

Contents

1	CONFIGURACIÓN OPENLDAP	2
1.1	Propósito	2
1.2	Paquetes Necesarios	2
1.3	Configuración e Instalación de ODBC	3
1.4	Configuración e Instalación de OpenLDAP	5
1.5	Configuración de Apache Directory Studio	6
1.6	Instalación y LDAPsearch en Node.js	7
2	CONFIGURACIÓN JBMP	8
2.1	Propósito	8
2.2	Paquetes necesarios	8
2.3	Instalación de jBPM	8
2.4	Configuración de PostgreSQL en jBPM	9

1 CONFIGURACIÓN OPENLDAP

1.1 Propósito

El presente tema “Configuración de OpenLdap” permite configurar Node.js, Postgres, OpenLdap, ODBC, Apache Directory Studio, con el fin de poder realizar consultas desde Node.js por medio del servidor OpenLdap a la base de datos Postgres. Este tutorial esta hecho para personas que ya tienen conocimiento básico de manejo de linux y open ldap ya que indica donde se coloquen los usuarios y contraseñas sin dar ejemplos, en caso de querer obtener con ejemplos se recomienda que se trabaje con el Readme.md del siguiente GitHub: <https://github.com/Andresit0/LDAP-Postgres-Connection.git> el cuál contiene formas de probar conexiones, además que indica como crear grupos y subgrupos en Apache Directory Studio.

1.2 Paquetes Necesarios

Debido a la necesidad de algunos paquetes en Linux o errores que pueden darse durante la instalación y configuración de LDAP detallamos a continuación los mismos con sus soluciones, recomendando instalar todos en este momento para no tener posteriores problemas:

```
configure: error: C compiler cannot create executables
sudo apt-get install g++
```

```
configure: error: readline library not found
If you have readline already installed, see config.log for details on the
failure. It is possible the compiler isn't looking in the proper directory.
Use --without-readline to disable readline support.
sudo apt-get install libreadline6-dev
```

```
configure: error: zlib library not found
If you have zlib already installed, see config.log for details on the
failure. It is possible the compiler isn't looking in the proper directory.
Use --without-zlib to disable zlib support.
sudo apt-get install zlib1g-dev
```

```
checking for CRYPTO_new_ex_data in -lcrypto... no
configure: error: library 'crypto' is required for OpenSSL
sudo apt-get install libssl-dev
```

```
configure: error: libpq library version >= 9.2 is required
sudo apt-get install libpq-dev
```

```
configure: error: could not locate libtool ltdl.h
sudo apt install libltdl-dev
```

```
connections on Unix domain socket "/tmp/.s.PGSQL.5432"?
```

Instalar postgres en: /usr/local/pgsql

```
createdb: /usr/local/pgsql/lib/libssl.so.1.0.0: no version information available (required by
/usr/local/lib/libldap_r-2.4.so.2)
sudo ln -fs /usr/lib/liblber-2.4.so.2 /usr/local/lib/
sudo ln -fs /usr/lib/libldap_r-2.4.so.2 /usr/local/lib/
```

1.3 Configuración e Instalación de ODBC

Para la instalación de unixODBC ejecutar en el terminal:

```
cd /usr/local/src/
sudo wget http://www.unixodbc.org/unixODBC-2.3.4.tar.gz
sudo tar xzf unixODBC-2.3.4.tar.gz
cd unixODBC-2.3.4/
sudo ./configure
sudo make
sudo make install
```

Al finalizar la instalación, es necesario agregar el directorio de instalación bin de Postgres (en este caso /usr/local/pgsql/bin) a la variable de entorno PATH, y configurar correctamente la variable de entorno LD_LIBRARY_PATH:

```
Entonces ejecutar en el terminal sudo nano /etc/profile.d/path.sh y pegar:
PATH="${PATH}:/usr/local/pgsql/bin"
export PATH
LD_LIBRARY_PATH="/usr/local/lib:/usr/local/pgsql/lib"
export LD_LIBRARY_PATH
```

Actualizar el perfil Bash ejecutando:

```
sudo source /etc/profile.d/path.sh
```

Para la instalación de psqlODBC ejecutar en el terminal:

```
cd /usr/local/src/  
sudo wget https://ftp.postgresql.org/pub/odbc/versions/src/psqlodbc-10.00.0000.tar.gz  
sudo tar xzf psqlodbc-10.00.0000.tar.gz  
cd psqlodbc-10.00.0000/  
sudo ./configure --help  
sudo ./configure --with-unixodbc  
sudo make  
sudo make install
```

Para configurar ODBC establecer correctamente los valores para el usuario, contraseña (los cuales deben ser creados en la base de datos Postgres también) y base de datos:

Entonces ejecutar en el terminal `sudo nano /usr/local/etc/odbc.ini` y pegar:

```
;  
; odbc.ini  
;  
[ODBC Data Sources ]  
PgSQL=PostgreSQL  
  
[PgSQL ]  
; WARNING: The old psql odbc driver psqlodbc.so is now renamed psqlodbcw.so  
; in version 08.x. Note that the library can also be installed under an other  
; path than /usr/local/lib/ following your installation.  
Driver=/usr/local/lib/psqlodbcw.so  
Description=Connection to LDAP/POSTGRESQL  
Servername=localhost  
Port=5432  
Protocol=6.4  
FetchBufferSize=99  
Username=nombreUsuarioPostgres  
Password=claveDeUsuarioPostgres  
Database=nombreBaseDatosPostgres  
ReadOnly=no  
Debug=1  
CommLog=1  
  
[ODBC ]  
InstallDir=/usr/local/lib
```

Resta configurar el driver ODBC “Postgres”:

```

Entonces después de insertar sudo nano /usr/local/etc/odbcinst.ini en el terminal, pegar: ;
; odbcinst.ini
;
[PostgreSQL ]
Description=ODBC for PostgreSQL
; WARNING: The old psql odbc driver psqlodbc.so is now renamed psqlodbcw.so
; in version 08.x. Note that the library can also be installed under an other
; path than /usr/local/lib/ following your installation.
Driver=/usr/local/lib/psqlodbcw.so

[ODBC ]
Trace=1
Debug=1
Pooling=No

```

1.4 Configuración e Instalación de OpenLDAP

Con la finalidad de instalar OpenLDAP ejecutar los siguientes comandos:

```

cd /usr/local/src/
sudo wget http://gpl.savoirfairelinux.net/pub/mirrors/openldap/openldap-release/openldap-2.4.45.tgz
sudo tar xzf openldap-2.4.45.tgz
sudo chown -R root:staff openldap-2.4.45
cd openldap-2.4.45/
sudo apt-get install libtool libssl-dev
sudo ./configure --enable-crypt --enable-modules --enable-rlookups --enable-sql=yes --with-tls=openssl --without-cyrus-sasl --disable-bdb --disable-hdb --disable-ipv6
make depend
sudo make
sudo make install

```

Para configurar OpenLDAP acceder al archivo slapd.conf y poner los datos solicitados

```

cd /usr/local/etc/openldap/
sudo cp /usr/local/src/openldap-2.4.45/servers/slapd/back-sql/rdbms_depend/pgsql/slapd.conf .
sudo nano slapd.conf

```

Y realizar las modificaciones a los siguientes parámetros acorde a la configuración ODBC:

```

database sql
suffix crearDirectorioVirtualLDAP(ej: "dc=example,dc=com")
rootdn crearUsuarioLDAP(ej: "cn=root,dc=example,dc=com")

```

```
rootpw crearClaveUsuarioLDAP
dbname PgSQL
dbuser nombreUsuarioPostgres
dbpasswd claveDeUsuarioPostgres
```

A continuación se debe crear la estructura de base de datos para LDAP. Esta estructura es necesaria para que LDAP funcione con SQL como backend. Para ello se debe reemplazar en la siguiente dirección los siguientes scripts `backsql_create.sql`, `testdb_create.sql`, `testdb_metadata.sql` por los mismos que se encuentran en la siguiente dirección de github <https://github.com/Andresit0/LDAP-Postgres-Connection.git> cambiando primeramente la línea 56 del `testdb_metadata.sql` por los datos que se hayan metido en la respectiva configuración de LDAP.

```
cd /usr/local/src/openldap-2.4.45/servers/slapd/back-sql/rdbms_depend/pgsql/
```

con la finalidad de configurar el backend se puede digitar en el terminal(estando en la carpeta `pgsql`):

```
sudo psql -d pg_ldap < backsql_create.sql
sudo psql -d pg_ldap < testdb_create.sql
psql -d pg_ldap | testdb_metadata.sql
```

Los errores arrojados por ambos scripts se deben a que intentan eliminar tablas que aún no existen (están pensados para limpiar una instalación preexistente). Simplemente ignorar estos errores.

Finalmente se deben dar permisos a la base de datos `pg_ldap` creada anteriormente, para ello en el terminal entramos como usuario `postgres`:

```
psql -d pg_ldap
grant all on ldap_attr_mappings,ldap_entries,ldap_entry_objclasses,ldap_oc_mappings,referrals,certs
to ldap;
grant all on ldap_attr_mappings_id_seq,ldap_entries_id_seq,ldap_oc_mappings_id_seq,referrals_id_seq
to ldap;
grant all on authors_docs,documents,institutes,persons,phones to ldap;
grant all on documents_id_seq,institutes_id_seq,persons_id_seq,phones_id_seq to ldap;
```

1.5 Configuración de Apache Directory Studio

Descargar de la página <http://directory.apache.org/studio/> descomprimirlo y ejecutarlo. Posteriormente ir a nueva conexión e insertar los datos de acuerdo a los puestos en la configuración del LDAP, con ello ya se podrá hacer modificaciones en la base de datos de Postgres.

1.6 Instalación y LDAPsearch en Node.js

Para proceder a realizar un search en LDAP con Node.js primero lo instalamos y luego instalamos el paquete ldapjs que se utilizará en un posterior script.

```
sudo curl -sL https://deb.nodesource.com/setup_10.x | sudo -E bash -  
sudo apt-get install -y nodejs  
sudo npm install ldapjs
```

Para poder realizar una búsqueda se puede utilizar el siguiente script llamado “ldapSearch.js” el mismo que debe ser modificado de acuerdo a los datos insertados en la configuración del LDAP y a lo que uno quiera buscar mediante el filtro:

```
1 var ldap = require('ldapjs');  
2 var client = ldap.createClient({  
3   url: 'ldap://127.0.0.1:389'  
4 });  
5  
6 var opts = {  
7   filter: '(telephoneNumber= 222-3234)',  
8   scope: 'sub',  
9   paged: true,  
10  password: 'secret',  
11  sizeLimit: 200,  
12  attributes: ['dn', 'sn', 'cn', 'userPassword']  
13 };  
14  
15 client.search('dc=example,dc=com', opts, function(err, res) {  
16   res.on('searchEntry', function(entry) {  
17     console.log('entryJsonFormat: ' + JSON.stringify(entry.object));  
18   });  
19   res.on('searchReference', function(referral) {  
20     console.log('referral: ' + referral.uris.join());  
21   });  
22   res.on('error', function(err) {  
23     console.error('error: ' + err.message);  
24   });  
25   res.on('end', function(result) {  
26     console.log('status: ' + result.status);  
27   });  
28 });
```

Y para probar su funcionamiento se debe primero activar el OpenLDAP y de allí correr el Script como se indica a continuación:

```
sudo /usr/local/libexec/slapd
node IS.js
```

Lás páginas de dónde se ha podido desarrollar el presente tutorial con las que se podría entender mejor el mismo se detallan a continuación:

<https://www.youtube.com/watch?v=Zmj6A5ggcgg>

<http://ldapjs.org/client.html>

<http://ldapjs.org/>

<https://stackoverflow.com/questions/13255389/ldapjs-authentication-user-login-setup>

<https://www.codeday.top/2017/01/09/17648.html>

<https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-openldap-and-phpldapadmin-on-an-ubuntu-14-04-server>

<https://www.digitalocean.com/community/tutorials/how-to-manage-and-use-ldap-servers-with-openldap-utilities>

<http://openaccess.uoc.edu/webapps/o2/bitstream/10609/27422/6/pllorensvTFM0114memoria.pdf>

2 CONFIGURACIÓN JBPM

2.1 Propósito

El presente tema tiene como objetivo instalar el Administrador de Procesos de Negocio jBPM en Ubuntu. La utilidad de los BPM's es la de facilitar el diseño y manejo de los procesos dentro de una empresa, haciendo visible el flujo de tareas para los desarrolladores, analistas y usuarios en general.

2.2 Paquetes necesarios

Para la instalación de jBPM se debe contar con Java JDK 1.6+ y Ant 1.7+.

En caso de no tenerlos instalados visitar:

- Java: <http://java.sun.com/javase/downloads/index.jsp>
- Ant: <http://ant.apache.org/bindownload.cgi> (se puede instalar por línea de comandos con `sudo apt-get install ant`)

2.3 Instalación de jBPM

- Descargar jBPM 6.3.0 desde <https://sourceforge.net/projects/jbpm/files/jBPM%206/jbpm-6.3.0.Final/>
- En la carpeta donde se ha descargado jBPM 6.3.0 ubicar el archivo `.zip` y descomprimirlo. Luego ingresar a la nueva carpeta descomprimida y ejecutar `ant install.demo.noclipse`. Esto hará las siguientes cosas:

- Descargar JBoss AS
- Configurar la herramienta web para jBPM
- Una vez instalado ejecutar `ant start.demo.noclipse`. Esto hará:
 - Iniciar el servidor de la base de datos H2 (luego cambiaremos a PostgreSQL)
 - Iniciar JBoss AS

Para comprobar que está correctamente instalado se debe comprobar que el puerto 8080 del computador no esté ocupado por otro software. Luego se debe ir a <http://localhost:8080/jbpm-console> unos 2 minutos después de haber ejecutado la línea `ant start.demo.noclipse`. Debería salir la pantalla inicial de KIE Workbench. Se puede ingresar con las credenciales usuario: krisv, password: krisv.

Todo esto debería ocurrir para saber que se ha instalado correctamente. Caso contrario re-intentar.

2.4 Configuración de PostgreSQL en jBPM

- Ir al archivo `build.properties` y:
 - Comentar lo correspondiente a H2
 - Descomentar lo correspondiente a PostgreSQL
 - Reemplazar lo correspondiente a PostgreSQL por:

```
db.name=postgresql
db.driver.module.prefix=org/postgresql
db.driver.jar.name=postgresql-9.4.1212.jre6.jar
db.driver.download.url=https://jdbc.postgresql.org/download/postgresql-9.4.1212.jre6.jar
```

- Ir al archivo `db/jbpm-persistence-JPA2.xml` y modificar la línea

```
<property name="hibernate.dialect" value="org.hibernate.dialect.H2Dialect" />
```

por

```
<property name="hibernate.dialect" value="org.hibernate.dialect.PostgreSQLDialect" />
```

- Ir al archivo `standalone-full-wildfly-8.1.0.Final.xml` y cambiar el bloque

```
<datasource jta="true" jndi-name="java:jboss/datasources/jbpmDS" pool-name="H2DS"
  enabled="true" use-java-context="true" use-ccm="true">
```

```

        <connection-url>jdbc:h2:tcp://localhost/~jbpm-db;MVCC=TRUE</connection-url
        >
        <driver>h2</driver>
        <security>
            <user-name>sa</user-name>
        </security>
    </datasource>
    <drivers>
        <driver name="h2" module="com.h2database.h2">
        <xa-datasource-class>org.h2.jdbcx.JdbcDataSource</xa-datasource-class>
        </driver>
    </drivers>

```

por

```

<datasource jta="true" jndi-name="java:jboss/datasources/jbpmDS" pool-name="
    PostgreSQLDS" enabled="true" use-java-context="true" use-ccm="true">
    <connection-url>jdbc:postgresql://localhost:5432/[BASE DE DATOS X DE
        POSTGRES]</connection-url>
    <driver>postgresql</driver>
    <security>
        <user-name>[USUARIO X DE POSTGRES]</user-name>
        <password>[PASSWORD DEL USUARIO X]</password>
    </security>
</datasource>
<drivers>
    <driver name="h2" module="com.h2database.h2">
    <xa-datasource-class>org.h2.jdbcx.JdbcDataSource</xa-datasource-class>
    </driver>
    <driver name="postgresql" module="org.postgresql">
    <xa-datasource-class>org.postgresql.xa.PGXADatasource</xa-datasource-class>
    </driver>
</drivers>

```

- Descargar el jdbc de PostgreSQL desde <https://jdbc.postgresql.org/download/postgresql-9.4.1212.jre6.jar> y colocarlo en db/driver
- Crear el archivo db/postgresql_module.xml y poner:

```

<module xmlns="urn:jboss:module:1.0" name="org.postgresql">
    <resources>
        <resource-root path="postgresql-9.4.1212.jre6.jar"/>
    </resources>

    <dependencies>
        <module name="javax.api"/>
    </dependencies>
</module>

```

```
        <module name="javax.transaction.api"/>
      </dependencies>
</module>
```

- Luego de todo esto, se debe ejecutar

```
ant stop.demo
ant clean.demo
ant install.demo.noclipse
ant start.demo.noclipse
```

- Ir de nuevo a <http://localhost:8080/jbpm-console> unos 2 minutos después de ejecutar la última línea y comprobar que salga la pantalla de inicio del KIE Workbench.

Para comprobar que la conexión a PostgreSQL es exitosa se debe crear un proyecto de prueba con un script que intente conectarse a la base de datos que hemos configurado en este tutorial.

- Para más información puede visitar:

- <http://docs.jboss.org/jbpm/v6.3/userguide/ch03.html>