OCSSW Web Services Developer Manual

Aynur Abdurazik

May 24, 2017

1 OCSSW

1.1 OCSSW Installation Locations

There are four possible installation locations for OCSSW in the following order:

1. The path given as the value of seadas.ocssw.root variable in the seadas.config file
2. The value of the OCSSWROOT system variable specified in the script files
3. The seadas.home directory specified in the seadas.config file
4. user.home/ocssw directory

These locations and their order of selection is implemented as the following:

```
String ocsswInstallDirPath = RuntimeContext.getConfig().getContextProperty("ocssw.root", System.gete

                if (dirPath == null) {
                    dirPath = RuntimeContext.getConfig().getContextProperty("home", System.getProp
```

2 Packaging and Deployment

2.1 Packaging

The ***ocsww client-server module*** is independent of the rest of SeaDAS. The jar file that will be deployed on the ocssw server is packaged using the following command in the $SEADAS_HOME/seadas/seadas-ocsswrest directory:

```
mvn install assembly:assembly
```

2.2 Services and Application Description

2.2.1 Adding New REST Services

How to register this new service?

3 OCSSW Server Side Representation

3.1 OCSSW programs

3.2 Execution

The jar file is packaged on a developer machine and deployed on the server.

Run the following command from the command line to execute:

```
java -Xmx2048m -jar seadas-ocsswserver-jar-with-dependencies.jar
```

4 Virtual Box Configuration

4.1 Basic Configuration

1. Need to install "guest editions" to be able to resize the vm window and also to enable file sharing between the host and the guest.
2. Choose "Fixed" virtual box for performance. Eloborate more later.
3. Need to install Java to run the ocssw rest server.

4.1.1 How to install guest additions

The guest additions are actually included (invisibly) in the binary package. You get at them by choosing Install Guest Additions from the Devices menu. You then install them by mounting the Guest Additions iso file, which will show up in the list of available images. The Guest Additions are not available separately and don't need to be.

4.1.2 Java Installation on the Ubuntu Linux Virtual Box

1. Check to see if your Ubuntu Linux operating system archtecture is 32-bit or 64-bit, open up a terminal and run the following command below:

   ```
   file /sbin/
   ```

2.

4.2 File Sharing

Usage of disk space. Does it take virtual box disk space? How to increase disk space when needed for "fixed" virtual boxes?

How disk space is used on virtual machine, i.e., how is it allocated?

error message when no disk space left: "segmentation error"

error message for incorrect way of increasing disk space:

VBoxManage modifyhd "/Users/aabduraz/VirtualBox VMs/OCSSW$_{U}buntu_{S}erver/OCSSW_{U}buntu_{S}erve
$-resize819200Progressstate : VBOX_{EN}OT_{S}UPPORTEDVBoxManage : error : Resizeharddiskoperatio$

The reason for the error was there are two disk space measures, virtual size and actual size. The resize amount has to be greater than the virtual size specified in the "Storage" configuration of the virtual machine.

- Manually sharing a folder between host and guest machines:
  In VirtualBox **Devices → Shared Folder Settings... → Shared Folders → Machine Folders**, select the folder from the host to be shared with the guest.

  1. sudo rm /sbin/mount.vboxsf
  2. sudo ln -s /opt/VBoxGuestAdditions-4.3.20/lib/VBoxGuestAdditions/mount.vboxsf /sbin/mount.vboxsf
  3. mkdir ocsswws
  4. sudo mount -t vboxsf seadas-ocsswws /home/aabduraz/ocsswws
  5. To be able to write in the shared directory, it needs to be mounted in this way:
     sudo mount -t vboxsf -o uid=1000,gid=1000 seadas-ocsswrest /home/aabduraz/ocsswrest

- Commands to manually mount a directory:

  ```
  sudo mount -t vboxsf seadas-ocsswrest /home/aabduraz/ocsswrest
  ```

  where $seadas - ocsswrest$ is the name of folder, which has the development source code for web services, shared from the host machine, and $/home/aabduraz/ocsswrest$ is an empty folder in the virtual machine. The $seadas - ocsswrest$ is shared to deploy the jar file from its *target* directory after each build.

- need to install git (error message:Error - Could not execute system command "git –version > /dev/null")

4.3 Network Configuration

1. The server must use 0.0.0.0 as its IP address.
2. The client should still use *localhost*
3. The virtual machine uses "NAT" port-forwarding, which is set through **Devices → Network → Network Settings ...** .
4. Between SeaDAS and OCSSWWS, we chose to use port number 6400 and 6401. The server side presents services using address "0.0.0.0 : 6401", and a SeaDAS client will access the services using "$http : //localhost : 6400$".

5 Security Concepts and Implementation

5.1 Security Concepts

5.1.1 Java Keystore

Java keystore is a repository of security certificates. JDK provides a tool named {*keytool*} to manipulates keystores. Java *keytool* stores the keys and certificates in a keystore, protected by a keystore password.

TrustManager: Determines whether the remote authentication credentials *andthustheconnection* should be trusted.

KeyManager: Determines which authentication credentials to send to the remote host.

5.1.2 Security Key Generation

1. Create a keystore for server

   ```
   keytool -genkey -alias server -keyalg RSA -keystore server.jks
   ```

   My password for server keystore is "ocsswws". The generated file is "*server.jks*".

2. Create a keystore for client

   ```
   keytool -genkey -alias client -keyalg RSA -keystore client.jks
   ```

   My password for server keystore is "ocsswwsclient". The generated file is "*client.jks*".

3. View the content of keystore files:

   ```
   keytool -list -v -keystore server.jks -storepass ocsswws
   keytool -list -v -keystore client.jks -storepass ocsswwsclient
   ```

4. Get server's self signed public key certificate and store it in client's keystore.

   ```
   keytool -export -file server.cert -keystore server.jks -storepass ocsswws -alias server
   ```

5. Get client's self signed public key certificate and store it in server's keystore.

   ```
   keytool -export -file client.cert -keystore client.jks -storepass ocsswwsclient -alias client
   ```

   Note: First we needed to export both server and client public key certificates into files.

6. Use following commands to view certificate contents.

```
keytool -printcert -v -file server.cert
keytool -printcert -v -file client.cert
```

7. As the last step, import server.cert into client keystore and client.cert into server keystore.

   - store client's self signed public key certificate(client.cert) in server.jks against the alias "client".

     ```
     keytool -import -file client.cert -keystore server.jks -storepass ocsswws -alias client
     ```

   - store server.cert within client.jks against the alias "server".

     ```
     keytool -import -file server.cert -keystore client.jks -storepass ocsswwsclient -alias ser
     ```

8. View the content of both keystore again using following commands.

   ```
   keytool -list -v -keystore server.jks -storepass ocsswws
   keytool -list -v -keystore client.jks -storepass ocsswwsclient
   ```

5.1.3 Setting up SSL Configuration on OCSSW (Jersey) Client

The SSL configuration is setup in the ClientBuilder class. The client builder contains methods for definition of KeyStore, TrustStore or entire SslContext.

- *KeyStore* - Represents a storage facility for cryptographic keys and certificated; Manages different types of entries. The keystore in javax.net.ssl.keyStore contains private keys and certificates.
- *TrustStore* - The *javax.net.ssl.trustStore* contain CA certificates that a server trusts when a remote party presents its certificate.
- *SslContext* -

```
SslConfigurator sslConfig = SslConfigurator.newInstance()
    .trustStoreFile("truststore.jks")
    .trustStorePassword("asdfgh")
    .trustStoreType("JKS")
    .trustManagerFactoryAlgorithm("PKIX")

    .keyStoreFile("keystore.jks")
    .keyPassword("asdfgh")
    .keyStoreType("JKS")
    .keyManagerFactoryAlgorithm("SunX509")
    .keyStoreProvider("SunJSSE")

    .securityProtocol("SSL");

 SSLContext sslContext = sslConfig.createSSLContext();
```

6 Virtualbox Related FAQ

A Virtual Box Ubuntu behaves very similar to Ubuntu on bare metal. Therefore the solutions from this question

How to fix "The system is running in low-graphics mode" error? mostly hold true as well. However note, that you do not have a graphics card or proprietary drivers in a VM. This will all be handled by the guest additions.

We can enter a TTY for command line repair by press and hold the HOST-key, which in your case is the right Ctrl-key and simultanously press F1 trough F8.

In case your virtual hard drive runs short of memory try first with

sudo apt-get autoremove sudo apt-get autoclean to remove no longer needed packages.

6.1 How to free up disk space

- empty the directory /tmp
- dpkg -l — grep $^{r}c|cut-d''-f3|xargssudoapt-getpurge-ysudoapt-getautoremove$

- Synaptic
- sudo apt-get clean sudo apt-get autoclean
- sudo apt-get remove –purge linux-image-X.X.XX-XX-generic – delete old kernels to decide which kernel version to remove: dpkg –get-selections — grep linux-image