EPSG:310702807 EPSG:310700806 EPSG:310547809 EPSG:310706808 EPSG:310642810 EPSG:310642801 EPSG:310642812
      EPSG:310032811 EPSG:310642813 EPSG:2986/g' osm-default.map
sed -i -e 's/OSM_NAME_COLUMN/name/g' osm-default.map
```

Despues de realizar la compilación, verificar que los archivos con las extensiones "dbf, prj, qix, shp, shx" se encuentren dentro de la carpeta "/opt/OSM/basemaps/data":

```
boundaries.dbf
boundaries.prj
boundaries.qix
boundaries.shp
boundaries.shx
Makefile
ne_10m_admin_0_boundary_lines_land.dbf
ne_10m_admin_0_boundary_lines_land.prj
ne_10m_admin_0_boundary_lines_land.qix
ne_10m_admin_0_boundary_lines_land.README.html
ne_10m_admin_0_boundary_lines_land.shp
ne_10m_admin_0_boundary_lines_land.shx
10:35 ne_10m_admin_0_boundary_lines_land.VERSION.txt
10:32 ne_10m_admin_0_boundary_lines_land.zip
processed_p.dbf
processed_p.index
processed_p.qix
processed_p.shp
processed_p.shx
processed_p.tar.bz2
Readme.txt
seed.dbf
seed.prj
```

```
seed.shp
seed.shx
shoreline_300.dbf
shoreline_300.index
shoreline_300.qix
shoreline_300.shp
shoreline_300.shx
shoreline_300.tar.bz2
TM_WORLD_BORDERS-0.3.dbf
TM_WORLD_BORDERS-0.3.prj
TM_WORLD_BORDERS-0.3.qix
TM_WORLD_BORDERS-0.3.shp
TM_WORLD_BORDERS-0.3.shx
TM_WORLD_BORDERS-0.3.zip
wget-log
```

El archivo ha sido generado con extensión .map, en el que se incluye una serie de parámetros que definen las capas disponibles en el servicio, el estilo con que se representarán, su simbología, formato se generará la imagen, el sistema de referencia, etc y se encuentra en "/opt/OSM/basemaps/osm-default.map".

El archivo .map consta de varias secciones. Cada sección se inicia con el nombre de la sección y termina con la palabra END. El contenido de las secciones consiste en la definición de determinados parámetros del tipo atributo - valor.

7 Publicación del fondo de mapa vía Apache

La última etapa corresponde a la publicación del fondo de mapa vía el servidor web Apache.

7.1 Instalación y configuracion de apache

Instalamos el servidor web apache2

```
yum -y install httpd
```

Copiamos el script de mapserver en la carpeta de scripts de Apache2

```
cp /usr/bin/mapserv /var/www/cgi-bin
```

Creamos, con los derechos adecuados, la carpeta donde se generaran los archivos de salida del script de MapServer

```
mkdir /tmp/ms_tmp
chmod 777 /tmp/ms_tmp
```

Para que está carpeta se cree a cada reinicio del servidor (porque se puede borrar la carpeta /tmp), añadimos está creación automática

```
nano /etc/rc.local
```

Contenido del archivo

```
# Crear la carpeta /tmp/ms_tmp necesitada por MapServer
mkdir /tmp/ms_tmp
chmod 777 /tmp/ms_tmp
```

Para verificar que funciona el vínculo entre Apache y MapServer, probamos en un navegador de acceder a http://direcciónIP/cgi-bin/mapserv, en este caso sin parámetros. La respuesta debería ser

```
No query information to decode. QUERY_STRING is set, but empty.
```

7.2 Acceso al fondo de mapa

Para verificar que la generación del fondo de mapa es correcta, abrir un navegador de internet y acceder a la URL.

```
http://direcciónIP/cgi-bin/mapserv?map=/opt/OSM/basemaps/osm-default.map&mode=browse&template=openlayers&layers=all
```

No se puede visualizar el mapa, debido a un problema con la conexión entre mapserver y postgis, el error es:

```
msPostGISLayerOpen(): Query error. Database connection failed (could not connect to server: Permission denied
Is the server running on host "localhost" (::1) and accepting
TCP/IP connections on port 5432?
could not connect to server: Permission denied
Is the server running on host "localhost" (127.0.0.1) and accepting
TCP/IP connections on port 5432?
with connect string 'host=localhost dbname= user= password=****** port=5432'
Is the database running? Is it allowing connections? Does the specified user exist? Is the password valid? Is the database on
      the standard port?
msDrawMap(): Image handling error. Failed to draw layer named 'places2'.
```

Solucionando el problema:
Verificar la conexión a la base de datos desde el servidor utilizando el comando "ogrinfo".

```
ogrinfo PG:"host=127.0.0.1 user=USER password=PASSWORD dbname=DATABASE port=5432"
```

La respuesta al comando debe ser exitosa

```
INFO: Open of 'PG:host=127.0.0.1 user=USER password=PASSWORD dbname=DATABASE port=5432'
    using driver 'PostgreSQL' successful.
```

Si todavía persiste el problema, verificar si "selinux" esta habilitado.

```
/usr/sbin/setenforce -v
```

El resulto del comando, muestra que se encuentra habilitado

```
SELinux status:          enabled
SELinuxfs mount:         /selinux
Current mode:            enforcing
Mode from config file:   disabled
Policy version:          24
Policy from config file:     targeted
```

Para deshabilitar "selinux" editamos el archivo "grub.conf" y reniciamos la maquina.

```
nano /etc/grub.conf
```

Especificar el parámetro "setenforce 0" al final del archivo.

```
setenforce 0
```

Verificar que se encuentra deshabilitado

```
/usr/sbin/sestatus -v
```

El resultado del comando indica que se encuentra deshabilitado.

```
/usr/sbin/setenforce: SELinux is disabled
```

Probar de nuevo la visualización del mapa desde una navegador web.

8 Actualización periódica de la base de datos

Los datos de OpenStreetMap están enriquecidos cada día por su comunidad, lo que implica que su calidad mejora regularmente. Para sacar provecho de esta mejora constante de los datos, se puede configurar la descarga y importación de los nuevos datos en la base de datos, vía la utilidad cron del sistema. Para automatizar la actualización de los datos, creamos un script que, cada determinado tiempo, descargue los datos de un espejo de OpenStreetMap y los suba de forma automática en la base de datos. Para que los archivos sean más organizados, creamos unos directorios en la siguiente dirección /opt/OSM/.

```
mkdir -p /opt/OSM/datos /opt/OSM/scripts /opt/OSM/logs /opt/OSM/cache
```

El directorio datos almacenará los archivos .osm.pbf descargados, El directorio scripts almacenará el script de actualización (extensión .sh), El directorio logs almacenará los archivos .log los cuales serán generados de forma automática al ejecutarse el script y contendrán información de todo el proceso del script El directorio cache será el directorio donde se almacenarán los archivos temporales generados cuando se lanza la subida de datos OSM a la base de datos PostgreSQL. Creamos el script de actualización de los datos en el directorio scripts

```
nano import_dat_osm.sh
```

Dentro del archivo colocamos el siguiente código

```
#! /bin/sh
# Descargamos datos desde OSM
## hacemos una copia de backup del archivo anterior
mv /opt/OSM/datos/south-america-latest.osm.pbf /opt/OSM/datos/south-america-latest.osm.pbf.old
## descargamos el nuevo archivo
wget -O /opt/OSM/datos/south-america-latest.osm.pbf -c
      http://download.geofabrik.de/openstreetmap/south-america-latest.osm.pbf
# Analizamos el archivo y lo importamos en la base de datos
cd /opt/OSM/cache/
imposm3 import -config config-imposm.json -deployproduction -read south-america-latest.osm.pbf -write
```

Le damos permisos de ejecución al archivo .sh

```
chmod +x import_dat_osm.sh
```

Ahora configuramos CRON ya que será quien active de forma automática el script.

```
crontab -u root -e
```

Dentro del archivo en la última línea colocamos lo siguiente.

```
# activar cada 23 horas con 10 min. (23h10)
10 23 * * * /opt/OSM/scripts/import_dat_osm.sh > /opt/OSM/logs/import_dat_osm_`date +\%Y_\%m_\%d`.log --
      >dejar_un_espacio_vacío_al final_de_la_línea
```