Installing DSpace 1.6 on Red Hat Enterprise Linux 5

Dspace 1.6.x on Red Hat Enterprise Linux 5 (RHEL5)

This documentation outlines how to install Dspace (1.6.2) on a Red Hat Enterprise Linux 5 (RHEL5) server with an emphasis on sustainability from the system administrative point of view. This is not necessarily THE way to do it, but if you follow this document, you will have a basic DSpace server up in running in the RHEL environment, and will really not need to worry about software updates.

The hardest issues to resolve were finding the prerequisite packages for **Maven2** and **Ant 1.7** (and resolving the dependencies of these packages) which are not in RHEL repos, *AND* still **use RPMs** to maintain the packages. The following is my attempt to use an external repository (JPackage) with RHEL5 and the only unmaintained packages are Dspace itself and possibly the Java JAI.

Goals:

1. Use package management for all requisite software, save the Dspace code itself.
2. Automatic updates must work.
3. Dependency problems between RHEL5 and the JPackage 5 repository must be resolved so it doesn't prevent the previous step. (auto-updates)
4. Remove the "dspace" user from the system and run the service as the existing user "tomcat".

Assumptions

- **Standard RHEL5 (32-bit) server** - This build/document assumes you are running RHEL5_, but it could possibly be adapted to a CentOS5 system._
    - **RHEL Supplementary channel** (v. 5 for 32-bit x86) - This is what provides a source for Sun Java updates. It is assumed you have a valid Red Hat Network account that you can manage your systems.
- **Package Management** - All required software for DSpace is provided and managed using package management, in our case RPM's. It is assumed software from external repositories; get regular updates, are trusted as secure, and all packages are maintained regularly.

Install

Official Dspace 1.6 Install Documentation: http://www.dspace.org/1_6_0Documentation/ch03.html
(*for reference purposes and the instructions I used to write this document*)

Check and Install Pre-requisite software

(*A list of software requirements for this service.*)

*Login as **root** if you are not already.*

- **Requirement -** Perl (no version listed.) – ***RHEL5*** *currently provides perl 5.8.8.* and was Installed by our default RHEL5 build.
- **Requirement -** Sun Java 1.5 or better, we will install Sun Java 1.6 -
  (***RHEL5*** *currently provides Sun Java 1.5 & 1.6* through the RHEL Supplementary (v. 5 for 32-bit x86) channel)
    1. **Requires** RHEL Supplementary (v. 5 for 32-bit x86) channel to get access to the Sun Java.
       This is done through the web interface @ https://rhn.redhat.com
    2. Install Sun Java 6:

       ```
       yum install java-1.6.0-sun java-1.6.0-sun-devel -y
       ```

       Output for Sun Java 6 install:

```
Dependencies Resolved

================================================================================
 Package        Arch   Version           Repository                       Size
================================================================================
Installing:
 java-1.6.0-sun i586   1:1.6.0.20-1jpp.1.el5
                                         rhel-i386-server-supplementary-5  37 M
 java-1.6.0-sun-devel
                i586   1:1.6.0.20-1jpp.1.el5
                                         rhel-i386-server-supplementary-5  20 M
Installing for dependencies:
 jpackage-utils noarch 1.7.3-1jpp.2.el5  rhel-i386-server-5                61 k

Transaction Summary
================================================================================
Install       3 Package(s)
Upgrade       0 Package(s)

Total download size: 57 M
```

Notice here

Installing Java the jpackage-utils 1.7 is also installed which is maintained by Red Hat. Later on it will be replaced by the JPackage maintained version and can/will cause some problems after we install and start using the JPackage repo.

- **Requirement -** Relational Database, PostgreSQL 7.4 or later - **RHEL5** currently provides postgres 8.1.18(default) (8.4 is available too.)
  1. Install Postgresql:

```
yum install postgresql-server -y
```

This will install the postgresql dependency.
Output for Postgresql Server install:

```
Dependencies Resolved

================================================================================
Package              Arch    Version               Repository             Size
================================================================================
Installing:
postgresql-server    i386    8.1.21-1.el5_5.1      rhel-i386-server-5    3.8 M
Installing for dependencies:
postgresql           i386    8.1.21-1.el5_5.1      rhel-i386-server-5    2.9 M

Transaction Summary
================================================================================
Install       2 Package(s)
Upgrade       0 Package(s)

Total download size: 6.7 M
```

Clean Install

We have a Clean install, using only RHEL provided packages, up to this point.

Install and Configure the JPackage repository

(*Ok, this is somewhat **dirty** and I don't know if this work work in a sustainable long-term way, but here goes...*)

Objective Statement: Only packages that are specifically needed, either explicitly or to resolve a dependency on JPackage will be used, otherwise the RHEL channel provided package will be used.

Footnote STUB: I go into detail why I didn't use *_yum-priorities_* in my more ideal install attempt:[\#alternative_install_methodyum-protectbase]. Contact me if you want more information on this.

1. Install a custom repo file to use Jpackage repository v5.0:

```
wget -P/etc/yum.repos.d http://librhn.unm.edu/pub/jpackage50-ant17-maven2-mod.repo
```

   The contents of the config is listed at the end of this document: Config Files
2. Install the GPG key from JPackage:

```
rpm --import http://www.jpackage.org/jpackage.asc
```

3. Update the system to use the new jpackage-utils 5.0 package:

```
yum update -y
```

   Notice

   *Unfortunately this has a bad side effect of removing **/usr/bin/rebuild-security-providers**, which was maintained in the RHEL 5 jpackage-utils 1.7, but not in the new one provided by Jpackage.*

   This is also a known RHEL5 + JPackage bug: https://bugzilla.redhat.com/show_bug.cgi?id=497213
4. Now we need to install a workaround, to address the missing **/usr/bin/rebuild-security-providers** dependency:

```
rpm -ivh http://plone.lucidsolutions.co.nz/linux/centos/images/jpackage-utils-compat-el5-0.0.1-1.
noarch.rpm
```

   It's a mess... but it's working.
   Footnote STUB: I used directions I found for CentOS 5: http://plone.lucidsolutions.co.nz/linux/centos/jpackage-jpackage-utils-compatibility-for-centos-5.x This link is also useful, and somewhat related to getting the repo config file we used to work: http://plone.lucidsolutions.co.nz/linux/centos/jpackage-rpm-repository-for-centos-rhel-5.x

Install Tomcat5, Ant 1.7 and Maven 2 with jpackage filters

(*We continue installing prerequisite software for Dspace 1.6 on RHEL5...*)

- **Requirement** Jakarta Tomcat 4.x or later - ***RHEL5*** *currently provides Tomcat 5.5.23.*
  1. Install Tomcat5 and postgres-jdbc connector:

```
yum install tomcat5 tomcat5-webapps postgresql-jdbc geronimo-javamail-1.4-api -x classpathx-mail -y
```

     *This will be a **large** list of dependencies to be installed, BUT the majority of the packages will be installed from official RHEL 5 channels. This is a good thing. Also removed/excluded the default java mail package classpathx-mail (gnu) as it wasn't working with Dspace, replaced with geronimo java mail. You may get an error here when tomcat5-common-lib is installed, but I don't think it is a game-stopper.*
  2. Since we changed the java mailer, we need to update **alternatives** to use the Geronimo version, which doesn't configure itself upon install:

```
alternatives --install /usr/share/java/javamail.jar javamail /usr/share/java/geronimo-javamail-1.4-
api-1.1.jar 666
```

     This step is necessary, otherwise Tomcat reports errors every time it starts up.
- **Requirement** Apache Ant 1.7 or later (Java build tool) – ***RHEL5*** *currently provides ant-1.6.5.* – Ant 1.7 was installed as a dependency for Tomcat5 via the jpackage repo, per the work we did in the jpackage repo config file.(*previous step*)Let's install a missed Ant related package which is required for a Dspace 1.6 build:

```
yum install ant-apache-regexp -y
```

- **Requirement** Apache Maven 2.0.8 or later (Java build tool) – The Maven2 install turns out to be the most complicated package to install on this system because the **HUGE** number of dependent packages. From the work done when creating the custom jpackage repo file(*previous steps*), we can install:

```
yum install maven2 -y
```

Java Advanced Imaging (JAI) package install

(*This package was created in a previous build, and was declared required for out install, so all we need to do is install it. I'm assuming it works. This is apparently what rescales* TIFF *images into jpg thumbnails.*)

1. Install the Java Advanced Imaging (JAI) rpm:

```
rpm -ivh http://librhn.unm.edu/pub/jai-1.1.2.01-1jpp.i586.rpm
```

Something to keep in mind: As an Alternative to JAI, possibly use ImageMagick as a sustainable (maintained by the OS) package for generating thumbnails from TIFF files. — [Jamin Ragle] *2010/06/07 16:28*

Configure the RHEL5 Environment for Dspace 1.6

(*At this point, we should have all the pre-requisite software installed. Now we need to make a few tweaks to make the System a good and sustainable environment for Dspace 1.6.*)

Configure the Web Services

1. Update */etc/tomcat5/tomcat5.conf* to turn on UTF-8 and some Java memory settings in Tomcat5:

```
vi /etc/tomcat5/tomcat5.conf
```

```
#JAVA_OPTS="-Xminf0.1 -Xmaxf0.3"
JAVA_OPTS="-Xmx512M -Xms64M -Dfile.encoding=UTF-8"
```

2. Edit Tomcat server config to use the **appBase dspace webapps** directory and use **UTF-8** file encoding:

```
vi /etc/tomcat5/server.xml
```

Add URIEncoding="UTF-8", when finished it looks like the following:

```
<Connector port="8080" URIEncoding="UTF-8" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" redirectPort="8443" acceptCount="100"
connectionTimeout="20000" disableUploadTimeout="true" />
```

Also URIEncoding="UTF-8" for port 8443 & the AJP 1.3 Connector:

```
<!-- Define a SSL HTTP/1.1 Connector on port 8443 -->
<!--
<Connector port="8443" URIEncoding="UTF-8" maxHttpHeaderSize="8192"
maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
enableLookups="false" disableUploadTimeout="true"
acceptCount="100" scheme="https" secure="true"
clientAuth="false" sslProtocol="TLS" />
-->

<!-- Define an AJP 1.3 Connector on port 8009 -->
<Connector port="8009" URIEncoding="UTF-8"
enableLookups="false" redirectPort="8443" protocol="AJP/1.3" />
```

Change the default appBase to use the dspace/webapps directory and add some new Context path's:

```
<Host name="localhost" appBase="/opt/dspace/webapps"
unpackWARs="true" autoDeploy="true"
xmlValidation="false" xmlNamespaceAware="false">

<!-- DEFINE A CONTEXT PATH FOR DSpace JSP User Interface  -->
<Context path="/jspui" docBase="/opt/dspace/webapps/jspui" debug="0" reloadable="true" cachingAllowed="
false" allowLinking="true"/>

<!-- DEFINE A CONTEXT PATH FOR DSpace XML User Interface (Manakin) -->
<Context path="/xmlui" docBase="/opt/dspace/webapps/xmlui" debug="0" reloadable="true" cachingAllowed="
false" allowLinking="true"/>

<!-- DEFINE A CONTEXT PATH FOR DSpace OAI User Interface  -->
<Context path="/oai" docBase="/opt/dspace/webapps/oai" debug="0" reloadable="true" cachingAllowed="
false" allowLinking="true"/>
```

3. Configure **mod_proxy_ajp** connector, */etc/httpd/conf.d/proxy_ajp.conf*:

```
vi /etc/httpd/conf.d/proxy_ajp.conf
```

Append this to the end of the file:

```
# Always keep the host header
ProxyPreserveHost On

# Dspace related rules
#ProxyPass /do_not_touch         \!
ProxyPass /                     ajp://localhost:8009/
ProxyPassReverse /              ajp://localhost:8009/
ProxyPassReverseCookiePath /    /
```

Note: If you want to block the ajp connector from connecting to something, the "!" <bang> sets Apache web-server to NOT REDIRECT everything under */do_not_touch*

4. Configure the default vhost:

```
vi /etc/httpd/conf.d/dspace.conf
```

Pretty simple configuration, don't forget to use your own hostname. 🙂

```
# Dspace vhost
<VirtualHost _default_:80>
ServerName repository.unm.edu
DocumentRoot "/opt/dspace/webapps"
ErrorLog logs/repository-error_log
CustomLog logs/repository-access_log common

RewriteEngine On
RewriteRule \^/dspace/(.*) /$1 [R=permanent]
</VirtualHost>
```

Configure the Postgresql DB Service

1. Create a override config for Dspace to relocate the DB to /opt/dspace/database:

```
vi /etc/sysconfig/pgsql/postgresql
```

Contents are as follows:

```
# New Location for the Dspace install of the database
PGDATA=/opt/dspace/database
```

2. Start the PostgreSQL Service to generate the initial config files:

```
service postgresql start
```

3. Edit /opt/dspace/database/postgresql.conf:

```
vi /opt/dspace/database/postgresql.conf
```

uncomment the line #listen_address = 'localhost'

```
listen_addresses = 'localhost'
```

4. Edit /opt/dspace/database/pg_hba.conf:

```
vi /opt/dspace/database/pg_hba.conf
```

and modify the "local" connections like the following:

```
local   all         all                               trust
host    all         all         127.0.0.1/32          trust
```

Warning: Setting these connections to "trust" is less than ideal from a security perspective. This works for us as we do not allow any external connections to the database. If you want to allow external connections, this setting should be at least md5. The "trust" setting is still relatively safe as long as you **only** allow local connections.

5. Then restart PostgreSQL:

```
service postgresql restart
```

6. Create the dspace database user and initial database:

```
su -c "createuser -U postgres -d -A -R -P dspace" postgres;
su -c "createdb -U dspace -E UNICODE dspace" postgres
```

You will be prompted for a password for the DSpace database.
Note: You may get a error if you were still in the /root home directory when you ran the commands. This can be safely ignored.

Setup the user environment

(In this section, I will set up appropriate sudo access for our web admins and configure a few initial users.)

1. Setup sudo access for WEBADMINS:

```
visudo
```

Append this to the end of the file:

```
## Sudo rules for the Postgresql service
User_Alias DBADMINS = <insert your users or groups>
Cmnd_Alias PSQL_CTRL = /sbin/service postgresql *
Cmnd_Alias POSTGRES_USR = /bin/su - postgres
DBADMINS   ALL= PSQL_CTRL, POSTGRES_USR

## Sudo rules for people who are Web Admins:
## They can influence Apache, Tomcat5, and the Dspace Handle Service.
User_Alias WEBADMINS = <insert your users or groups>
Cmnd_Alias WEB_EDIT_CFGS = /usr/bin/rvim /etc/php.*, /usr/bin/rvim /etc/httpd/*, /usr/bin/rvim /etc
/tomcat5/\*
Cmnd_Alias APACHE_CTRL = /sbin/service httpd *
Cmnd_Alias TOMCAT5_CTRL = /sbin/service tomcat5 *
Cmnd_Alias HANDLE_CTRL = /sbin/service dspace-handle *
WEBADMINS  ALL= APACHE_CTRL, TOMCAT5_CTRL, HANDLE_CTRL, WEB_EDIT_CFGS

## Some Dspace specific rules, required since we don't have a "dspace" user.
User_Alias DS_ADMINS = <insert your Dspace developer(s)>
Cmnd_Alias DS_BUILD = /usr/bin/ant *
Cmnd_Alias DS_CMDS = /opt/dspace/bin/\*
Cmnd_Alias DS_HANDLE_CFG = /usr/bin/rvim /opt/dspace/handle-server/config.dct
```

```
DS_ADMINS ALL=(tomcat) DS_BUILD, DS_CMDS, DS_HANDLE_CFG
DS_ADMINS ALL= /opt/dspace/sbin/make_xmlui_active
```

2. Add a helper script when DSpace updates occur:

```
su -c "mkdir /opt/dspace/sbin" tomcat; vi /opt/dspace/sbin/make_xmlui_active
```

The contents are simply:

```
# Make the xmlui the default ROOT, since any DSpace updates clobber the old one.
su -c "cd /opt/dspace/webapps; ln -s ./xmlui ROOT" tomcat
```

3. Make sure we flag the execute bit:

```
chmod +x /opt/dspace/sbin/make_xmlui_active
```

Note: There is a matching sudo rule for this above, which ends up running as **root** to force ownership as tomcat. This file is owned by root and cannot be modified by anyone else on the system, other than root. It's purpose is to put the ROOT link back to xmlui in /opt/dspace/webapps after every update. Unfortunately it couldn't be placed in dspace/bin, since DSpace nukes that directory regularly.

Dspace 1.6 Install

(*The following configure's Dspace 1.6 for our UNM environment.*)

- Create the directory for the DSpace source.

```
chown tomcat:tomcat /opt/dspace; su -c "mkdir -p /opt/dspace/src" tomcat;
```

- Get the Dspace 1.6 Source:

```
su -c "wget -P/opt/dspace/src http://sourceforge.net/projects/dspace/files/DSpace%20Stable/1.6.2/dspace-1.6.2-src-release.tar.gz/download" tomcat
```

As a **non-root** user. (Usually *your* account.)

1. Slightly adjust your $HOME permissions so the tomcat user will be able to read the source:

```
chmod +rx $HOME
```

2. Unpackage the source to *your* home directory:

```
mkdir ~/build; tar -zxvf /opt/dspace/src/dspace-1.6.2-src-release.tar.gz -C ~/build
```

3. Edit [dspace-source]/config/dspace.cfg, in particular you'll need to set these properties:

```
vi ~/build/dspace-1.6.2-src-release/dspace/config/dspace.cfg
```

Anything not listed, leave as the default setting.

- **dspace.dir** – must be set to the [dspace] (installation) directory
  dspace.dir = /opt/dspace
- **dspace.hostname** – fully-qualified domain name of web server
  dspace.hostname = repository.unm.edu
- **dspace.baseUrl** – complete URL of this server's DSpace home page
  dspace.baseUrl = http://repository.unm.edu
- **dspace.url** – DSpace base URL
  Unknown macro: {dspace.baseUrl}
- **dspace.name** – Proper name of your server
  dspace.name = UNM Digital Repository
- **db.password** – the database password you entered in the previous step for the 'dspace' user.
  db.password = XXXXXXXX
- **mail.server** – fully-qualified domain name of your outgoing mail server.
  mail.server=smtp.unm.edu

- **mail.server.username** – SMTP mail server authentication username.mail.
  server.username = reposit
- **mail.server.password** – SMTP mail server authentication password. (Hopefully you remember this.)
  mail.server.password =
- **mail.from.address** – the "From:" address to put on email sent by DSpace.
  mail.from.address = reposit@unm.edu
- **feedback.recipient** – mailbox for feedback mail.
  feedback.recipient = reposit@unm.edu
- **mail.admin** – mailbox for DSpace site administrator.
  mail.admin = reposit@unm.edu
- **alert.recipient** – mailbox for server errors/alerts (not essential but very useful!)
  alert.recipient = XXXX@unm.edu

4. Start the build of Dspace:

```
cd ~/build/dspace-1.6.2-src-release/dspace; mvn package
```

This will run for a long time.
Note: The "mvn package" command needs to be run as a real user, this can be either root or your own personal account. I do not know the reason maven insists on creating a $HOME/.m2 directory where more packages are downloaded from the maven repository. (Not the authority in this JAVA development world...)

5. Build Dspace and initialize the Database:

```
cd target/dspace-1.6.2-build.dir; sudo -u tomcat ant fresh_install
```

Hopefully you will see something similar to this:

```
[echo] ====================================================================
[echo] The DSpace code has been installed, and the database initialized.
<snip> ...
BUILD SUCCESSFUL
Total time: 44 seconds
```

6. Make an initial administrator account (an e-person) in DSpace:

```
sudo -u tomcat /opt/dspace/bin/dspace create-administrator
```

Note: *This will be your primary and/or first Dspace administrator, most likely a real person or some shared administrator account at your location. You will be prompted for this person's email, name and initial password.*

7. Make the xmlui the default UI:

```
sudo /opt/dspace/sbin/make_xmlui_active
```

8. Start Tomcat5 and Apache:

```
sudo /sbin/service tomcat5 start; sudo /sbin/service httpd start
```

9. See if we have a web page: http://repository.unm.edu
   Use your own URL/Hostname here.

Dspace Handle Service

(*We use the Dspace handle server at UNM, which really was never set up by a system admin in our previous build.(a planning/communication oversight. oops.) Our Dev struggled with it being problematic and I had assumed it was started by tomcat... Anyway, this time we are implementing it correctly including a real startup script for the Handle server.*)

*Login as **root** again, if you are not already.*

1. Create a new startup script that works with chkconfig:

```
vi /etc/init.d/dspace-handle
```

{_}And the contents are:

```
#!/bin/bash
#
# chkconfig: - 86 14
# description: Starts and stops the Dspace handle service
# pidfile: /var/run/dspace-handle.pid
```

```
#
# - Originally written for RHEL5 platform by Gary Browne per
# [http://www.mail-archive.com/dspace-tech@lists.sourceforge.net/msg04943.html]
# - Heavily modified and adapted by Jamin Ragle (jragle At unm edu), 8Jun2010
# University Libraries, University of New Mexico
#
# NOTE: Don't forget to change/update your dspace install location and run user!
#

# Source function library.
. /etc/rc.d/init.d/functions

# Load any dspace environment changes
if [ -f /etc/sysconfig/dspace ]; then
. /etc/sysconfig/dspace
fi

# Start handle in the C locale by default.
HANDLE_LANG=${HANDLE_LANG-"C"}

# Dspace environment
RUN_AS=tomcat
handle=${HANDLE-/opt/dspace/bin/start-handle-server}
prog=handle
pidfile=${PIDFILE-/var/run/dspace-handle.pid}
lockfile=${LOCKFILE-/var/lock/subsys/dspace-handle}
RETVAL=0

# Running 'start' again, when the process is already running will not result
# in duplicate processes. Just an immediate return from the daemon function.
start() {
echo -n $"Starting $prog server: "
LANG=$HANDLE_LANG daemon --user=${RUN_AS} --pidfile=${pidfile} $handle $OPTIONS
RETVAL=$?
echo
# give it long enough to start the stupid java process. :-( sleeping for 1 seems to work.
sleep 1;
echo `pgrep -f -u $RUN_AS net.handle.server.Main` > $pidfile
[ $RETVAL = 0 ] && touch ${lockfile}
return $RETVAL
}

stop() {
echo -n $"Stopping $prog server: "
killproc -p ${pidfile} -d 10 $handle
RETVAL=$?
echo
[ $RETVAL = 0 ] && rm -f ${lockfile} ${pidfile}
}

case "$1" in
start)
start
;;
stop)
stop
;;
restart)
stop
start
;;
status)
status -p ${pidfile} $handle
RETVAL=$?
;;
*)
echo $"Usage: $0 {start|stop|restart|status}"
RETVAL=3
esac
exit $RETVAL
```

Note: This script is a huge improvement from what we started with, but I think there are some bugs:

a. i. Had to put in a ~~sleep 5~~ 'sleep 1' in the start, otherwise sometimes the handle service starts, but the pidfile is empty (no pid listed) and the restart would not work properly without it.
ii. The way I create the pidfile, searching the process table after the exec, I think is a hack. The 'pgrep -u dspace java' is better than the original search that Gary used... well its shorter anyway. :)

2. Adjust the permissions and add it with chkconfig:

```
chmod +x /etc/init.d/dspace-handle; chkconfig --add dspace-handle; \
chkconfig dspace-handle on; chkconfig --list dspace-handle
```

*Confirm the service is **on** _*for some of the run-levels:

```
# chkconfig --list dspace-handle
dspace-handle          0:off       1:off       2:on        3:on        4:on        5:on        6:off
```

3. **Not** Following the Official instructions, configure your install to run the handle server:

```
su -c "/opt/dspace/bin/dsrun net.handle.server.SimpleSetup /opt/dspace/handle-server" tomcat
```

You will be prompted to answer a few questions. The defaults _are fine, except for the steps listed here._Please enter a short description of this server/site:

```
 UNM Digital Repository

Please enter the name of your organization: University Libraries, University of New Mexico

Please enter the name of a contact person
for University Libraries, University of New Mexico (optional) [(none)|(none)]: xxxxx@unm.edu

Please enter the email address of bfreels@unm.edu or of University Libraries, University of New Mexico:
xxxxxx@unm.edu

The Handle System can communicate via UDP and/or TCP sockets.
Since UDP messages are blocked by many network firewalls, you may
want to disable UDP services if you are behind such a firewall.

Would you like to disable UDP services?(y/n) [n]: y

Generating keys for: Server Certification

The private key that is about to be generated should be stored
in an encrypted form on your computer. Encryption of the
private key requires that you choose a secret passphrase that
will need to be entered whenever the server is started.
Note: Your private key may be stored unencrypted if you so choose.
Please take all precautions to make sure that only authorized
users can read your private key.

Would you like to encrypt your private key?(y/n) [y]: n

Generating keys for: Administration

The private key that is about to be generated should be stored
in an encrypted form on your computer. Encryption of the
private key requires that you choose a secret passphrase that
will need to be entered whenever the server is started.
Note: Your private key may be stored unencrypted if you so choose.
Please take all precautions to make sure that only authorized
users can read your private key.

Would you like to encrypt your private key?(y/n) [y]: n
```

At the end, it says you need to "register in the Global Handle Registry (GHR).. but we already did this, so ignore that.

4. Edit the config to include our Dspace Naming Authority(300:0.NA/1928):

```
vi /opt/dspace/handle-server/config.dct
```

Use search and replace: (replace #### with your number)

```
:%s/YOUR_NAMING_AUTHORITY/####/g
```

*Or update by hand, there will be **3 places** to change. (snippet*)

```
"max_session_time" = "86400000"
"this_server_id" = "1"
"max_auth_time" = "60000"
"backup_admins" = (
"300:0.NA/1928"
)
```

Add the following two lines into the "server_config" clause:

```
"storage_type" = "CUSTOM"
"storage_class" = "org.dspace.handle.HandlePlugin"
```

5. Start the handle server for the first time:

```
service dspace-handle start
```

6. Check to make sure it is running properly:

```
service dspace-handle status
```

You should see something like:

```
# service dspace-handle status
start-handle-server (pid  5115) is running...
```

Finishing Up

Almost finished, stick with me here and you will be ready to use your Dspace instance in no time.
Configure the local Firewall

1. Port **80** and **443**, should already be configured. (Among others.)
2. Open port **2641** for the "handle server" by adding the following to /etc/sysconfig/iptables (or use system-config-securitylevel GUI tool):

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

Please insert the rule in port order.

```
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 2641 -j ACCEPT
```

3. Restart iptables:

```
service iptables restart
```

Configure the cron environment

(*There are a lot of scheduled events that need to happen, which include DB dumps for backups and other cleanup tasks. Here is how they are set up.*)

1. Make a home for DB backups:

```
su -c "mkdir -m 770 /opt/dspace/database/backups" postgres
```

2. Install **postgres** & **tomcat** run cron jobs in /etc/cron.d:

```
vi /etc/cron.d/dspace-tasks.cron
```

and the contents of the file...

```
#########################################################################
# DSpace Database cron jobs                                             #
# [http://www.dspace.org/1_6_0Documentation/ch03.html#N10A9A]   #
#########################################################################

# DSPACE backups - DB dump to file: Updated 9/3/10 by jragle
10 1 * * * postgres pg_dump -Ft -U dspace -f /opt/dspace/database/backups/dspace-nightly-dump.pgb

# Clean up the database nightly at 4.20am
20 4 * * * postgres vacuumdb -q --analyze dspace

#########################################################################
# DSpace Application cron jobs (run as _tomcat_ user)           #
# [http://www.dspace.org/1_6_0Documentation/ch03.html#N10A9A]   #
#########################################################################

# Send out subscription e-mails at 1 AM every morning
0 1 * * * tomcat /opt/dspace/bin/sub-daily

# Run the media filter at 2 AM every day
0 2 * * * tomcat /opt/dspace/bin/filter-media

# Run the checksum checker at 3 AM
0 3 * * * tomcat /opt/dspace/bin/checker -lp

# Run embargo lifter at 4 AM
0 4 * * * tomcat /opt/dspace/bin/dspace embargo-lifter

# Mail the results to the sysadmin at 04:05 AM
05 4 * * * tomcat /opt/dspace/bin/dsrun org.dspace.checker.DailyReportEmailer -c

# Collection strengths:
0 5 * * * tomcat /opt/dspace/bin/dsrun org.dspace.browse.ItemCounter

# Generate sitemaps at 6am
0 6 * * * tomcat /opt/dspace/bin/generate-sitemaps
```

Configure proper log rotation for Dspace logs

(*Really I'd rather these logs be in /var/log/dspace, but for now, I think I'll just create a proper logrotate.d rule.*)

1. I guess this step is a **stub** for some future work. See my notes in the box below for the reasons log rotation is being skipped in this build... for now.

> **Notes on log rotation:** Chatting with our Dspace dev (Brian), he tried to explain to me the complexities of Dspace and logs. The short of the story is "*the re is a 500-file limit, and a file size limit, so it keeps them as long as that.*" I'm sure I'd have to get a full historical lesson on Dspace to really understand why they have the log methodology they are using right now.
>
> I really don't see why it can't be similar to the default apache setup, which is "Rotate weekly, keep 52 copies (1 year), compress older files" and should be fine for any statistical purposes. You let the system, specifically **logroate.d**, handle the log management and retention. I hope this can be addressed in the future.

Make services persistent through reboots

1. Make sure that Apache, Tomcat, PostgreSQL and Dspace Handle services are set to start up automatically at boot-time:

```
chkconfig httpd on; chkconfig tomcat5 on; chkconfig postgresql on
```

STOP HERE

- **Congratulations! -** You are finished!
(*With a generic install of Dspace 1.6*)

Todo/Problems to resolve in this document:

- Figure out how to do footnotes in Confluence.
- Figure out how to do hover-over notes.
- Figure out how to collapse chunks of quoted text. (for output of commands)
- place missing repo file on this page in case it is not downloadable by external (to UNM) readers/users.
- Insert missing quoted text. (output of commands typically)
- Insert missing notes on Migrations and Updates
- Include Brian's notes on modifying the build.xml file to clean out the tomcat cache/work directory upon Updates.
- Post notes on how to build your own Java JAI rpm.
- Next time notes on what to fix. (Next build)