JAVA, JBOSS AND DATABASE SERVER CONFIGURATION

Postgresql Server installation & Configuration

The postgresql-server package includes the programs needed to create and run a PostgreSQL server, which will in turn allow to create and maintain PostgreSQL databases. PostgreSQL is an advanced Object-Relational database management system (DBMS) that supports almost all SQL constructs (including transactions, subselects and user-defined types and functions). We should install postgresql-server if we want to create and maintain our own PostgreSQL databases and/or our own PostgreSQL server. We also need to install the postgresql package.

We also need to install client package called postgrsql. This package contains the docs in HTML for the whole package, as well as command-line utilities for managing PostgreSQL databases on a PostgreSQL server.

PACKAGES AND OS REQUIREMENTS FOR POSTGRESQL SERVER

Operating Systems: RHEL 6.x.x
Install Postgresql Server
Login as the root user and enter the command:

```
# yum install postgresql postgresql-server
```

Start Postgresql:
Type the following two command:

```
# chkconfig postgresql on
# service postgresql start
```

Connect to Postgresql Server using the below command.

```
# su - postgres
$ psql -d template1 -U postgres
```

Finally make sure that iptables is allows TCP port 5432 for remote access to Postgresql server. if not allow it in /etc/sysconfig/iptables file

```
# vi /etc/sysconfig/iptables
```

Append following line before COMMIT line to open port 5432 .

```
-A -INPUT -m state --state NEW -m tcp -p tcp --dport 5432 -j ACCEPT
```

Save and close the file. Restart the firewall:

```
# service iptables restart
```

PostgreSQL: To create database, add a user account and grant permission for database

To create a normal user and an associated database we need to type the following commands.
The easiest way to use is to create a Linux IDENT authentication i.e. add user 'integra' to Linux system first.

Step # 1: Add a Linux user called 'integra'

```
#useradd integra
#password integra
```

Step # 2: Becoming a superuser

We need to login as database super user under postgresql server. Again the simplest way to connect as the postgres user is to change to the postgres unix user on the database server using su command as follows:

```
# su - postgres
```

Step #3: Now connect to database server
Type the following command

```
$ psql -d template1 -U postgres
```

Output:

```
Welcome to psql 9.1.16, the PostgreSQL interactive terminal.
Type:  \\copyright for distribution terms
       \\h for help with SQL commands
       \\? for help on internal slash commands
       \\g or terminate with semicolon to execute query
       \\q to quit
template1=#
```

Step #4: Add a user called integra
Type the following command to create a user called integra with a password called myPassword.

```
template1=# CREATE USER integra WITH PASSWORD 'myPassword';
```

Step #5: Add a database called iask
Type the following command.

```
template1=# CREATE DATABASE iask;
```

Now grant all privileges on database

```
template1=# GRANT ALL PRIVILEGES ON DATABASE iask to integra;
```

Type \q to quit:

```
template1=# \q
```

Step #6: Test integra user login
In order to login as integra you need to type following commands. Login as 'integra' or use su command:

```
$ su - integra
$ psql -d iask -U integra
```

Step #7: To enable remote access to PostgreSQL database server?
By default, PostgreSQL database server remote access disabled for security reasons. However, some time we need to provide the remote access to database server from application server or from web server.
Step #1. **Enable client authentication**
We need edit the PostgreSQL configuration file, edit the PostgreSQL configuration file **/opt/postgres/9.1/data/pg_hba.conf** using a text editor such as vi.

Edit the file:

```
/opt/postgres/9.1/data/pg_hba.conf
```

Append the following configuration lines to give access to 10.10.11.2 I/P address

```
host    all         all         10.10.11.2/32          md5
```

Save and close the file. Make sure you replace 10.10.11.2 with actual network IP address range of the clients system in your own network

JBOSS AND JAVA CONFIGURATION

Step 1
Add the user for jboss

```
[root@localhost ~]#useradd jboss
[root@localhost ~]#passwd jboss
```

Step 2
Installing the java package
Download the package from following url (http://www.oracle.com/technetwork/java/)
Use following command to install.

```
[root@localhost ~]#./jdk-6u45-linux-x64.bin
```

Step 3

Installing & Configuration JBOSS

Step 4
Switch to jboss user using following command

```
[root@localhost ~]# su – jboss

[jboss@localhost ~]$ cp -av jboss-6.0.0.tar.gz  /home/jboss/
[jboss@localhost ~]$exit ( Exit from jboss user)
[root@localhost ~]#cd /etc/alternatives/
[root@localhost alternatives]# rm -f java  javac keytool
[root@localhost alternatives]# ln -s /usr/java/jdk1.6.0_45/bin/java .
[root@localhost alternatives]# ln -s /usr/java/jdk1.6.0_45/bin/javac .
[root@localhost alternatives]#  ln -s /usr/java/jdk1.6.0_45/bin/keytool .
[root@localhost ~]# su -  jboss
[jboss@localhost ~]$ vi .bash_profile
export JBOSS_HOME=$HOME/jboss-6.0.0 ( Note* :- last don't give slash)
export JAVA_HOME=/usr/java/jdk1.6.0_45 ( Note* :- last don't give slash)
:wq!
[root@localhost ~]# java -version (Enter) ( Here you can see java latest version what you installed.)
[jboss@localhost ~]$ source .bash_profile
[jboss@localhost ~]$echo $JBOSS_HOME
[jboss@localhost ~]$ echo $JAVA_HOME
[root@localhost ~]#  touch /var/log/jboss
[root@localhost ~]#chown jboss:jobss  /var/log/jboss
[root@localhost ~]# su -l jboss
[jboss@localhost ~]$cd jboss/jboss-6.0.0/bin
[jboss@localhost ~]$./run.sh ( After running jboss,  in same terminal you can see  some values. Eg:
16s:175ms .That means jboss started successfully.)
then you  can check it in  browser  ( http://localhost:8080/  or http://serverip:8080/ )
```

Step 5
Setting JAVA CLASS PATH

```
[root@localhost ~]# vi /etc/bashrc
export PATH=$PATH:/usr/java/jdk1.6.0_45/bin/
export CLASS=$CLASSPATH:/usr/java/jdk1.6.0_45/jre/lib/
:wq! ( to save)
[root@localhost ~]# source /etc/bashrc
[root@localhost ~]#echo $CLASS
[root@localhost ~]#echo $PATH
```

Step #6 **Create startup Script for jboss:**
In the jboss script (shown completed below), make the following changes:

1.Set the JBOSS_HOME to where we unpacked JBoss in step 2 above:

JBOSS_HOME=${JBOSS_HOME:-"**/home/jboss/jboss-6.0.0**"}
2. Set the **JAVA_HOME** to where we installed the JDK in step 1 above:
JAVAPTH=${JAVAPTH:-"**/usr/java/jdk1.6.0_45**"}
3.which sets the JBOSS_HOST to 0.0.0.0,allowing JBoss to bind to any IP. **JBOSS_HOST="0.0.0.0"**

[root@localhost ~]#vi /etc/init.d/jboss (Paste following lines into jboss file and save it.)

```
#!/bin/sh
#
# description: JBoss Start Stop Restart
# processname: jboss6
# chkconfig: 234 20 80
#
# $Id: jboss_init_redhat.sh 81068 2008-11-14 15:14:35Z dimitris@jboss.org $
#
# JBoss Control Script
#
# To use this script run it as root - it will switch to the specified user
#
# Here is a little (and extremely primitive) startup/shutdown script
# for RedHat systems. It assumes that JBoss lives in /usr/local/jboss,
# it's run by user 'jboss' and JDK binaries are in /usr/local/jdk/bin.
# All this can be changed in the script itself.
#
# Either modify this script for your requirements or just ensure that
# the following variables are set correctly before calling the script.

#define where jboss is - this is the directory containing directories log, bin, conf etc
JBOSS_HOME=${JBOSS_HOME:-"/home/jboss/jboss-6.0.0"}

#define the user under which jboss will run, or use 'RUNASIS' to run as the current user
JBOSS_USER=${JBOSS_USER:-"jboss"}

#make sure java is in your path
JAVAPTH=${JAVAPTH:-"/usr/java/jdk1.6.0_45"}

#configuration to use, usually one of 'minimal', 'default', 'all'
JBOSS_CONF=${JBOSS_CONF:-"default"}

#if JBOSS_HOST specified, use -b to bind jboss services to that address
JBOSS_HOST="0.0.0.0"
JBOSS_BIND_ADDR=${JBOSS_HOST:+"-b $JBOSS_HOST"}

#define the classpath for the shutdown class
JBOSSCP=${JBOSSCP:-"$JBOSS_HOME/bin/shutdown.jar:$JBOSS_HOME/client/jnet.jar"}

#define the script to use to start jboss
```

```
JBOSSSH=${JBOSSSH:-"$JBOSS_HOME/bin/run.sh -c $JBOSS_CONF $JBOSS_BIND_ADDR"}

if [ "$JBOSS_USER" = "RUNASIS" ]; then
  SUBIT=""
else
  SUBIT="su - $JBOSS_USER -c "
fi

if [ -n "$JBOSS_CONSOLE" -a ! -d "$JBOSS_CONSOLE" ]; then
  # ensure the file exists
  touch $JBOSS_CONSOLE
  if [ ! -z "$SUBIT" ]; then
    chown $JBOSS_USER $JBOSS_CONSOLE
  fi
fi

if [ -n "$JBOSS_CONSOLE" -a ! -f "$JBOSS_CONSOLE" ]; then
  echo "WARNING: location for saving console log invalid: $JBOSS_CONSOLE"
  echo "WARNING: ignoring it and using /dev/null"
  JBOSS_CONSOLE="/dev/null"
fi

#define what will be done with the console log
JBOSS_CONSOLE=${JBOSS_CONSOLE:-"/dev/null"}

JBOSS_CMD_START="cd $JBOSS_HOME/bin; $JBOSSSH"
JBOSS_CMD_STOP=${JBOSS_CMD_STOP:-"java -classpath $JBOSSCP org.jboss.Shutdown
--shutdown"}

if [ -z "`echo $PATH | grep $JAVAPTH`" ]; then
  export PATH=$PATH:$JAVAPTH
fi

if [ ! -d "$JBOSS_HOME" ]; then
  echo JBOSS_HOME does not exist as a valid directory : $JBOSS_HOME
  exit 1
fi

echo JBOSS_CMD_START = $JBOSS_CMD_START

case "$1" in
start)
    cd $JBOSS_HOME/bin
    if [ -z "$SUBIT" ]; then
        eval $JBOSS_CMD_START >${JBOSS_CONSOLE} 2>&1 &
    else
```

```
        $SUBIT "$JBOSS_CMD_START >${JBOSS_CONSOLE} 2>&1 &"
    fi
    ;;
stop)
    if [ -z "$SUBIT" ]; then
        $JBOSS_CMD_STOP
    else
        $SUBIT "$JBOSS_CMD_STOP"
    fi
    ;;
restart)
    $0 stop
    $0 start
    ;;
*)
    echo "usage: $0 (start|stop|restart|help)"
esac
```

```
:wq ( to save)
[root@localhost ~]# ( After saving you will get this root prompt )
```

Step 7 Run jboss as a service

To run JBoss as a service and enable start up at boot, make the script we created above executable and add it to our chkconfig so it starts at boot

```
[root@localhost ~]# chmod 755 jboss
[root@localhost ~]# chkconfig --add jboss
[root@localhost ~]# chkconfig --level 2345 jboss on
```

```
We should now be able to Start, Stop, and Restart JBoss as a service.
Start JBoss:
```

Note: JBoss can take some time to start.

```
[root@localhost ~]#service jboss start
JBOSS_CMD_START = cd /usr/share/jboss-6.0.0/bin; /usr/share/jboss-6.0.0/bin/run.sh -c default -b 0.
0.0.0
```

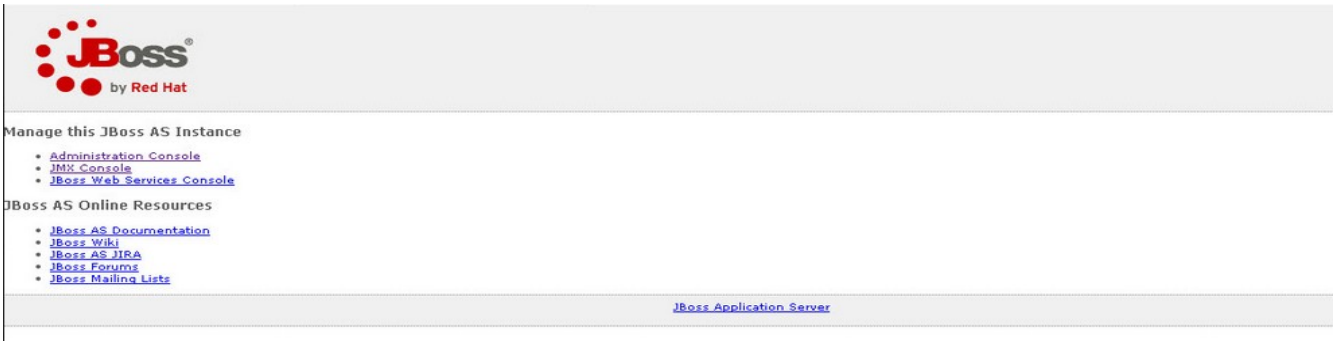
Stop JBoss:

```
[root@localhost ~]#service jboss stop
JBOSS_CMD_START = cd /usr/share/jboss-6.0.0.Final/bin; /usr/share/jboss-6.0.0.Final/bin/run.sh -c
default -b 0.0.0.0
Shutdown message has been posted to the server.
Server shutdown may take a while - check logfiles for completion
```

Access the Jboss Admin:

Make sure JBoss is started and you should now be able to access the Jboss Console at:

http://yourdomain.com:8080 or http://yourip:8080
The default user name and password for the JBoss Admin Console is admin/admin



For the Further reference please check the following url:-
http://wiki.jboss.org/wiki/Wiki.jsp?page=StartJBossOnBootWithLinux

Step 8
Edit following file and make appropriate changes in the datasource configuration.

```
[root@localhost ~]#vi  /home/jboss/jboss-6.0.0/server/default/deploy/mysql-ds.xml
eg:-
   <connection-url>jdbc:postgresql://serverip:5432/databasename</connection-url>

   <driver-class>org.postgresql.Driver</driver-class>

   <user-name>dbusername</user-name>

   <password>dbpassword</password>
```

Step 9
Copy filename.war file to particular path using following command

```
[root@localhost ~]#cp -v iASKWeb  /home/jboss/jboss-6.0.0/server/default/deploy/
```

Making your Jboss web application SSL enabled

Introduction:

With the growing need for information security in today's digital systems, SSL has become the universally accepted method for setting up authenticated and encrypted communication between clients and servers.

You can use your web server's SSL security features for two types of authentication. You can use a *server certificate* to allow users to authenticate your Web site before they transmit personal information, such as a credit card number. Also, you can use *client certificates* to authenticate users that request information on your Web site.

The security certificate can either be purchased from a Certificate Authority(CA) or it can generated on your own. This article assumes that you are creating your own self-signed certificate.

SSL configuration steps:

Step:1

Generate a self signed certificate using the "keytool" command as follows:

```
[root@localhost ~]# keytool -genkey -alias mohan –keyalg RSA –keystore training.keystore –validity 365

Enter keystore password:  password

What is your first and last name?

  [Unknown]:  TYPE NAME

What is the name of your organizational unit?

  [Unknown]:  IMFAST

What is the name of your organization?

  [Unknown]:  INTEGRA

What is the name of your City or Locality?

  [Unknown]:  Bangalore

What is the name of your State or Province?

  [Unknown]:  Karnataka

What is the two-letter country code for this unit?

  [Unknown]:  IN

Is CN=NAME, OU=IMFAST, O=INTEGRA, L=Bangalore, ST=Karnataka, C=IN correct?

  [no]:  yes


Enter key password for <somename>

        (RETURN if same as keystore password):  password

[root@localhost~]#
```

Step 2
To create a self-signed certificate using following command.

[root@localhost~]#keytool -selfcert -alias somname -keypass password -dname "CN=NAME, OU=IMFAST, O=INTEGRA, L=Bangalore, ST=Karnataka, C=IN" (Enter)
Enter keystore password: password

[root@localhost ~]#

Step 3
A keystore will be generated in the root home directory. If you do not give any name for your keystore, the default name will be ".keystore" .
Eg: .keystore path will be in /root/.keystore
Place this .keystore in "/home/jboss/jboss-6.0.0/server/default/conf/" directory.
[root@localhost ~]# chown jboss:jboss /home/jboss/jboss-6.0.0/server/default/conf/.keystore

Step 4
[root@localhost ~]#vi /home/jboss/jboss-6.0.0/server/default/deploy/jbossweb.sar/server.xml file and do the following:

```
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"
maxThreads="150" scheme="https" secure="true"
clientAuth="false"
keystoreFile="${jboss.server.home.dir}/conf/.keystore"
keystorePass="password"
truststoreFile="${jboss.server.home.dir}/conf/.keystore"
truststorePass="password" sslProtocol="TLS"/>
```

:wq (To save)

Then you can stop and start jboss again using following command.
[root@localhost ~]# su -l jboss (Enter)
[jboss@localhost ~]$ cd jboss-6.0.0/bin/ (Enter)
[jboss@localhost ~]$ sh shutdown.sh (To stop jboss)
[jboss@localhost ~]$ sh run.sh (To start jboss)

Then type following url in windows machine:-
https://serverip:8443/warfilename

Notes:-
Change the "keystorefile" name entry to the name you have given when you created the keystore.
The default password is "changeit", but it can changed while creating the keystore.
Since the certificate is self signed, your browser will not automatically identify the CA the first time

you access the page. Therfore you need to install the certificate into your browser's certificate store. This can done by clicking on the error message and then on Install Certificate.

Known Issues With Self signed certificates

IE7 and SSL Certificates:

The Internet Explorer 7 issues the following warning page whenever you access a web page with self signed certificate:

There is a problem with this website's security certificate.
The security certificate presented by this website was issued for a different website's address.
Security certificate problems may indicate an attempt to fool you or intercept any data you send to the server. We recommend that you close this webpage and do not continue to this website.

- Click here to close this webpage.
- Continue to this website (not recommended).

The fix is to open Internet Options in IE7, and then to uncheck the "Warn about certificates address mismatch":

Tools->Internet Options->Advanced->"Warn about certificate address mismatch"
Restart IE7, and the error will now go away.