1 Introduction

1.1 General Information

Required Ports

- 80 – HTTP
- 22 – SSH
- 443 – HTTPS
- 5432 – PostgreSQL port
- 8080 – Tomcat HTTP
- 8443 – Tomcat HTTPS
- 2461 – Handle Server native
- 8000 – Handle Server HTTP

Operating System: Red Hat Enterprise System 3

Java Version: 1.5.0_06

Ant Version: 1.6.2

PostgreSQL Version: 7.4.2

DSpace Version: 1.3.2

1.2 Notation and Fonts for this Document

Normal Sans-Serif Font - The main body of the text of this document

Italic Sans-Serif Font - Notes that should be paid attention to.

Fixed-Width Font - Examples of commands or items that you might see on the computer screen. For example, directory and file names as well as installation commands.

Italic Fixed-Width Font - The user that you should be logged in as when performing tasks.
You should always be logged in as the user most recently specified on the left-hand side of the page in the current section.

1.3 Short Hand for this Document
The following short-hand notations are used throughout this document to represent certain types of information. When these are encountered in the main text they should always be replaced with the values found/inserted in this section.

[postgres] = The location of the PostgreSQL installation

dspace = The location of the DSpace installation

dspace-source = The location of the source code used to create the DSpace installation

dspace-home = The dspace user's home directory on the system

database-pw = The dspace database password for the dspace user

[admin-email] = The email address of the system administrator who will receive all administrative requests and emails from the site.

[admin-fn] = The First Name of the administrator's account

[admin-ln] = The Last Name of the administrator's account

[admin-pw] = The DSpace administrator's login password

[handle] = The Handle Prefix given to the organisation by CNRI

tomcat = The installation directory of the Tomcat web-server

[handle-server] = The installation directory of the Handle Server

[machine-ip] = The IP address of the machine on which your DSpace installation is going

### 1.4 System Users

Users that will have to exist on the machine onto which DSpace is to be installed are as follows:

- root - the super-user for the machine.
• *dspace* - the user who will be the default DSpace user, and will own the tomcat instance etc. This is the most used user.

• *postgres* - the user who will own the database software

The user you need to be logged in as when performing the actions laid out in this document will be given on the left hand side of the page before the set of actions which this user needs to perform. For example:

*postgres*

1. Create the database using: `createdb -U dspace dspace`

indicates that before executing instruction (1), you should be logged in as the *postgres* user.
2 Full Installation Overview

This section details the order in which things ought to be done in order to have a fully working DSpace with Tapir in the form of the Edinburgh Research Archive.

1. Before starting the installation ensure that the machine has the prerequisite software for DSpace installed:
   a. Operating System
   b. Tomcat 5.5.9
   c. Java 1.5.0_06
   d. Ant 1.6.5

   The installation and configuration of this software is outside the scope of this document.

2. Install PostgreSQL as per the section Installing PostgreSQL from Source (3.1). Once this has been done you should configure it as per the section PostgreSQL Configuration (3.2) before starting the database server as per the section Starting and Stopping PostgreSQL (3.3).

3. Set up the PostgreSQL cron job as per the section PostgreSQL Cron Jobs (3.4).

4. Install DSpace as per the section Installing DSpace from Source (4.1).

5. Generate and install a Signed SSL Certificate with OpenSSL. To use a self-signed certificate, see the section Creating a Self-Signed Certificate with OpenSSL (9.4).

6. Configure Tomcat to use the SSL Certificate as per the section Configuring Tomcat SSL (8.4)

7. Configure the pre-installed handle server as per the section Handle Server Configuration (10.1) then start it as per the section Starting and Stopping the Handle Server (10.2).

8. Test that the new OAI interface is working correctly as per the section Configuring the OAI Interface (11.1).

9. Import the data from another instance of DSpace as per the section Data Import (12).
3 PostgreSQL

3.1 Installing PostgreSQL from Source

1. Download the postgres source from www.postgres.org

   postgres

2. Unzip and Untar postgres source

   tar -zxvf postgresql-source.tar

3. Configure PostgreSQL for installation:

   In [postgres]/postgresql-source:

   ./configure --with-java --prefix=[postgres]

4. Build the PostgreSQL source code. In [postgres]/postgresql-source:

   gmake

5. Check that this build has worked

   gmake check

6. Install PostgreSQL

   gmake install

7. Add PostgreSQL to the PATH variable for all users. In /etc/profile add the line

   PATH = PATH:[postgres]/bin

8. Initialise the database

   [postgres]/bin/initdb -D [postgres]/data

9. Place the postgres.jar file produced by the build process into the [dspace]/lib directory.
3.2 Configuration

The following settings need to be entered in the file 
\texttt{[postgres]/data/postgresql.conf}:

1. \texttt{tcpip_socket = true}

To allow PostgreSQL to be accessed via TCP/IP

2. \texttt{max_connections = 400}

To allow 400 connections to be opened to the database

3. \texttt{shared_buffers = 3000}

To provide normal usage in a live environment

Note that if these values have to go any larger then the value of the system variable \texttt{SHMMAX} which limits segment size must be increased.

3.3 Starting and Stopping PostgreSQL

\textit{postgres}

\begin{itemize}
\item Start the database server (in the background):

\texttt{[postgres]/bin/postmaster –D [postgres]/data &}
\end{itemize}

\begin{itemize}
\item Stop the database server:

\texttt{pg_ctl stop -m fast -D [postgres]/data}
\end{itemize}

or (only if above fails):

\texttt{kill `cat [postgres]/data/postmaster.pid`}

3.4 PostgreSQL Cron Jobs

\textit{postgres}

\begin{itemize}
\item Check that the file \texttt{mycron} exists in the directory \texttt{[postgres-home]} to contain all the cron information for the \texttt{postgres} user. If not, create it.

\item Add the following lines into the \texttt{mycron} file:
\end{itemize}
# vacuum full analyze the database every night
0 2 * * * [postgres]/bin/vacuumdb -f -z

3. Export the new cron job to the postgres user’s crontab:

crontab mycron

3.5 Backing up and restoring PostgreSQL Data

postgres

- Backup the data to a plain text file:

  
  pg_dump dspace > postgres-backup

- Restore the data to Postgres:

  
  psql dspace < postgres-backup
4. DSpace

4.1 Installing DSpace from Source

1. Download the DSpace source from http://sourceforge.net/projects/dspace

2. Create source directory: `[dspace-source]`, and insert the uncompressed source from http://sourceforge.net/projects/dspace

3. Create the target installation directory `[dspace]`

4. Ensure the the additional postgres library is in `[dspace-source]/lib`

5. Create a dspace database user for PostgreSQL (the database server must be running in order for these commands to work). In the directory `[postgres]/bin`

   ```
   ./createuser –U postgres –d –A –P dspace
   ```

   Password: [database-pw]

6. Create the database to be used by dspace

   ```
   ./createdb –U dspace -E unicode dspace
   ```

7. Prepare the configuration files for build in `[dspace-source]/config`. There are more details in the section DSpace Configuration (5)

8. Compile and install the code. In `[dspace-source]`

   ```
   ant fresh_install
   ```

9. Copy the `dspace.war` and `dspace-oai.war` files, which are created in the directory `[dspace-source]/build` into the tomcat webapps directory: `[tomcat]/webapps`

10. Create the dspace system administrator. In `[dspace]/bin`

   ```
   ./create-administrator
   ```

   Using the values

   ```
   email: [admin-email]
   ```
11. Set up Tomcat as the web-server as per the section **Configuration Tomcat as a Standalone Web-Server (8.1)**

12. Start Tomcat as per the section **Starting and Stopping Tomcat (8.2)**

13. Set up the cron jobs associated with DSpace as per the section **DSpace Cron Jobs (4.3)**

### 4.2 DSpace Cron Jobs

**dspace**

1. Check that the file `mycron` exists in the directory `[dspace-home]` to contain all the cron information for the `dspace` user. If not, create it.

2. Add the following lines into the `mycron` file:

```
JAVA_OPTS=-Xmx512M -Xms64M -Dfile.encoding=UTF-8
PATH=/usr/java/j2sdk1.4.2_01/bin
JAVA_HOME=/usr/java/j2sdk1.4.2_01

10 00 * * * /dspace/bin/index-all
30 00 * * * /dspace/bin/filter-media
00 03 * * * /dspace/bin/stat-monthly
10 03 * * * /dspace/bin/stat-report-monthly
30 03 * * * /dspace/bin/stat-general
00 04 * * * /dspace/bin/stat-report-general
00 07 * * * /dspace/bin/sub-daily
```

3. Export the new cron job to the `dspace` user's crontab:

```
crontab mycron
```

### 4.3 Updating a DSpace Instance

**dspace**

1. In `[dspace-source]` run the update:
2. Move the new `dspace.war` and `dspace-oai.war` files into `[/tomcat]/webapps`, replacing the ones there already.

3. Remove the directories (and their contents):
   1. `/tomcat/webapps/dspace`
   2. `/tomcat/webapps/dspace-oai`
   3. `/tomcat/work/Catalina/localhost/dspace`
   4. `/tomcat/work/Catalina/localhost/dspace-oai`

If you wish to regularly build instances of DSpace then another ant build script can be created to run a full update in one process. The contents of that file (`mybuild.xml`) should be:

```xml
<project name="dspace-builder" default="deploy" basedir=".">
  <property name="tomcat" value="/opt/tomcat/webapps" />
  <property name="source" value="/opt/dspace" />
  <property name="v" value="dspace-source" />
  <property name="n" value="dspace" />

  <target name="deploy" description="deploy the dspace instance">
    <copy file="${source}/${v}/build/dspace.war"
          tofile="${tomcat}/${n}.war" overwrite="true" />
    <copy file="${source}/${v}/build/dspace-oai.war"
          tofile="${tomcat}/${n}-oai.war" overwrite="true" />
    <delete dir="${tomcat}/${n}" failonerror="false" />
    <delete dir="${tomcat}/${n}-oai" failonerror="false" />
  </target>
</project>
```

This can then be called just using:

```
ant -buildfile mybuild.xml
```

It can also be used to rename the war files as they are deployed by passing a new value of `n` to the script:

```
ant -buildfile mybuild.xml -Dn=newname
```

### 4.4 Updating DSpace Configurations

`dspace`

1. Edit the live DSpace config file `[/dspace]/config` to meet your new criteria (you should ensure that your backup copy is kept in sync)

2. Restart tomcat as per the section `Starting and Stopping Tomcat (8.3)`
4.5 Installing new configuration files

If a DSpace instance requires new configuration files (for example for new modules) it is necessary to manually copy them into \[dspace\]/config.

4.6 Identifying Problems with DSpace

To find out what the problem is, view the page source of the page with the Internal System Error. Scan down the source until you reach the section which starts:

```html
<P align=center>
   <A HREF="/dspace/">Go to the DSpace home page</A>
</P>
```

Following this there will be a stack trace for the place in the software where the error occurs, or a message saying:

```
No stack trace available
```

If no stack trace is available, you will be able to find the error message by looking at the end of the file:

\[dspace\]/log/dspace.log

There are a several situations which you may come across:

**No stack trace available**

This can be for a number of reasons, usually that the database is not up. If this is the case look at the section **Recovering from an unknown error**

*Exception: org.apache.commons.dbcp.SQLNestedException: Connect failed, cause: java.util.NoSuchElementException: Timeout waiting for idle object …*

The database has run out of connections and needs to be restarted. Try stopping and then starting PostgreSQL as per the installation and systems administration document section 3.3. If this fails look at the section **Recovering from an unknown error**

*org.apache.commons.dbcp.SQLNestedException: Cannot get a connection, pool exhausted*

*org.apache.commons.dbcp.SQLNestedException: Cannot get a connection, pool exhausted at org.apache.commons.dbcp.PoolingDriver.connect(PoolingDriver.java:183)*
The database connection pool has run out of spaces for new connections – there are too many users accessing the system simultaneously. Stopping and starting tomcat as per the section 8.2 will fix the problem initially, but will throw users out of the system. For a long term fix, increase the `db.maxconnections` parameter in `config/dspace.cfg` then restart tomcat.

**Any other error**

Most errors that can happen at the moment are due to either database failure or web server failure. These should all be solvable by following the steps laid out in Recovering from an unknown error (4.9).

### 4.7 Recovering from an Unknown Error

Most problems that you may encounter can be remedied either by restarting postgres or restarting the web server (tomcat).

First try restarting postgres, by following the instructions in the dspace installation and systems administration manual section 3.3

If restarting postgres then <shift>+reload of the page does not get rid of the error then restart the web server as per the dspace installation and systems administration manual section 5.3

Logged in as dspace you should be able to issue the commands `starttomcat` and `stoptomcat` from anywhere in the shell. Leave approximately 10 seconds in between stopping and starting to ensure that the restart is clean.
5 DSpace Configuration

This section deals with the advanced configuration of DSpace. Most of the standard configuration options are well documented in the dspace.cfg file, and will not be discussed here.

5.1 LDAP

If you are using LDAP for authenticating your institution's users, you can instruct DSpace to use the same authentication database. If LDAP is enabled then new users will be able to register simply by entering their username and password. It does not mean that users without LDAP details cannot register as before, though.

First, you must enable the ldap registration process

```
ldap.enable = true
```

You must also specify the location of the LDAP server (whether to include o=myu.edu will depend on the configuration of the server:

```
ldap.provider_url = ldap://ldap.myu.edu/o=myu.edu
```

To allow the system to correctly query the LDAP server, you must set the object and search contexts. Again, these are specific to each LDAP implementation, although they are likely of the form as follows:

```
ldap.object_context = ou=people,o=myu.edu
ldap.search_context = ou=people
```

Finally, we can set the fields that the LDAP code should look in in order to populate the user object. These are the userid, email, surname, given name and telephone number fields:

```
ldap.id_field = uid
ldap.email_field = mail
ldap.surname_field = sn
ldap.givenname_field = givenName
ldap.phone_field = telephoneNumber
```

If you wish your LDAP users to be automatically registered when they first log in (recommended), you can specify:

```
webui.ldap.autoregister = true
```
5.2 Custom Item View

This configuration allows you to choose how you would like your items displayed in the simple item view mode. You can choose which fields you wish to display, and gives you some small control over how they are displayed.

The form of the configuration line is:

```
<schema>.<element>[.<qualifier>]*[(date)|(link)], ... [all other fields]
```

For example:

```
# Dublin Core element 'title' (unqualified)
dc.title

# DC element 'title', qualifier 'alternative'
dc.title.alternative

#All fields with Dublin Core element 'title' (any or no qualifier)
dc.title.*

# DC identifier.uri, render as a link
dc.identifier.uri(link)

#DC date.issued, render as a date
dc.date.issued(date)
```

The full example configuration is then given as (where \ is inserted to indicate that although a line break is present, the following line should be appended to the end of the current):

```
webui.itemdisplay.default = dc.title, dc.title.alternative, dc.contributor.*, \
dc.subject, dc.date.issued(date), dc.publisher, \
dc.identifier.citation,dc.relaiton.ispartofseries,\ndc.description.abstract, dc.description, \
dc.identifier.govdoc, dc.identifier.uri(link), \
dc.identifier.isbn, dc.identifier.issn, \ndc.identifier.ismn, dc.identifier
```

This would then produce an item listing presenting each of those elements in order, with date.issued formatted as a date, and identifier.uri rendered as a link to the value of that field.

5.3 Search Indexing

If you are customising your metadata intake at all, it is also wise to customise your indexing data. This is done by providing configuration lines of the form:
This will modify the way that the contents of the database are indexed for searching (not browsing). If you wish to change the advanced search system to reflect these changes it is necessary to modify the UI. The default configuration for DSpace is as follows:

search.index.1 = author:contributor.*
search.index.2 = author:creator.*
search.index.3 = title:title.*
search.index.4 = keyword:subject.*
search.index.5 = abstract:description.abstract
search.index.6 = author:description.statemofresponsibility
search.index.7 = series:relation.ispartofseries
search.index.8 = abstract:description.tableofcontents
search.index.9 = mime:format.mimetype
search.index.10 = sponsor:description.sponsorship
search.index.11 = id:identifier.*

This gives you the possibilities of searching on author, title, keyword, abstract, series, mimetype, sponsor and identifier, with more than one metadata element tied to some of these fields (e.g. author is contributor.*, creator.* and description.statemofresponsibility.

### 5.4 Custom submission forms

To customise how your submission forms look, you can modify the input-forms.xml file to present the relevant information.

The input forms submission process definitions are structured as follows:

```xml
<form-definitions>
  <form name="[name]">
    <page number="[number]">
      <field>
        <dc-element>element</dc-element>
        <dc-qualifier>qualifier</dc-qualifier>
        <repeatable>true | false</repeatable>
        <label>Label</label>
        <input-type>type</input-type>
        <hint>Some hint</hint>
        <required></required>
      </field>
    </page>
  </form>
</form-definitions>
```

The submission can have any number of pages, and each page number should be defined in the `number="[number]"` attribute. These will be presented in order. Each field in the submission should then be defined using the values here. Most are self explanatory, and
extensive documentation is provided with DSpace to describe how each of these fields is defined. Note that the presence of `<required></required>` does not indicate that the field is required. A field is only taken to be required when this tag pair contain a value, which is the value to print to the screen if the form is submitted without completing this field.

You may also specify multiple form definitions (ensure they are called different things in the name="[name]" attribute of the form. If you do this, then you can specify collections by handle to which the different submission systems will be applied. For example:

```
<form-map>
  <name-map collection-handle="default" form-name="traditional" />
  <name-map collection-handle="123456789/345" form-name="custom" />
</form-map>
```

5.5 Statistics

The configuration of the statistics package (dstat.cfg) is pretty standard, and should need almost no modification prior to use. Primarily you will just need to specify the host name and host url of the service:

```
# the name and url of the service being reported on
host.name = My DSpace Host
host.url = http://www.mydspace.edu/
```

There are four sections to the rest of the dstat.cfg. First there are the settings for which actions to place in the general summary. These are default configured thus:

```
# actions to place in the general summary
general.summary=archive_item
general.summary=view_bitstream
general.summary=view_item
general.summary=view_collection
general.summary=view_community
general.summary=login
general.summary=search
general.summary=reject_licence
general.summary=oai_request
```

These are presented at the top of the stats for each section, in the order that they are specified here.

The next section are the exclude terms. This excludes words, search type identifiers, and special characters to be excluded from search term analysis:

```
# words to exclude from the search words report
exclude.word=the
exclude.word=of
```
You may need to localise these for your language, and monitor the search results for any terms or characters which slip through unwanted.

The statistics system will also analyse the current archive contents, and is capable of breaking down the contents based on keywords specified in the config. The default is:

```plaintext
# The item types in the archive that you need number breakdowns on
item.type=Thesis or Dissertation
item.type=Preprint
item.type=Research Paper
```

This will provide a list of item types and the numbers of them that appear in the archive at the time of the report.

Finally, there are the floor values. These tell the system when to stop counting items for display. That is, when an item has been viewed fewer than, say, 20 times, it will not be
presented as part of the statistics, for purposes of keeping the statistics as short as is reasonable. The same functionality exists for presenting search word analyses. A careful balance should be made so that the floor value works well for both the general listings page and the monthly listings page, as the same value is used in each (which could do with being fixed):

```
# floor values (don't display things that have been activated fewer times
# than this) for the reports
item.floor=20
search.floor=5
```

There is also an option in the configuration to limit how many of the items displayed should have their details looked up. If an item does not have its details looked up, then only the handle will be provided, which is of limited interest. The reason for this configuration existing is that it used to deal with the event that the java environment had to be fired up from within a Perl script, which took some time. Now that the entire system runs in one java call, this field is of less use. Nonetheless, in very large result sets, there will be a corresponding performance hit. This variable should therefore be set very large, depending on whether you care how long the statistics take to generate. Since they are non-user-facing when they are generated, this is probably not a problem. Therefore set (to look up the first 1 million items):

```
# limit the number of lookups of titles and authors to the first X. Lookup
# invokes the java environment so has quite an impact on performance.
item.lookup=1000000
```
6 Tomcat

6.1 Tomcat Configuration as Standalone Web-Server

Since HTTP requests come in on port 80 by default we need to route all traffic for that port to port 8080 where Tomcat is listening. We cannot run tomcat on port 80 as it will only run as root, which is a security risk. We therefore use IPTables to perform port pre-routing to overcome this problem. See the section Web-Server Issues for more information.

To set up IP tables to route 80 to 8080 and 443 to 8443:

```bash
iptables -t nat -A PREROUTING -p tcp -i eth0 -d [machine-ip] --dport 80 -j DNAT --to [machine-ip]:8080
iptables -A FORWARD -p tcp -i eth0 -d [machine-ip] --dport 8080 -j ACCEPT
iptables -t nat -A PREROUTING -p tcp -i eth0 -d [machine-ip] --dport 443 -j DNAT --to [machine-ip]:8443
iptables -A FORWARD -p tcp -i eth0 -d [machine-ip] --dport 8443 -j ACCEPT
```

The web server will now route communication from ports 80 and 443 to 8080 and 8443 respectively.

Set up the connectors in `server.xml` as follows, to allow for unsecured and secured connections:

```xml
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
<Connector port="8080" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" redirectPort="8443" acceptCount="100"
    connectionTimeout="20000" disableUploadTimeout="true"
    URIEncoding="UTF-8" />

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

dspace

Before starting or stopping tomcat for the first time, create shell scripts to do the job for you, as this will be much quicker.

1. In the directory \[\text{dspace-home}/bin\] create a file called \texttt{starttomcat} containing the line:

\[\text{[tomcat]/bin/startup.sh}\]

2. In the directory \[\text{dspace-home}/bin\] create a file called \texttt{stoptomcat} containing the line:

\[\text{[tomcat]/bin/shutdown.sh}\]

3. In the directory \[\text{dspace-home}/bin\] you need to make the files \texttt{starttomcat} and \texttt{stoptomcat} executable. To do this use:

\texttt{chmod 764 starttomcat stoptomcat}

4. Place this directory in the \texttt{dspace} user’s \texttt{PATH} environment variable so that they can be run from anywhere on the system. In the file \[\text{dspace-home}/.bash_profile\] add the line:

\texttt{PATH=$PATH:$HOME/bin}

You can now start and stop tomcat from anywhere on the system by issuing one of the commands:

\[\#> \texttt{starttomcat}\]

\[\#> \texttt{stoptomcat}\]

Note that it is worth restarting tomcat after every system change as otherwise updates may not be reflected.
7 SSL Certificates

This section presents a number of options with regard to creating SSL certificates for use with DSpace.

7.1 Generating SSL Certificates with OpenSSL

If you are signing the certificate yourself, see the section Creating a Self-Signed Certificate with OpenSSL (6.2)

dspace

1. Generate the key in [dspace-home]/certificates:

```bash
openssl genrsa -des3 -out pass.key 1024
```

2. You can now generate a certificate request with:

```bash
openssl req -new -key pass.key -out pass.csr
```

7.2 Creating a Self-Signed Certificate with OpenSSL

This is the method that has been used to generate the certificate that is currently in use in ERA.

dspace

1. Generate the key:

```bash
openssl genrsa -des3 -out pass.key 1024
```

2. Generate the server key:

```bash
openssl rsa -in pass.key -out server.key
```

3. Sign the certificate yourself (valid for 999 days):

```bash
openssl req -new -key server.key -x509 -out server.crt -days 999
```

4. Generate the DER key file:

```bash
openssl pkcs8 -topk8 -nocrypt -in server.key -out server.key.der -outform der
```
5. Generate the DER certificate file:

```bash
openssl x509 -in server.crt -out server.crt.der -outform der
```

6. Use the ImportKey utility in Java to import the key into the keystore (you will need to obtain the importkey.jar file if you do not already have it):

```bash
java -cp importkey.jar comu.ImportKey server.key.der server.crt
```

### 7.3 Creating a Self-Signed Certificate with the Java Keytool

**dspace**

1. Generate the certificate:

```bash
keytool -genkey -alias tomcat -keyalg RSA
```

2. Self-sign the certificate:

```bash
keytool -selfcert -alias tomcat
```

3. If you wish to sign the certificate via a certificate authority at a later date, a certificate for signing can be exported thus:

```bash
keytool -certreq -keyalg RSA -alias tomcat -file request.csr
```

### 7.4 Importing the Signed Certificate using the Java Keytool

**dspace**

1. Import the .pem file using the command

```bash
keytool -import -alias dspace -file [dspace-home]/certificates/dspace.pem
```

When a password is requested use: `changeit`

The resulting keystore file is located here:

```bash
[dspace-home]/.keystore```
8 Handle Server

8.1 Handle Server Configuration

dspackage

1. Make the handle server with the DSpace native commands. In 
   dspace/bin:

   ./make-handle-config

   Note: This may fail with an error like "Warning: data not encrypted". In this case it is
   necessary to do the installation directly:

   ./dsrun net.handle.server.SimpleSetup [handle-server]

2. The following are the answers to the questions that should be given during handle-
   server setup:

   Caching or Regular: 1
   Primary Server: y
   IP Address: [your host name]
   Port Number: 2641
   HTTP Port: 8000
   Log all access: y
   Version/Serial No: 1
   Description: [your description]
   Disable UDP: y
   Server Key: n
   Admin Key: n

3. This generates a number of files, including a setup file that needs to be sent to CNRI. 
   Find the file:

   [handle-server]/sitebndl.zip

   and email to CNRI quoting the identifier number registered

4. While waiting for the site bundle to be applied we need to perform the local handle
   server configuration. In the file [handle-server]/config.dct

   Find the section server_config and do the following updates:

   "storage_type" = "CUSTOM"
   "storage_class" = "org.dspace.handle.HandlePlugin"
In addition, replace all instances of `YOUR_NAMING_AUTHORITY` with `[handle]` for the whole file.

5. Now start the handle server as per the section **Starting and Stopping the Handle Server (10.2)**

6. Note that the handle server caches previous settings for a while, so it is necessary to wait for 24 hours or so before changes noticeably take effect.

### 8.2 Starting and Stopping the Handle Server

**dspace**

To start the handle server, in `[dspace]/bin` use the command:

```
./start-handle-server
```

To stop the handle server it is necessary to use the unix command `kill`. Get the process ID for the line starting:

```
/bin/sh /u01/dspace/bin/dsrun -Dlog4j.configuration =log4j-handle-plugin.properties net.handle.server.Main ...
```

and use:

```
kill <process id>
```
9 Data Import

This section describes how we move data from one instance of DSpace on one machine to another instance on another machine. This process should be adaptable to most import or upgrade tasks.

This procedure assumes similar structure on each machine, and notation is relative to the machine in question.

9.1 Taking the Data from a Source DSpace

dspace

1. Shut down Tomcat to prevent any further changes to the data or the database.

2. In [dspace] bundle up the assetstore for moving:

   tar -zcvf assetstore.tar assetstore/

   This produces a file called assetstore.tar.gz which contains all of the archived files. This file can be very large (approximately 5Mb for each item in the archive).

postgres

3. Back up the contents and structure of the database. In [dspace] run:

   pg_dump dspace > data

   This produces a file called data which contains all of the database. This file is generally of reasonable size (approximately 13Kb for each item in the archive).

dspace

4. Compress the data file for convenience:

   gzip -v9 data

5. Backup assetstore.tar.gz and data.gz to another machine (ideally not the source or the target machines).

6. You can now restart Tomcat. Changes made after this will not be reflected in the target machine, and this procedure must be performed again for new updates.
9.2 Installing the Data in a Target DSpace

dspace

1. Stop Tomcat.

2. Upload the files `assetstore.tar.gz` and `data.gz` created in the previous section to the directory `[dspace-home]/data/`

3. In `[dspace]` rename the old assetstore:

   ```
   mv assetstore assetstore_old
   ```

4. Copy `[dspace-home]/data/assetstore.tar.gz` to `[dspace]`

5. Unzip and untar the new assetstore:

   ```
   tar -zxvf assetstore.tar
   ```
   This will create a new directory called `assetstore` which is an exact copy of the one in the source DSpace installation.

postgres

6. Backup the old data in the database. In `[dspace]` run:

   ```
   pg_dump dspace > data-old
   ```
   This generates a file called `data-old` which contains the old database contents.

7. Prepare to drop the old `dspace` database by stopping and starting postgres to kill all latent connections to the database:

   ```
   pg_ctl stop -m fast -D [postgres]/data
   postmaster -D [postgres]/data
   ```

8. Drop the old DSpace database:

   ```
   dropdb dspace
   ```

dspace

9. Now log in as `dspace` and reinitialise the DSpace database as you would do in a normal installation. In `[postgres]/bin` run:
10. Prepare to import the data into the DSpace database. Copy 
    `data/data.gz` to `postgres`

11. Unzip the `data.gz` file:

```
gunzip data.gz
```

12. Import the data into the DSpace database:

```
psql dspace < data
```

This will create the database as an exact replica of the source installation's
database.

13. You can now restart Tomcat.