CERTIFICATES FOR NON-TECHIES

In the ideal world it should go like this

NEW INSTALL

Create keystore with a self signed certificate.

The keystore password should be the same which is in NSA in Server properties on the security page.
Create a certificate signing request (CSR) and send it to a Certificate authority (CA) and get back a certificate signed by the CA.

Import that to the keystore.

Export the cert from the Clarity keystore and import the cert also to the jdk cacerts keystore.

CERTIFICATE UPDATE

With the data CA already has, place an order for the expired certificate.
Install as above.

The result of a successful process is a chain of certificates from root via intermediate certificates to the final certificate.

When it does not go trouble shooting is definitely no something for the non-techie.

The basic instructions are below (the way I understand them, which may be wrong) More details are in the Installation guide or in Enable Secure Sockets Layer.doc.

Before you do these, remember to stop the services and backup you existing .keystore file.

PROBLEMS

Cannot access your Clarity system after certificate install (new or upgrade) If you cannot access your Clarity after the certificate is imported then, as Nick Darlington puts it, either it is not there or there is something wrong with it.

If you did not get any error messages when importing the certificate then and you have verified with the list command that the certificate is there, then the assumption is that there is something wrong in it.

That also means that Clarity does not see the proper certificate.

LISTING THE CERTIFICATES IN THE KEYSTORE

You can list the certificates in the keystore to a file with the following command
keytool -list -v -keystore D:\Clarity\config\.keystore > d:\20130303_certs_in_keystore.txt

HELP FROM LOGS

What Clarity might do is to create the following error message in app-system.log
No available certificate or key corresponds to the SSL cipher suites which are enabled.
That confirms that something is missing

SPECIFYING WHICH CERTIFICATE TO BE USED

Normally the proper certificate would be automatically used. Tomcat documentation tells how you can specify the certificate to be used.

When Tomcat starts up, I get an exception like "java.net.SocketException: SSL handshake errorjavax.net.ssl.SSLException: No available certificate or key corresponds to the SSL cipher suites which are enabled."
A likely explanation is that Tomcat cannot find the alias for the server key within the specified keystore. Check that the correct keystoreFile and keyAlias are specified in the <Connector> element in the Tomcat configuration file. REMINDER - keyAlias values may be case sensitive!

The configuration file is
D:\clarity\tomcat-app-deploy\conf\server.xml

and in it

<Connector port="443" secure="true" SSLEnabled="true" connectionTimeout="86400000" maxThreads="200" minSpareThreads="1" maxSpareThreads="50" keystoreFile="D:\clarity\config\.keystore" keystorePass="xxxxxx" sslProtocol="TLS" maxHttpHeaderSize="8192" enableLookups="false" disableUploadTimeout="true" acceptCount="100" scheme="https" clientAuth="false" URIEncoding="UTF-8" useBodyEncodingForURI="true" />
<Engine name="Catalina" defaultHost="localhost">

that is where you would add keyAlias="mykey"

If after inserting the keyAlias attribute you get in app-system.log "alias mykey not found" that definitely confirms that there is something missing.

MORE DETAILS ON THE INSTALL PROCESS

In practice the key strength of the certificate has to be 2048. That is, if it expires after December 31 2013.
It is the keystore where you specify the strength not the certificate nor the CSR.
If key length of your existing keystore is less than 2048 you will get an error about the length and expiry date when you validate the CSR
See below for Verisign/Symantec validation URL

The command to generate the keystore is
keytool -genkey -alias <alias_name> -keyalg RSA keysize 2048 -keystore <yourdomain.keystore>
With Clarity the name of the keystore is .keystore and the certificate alias is mykey.
Apparently the order of the parameters does not matter
Note that you specify the key length for the keystore not for the certificate request.

Once you have the keystore you will create certificate signing request with the following command
keytool -certreq -keystore /<clarity home>/config/.keystore -keyalg RSA -file <your filename>.csr
With that csr file you purchase a valid certificate from a CA.

After receiving a certificate from a CA, install it with the following command
keytool -import -keystore /<clarity home>/config/.keystore -keyalg RSA -file <cert_name>.cer -trustcacerts
Note that you must install the certificate to the same keystore from which the CSR was created.

ERROR MESSAGES WHEN IMPORTING THE NEW CERT

If you have expired certificate or a selfsigned certificate in the keystore which has the same alias as the new certificate you get an error that alias my already exists. In that case you delete old alias or rename it.

If there are keys or certificates missing you will get an error message
no matching public key found
suggests that you have not the root and intermediate certificates

java.lang.Exception: Failed to establish chain from reply
means that the certificates do not match and don't form a chain.

In the certificate listing you should have the chain length given
Alias name: mykey
Creation date: Jan 22, 2013
Entry type: PrivateKeyEntry
Certificate chain length: 4
when the last line is missing that means that the certificates do not match and don't form a chain.

VERIFYING THE CERTIFICATE PATH

One more way to test certificates in Windows operating system is to export the certificate from the keystore and to right click the certificate and verify what is on the Certificate Path.

CERTIFICATE OK BUT STILL NO ACCESS

It can happen that after some problems and various attempts you you are convinced that the keystore and certificates are OK, but instead of getting to your system you get a blank page.
The reason can be that the browser cache contains invalid items the prevent the connection. Clear the cache.

MULTIPLE SERVERS IN A CLUSTER OR FOR TESTING

If you have more servers like in a cluster or for testing you can copy the whole keystore.
That will work if the systems are isolated from each other like in testing.
In a cluster the CN (Common Name) in the certificate has to be different for each server.
If there is a load balancer they could be the same and point to the load balancer.

MORE INFORMATION

(If you have Verisign/Symatec certificate)

On October 10, 2010, Symantec migrated its public Root Certification Authorities from 1024-bit RSA keys to 2048-bit RSA keys. Certificates issued prior to October 10, 2010, from the 1024-bit RSA keys, have continued to operate correctly and securely after the migration to the 2048-bit RSA keys.
So any new roots now are 2048 bit strong. Likewise the new intermediate certificates are 2048 bit.

If you have Verisign/Symatec certificate
the install instructions for Tomcat are at
https://knowledge.verisign.com/support/ssl-certificates-support/index?page=content&actp=CROSSLINK&id=AR234

You can search for existing certificates at
https://securitycenter.verisign.com/celp/enroll/outsideSearch?application_locale=VRSN_US&originator=VeriSign:CELP

E.g. you can use the common name as search criteria

If you validate your certificate (if your system is accessible form open Internet) and you can validate you CSR at

https://knowledge.verisign.com/support/ssl-certificates-support/index?page=content&actp=CROSSLINK&id=AR1692

Click the link "here" to access the validation.

You can get the primary certificate for Verisign/Symantec from
https://knowledge.verisign.com/support/ssl-certificates-support/index?page=content&actp=CROSSLINK&id=AR1553

You can get the intermediate certificates for Verisign/Symantec from

https://knowledge.verisign.com/support/ssl-certificates-support/index?page=content&id=AR1735

The commands to import the intermediate certificates are
keytool -import -trustcacerts -alias primaryIntermediate -keystore d:\Clarity\config\.keystore -file primary_inter.cer
keytool -import -trustcacerts -alias secondaryIntermediate -keystore d:\Clarity\config\.keystore -file secondary_inter.cer

VERSION

Version 1.0 April 1, 2013 MKi

DISCLAIMER