SOLA Server Installation Guide

for Debian and Ubuntu

Revision History

Date	Version	Description	Author
Sep 30, 2013	0.1	Initial Draft	Andrew McDowell

Table of Contents

1. Introduction

This Installation Guide steps through installing the SOLA Server components on Debian using the **root** admin account. You can also use these instructions for installing on Ubuntu however you may need to prefix the commands listed in this document with **sudo**.

It is recommended that the computer hosting the server components of SOLA (Postgresql database and Glassfish) has a minimum of 300Gb free disk space for application data and backups.

1.1 Useful Commands

To determine if your server is 32bit or 64bit;

```
➢ uname -m
```

If the output is **x86_64** then 64bit, otherwise your server is 32bit (e.g. i386, i686, etc.).

To scroll output from a file to the screen;

```
➢ cat <filename> | more
```

To view the content of a file as it is being written to;

```
➢ tail -f <filename>
```

To run a command on a background thread use **&** at the end of the command. To bring a process into the foreground, use fg.

To check if a package is available to apt

```
➢ apt-cache search <package name>
```

2. Installation

2.1 Component Version Summary

This guide provides instructions for installing Java 7 (update 40), Glassfish 4, PostgreSQL 9.3 and PostGIS 2.1 into Debian and Ubuntu.

2.2 Installation Preparation

Debian and Ubuntu use the apt tool (Application Package Tool) to install and upgrade applications. apt is capable of downloading pre-configured installation packages from various online repositories however some of the packages required for SOLA are not directly available through these repositories due to licensing restrictions and/or the lack of a suitable install package. To assist with downloading the packages that are not available in the Debian repositories you should install aria2. This tool can perform multi-threaded downloads for large installation packages.

1) Use apt-get to install aria2

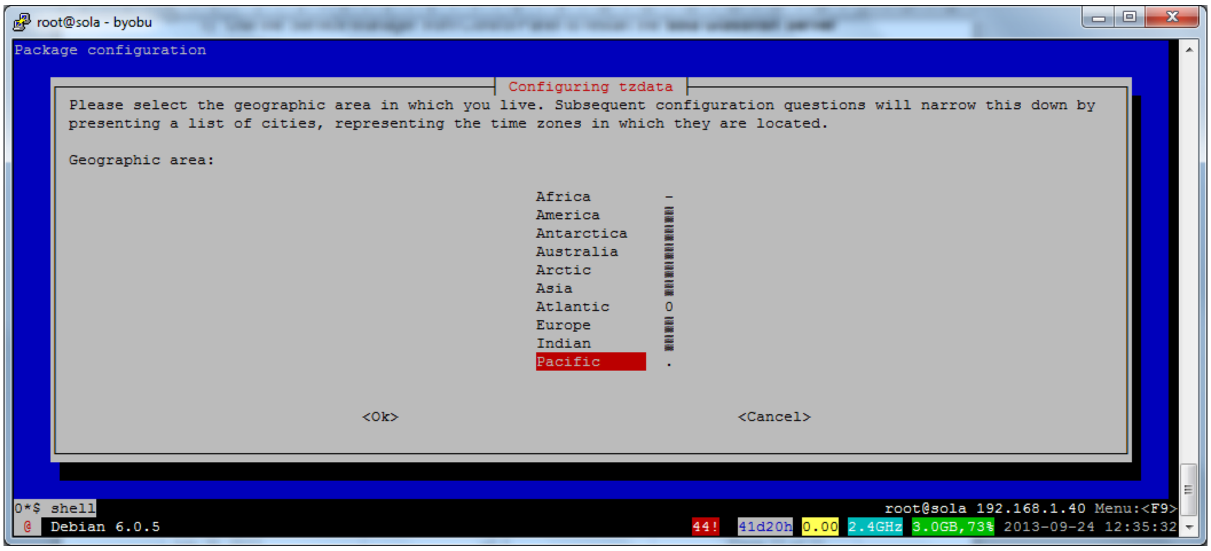
➢ `apt-get install aria2`

The security protocols used by SOLA are heavily reliant on the clock of the server computer and the client computer. Ideally these times will be synchronized by a Domain Network Time Service (NTS), however you should also verify the timezone on the server is set correctly.

2) Use the dpkg-reconfigure command to check the time zone and update it as necessary

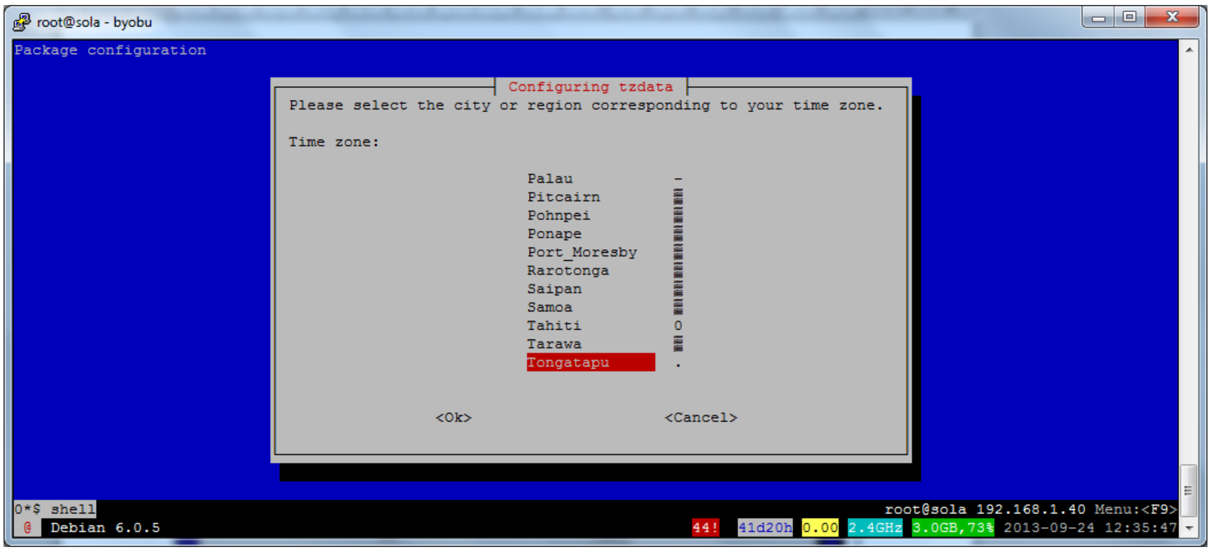
➢ `dpkg-reconfigure tzdata`

3) The dpkg-reconfigure command will display a command screen. Use the Arrow keys to select the correct area (e.g. Pacific) and press Enter



```
Package configuration

                          Configuring tzdata
 Please select the geographic area in which you live. Subsequent configuration questions will narrow this down by
 presenting a list of cities, representing the time zones in which they are located.

 Geographic area:

                                              Africa
                                              America
                                              Antarctica
                                              Australia
                                              Arctic
                                              Asia
                                              Atlantic
                                              Europe
                                              Indian
                                              Pacific

                          <Ok>                                        <Cancel>

0*$ shell                                                              root@sola 192.168.1.40 Menu:<F9>
 @  Debian 6.0.5                          44!  41d20h 0.00 2.4GHz 3.0GB,73% 2013-09-24 12:35:32
```

4) To complete the configuration, select the time zone (e.g. Tongatapu) and press enter



```
Package configuration

                    ┤ Configuring tzdata ├
 Please select the city or region corresponding to your time zone.

 Time zone:

                          Palau
                          Pitcairn
                          Pohnpei
                          Ponape
                          Port_Moresby
                          Rarotonga
                          Saipan
                          Samoa
                          Tahiti
                          Tarawa
                          Tongatapu

              <Ok>                        <Cancel>

0*$ shell                                   root@sola 192.168.1.40 Menu:<F9>
@ Debian 6.0.5     44! 41d20h 0.00 2.4GHz 3.0GB,73% 2013-09-24 12:35:47
```

5) You should optionally configure the firewall on your server to block everything apart from ports 22 (SSH), 80, 443 (HTTP/SSL), 4848, 8080, 8081 (Glassfish) and 5432 (Postgresql). The steps for doing this are beyond the scope of this document. The following article may provide some useful tips. https://wiki.debian.org/iptables

2.3 Install Java

OpenJDK is freely available for Linux distributions and can be installed using apt, however it is recommended to use the Oracle Java JDK virtual machine for SOLA. Due to licensing restrictions, the Oracle Java JDK cannot be installed using apt. You must manually download and install the Oracle Java JDK. The steps described here have been referenced from https://d.stavrovski.net/blog/installing-oracle-java7-on-debian-wheezy/

1) Download the Oracle Java JDK for Linux. Oracle requires the user to acknowledge the Oracle license before proceeding. This can be achieved using the following aria2 command. The command will download the file in the background (&) and you can monitor progress of the download by using tail –f out.log

for 32 bit OS

```
➢ aria2c --http-no-cache=true --check-certificate=false --header
  "Cookie: gpw_e24=http%3A%2F%2Fwww.oracle.com%2F"
  "http://download.oracle.com/otn-pub/java/jdk/7u40-b43/jdk-7u40-linux-
  i586.tar.gz" > out.log 2>&1 &
```

for 64 bit OS

```
➢ aria2c --http-no-cache=true --check-certificate=false --header
  "Cookie: gpw_e24=http%3A%2F%2Fwww.oracle.com%2F"
  "http://download.oracle.com/otn-pub/java/jdk/7u40-b43/jdk-7u40-linux-
  x64.tar.gz" > out.log 2>&1 &
```

2) Create a directory in opt and unzip the tar package there

```
➢ mkdir /opt/java-oracle
```

for 32 bit OS

```
➢ tar -zxf /tmp/jdk-7-linux-xi586.tar.gz -C /opt/java-oracle
```

for 64 bit OS

```
➢ tar -zxf /tmp/jdk-7-linux-x64.tar.gz -C /opt/java-oracle
```

3) Set Oracle Java 7 to be used as a default Java on the system by using a higher priority with update-alternatives.

```
➢ JHome=/opt/java-oracle/jdk1.7.0_40
➢ update-alternatives --install /usr/bin/java java ${JHome%*/}/bin/java
  20000
➢ update-alternatives --install /usr/bin/java javac
  ${JHome%*/}/bin/java 20000
```

4) Check the default java JRE is the one you just install using the update-alternatives command and checking the java version

```
➢ update-alternatives --config java
➢ java -version
```

2.4 Install PostgreSQL

Install PostgreSQL 9.3 using the following command.

```
➢ apt-get install postgresql-9.3 postgresql-client-9.3 postgresql-
  contrib-9.3
```

This command may take some time to complete as it will download and install the PostgreSQL 9.3 database server. Once the installation is complete, PostgreSQL will be running as a service. You can start, stop and restart PostgreSQL, reload the configuration files and list the status of the server using the postgres command in **/etc/init.d** e.g.

```
➢ /etc/init.d/postgres start
➢ /etc/init.d/postgres restart
➢ /etc/init.d/postgres stop
➢ /etc/init.d/postgres reload
➢ /etc/init.d/postgres status
```

The configuration files for PostgreSQL will be installed in **/etc/postgres/9.3/main**
The data directory for PostgreSQL will be **/var/lib/postgres/9.3/main**

To connect to the PostgreSQL from the server and run SQL commands will need to sudo to the postgres user account;

```
➢ su – postgres
➢ psql
```

This will place you at the psql command prompt. Type **help** for details on using psql. To exit psql use **\q <enter>.** To exit the postgres login type **exit**.

2.4.1 Set a password for the postgres super user

The default installation of PostgreSQL uses trust authentication which allows you to login as the postgres super user without entering a password. To be able to access the PostgreSQL server from another application or computer, you need to configure a suitable password for the postgres user account. This can be achieve from psql with the ALTER USER command as follows

```
➢ su – postgres
➢ psql
➢ postgres=# ALTER USER postgres WITH PASSWORD '<your passwd>';
```

2.4.2 Configure PostgreSQL to allow access from other computers

By default PostgreSQL locked down to prevent access from computers other than the localhost. If your database server will be used as the production server, then it may be a good option to leave PostgreSQL in this locked down state. If the database will be used for testing or training, it may be desirable to access the database server from other computers. To allow these connections it is necessary to make some configuration changes.

1) Edit the **/etc/postgres/9.3/main/pg_hba.conf** file and add the following line to the bottom of the file

```
hostssl all all 192.168.0.0 255.255.0.0 md5
```

2) Edit the **/etc/postgres/9.3/main/postgresql.conf** file and uncomment the listen_addresses parameter and set it to * e.g. `listen_addresses = '*'`
3) Reload the configuration of PostgreSQL

```
➢ /etc/init.d/postgres reload
```

You should now be able to connect to your PostgreSQL database using PgAdmin from another computer.

2.5 Install PostGIS

PostGIS is used by SOLA to manage spatial data in the database. Unfortunately there aren't any recently created packages to install PostGIS into Debian so it is necessary to build the PostGIS libraries from the source code. This requires installing a number of extra packages that the PostGIS build scripts are dependent on. Be aware that PostgreSQL 9.3 has had some architectural changes since 9.2 and only PostGIS 2.1.0 and above is compatible with PostgreSQL 9.3

These steps are based on the instructions available here http://trac.osgeo.org/postgis/wiki/UsersWikiPostGIS20Debian60src.The steps noted below have been revised to describe installing PostGIS 2.1 into PostgreSQL 9.3

1) As the user **root,** install packages required for the PostGIS build. Note that you may need to exit your shell if you logged in as postgres.

```
➢ apt-get install build-essential postgresql-server-dev-9.3 libxml2-dev
  libproj-dev libjson0-dev xsltproc docbook-xsl docbook-mathml
```

2) Download the latest source code for GEOS. Any version above 3.3.2 is compatible with PostGIS, but versions 3.4+ are recommended to make full use of the PostGIS 2.1 functionality.

```
➢ aria2c "http://download.osgeo.org/geos/geos-3.4.2.tar.bz2" > out.log
  2>&1 &
```

3) Once the download is complete, compile and install GEOS

```
➢ tar xfj geos-3.4.2.tar.bz2
➢ cd geos-3.4.2
```

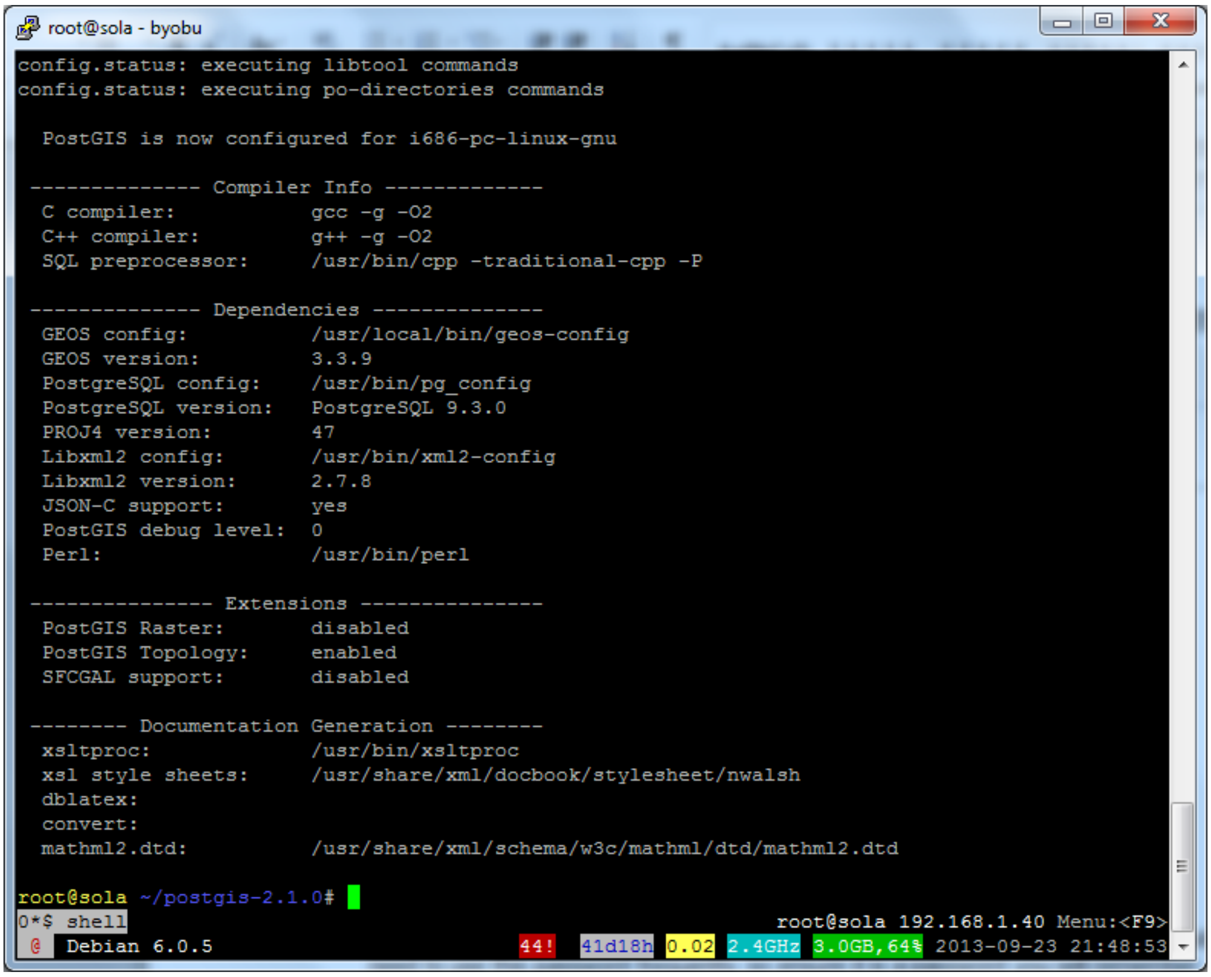
- `./configure`
- `make`
- `sudo make install`

4) Download the latest source code for PostGIS v2.1 (or above)

- `aria2c "http://download.osgeo.org/postgis/source/postgis-2.1.0.tar.gz" > out.log 2>&1 &`

5) Unzip and configure PostGIS. Note that SOLA does not use the raster image features of PostGIS, so the following steps will compile PostGIS without raster support. You should see output similar to the screenshot below after the configure command is complete.

- `tar xfz postgis-2.1.0.tar.gz`
- `cd postgis-2.1.0`
- `./configure --without-raster`



```
config.status: executing libtool commands
config.status: executing po-directories commands

  PostGIS is now configured for i686-pc-linux-gnu

 -------------- Compiler Info -------------
  C compiler:           gcc -g -O2
  C++ compiler:         g++ -g -O2
  SQL preprocessor:     /usr/bin/cpp -traditional-cpp -P

 -------------- Dependencies --------------
  GEOS config:          /usr/local/bin/geos-config
  GEOS version:         3.3.9
  PostgreSQL config:    /usr/bin/pg_config
  PostgreSQL version:   PostgreSQL 9.3.0
  PROJ4 version:        47
  Libxml2 config:       /usr/bin/xml2-config
  Libxml2 version:      2.7.8
  JSON-C support:       yes
  PostGIS debug level:  0
  Perl:                 /usr/bin/perl

 --------------- Extensions ---------------
  PostGIS Raster:       disabled
  PostGIS Topology:     enabled
  SFCGAL support:       disabled

 -------- Documentation Generation --------
  xsltproc:             /usr/bin/xsltproc
  xsl style sheets:     /usr/share/xml/docbook/stylesheet/nwalsh
  dblatex:
  convert:
  mathml2.dtd:          /usr/share/xml/schema/w3c/mathml/dtd/mathml2.dtd

root@sola ~/postgis-2.1.0#
0*$ shell                                       root@sola 192.168.1.40 Menu:<F9>
 @  Debian 6.0.5                    44! 41d18h 0.02 2.4GHz 3.0GB,64% 2013-09-23 21:48:53
```

Figure 1 - Output following compile of PostGIS

6) Compile and install PostGIS and supporting components

- `make`
- `sudo make install`
- `sudo ldconfig`
- `sudo make comments-install`

7) Create a template database and run the PostGIS enabler scripts. To do this you need to login as the **postgres** user.

```
➢ su - postgres
➢ createdb template_postgis_21
➢ createlang plpgsql template_postgis_21
➢ psql -d template_postgis_21 -c "UPDATE pg_database SET
  datistemplate=true WHERE datname='template_postgis_21'"
➢ psql -d template_postgis_21 -f
  /usr/share/postgresql/9.3/contrib/postgis-2.1/postgis.sql
➢ psql -d template_postgis_21 -f
  /usr/share/postgresql/9.3/contrib/postgis-2.1/spatial_ref_sys.sql
➢ psql -d template_postgis_21 -f
  /usr/share/postgresql/9.3/contrib/postgis-2.1/postgis_comments.sql
```

2.6 Install the uuid-ossp extension

The uuid-ossp PostgreSQL extension provides GUID generation functions. It must be installed as an extension into PostgreSQL. The source files for this extension are included in the postgresql-contrib-9.3 package which is installed as part of the PostgreSQL installation steps. To create the extension in the database you must be logged in as **postgres**.

```
➢ su - postgres
➢ psql -d template_postgis_21 -c "CREATE EXTENSION \"uuid-ossp\";"
```

2.7 Create the SOLA Database

These steps are only required the very first time you create a new SOLA database in Postgresql.

1) As **root**, restart the Postgresql database to release all connections to the template database

```
➢ /etc/init.d/postgres restart
```

2) To create the database you must be logged in as **postgres**. The name of your sola database should reflect the environment it will be used for. E.g. for a live/production environment the suggested name of the database is **sola_prod**. For a test environment, the suggested name of the database is **sola_test**, etc.

```
➢ su - postgres
➢ createdb -T template_postgis_21 <sola DB name>
```

3) Run the appropriate SOLA Database build script (e.g. create_soladb_tonga.bat) to build and populate the empty database or restore a database backup. It may be necessary to do this build across the network from a Windows computer if no suitable shell script is available. To perform a build across the network, you will need to configure PostgreSQL to allow access from an external computer. See section 1.5.2 for details.

2.8 Install Glassfish

Glassfish4 has been released in May 2013 and is the recommended version of Glassfish to use for SOLA. These steps are based on the installation instructions provided here http://www.physics.usyd.edu.au/~rennie/glassfish.html.

1) As root, create a user with restricted rights to run the Glassfish server.

```
➢  adduser --system --group --home /home/glassfish glassfish
```

2) Still as root, download the Glassfish4 zip bundle from the Glassfish web site. This download is approximately 97Mb.

```
➢ cd /home/glassfish
➢ aria2c "http://download.java.net/glassfish/4.0/release/glassfish-
  4.0.zip" > out.log 2>&1 &
➢ unzip glassfish-4.0.zip -d /opt
```

3) Change the name of the default domain from domain1 to sola

```
➢ mv /opt/glassfish4/glassfish/domains/domain1
  /opt/glassfish4/glassfish/domains/sola
```

4) Change the ownership of the files to the glassfish user and group and setup owner and group rights. Remove world/other rights from the glassfish files.

```
➢ chown -R glassfish:glassfish /opt/glassfish4
➢ chmod -R ug+rwx /opt/glassfish4/bin
➢ chmod -R ug+rwx /opt/glassfish4//glassfish/bin
➢ chmod -R ug+rwx /opt/glassfish4/glassfish/domains/sola/autodeploy
➢ chmod -R o-rwx /opt/glassfish4/bin
➢ chmod -R o-rwx /opt/glassfish4//glassfish/bin
➢ chmod -R o-w /opt/glassfish4/glassfish/domains/sola/autodeploy
```

5) Create an init script to automatically start and stop Glassfish as a service. The content of the init script is shown in the following table.

```
➢ vi /etc/init.d/glassfish
```

```
#! /bin/sh
### BEGIN INIT INFO
# Provides:          glassfish
# Required-Start:    $remote_fs $network $syslog
# Required-Stop:     $remote_fs $network $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Starts GlassFish
# Description:       Starts GlassFish application server
### END INIT INFO

# We need this, since NetworkServerControl uses $JAVA_HOME to find java
export JAVA_HOME=/opt/java-oracle/jdk1.7.0_40               # Or whatever
# We need this, since asadmin contains the line 'JAVA=${AS_JAVA}/bin/java'
export AS_JAVA="$JAVA_HOME"

GLASSFISH=/opt/glassfish4

case "$1" in
start)
  echo "Starting GlassFish from $GLASSFISH"
  sudo -u glassfish -E "$GLASSFISH/bin/asadmin" start-domain sola
  ;;
stop)
```

```
  echo "Stopping GlassFish from $GLASSFISH"
  sudo -u glassfish -E "$GLASSFISH/bin/asadmin" stop-domain sola
  ;;
restart)
  $0 stop
  $0 start
  ;;
status)
  echo "# GlassFish at $GLASSFISH:"
  sudo -u glassfish -E "$GLASSFISH/bin/asadmin" list-domains | grep -v
Command
  sudo -u glassfish -E "$GLASSFISH/bin/asadmin" list-domains | grep -q
"sola running"
  if [ $? -eq 0 ]; then
    sudo -u glassfish -E "$GLASSFISH/bin/asadmin" uptime | grep -v Command
    echo "\n# Deployed applications:"
    sudo -u glassfish -E "$GLASSFISH/bin/asadmin" list-applications --
long=true --resources | grep -v Command
    echo "\n# JDBC resources:"
    sudo -u glassfish -E "$GLASSFISH/bin/asadmin" list-jdbc-resources |
grep "jdbc/"
  fi
  ;;
*)
  echo "Usage: $0 {start|stop|restart|status}"
  exit 1
  ;;
esac

exit 0
```

6) Make the script executable and attempt to start Glassfish. Wait 30 seconds and try to access the Glassfish Admin console at port 4848 (e.g. localhost:4848).

```
➢ chmod ug+x /etc/init.d/glassfish
➢ /etc/init.d/glassfish start
```

Once you have confirmed that Glassfish is up and running, stop it using the following command

```
➢ /etc/init.d/glassfish stop
```

7) Add the init.d script to autostart so that Glassfish will startup when the server is rebooted.

```
➢ update-rc.d glassfish defaults
```

At this point you will have a Glassfish instance that is configured to run as a service under the glassfish user account. There are several additional security configurations you should complete if this instance of Glassfish will operate on the production server. For details on how to proceed to harden you glassfish instance, refer to the Hardening Glassfish topic on the SOLA Wiki http://wiki.flossola.org/wiki/Hardening_Glassfish.

3. Configure Glassfish

3.1 Copy PostgreSQL JDBC Driver and SSH Key Files

Before you start the new Glassfish domain, you should copy the PostgreSQL JDBC Driver and the default development SSH Key files into the domain folder. Glassfish will expand the JDBC Driver file on startup and as well as load the new SSH key files. If Glassfish is already running then restart Glassfish after you have completed copying the files so that Glassfish recognizes the changes.

1) Obtain a copy of the postgresql-9.2-1002.jdbc4.jar file (from http://jdbc.postgresql.org/download.html) and the SOLA cacerts.jks and keystore.jks SSH key files. You may need to request the SSH key files from a member of the SOLA Team.
2) Copy the postgresql-9.2-1002.jdbc4.jar into the **/opt/glassfish4/glassfish/domains/sola/lib** folder of the new sola domain.
3) Copy the cacerts.jks and keystore.jks SSH key files into the **/opt/glassfish4/glassfish/domains/sola/config** folder. You will need to replace the cacerts.jks and keystore.jks files in that directory.
4) Restart Glassfish .

The key files you replaced were the default key files for Glassfish. They do not include the necessary configuration to support message encryption. The key files you copied in have been generated as the default key files for SOLA development purposes and these files match the key files used by the SOLA Web Services Client library for message encryption. When deploying to a production environment, new key files should be generated for the production Glassfish instance. The task to generate new key files is beyond the scope of this document, but instructions on how to do this are provided in the SOLA Development Wiki. See http://wiki.flossola.org/wiki/Hardening_Glassfish.

3.2 Configure JDBC Connection

1) Browse to the Glassfish Admin console at **<server>:4848**. You will need to login to the Admin Console as admin using the password from section 1.3 step 9 c).
2) In the Glassfish Admin Console, locate the Resources > JDBC > JDBC Connection Pools node
3) Click the New button
4) In the New JDBC Connection Pool (Step 1 of 2)
    a. Pool Name = sola
    b. Resource Type =javax.sql.ConnectionPoolDataSource
    c. Database Driver Vendor = Postgresql
5) Click Next and verify the Datasource Classname is org.postgresql.ds.PGConnectionPoolDataSource
6) Scroll down to the Additional Properties and set the following values
    a. User = postgres
    b. DatabaseName = sola
    c. Password = <your postgres password>
7) Click Finish, then click the new Connection Pool you just created and on the Edit JDBC Connection Pool, click **Ping**. You should get a Ping Succeeded message.
8) It is now necessary to add a new JNDI resource for the connection pool. The JNDI name is the value used by the Java Naming Directory Interface (JNDI) resolution to reference/locate the sola connection pool resource. Go to the JDBC > JDBC Resources node and click New...
9) Set the JNDI Name to jdbc/sola. It is important to use this name as this is the value referenced by the MyBatis connection configuration files.
10) Select sola as the Pool Name and click OK to create the new JDBC Resource.

3.3 Configure Security Realm

SOLA delegates authentication of user credentials to a Glassfish JDBC Security Realm. It is possible to configure a variety of Security Realms in Glassfish, and any of these could be used for SOLA. The JDBC Security Realm is used because it references the user credentials directly from the sola database which greatly simplifies the administration of user details using SOLA Admin.

1) From the Configurations > server-config > Security > Realms node, select New...
2) Set the following values. All other values should remain blank.
    a. Name = SolaRealm
    b. Class Name = com.sun.enterprise.security.auth.realm.jdbc.JDBCRealm
    c. JAAS Context = jdbcRealm
    d. JNDI = jdbc/sola
    e. User Table = system.appuser
    f. User Name Column = username
    g. Password Column = passwd
    h. Group Table = system.user_roles
    i. Group Name Column = rolename
    j. Password Encryption Algorithm = MD5 (Only required for Glassfish 4+)
    k. Digest Algorithm = SHA-256
3) Click OK to save the new security realm.
4) Click the Configurations > server-config > Security node
5) Check the **Default Principal To Role Mapping** checkbox to enable this setting. SOLA does not include any specific role mapping configuration and relies on the default mapping provided by Glassfish (i.e. roles from the security realm have a one to one mapping to roles in the application based on role name).
6) Save this change

3.4 Configure JVM Settings

This section customizes a minimal set of JVM settings for the sola application domain. For production deployment, it may be necessary to further customize other JVM settings to improve the overall performance of Glassfish and to harden the Glassfish instance minimizing potential security risks.

Warning

- If you copy and paste the settings below (e.g. –server) be aware the hyphen (-) has been converted automatically by Word to a dash (–). You will need to revert it to a hyphen!

1) Click the Configurations > server-config > JVM Settings node
2) Click the JVM Options tab
3) Change –client to –server. This will ensure the Server JVM is used to run the application domain rather than the client JVM.
4) Change the –Xmx512m option to –Xmx1280m. This will allow the JVM to use up to 1.25GB of RAM.
5) Add a new JVM Option and set this option to –Xms640m. This will set the minimum size of the RAM required by the JVM. The default –Xms is 64m. When the JVM reaches 64m, it will allocate another block of 64m. To avoid the need to perform a number of memory allocations, it is generally recommended to set the –Xms to half of or equal to the –Xmx setting.

6) Save your changes. You should get a message indicating the save was successful. Changes to the JVM settings will also require a restart of Glassfish, however you should complete all of the configurations below before restarting.

3.5 Configure Logger Settings

For development purposes it is useful to capture the SQL queries that are sent from Glassfish to the database in the Glassfish Log. The Glassfish Log is displayed in the Output View by Netbeans making it very convenient to follow the SQL the application sends to the database as the application is running. Note that these settings are not required in production unless it is necessary to capture the database queries for debugging purposes.

1) Click the Configurations > server-config > Logger Settings node
2) Click the Log Levels tab
3) Click the Add Logger button
4) Enter the new logger name java.sql and set the Log Level to FINE.
5) Click the Add Logger button and add a logger called java.sql.Connection with the Log Level set to FINE. Note that the java.sql and java.sql.Connection loggers must both be set to FINE to log SQL statements.
6) Click the Add Logger button and add a logger called org.sola.services with the Log Level set to INFO. This is the logger used by the LogUtility in SOLA. Exception messages and other useful details are recorded in the Glassfish Log through this logger.
7) Click the Add Logger button and add a logger called java.sql.ResultSet with the Log Level set to OFF. This logger will capture all results returned from the database however logging every result has a significant performance impact. This logger should only be turned to the FINE level for debugging specific issues and then turned OFF again.

You can also view the Glassfish Log directly from the Glassfish Admin Console. Click the View Log Files on the server (Admin Server) node. You can also rotate the log if necessary.

3.6 Restart Glassfish

1) Restart Glassfish

➤ `/etc/init.d/glassfish restart`

2) Check the Admin Console is available to ensure the configurations are valid. You will need to login to the Admin Console using as admin using the password from section 1.3 step 9 c).

You now have an instance of Glassfish ready to deploy the sola-services-ear.ear into.