SIEMENS

# Configuring Teamcenter for SSL

v1.1

# Table of Contents

# Expectations of this guide:

- TC 11.2.3 version
- Monolithic installation. All components on the same single machine.
- Not a deep dive into SSL
- General overview and technical transfer based on docs and case work
- Cover one way SSL using a self-signed certificate.
  Difference: One way has client trust the server it is connecting to. 2-way has the client trust the server and the server trust the client.

# Details to know about the Certificate being used

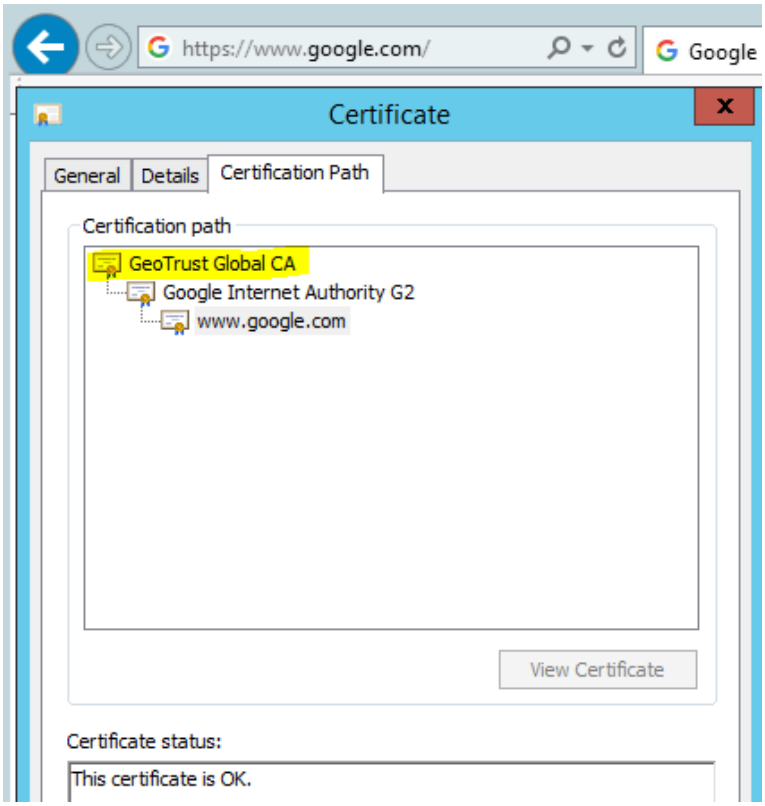
Is the certificate self-signed? A **self-signed certificate** is an identity **certificate** that is **signed** by the same entity whose identity it certifies. Not one signed by a well-known certificate authority (CA) like Entrust, VeriSign, and Thawte to name a few.

NOTE: Teamcenter documentation does not recommend using self-signed certificates in production environment. However, for testing, a self-signed certificate is used for this document.

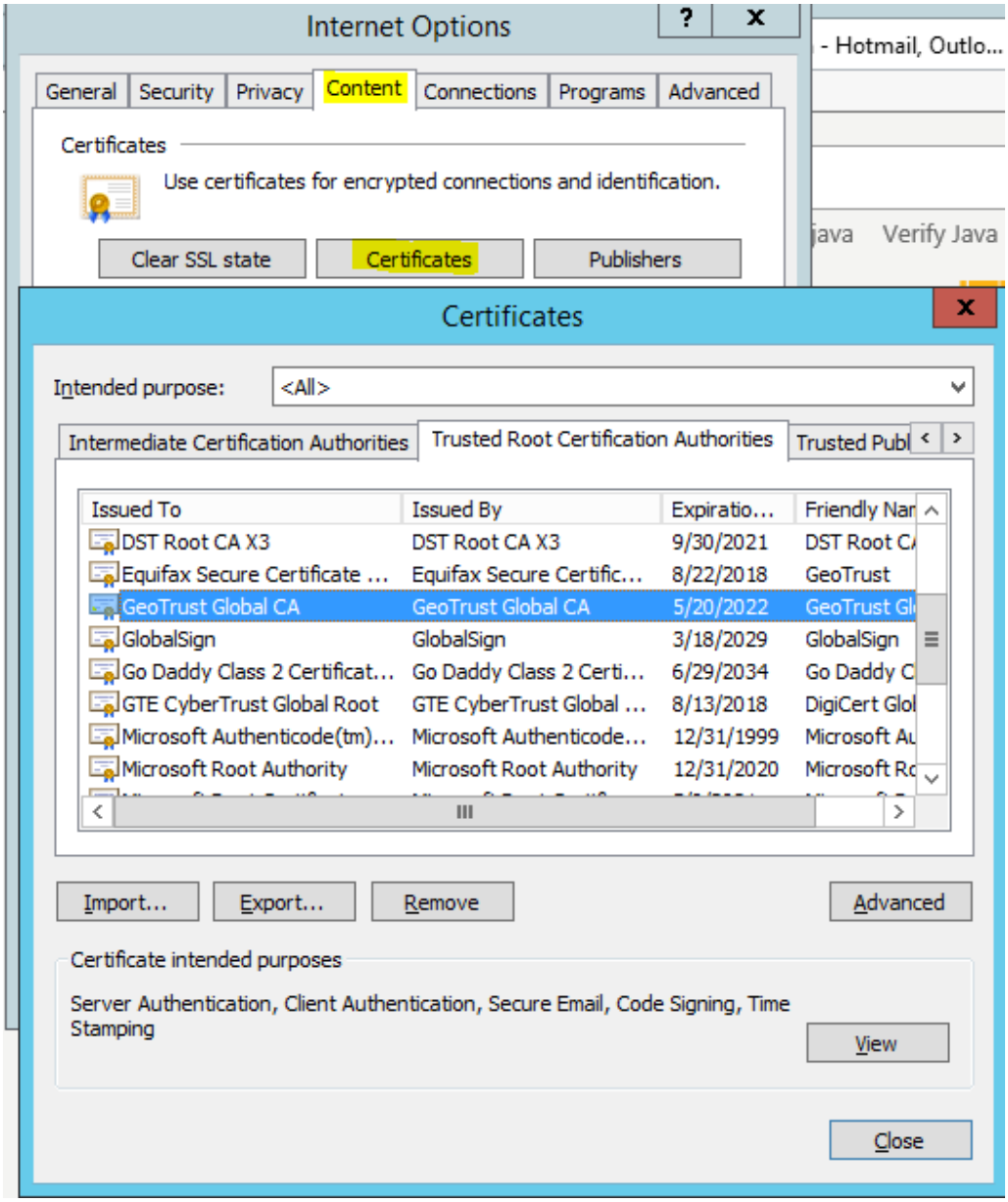
## Or is it signed by a Trusted CA? What does that mean?

The Browser and JAVA come pre-loaded with a list of trusted CA's. If the certificate is signed by one of those the browser displays the page without prompting the user to trust the server cert.

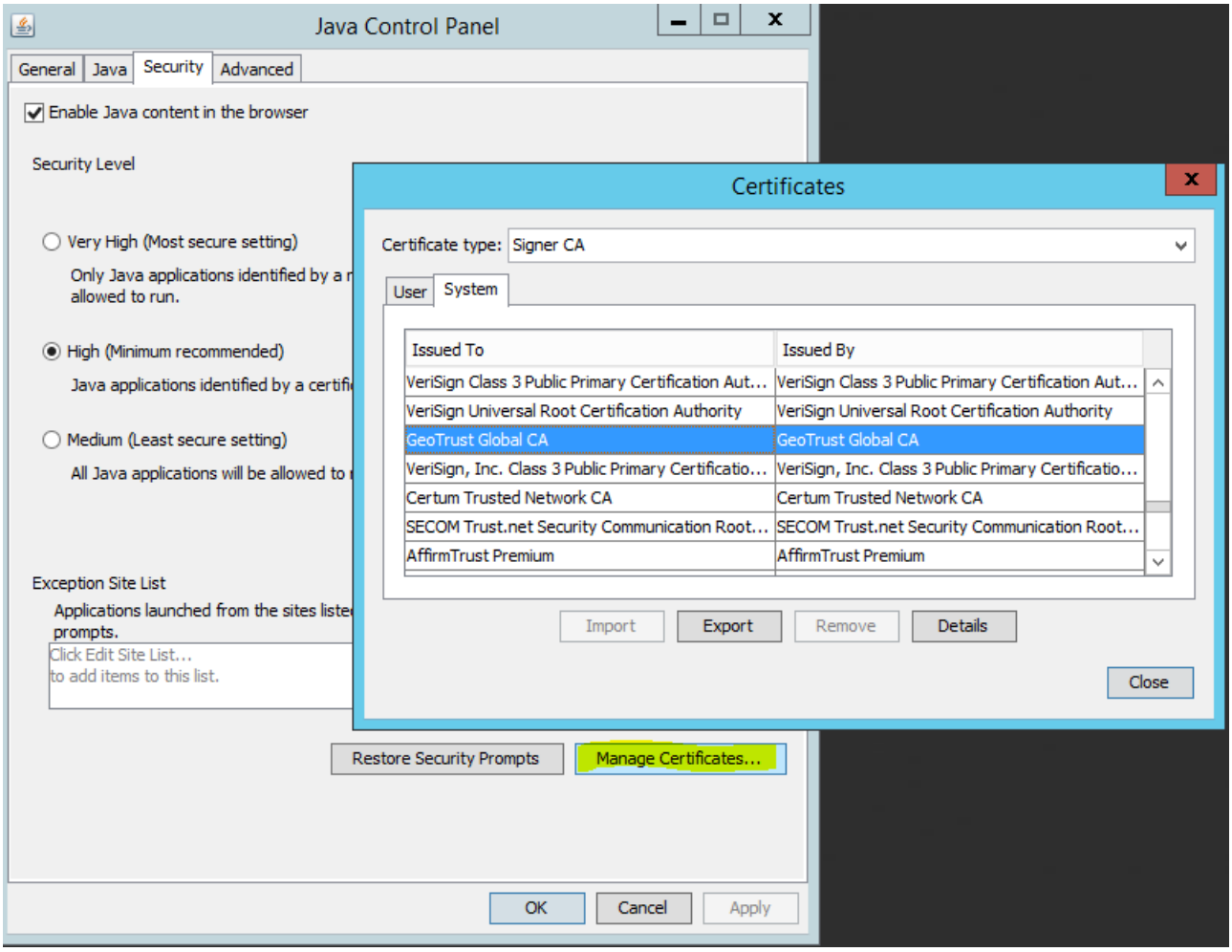
For example in IE:
Connecting the SSL based URL for Google (https://google.com) the user is presented the web page without any certificate related prompts. The certificate PATH for that ULR is:




Within IE that highlighted CA is Trusted:


Internet Options
General
Security
Privacy
Content
Connections
Programs
Advanced
Certificates
Use certificates for encrypted connections and identification.
Clear SSL state
Certificates
Publishers
- Hotmail, Outlo...
Verify Java
Certificates
Intended purpose:
<All>
Intermediate Certification Authorities
Trusted Root Certification Authorities
Trusted Publ
Issued To
Issued By
Expiratio...
Friendly Nar
DST Root CA X3
DST Root CA X3
9/30/2021
DST Root CA
Equifax Secure Certificate ...
Equifax Secure Certific...
8/22/2018
GeoTrust
GeoTrust Global CA
GeoTrust Global CA
5/20/2022
GeoTrust Gl
GlobalSign
GlobalSign
3/18/2029
GlobalSign
Go Daddy Class 2 Certificat...
Go Daddy Class 2 Certi...
6/29/2034
Go Daddy C
GTE CyberTrust Global Root
GTE CyberTrust Global ...
8/13/2018
DigiCert Glol
Microsoft Authenticode(tm)...
Microsoft Authenticode...
12/31/1999
Microsoft Au
Microsoft Root Authority
Microsoft Root Authority
12/31/2020
Microsoft Rc
Import...
Export...
Remove
Advanced
Certificate intended purposes
Server Authentication, Client Authentication, Secure Email, Code Signing, Time Stamping
View
Close

For example in JAVA the same CA listed as a Signer CA:

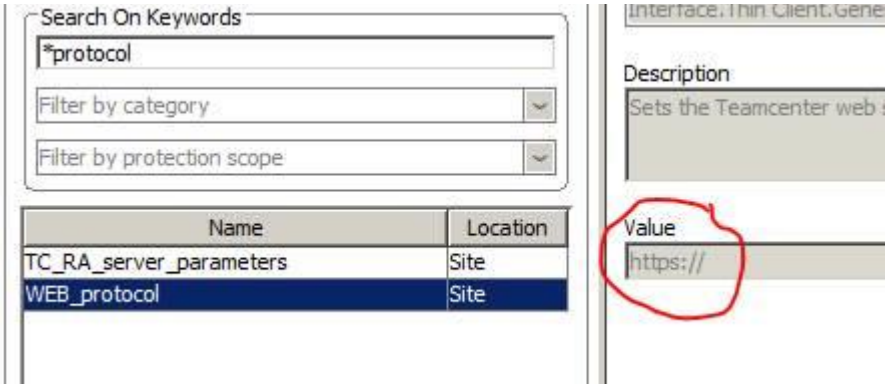



## What is the difference between a keystore and truststore?

The keystore is needed when you are setting up server side on SSL, it is used to store server's identity certificate, which the server will present to a client and the client will need to have that servers keystore certificate detail in its truststore in order for the connection to work. If your browser connects to a website over SS, it verifies the certificate presented by server against its truststore.

# Teamcenter Preference Updates

Update the 'Web_protocol preference to be 'https':




# Create a self signed certificate via Java keytool utility

The following command will generate a self signed certificate within the 'keystore' file specified by the '-keystore' option (C:\keystore).

```
%JAVA_HOME%\bin\keytool -keystore c:\keystore -alias gtac -genkey -keyalg RSA
```

The command will prompt for a number of values. In this example the alias for the cert is 'gtac'. The first prompt will be for 'First and Last name'. That value is the CN (common name) and should be the HOSTNAME of the Web app server. NOTE: The 'Common Name (CN)' of the web application hostname **MUST** match how that hostname will be provided in the URL. For example: If the URL that the end users will use is fully qualified (myserver.us.com) then the CN needs to be 'myserver.us.com'. If the CN name in the certificate does not match the hostname the connection will fail.

The remaining entries for the certificate used for this guide were entered as follows:

```
What is the name of your organizational unit?
  [Unknown]:  GTAC
What is the name of your organization?
  [Unknown]:  GTAC
What is the name of your City or Locality?
  [Unknown]:  Milford
What is the name of your State or Province?
  [Unknown]:  OH
What is the two-letter country code for this unit?
  [Unknown]:  US
Is CN=civjmsrv2012r21, OU=GTAC, O=GTAC, L=Milford, ST=OH, C=US correct?
  [no]:  y

Enter key password for <gtac>
        (RETURN if same as keystore password):
Re-enter new password:
```

The password entered above is 'changeit'.

# Export a certificate from the keystore using the Java keytool utility

This is a proactive step as the .crt generated will be required later for import into Teamcenter RAC client Java keystore. The keystore used in the command below is the 'C:\keystore' that was created in the last section.

```
%JAVA_HOME%\bin\keytool -exportcert -alias gtac -file c:\webappcert.crt -keystore c:\keystore -storepass changeit
```

The output from the utility run:

```
Certificate stored in file <c:\webappcert.crt>
```

# Enabling SSL within the Web Application

### JBoss EAP 6.3 (7.1 as well)

Update the Jboss standalone.xml file to include the location and connection information for the C:\keystore file that was generated in the 'Create a self signed certificate via Java keytool utility' section earlier in this guide. Added the highlighted section below:

```
<subsystem xmlns="urn:jboss:domain:web:1.1" native="false" default-virtual-server="default-host">
        <connector name="http" protocol="HTTP/1.1" scheme="http" socket-binding="http"/>
        <connector name="https" protocol="HTTP/1.1" scheme="https" socket-binding="https" secure="true">
        <ssl key-alias="gtac" password="changeit" certificate-key-file="C:/keystore" protocol="ALL" verify-client="false"/>
        </connector>
        <virtual-server name="default-host" enable-welcome-root="true">
            <alias name="localhost"/>
            <alias name="example.com"/>
        </virtual-server>
</subsystem>
```

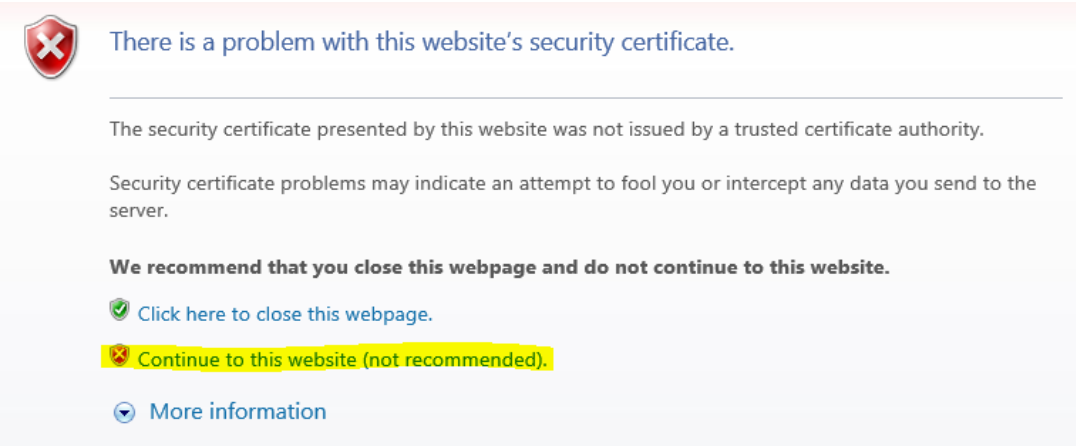
The default HTTPS port is listed at the bottom of the standalone.xml file. The default is '8443'. For example:


Start up Jboss and verify the SSL port.

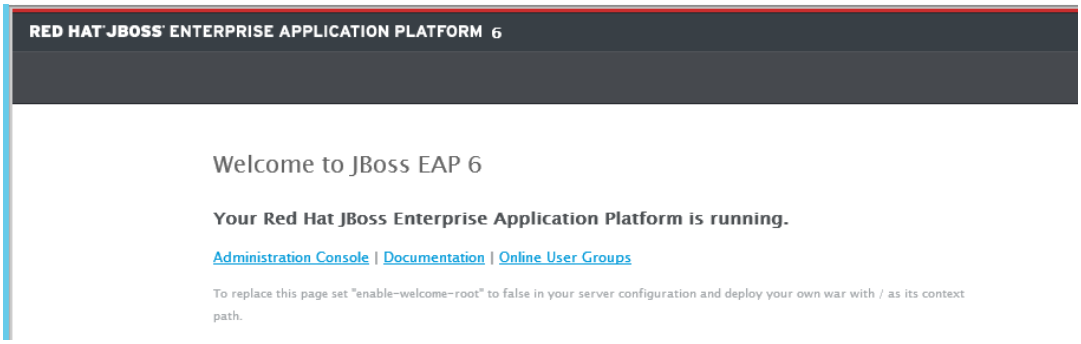
https://<web_app_servername>:8443

For example: For this guide the Web Application server name is 'gtac1' so the URL would be:
https://gtac1:8443

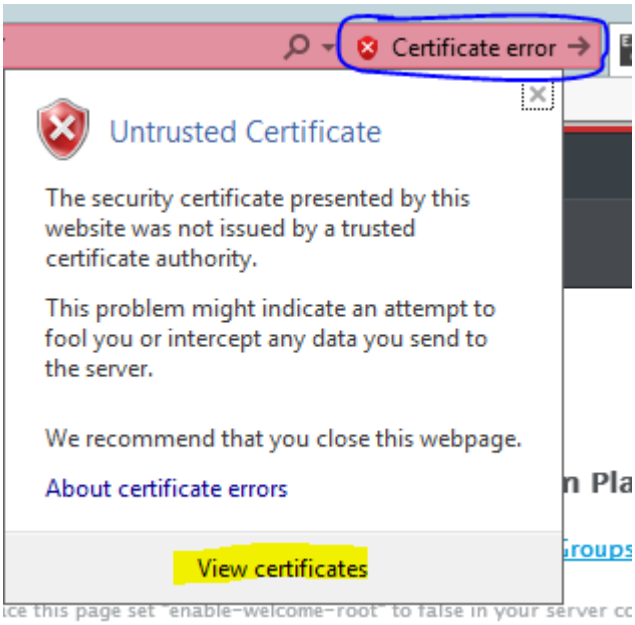
That should result in this page. Select the 'Continue' option:




That should bring up the Jboss splash screen:

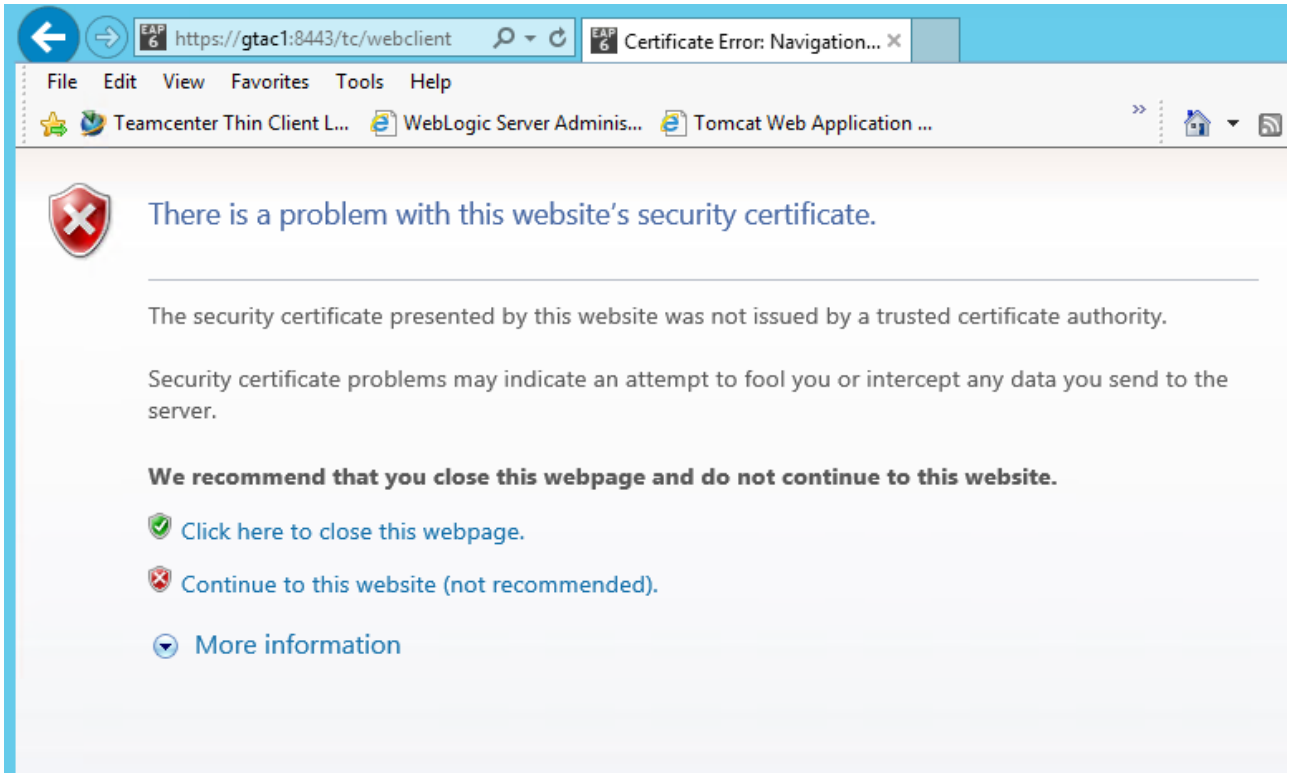



To review the certificate select the 'Certificate Error' / 'view certificate' to the right of the URL address bar.



# Thin Client validation

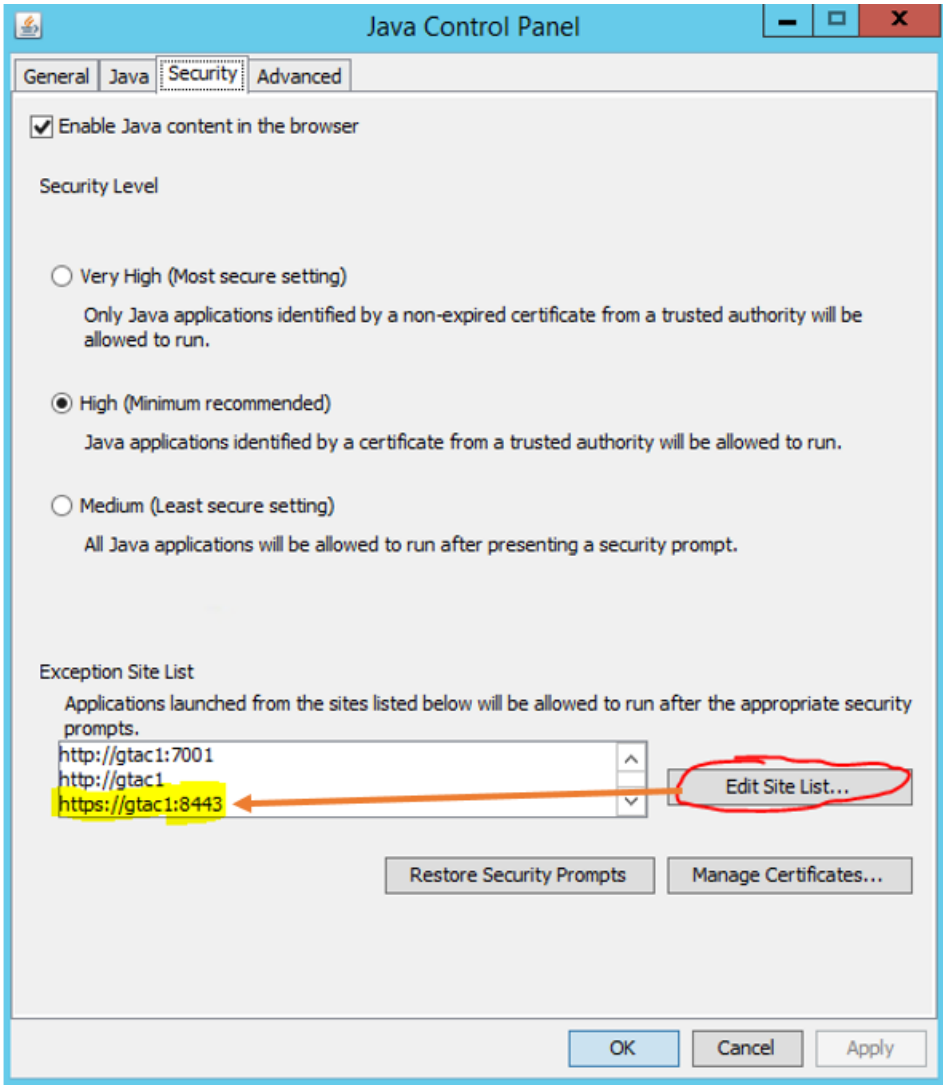
Launch the thin client URL. Until the certificate is imported into the Browsers truststore, the user will see the following page when launching the thin client:



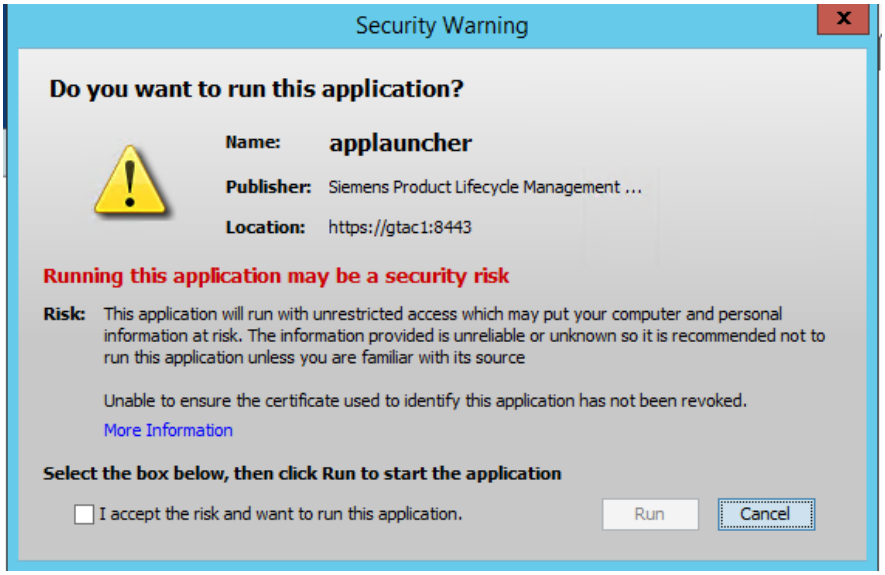

Select the 'Continue to the website' option. Then the user will be presented the login page.

One may see the following messages as well. These appear to be Java 'untrusted' related.

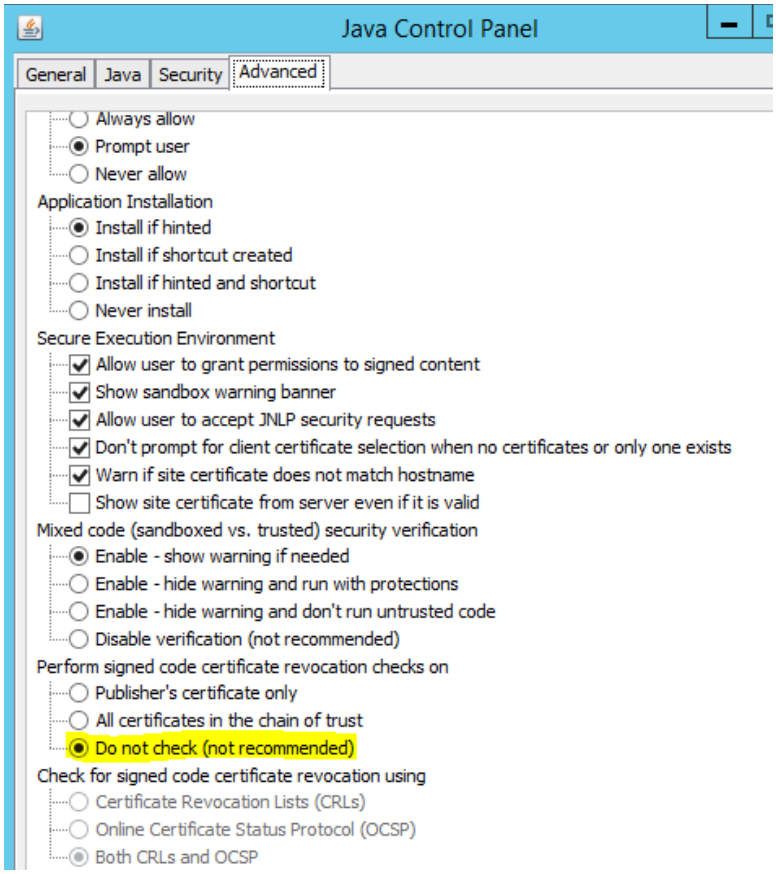
These are corrected by adding the server URL to exception list within the Java control panel.



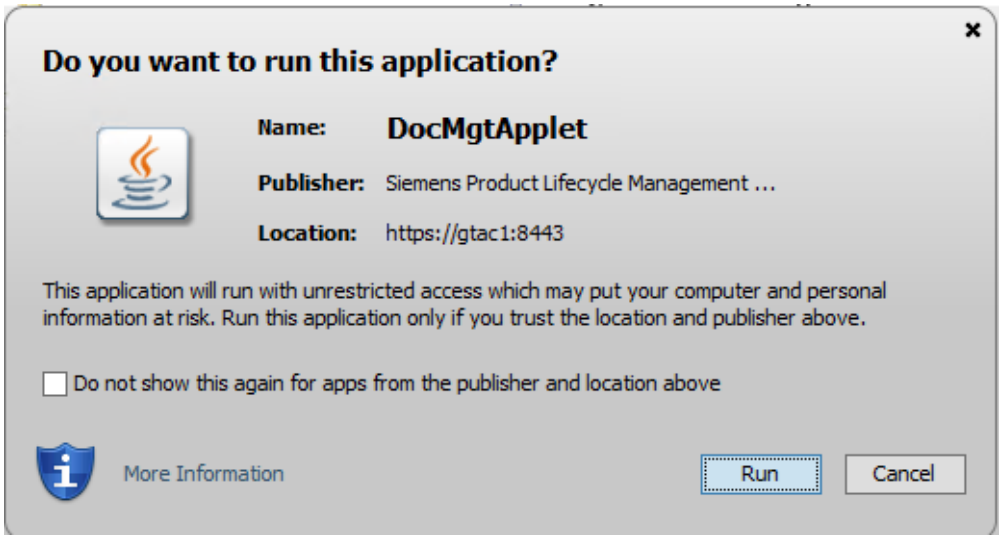
This one still appeared:




Disable cert validation in Java control panel kept that warning from appearing. Depending on the certificate this option may not be required. In this test use it is a result of the certificate being self-signed.



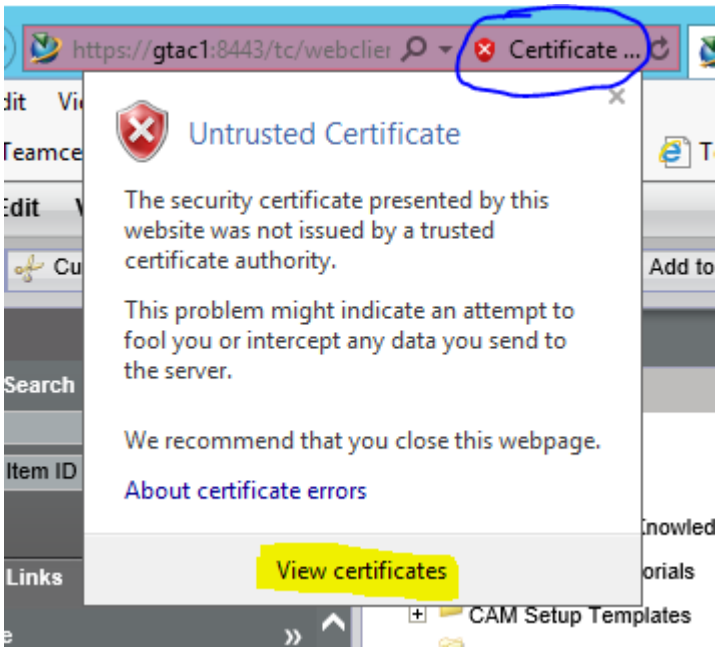
And now prompts for this:



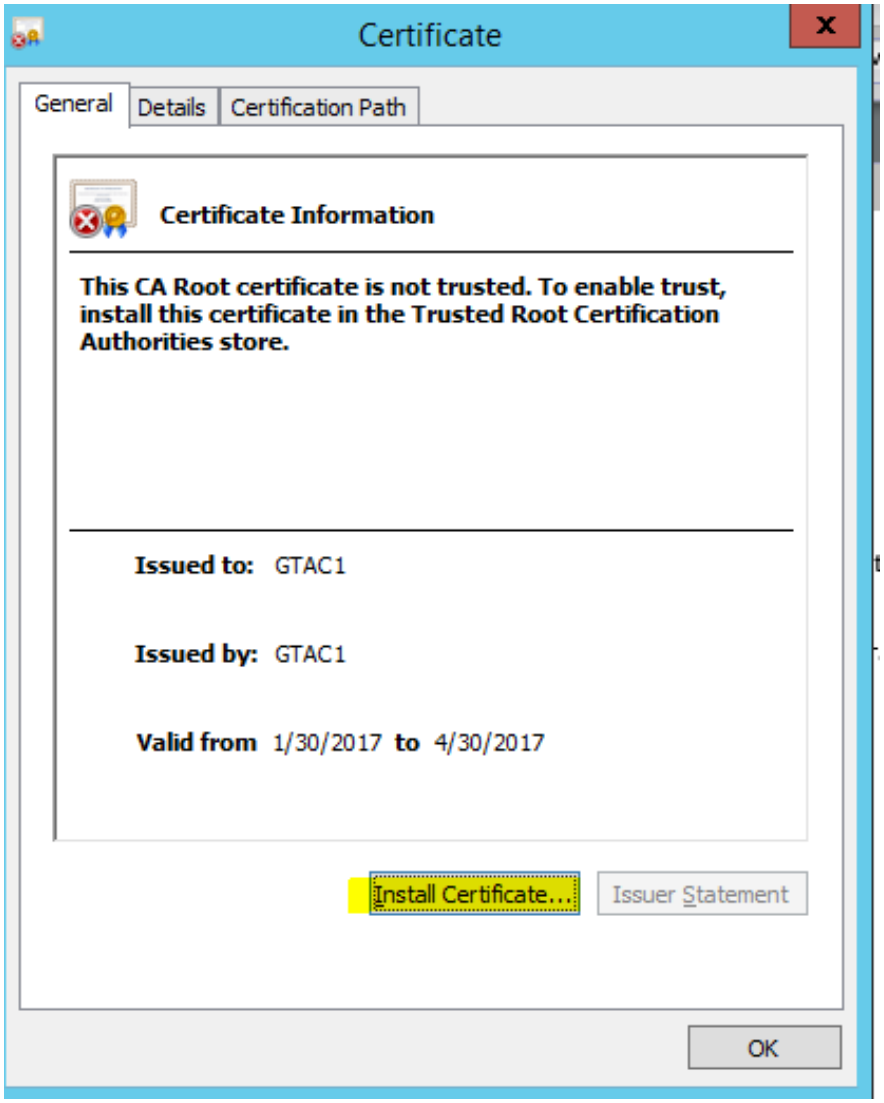

Have to select the check box in order to select 'RUN'.

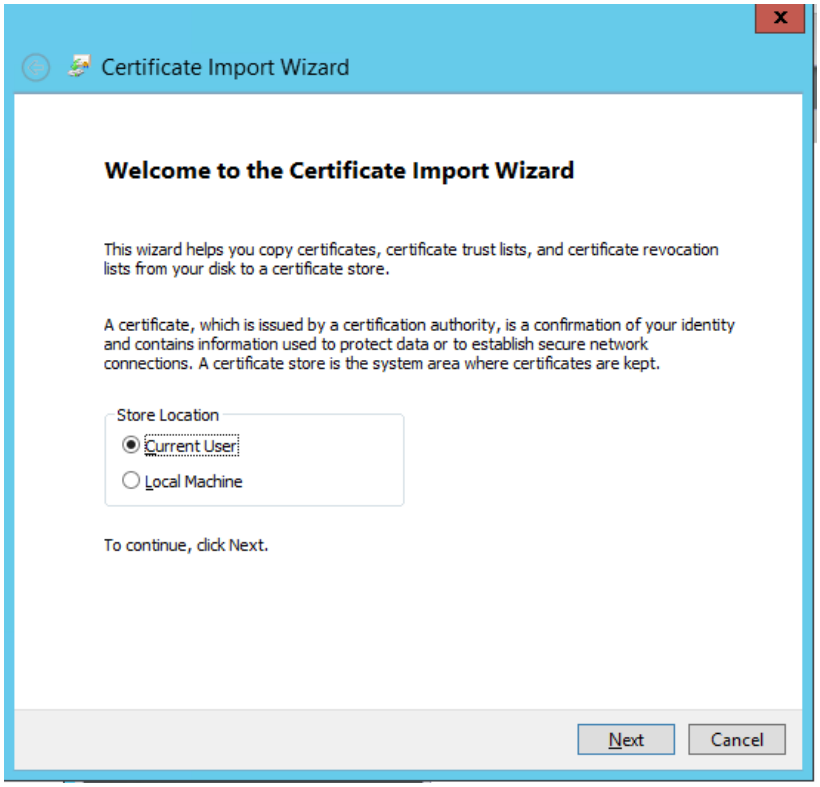
# Import Certificate into IE Browser Trust Store

To import the certificate into the IE Browser certificate store select the 'padlock' or Certificate warning located to the right of the URL address. Then select 'View certificates' option.

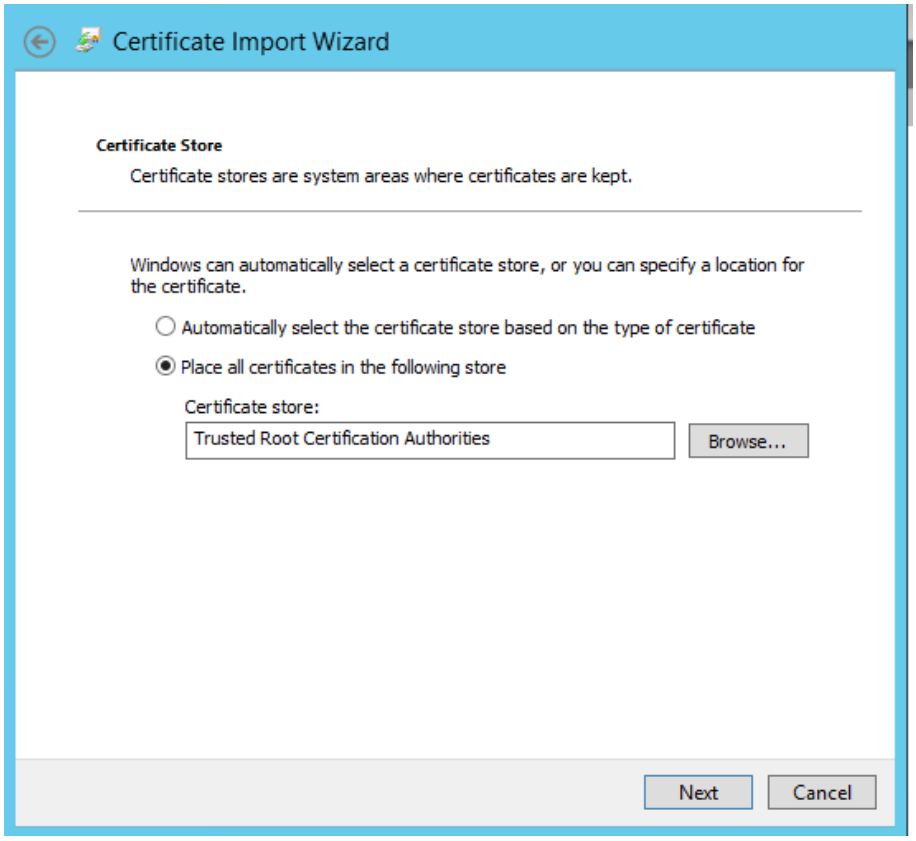


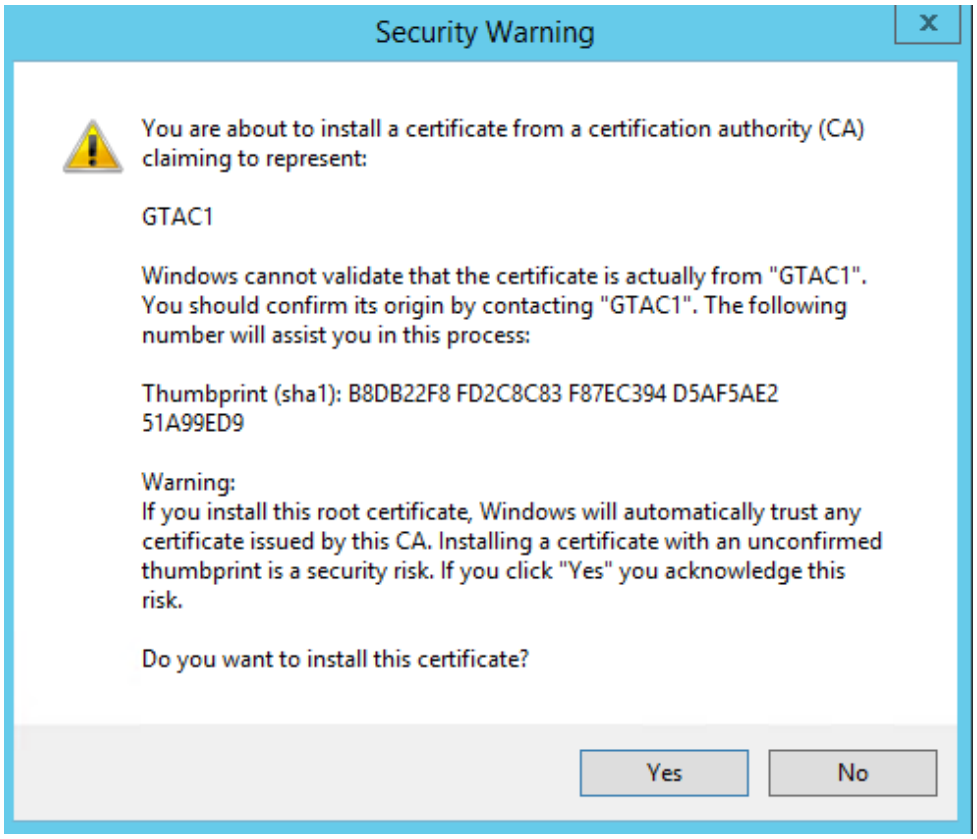
Select 'Install Certificate' option:



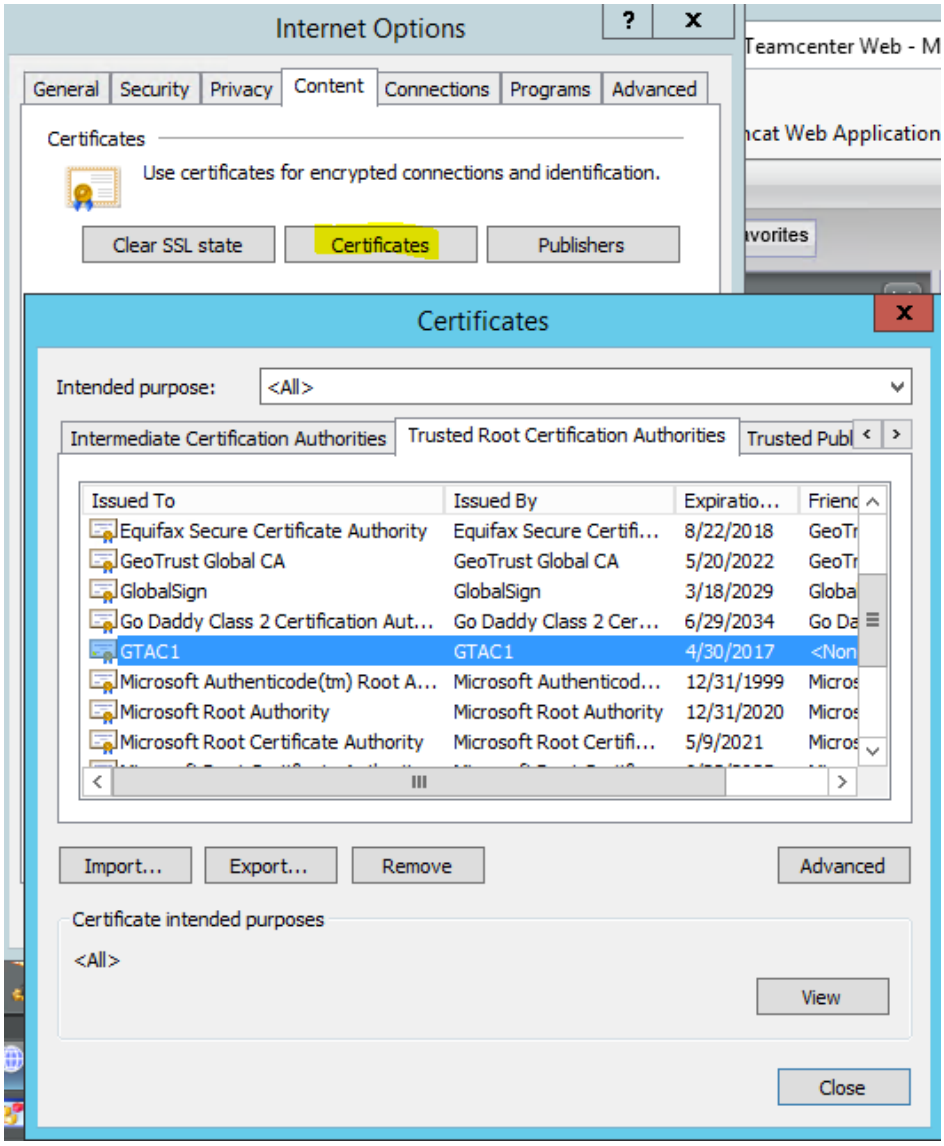



Import the certificate into the Trusted Root Certificate Authorities

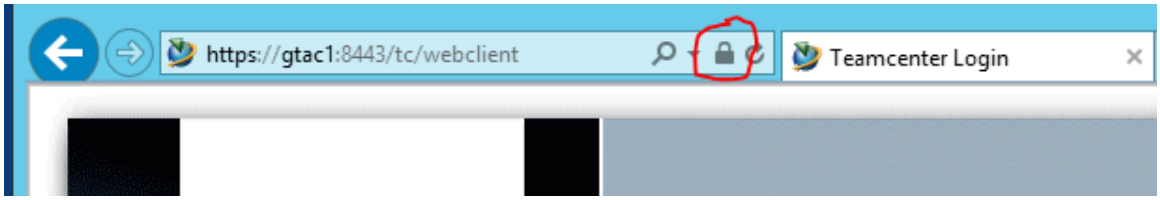



Security Warning
You are about to install a certificate from a certification authority (CA) claiming to represent:
GTAC1
Windows cannot validate that the certificate is actually from "GTAC1". You should confirm its origin by contacting "GTAC1". The following number will assist you in this process:
Thumbprint (sha1): B8DB22F8 FD2C8C83 F87EC394 D5AF5AE2 51A99ED9
Warning:
If you install this root certificate, Windows will automatically trust any certificate issued by this CA. Installing a certificate with an unconfirmed thumbprint is a security risk. If you click "Yes" you acknowledge this risk.
Do you want to install this certificate?
Yes
No

Once complete the cert will be listed in the Certificates trust store:




With that import completed, there will no longer be certificate warnings during thin client launch. Security padlock no longer RED



# Create a PEM formatted certificate file from Web Tier SSL config

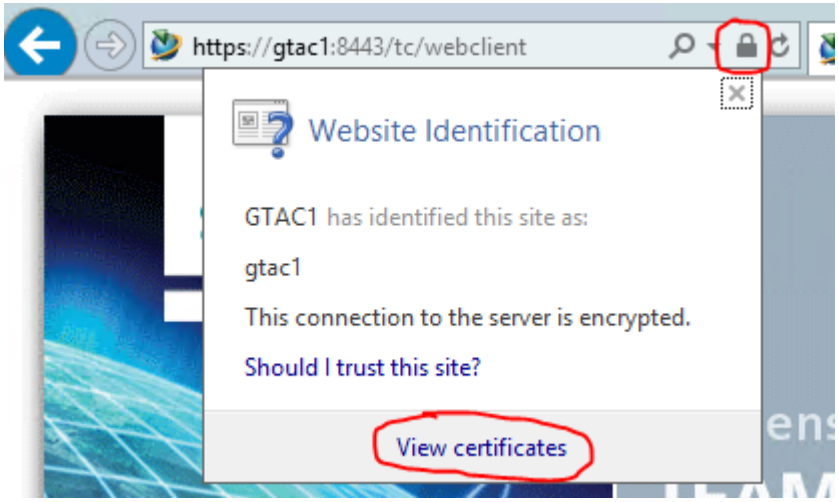
This is a proactive step as some integrations will require this specific format of the certificate.

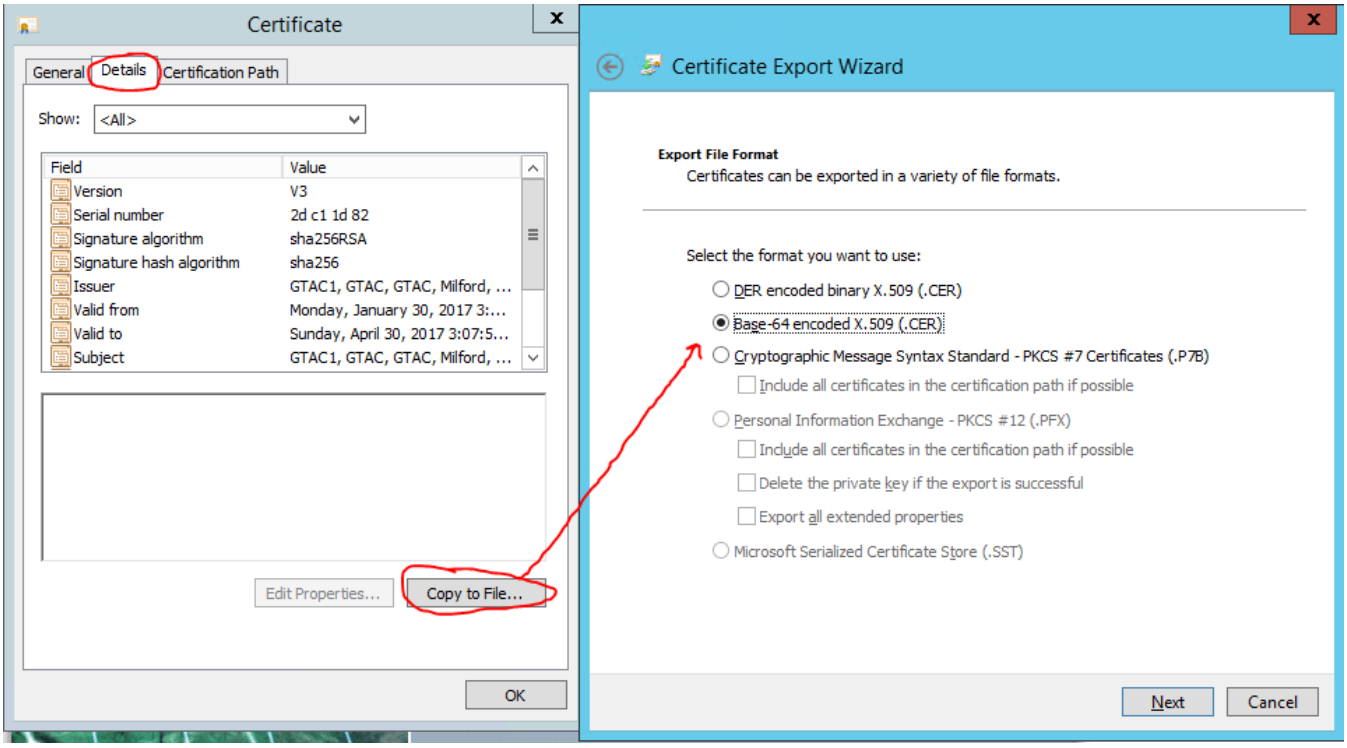
A List of integrations that will need this are:

- Native implementation of the FSCClientAgent
- NX Integration
- TCVis Integration

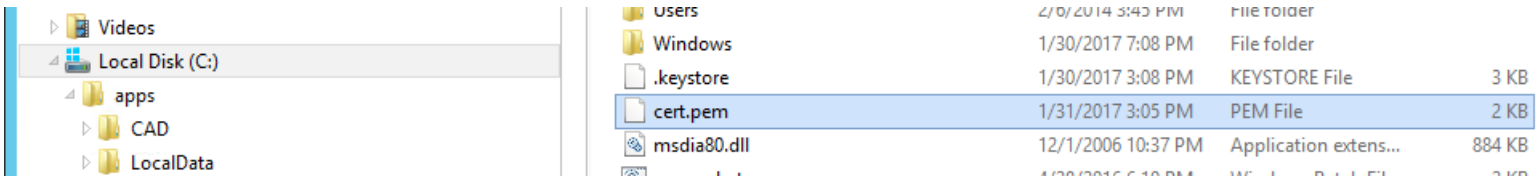
- **NOTE: The list of integration that use this will be added as confirmed**

1. One option to create a PEM formatted certificate file is to leverage the certificate already in use by the Web tier. Launch the web tier URL:



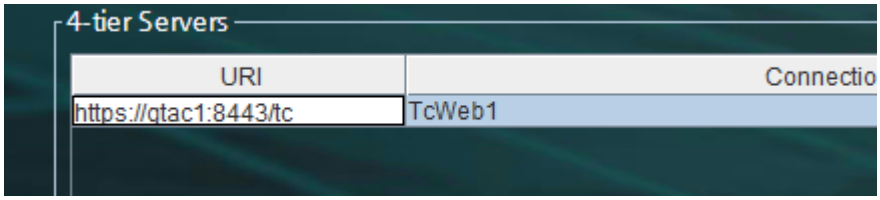



Make sure that the created PEM file has a .pem extension. From the above step the cert was saved as C:\cert.pem:



# 4-TIER updates to support Web Tier SSL

1. Update the web tier URI to be 'HTTPS' and reflect the correct SSL port:




2. Import the certificate from the 'C:\keystore' into the Teamcenter 4-tier client based Java cacerts file. The export process was completed earlier in this guide in the 'Export a certificate from the keystore using the Java keytool utility' section. From that export a 'c:\webappcert.crt' file was created. That cert file is the one that will be imported into the java keystore in the command below. %JAVA_HOME% = installation location of the Java that the 4-tier client is using.

   %JAVA_HOME%\bin\keytool -importcert -alias gtac -file c:\webappcert.crt -keystore %JAVA_HOME%\lib\security\cacerts -storepass changeit

   NOT A PROCESS TESTED IN THIS GUIDE but it may be possible to import directly out of the C:\keystore file. For example: %JAVA_HOME%\bin\keytool -importcert -alias gtac -file c:\keystore -keystore %JAVA_HOME%\lib\security\cacerts -storepass changeit

   The output from executing that command:

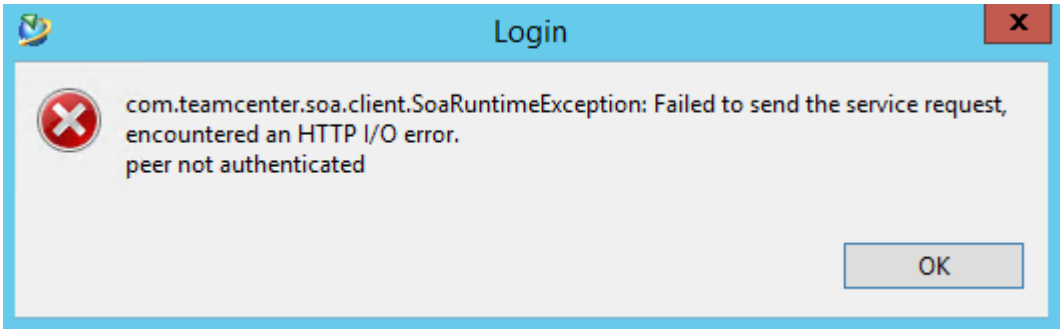
```
C:\jre780_64\bin>%JAVA_HOME%\bin\keytool -importcert -alias gtac -file c:\webappcert.crt -keystore
%JAVA_HOME%\lib\security\cacerts -storepass changeit
Owner: CN=GTAC1, OU=GTAC, O=GTAC, L=Milford, ST=OH, C=US
Issuer: CN=GTAC1, OU=GTAC, O=GTAC, L=Milford, ST=OH, C=US
Serial number: 2dc11d82
Valid from: Mon Jan 30 15:07:55 EST 2017 until: Sun Apr 30 16:07:55 EDT 2017
Certificate fingerprints:
     MD5:  47:DE:B1:FD:62:77:D6:82:40:38:EF:B6:A7:41:5E:BC
     SHA1: B8:DB:22:F8:FD:2C:8C:83:F8:7E:C3:94:D5:AF:5A:E2:51:A9:9E:D9
     SHA256: C8:49:86:5E:91:E1:9B:B8:93:2B:2D:F9:84:97:8C:00:08:A9:21:BF:41:31:69:81:EF:B1:2F:64:5E:08:A9:21
     Signature algorithm name: SHA256withRSA
          Version: 3
          Extensions:
          #1: ObjectId: 2.5.29.14 Criticality=false
          SubjectKeyIdentifier [
          KeyIdentifier [
          0000: 84 A6 BF AF 5A 55 FD F8   4E 71 53 11 3E 14 96 91  ....ZU..NqS.>...
          0010: C2 51 CE 9E                                        .Q..
          ]
          ]

          Trust this certificate? [no]:  y
          Certificate was added to keystore
```

C:\jre780_64\bin>

3. To confirm if the certificate is in the keystore that it was imported in or to see if it has already been imported previously use the '-list' option of the keytool utility. For example:
   %JAVA_HOME%\bin\keytool -list -keystore %JAVA_HOME%\lib\security\cacerts -storepass changeit
   Output highlighting the 'gtac' cert that was imported from step 2:

```
Certificate fingerprint (SHA1): C9:3C:34:EA:90:D9:13:0C:0F:03:00:4B:98:BD:8B:35:70:91:56:11
verisignclass2g2ca, Mar 25, 2004, trustedCertEntry,
Certificate fingerprint (SHA1): B3:EA:C4:47:76:C9:C8:1C:EA:F2:9D:95:B6:CC:A0:08:1B:67:EC:9D
gtac, Jan 31, 2017, trustedCertEntry,
Certificate fingerprint (SHA1): B8:DB:22:F8:FD:2C:8C:83:F8:7E:C3:94:D5:AF:5A:E2:51:A9:9E:D9
geotrustprimarycag3, Dec 10, 2009, trustedCertEntry,
Certificate fingerprint (SHA1): 03:9E:ED:B8:0B:E7:A0:3C:69:53:89:3B:20:D2:D9:32:3A:4C:2A:FD
geotrustprimarycag2, Dec 10, 2009, trustedCertEntry,
Certificate fingerprint (SHA1): 8D:17:84:D5:37:F3:03:7D:EC:70:FE:57:8B:51:9A:99:E6:10:D7:B0
swisssigngoldg2ca, Oct 31, 2008, trustedCertEntry,
```

NOTE: If the certificate is not imported into the Teamcenter 4-tier client Java cacerts file then the following error will appear during login:



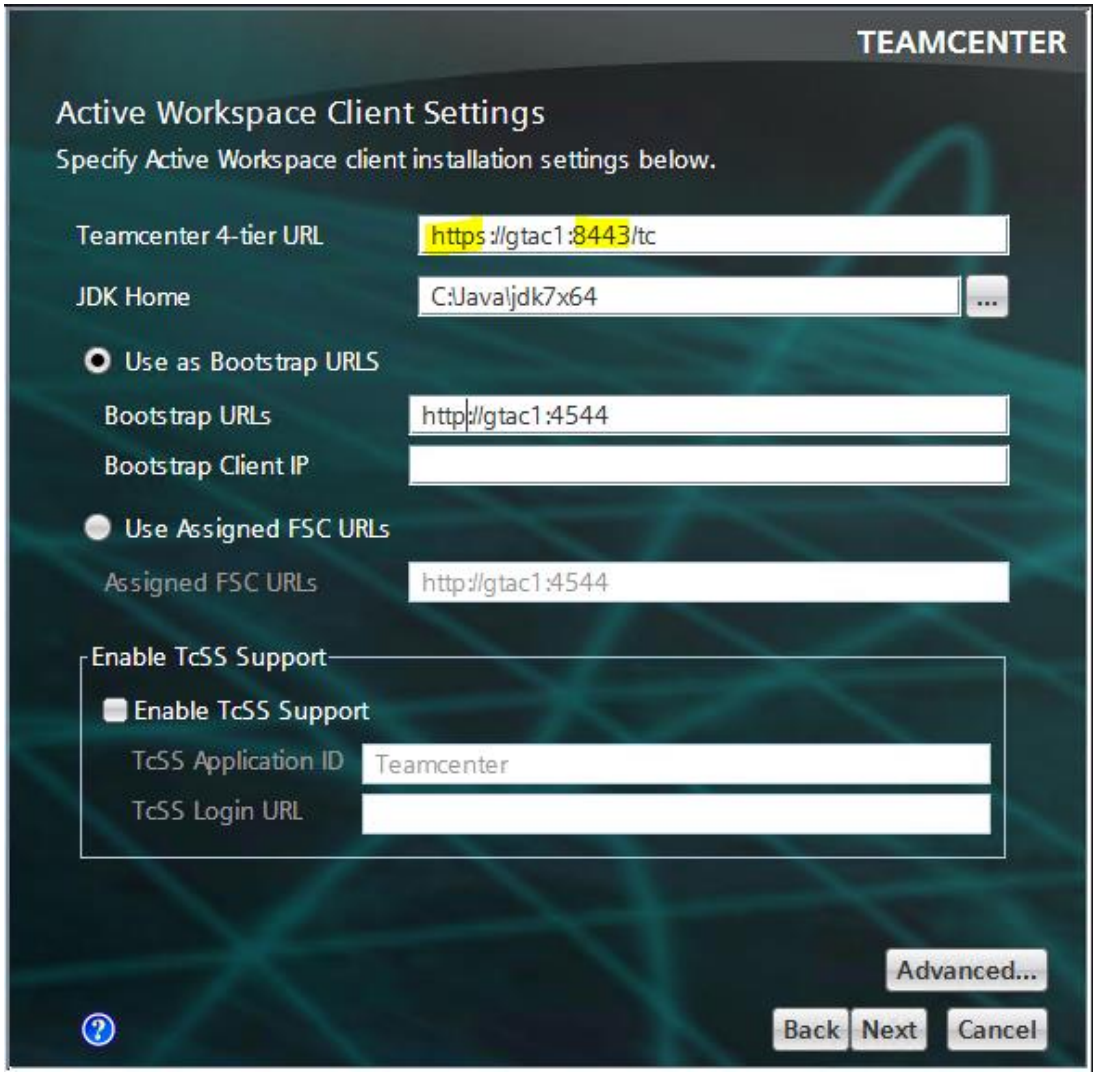

NOTE: If you have other 4-tier clients at the same java version you can copy this cacerts file to all other client java installations to save you from manually running the import command on each.

# 2-TIER updates to support Web Tier SSL

No updates required as it does not connect to web tier during login process

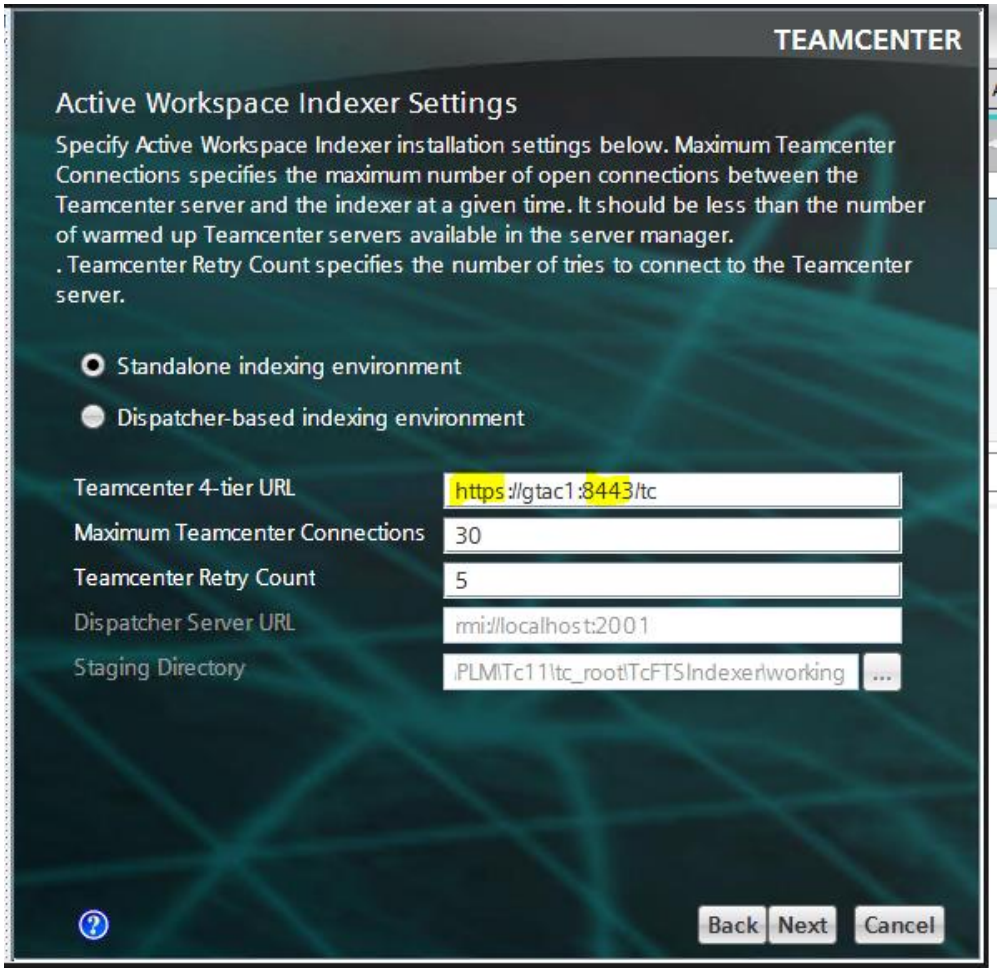
# Teamcenter Active Workspace Client updates for Web Tier SSL

Update the 'Teamcenter 4-tier URL' and FSC settings to reflect correct HTTPS protocol and port:

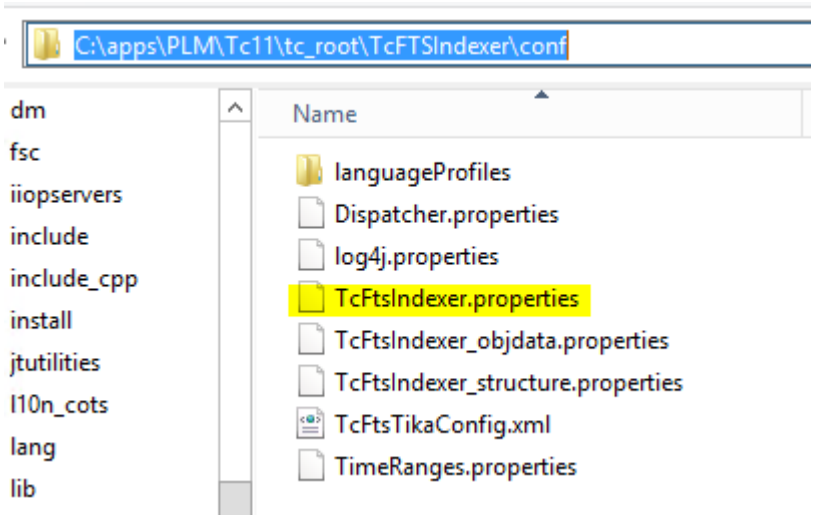


# Teamcenter Active Workspace FTS Indexer updates for Web Tier SSL

4-tier URL entry via TEM:



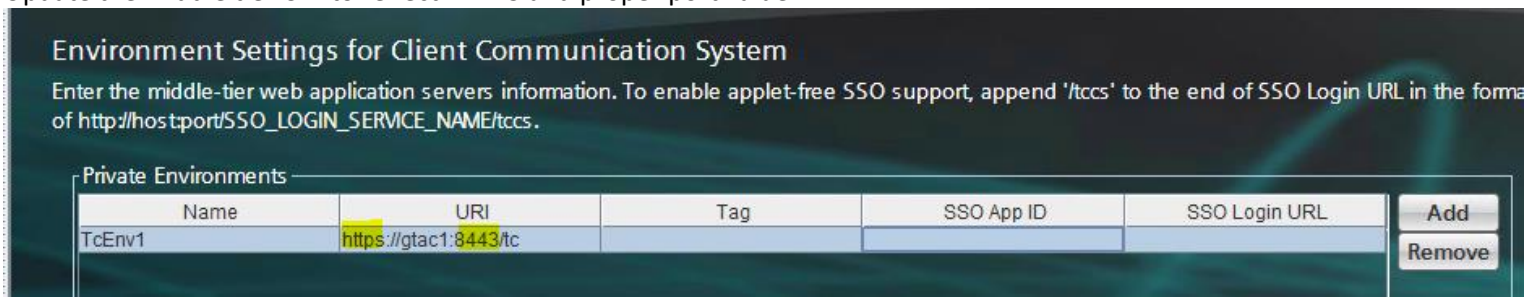
OR 4-tier URL entry in the 'TcFtsIndexer.properties' file:



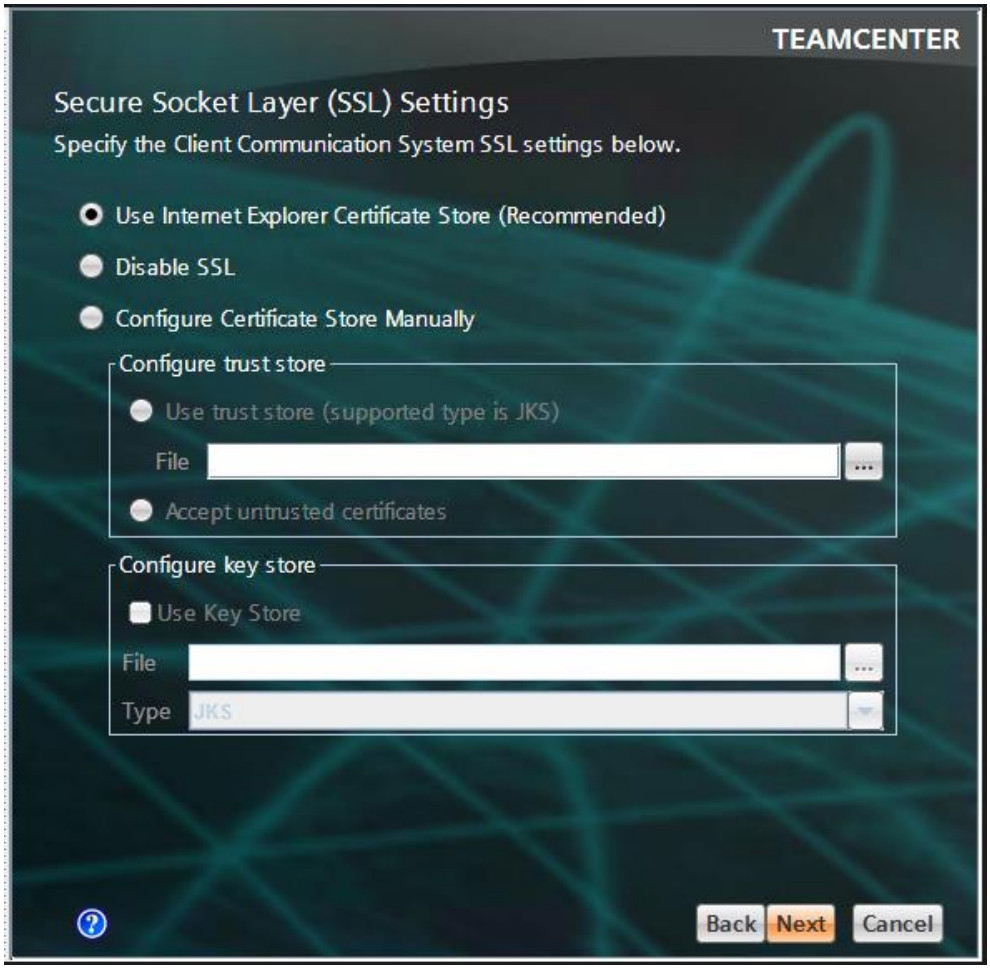

```
5  #==============================================================
6  # Description:
7  #    URL to connect to Teamcenter to extract data.
8  #==============================================================
9  #
10 Tc.URL=https://gtac1:8443/tc
11
```

# Teamcenter Client Communication (TCCS) updates for Web Tier SSL

Update the middle tier URI to reflect HTTPS and proper port value:

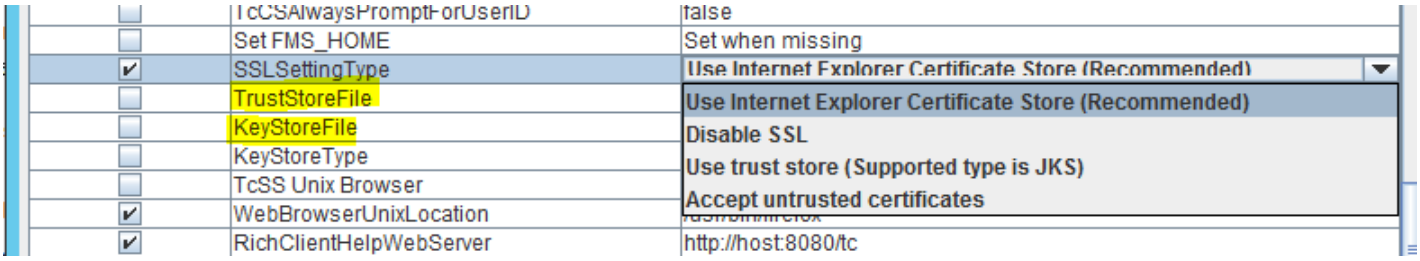



The default SSL setting is to reference the IE trust store for certificate validation. One can select to manually configure an alternate trust store file or key store file. Either of those options could be used to point to a Java cacerts file for example.

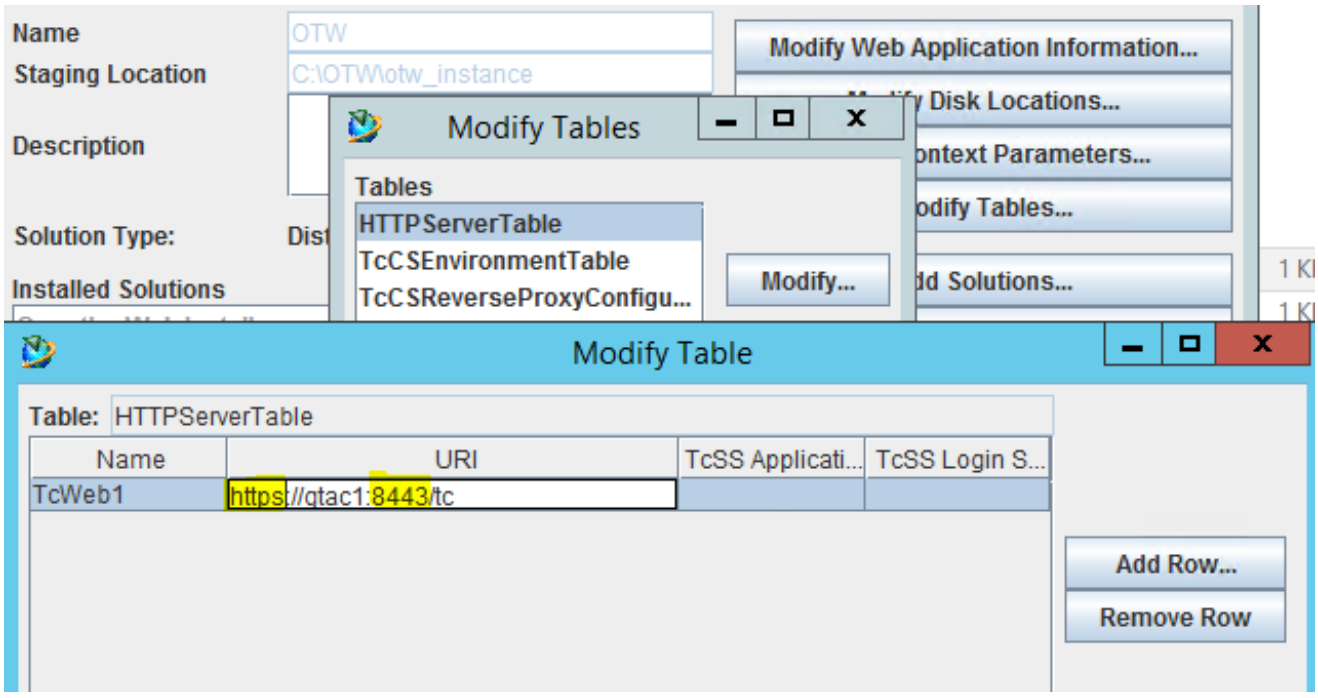


# Teamcenter Over the Web updates for Web Tier SSL

SSL Setting Type can be updated to reflect where the 4-tier client will look to validate the certificate. If selecting the 'trust store' option then enter proper values for the 'TrustStoreFile' or 'KeyStoreFile' path and file as located from the clients perspective.

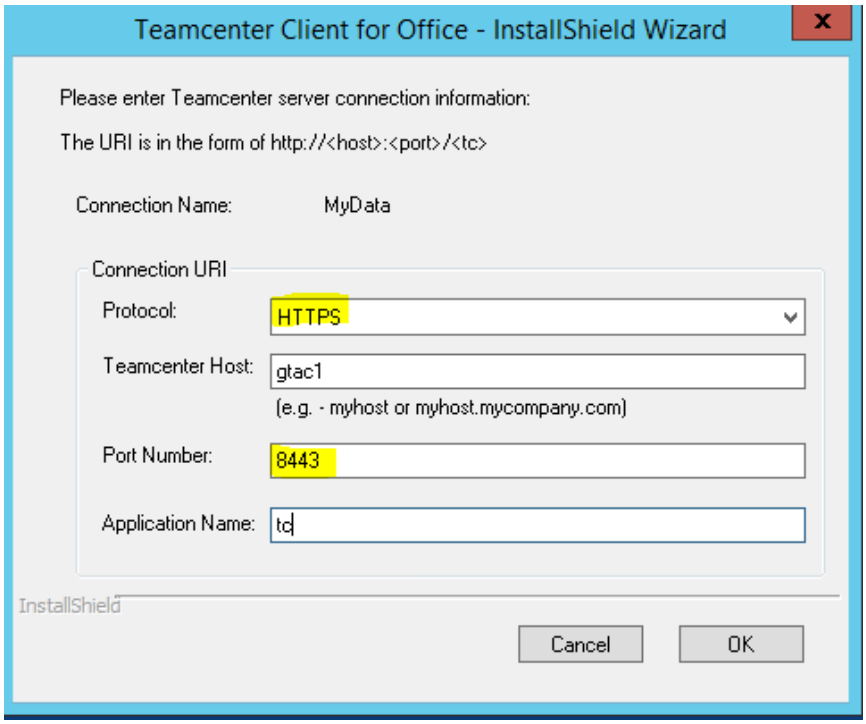



Update the middle tier URI to reflect HTTPS and proper port value:

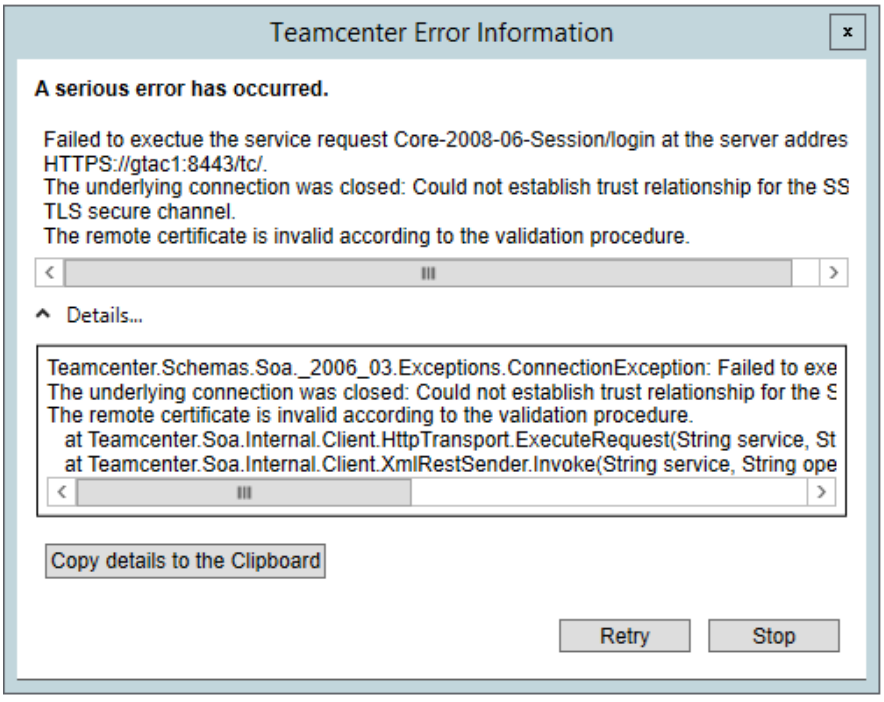


# Teamcenter Client for Office updates for Web Tier SSL

Update the middle tier URI to reflect HTTPS and proper port value:

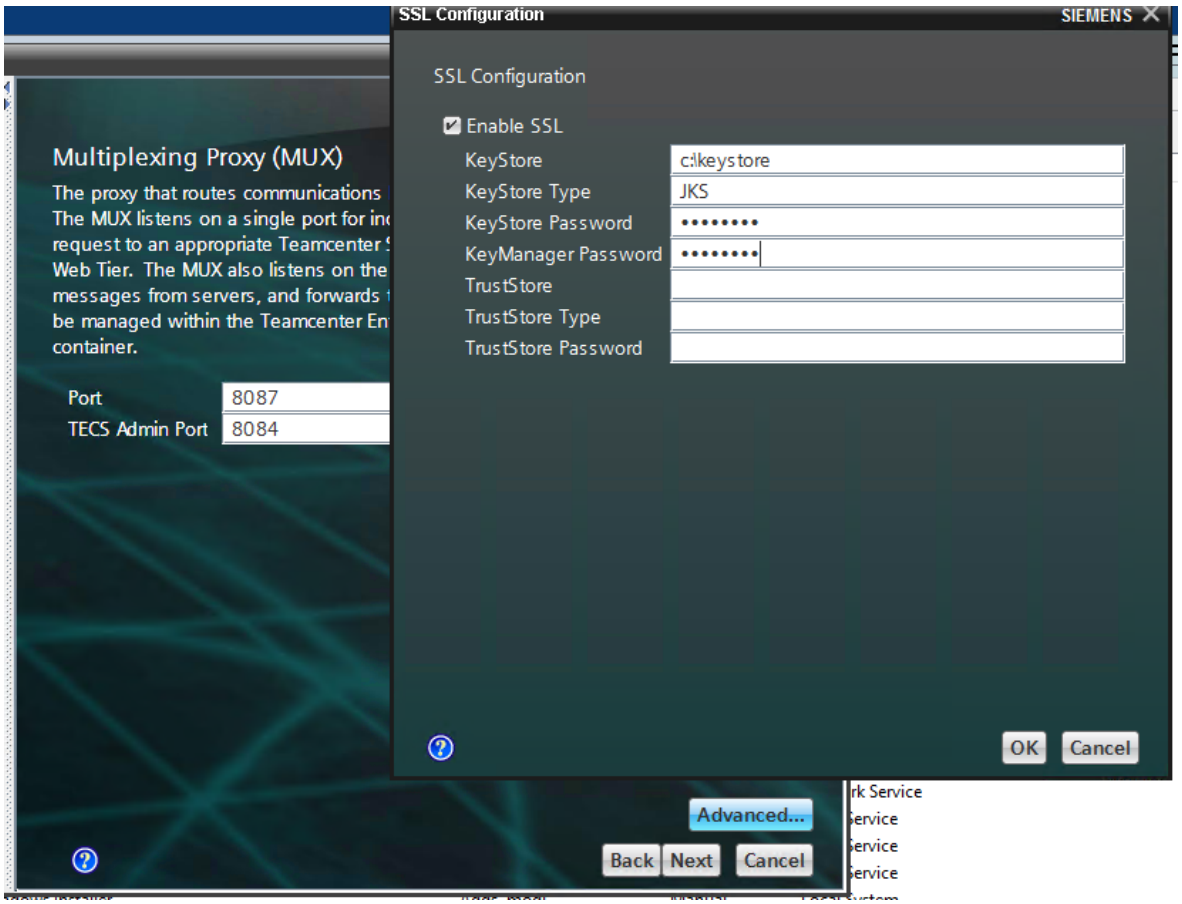



NOTE: The certificate validation is done through the browser trust store. So if the certificate is not in that trust store the following message during login will appear:

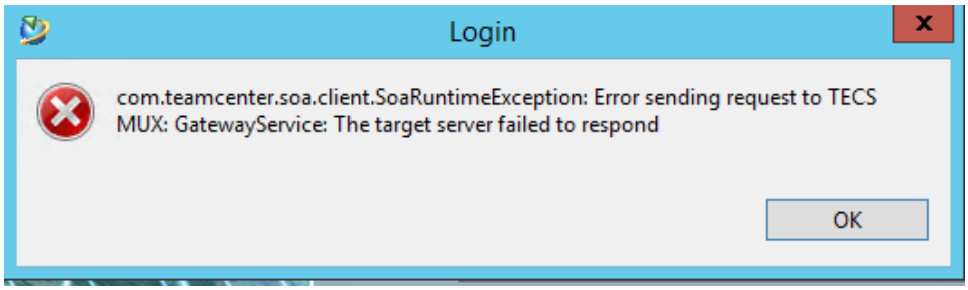


# Teamcenter MUX (J2EE web tier) Updates

- This is updated via TEM within the Java EE based Server Manager.
- Enable SSL in the ADVANCED panel and set the Keystore to reflect the same keystore file created earlier in this guide (c:\keystore). The format of that keystore file is JKS with a password of 'changeit'. KeyManager Password also set to 'changeit'.




This is the type of pop up the user will see if MUX is not configured correctly with SSL.




NOTE: Stopping the Server Manager may not stop the MUX. To shut it down run the 'tecsstop.bat' inside the Server Manager folder.

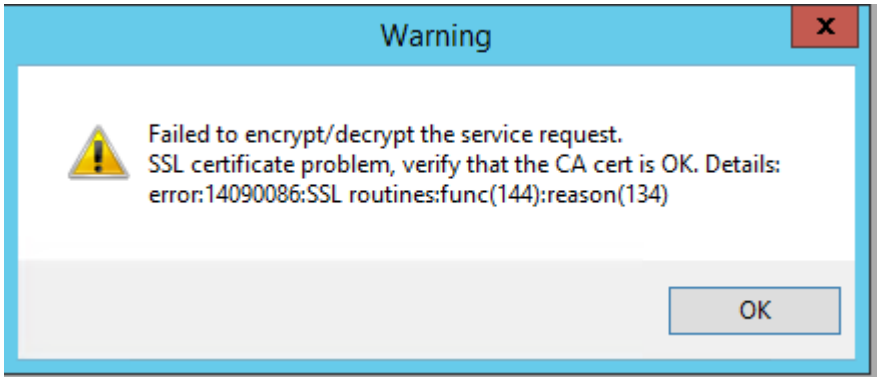
# Teamcenter Visualization Updates

## 2-tier

- Open in Standalone Vis or Embedded Vis within a 2-tier client does not require any additional configuration.

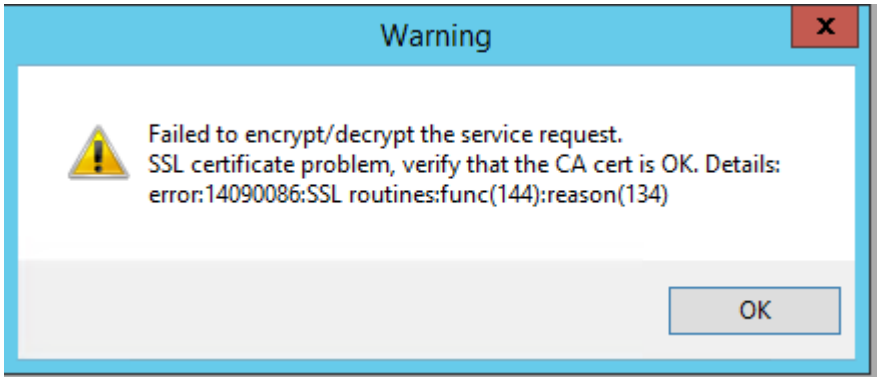
## 4-tier

- Opening in Standalone from within a 4-tier results in this message:



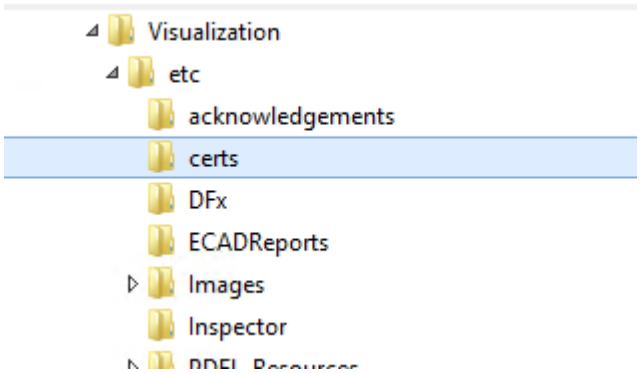

  Selecting OK on the above form but the same form keeps appearing. Have to kill Vis manually.

- Opening in Embedded Vis from within a 4-tier results in the same message:



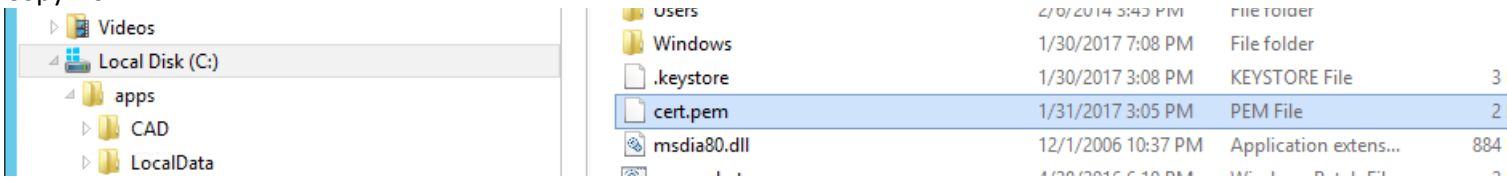
SOLUTION:

TCVis has its own 'certs' folder within its installation.

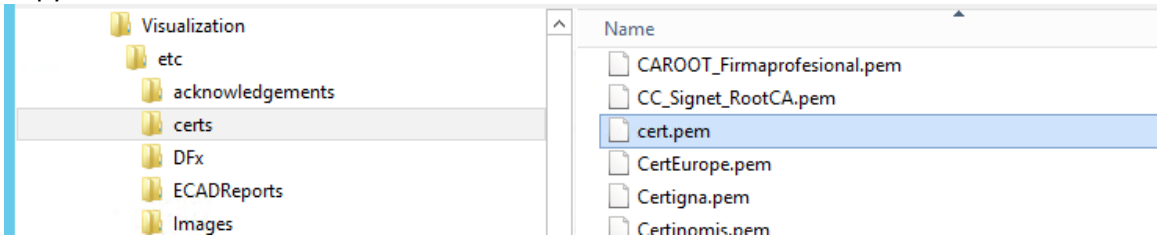



Convert the certificate into a PEM format (performed in the 'Create a PEM formatted certificate file from Web Tier SSL config' section of this guide) and copy it into this folder. For example copy the C:\cert.pem file, created from the previous section of this guide, into the TCVis 'cert' folder:

Copy from:




Copy to:



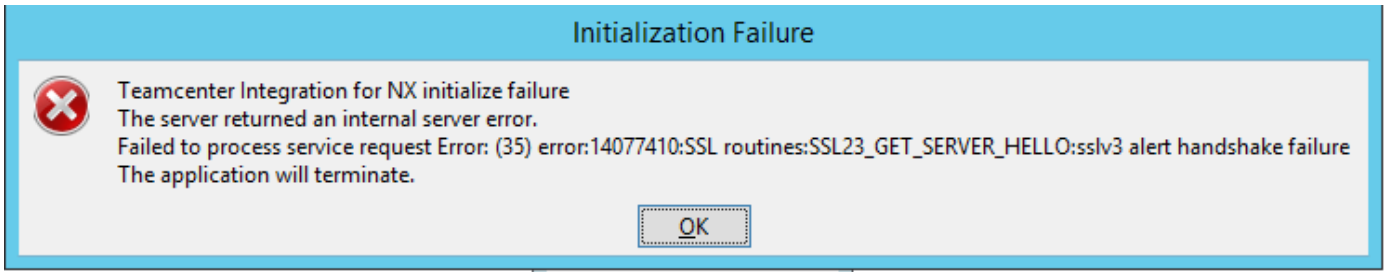
# NX Integration Updates

## 2-tier

- Open in NX does not require any additional configuration.

## 4-tier

- Open in NX failed with the following message:




Solution:

Set the following variable to point to the PEM formatted certificate file. PEM file creation outlined earlier in the 'Create a PEM formatted certificate file from Web Tier SSL config' of this guide. In that section the created pem file was: c:\cert.pem

```
set TEAMCENTER_SSL_CERT_FILE=c:\cert.pem
```

This can be set as a system variable on the client host or within the startnxmanager.bat file:

```
91  rem Start NX through the launcher program
92
93  set TEAMCENTER_SSL_CERT_FILE=c:\cert.pem
94  start "Teamcenter Integration for NX" /B "%UGII_ROOT_DIR%\ugs_router" -ugm -enable_cancel -
95  goto :ENDOFFILE
96
97  :nx_error
```

If setting that variable still shows the error, review the protocol setting in the https connector section of the Jboss standalone.xml. Setting it to 'ALL' was found to help resolve the issue.

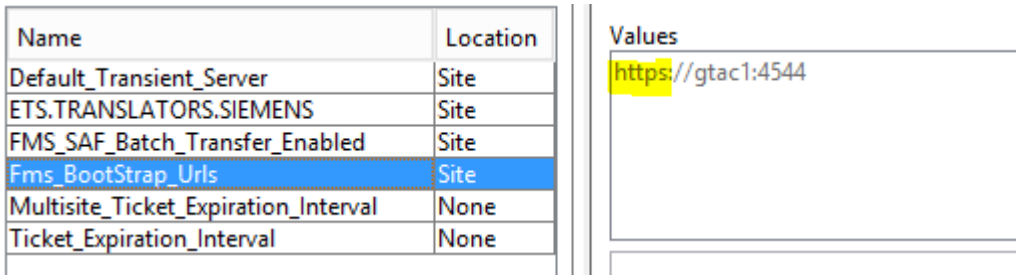
```
<subsystem xmlns="urn:jboss:domain:web:1.1" native="false" default-virtual-server="default-host">
      <connector name="http" protocol="HTTP/1.1" scheme="http" socket-binding="http"/>
       <connector name="https" protocol="HTTP/1.1" scheme="https" socket-binding="https" secure="true">
            <ssl key-alias="gtac" password="changeit" certificate-key-file="C:/keystore" protocol="ALL" verify-
         client="false"/>
       </connector>
    <virtual-server name="default-host" enable-welcome-root="true">
          <alias name="localhost"/>
          <alias name="example.com"/>
    </virtual-server>
</subsystem>
```

Setting that protocol to TLSv1 for example did not correct the issue. Others [SSLv2Hello, TLSv1, TLSv1.1, TLSv1.2]

# Configuring FMS for one way SSL

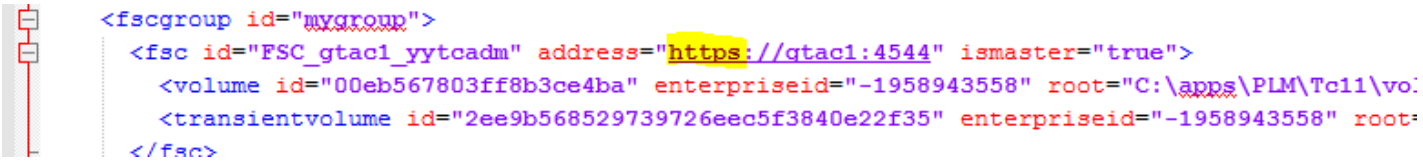
## Teamcenter Preference:

Add the URL of the local HTTPS host to the list of servers defined in the Fms_BootStrap_Urls preference.




## FMS Master:

Update the FMS master to reflect the correct SSL based URL address




## FSC configuration:

Update the fsc.FSC_<host>_<userid>.properties file in the FSC_HOME directory with the keystore information. The information used below is based on the keystore created in the 'Create a self-signed certificate via Java keytool utility' section of this guide.
Copy the 'fsc.properties.template' file to 'fsc.FSC_<host>_<userid>.properties' and open it for edit.

**NOTE:** Important that the properties filename be in the format listed above. Example for this would be:
fsc.FSC_gtac1_yytcadm.properties

Edit the keystore related fields. Notice the direction of the slashes (/). FSC will fail to start and complain about file not found if the slashes are not correct.

```
#
# These are required for ssl support:
#
com.teamcenter.fms.servercache.keystore.file=c:/keystore
com.teamcenter.fms.servercache.keystore.password=changeit
com.teamcenter.fms.servercache.keystore.ssl.certificate.password=changeit
#
```

Since this process is using a self signed certificate we need to also uncomment this line in the 'fsc.FSC_gtac1_yytcadm.properties' file as well:

```
#
# Optionally trust self signed server certificates:
com.teamcenter.fms.allowuntrustedcertificates=true
#
```

# FCC Configuraton:

Non-Native:
Update FCC.xml to reflect parent FSC's SSL based URL address.

```
<!-- default parentfsc - this is a marker that will be overwritten
<parentfsc address="https://gtac1:4544/" priority="0" />
</fccconfig>
```

With that in place the 4-tier client login appears to have hung and sits at the login page with the status bar continually scrolling across the page. Review of the tcserver.fscproxlog shows this:

```
Tue Jan 31 14:31:06 2017-com.teamcenter.fms.servercache.proxy.FSCNativeClientProxy Thr3196
  N:\units\fms\fms.11.2.3_wnti32\src\FSCNativeClientProxy\FSCClientAgent.cpp:96
  WARNING: FSCClientAgent.init(): https://gtac1:4544/mapClientIPToFSCs failed: 60 CURLE_SSL_CACERT: Problem with the CA cert(path?)
Tue Jan 31 14:31:36 2017-com.teamcenter.fms.servercache.proxy.FSCNativeClientProxy Thr3196
```

Two ways to correct this:

1. For 4-tier, set the following within the %TC_DATA%\tc_profilevars.bat file OR set as a system variable. In either case the Server Manager will need to be restarted after setting it:

   SET TC_USE_FSC_CPROXY_VIA_JVM=true

   For 2-tier, set the same variable in %TC_DATA%\tc_profilevars.bat or as a system variable as well.

2. Configure native FSC client proxy:
   Create the %FSC_HOME%/fsc.clientagent.properties file. Add the 'com.teamcenter.fms.curl.cacerts.file' property to it and set its value to the absolute path and file name to the PEM formatted file. For example based on the PEM file created in an earlier step of this guide ('Create a PEM formatted certificate file from Web Tier SSL config') the line would appear as follows:

   com.teamcenter.fms.curl.cacerts.file=C:\cert.pem

   In addition to that line, given that this is a untrusted self-signed certificate, add the following line as well:

   com.teamcenter.fms.allowuntrustedcertificates=true

   File content after edit will look like this:

```
#
#
com.teamcenter.fms.curl.cacerts.file=C:\cert.pem

com.teamcenter.fms.allowuntrustedcertificates=true

```

## FSCADMIN utility update:

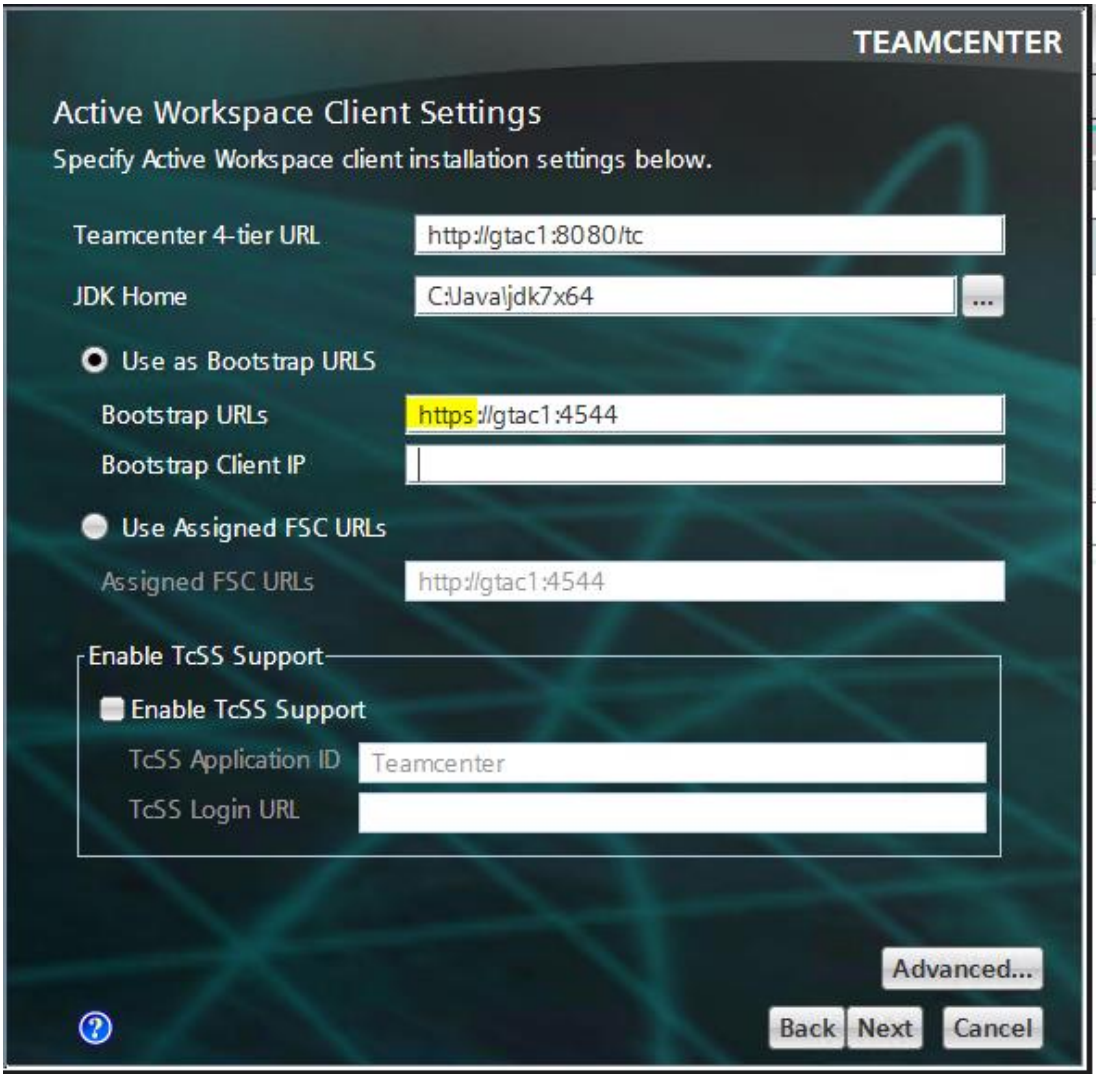
The fscadmin script most also be configured to allow untrusted certificates. Use the %FSC_HOME%\fscadmin.properties.template file to create a %FSC_HOME%\fscadmin.properties file. Within that file uncomment the line shown below:

```
# Optionally trust self signed server certificates:
com.teamcenter.fms.allowuntrustedcertificates=true
#
```

If the self-signed certificate has been imported into the Java cacerts file of the JAVA that the FSCAdmin utility is using, then this file and edit is not required. Otherwise, if the cert is untrusted the following type of message will appear when trying to run the fscadmin utility:

C:\apps\PLM\Tc11\tc_root\fsc>fscadmin -s https://gtac1:4544 ./cachesummary
OS name is Windows Server 2012 R2
JVM vendor is Oracle Corporation
Data model is 32
javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException: PKIX path validation failed:
java.security.cert.CertPathValidatorException: signature check failed
    at sun.security.ssl.Alerts.getSSLException(Unknown Source)
    at sun.security.ssl.SSLSocketImpl.fatal(Unknown Source)
    at sun.security.ssl.Handshaker.fatalSE(Unknown Source)
    at sun.security.ssl.Handshaker.fatalSE(Unknown Source)

# Teamcenter Active Workspace Client updates for FMS FSC SSL

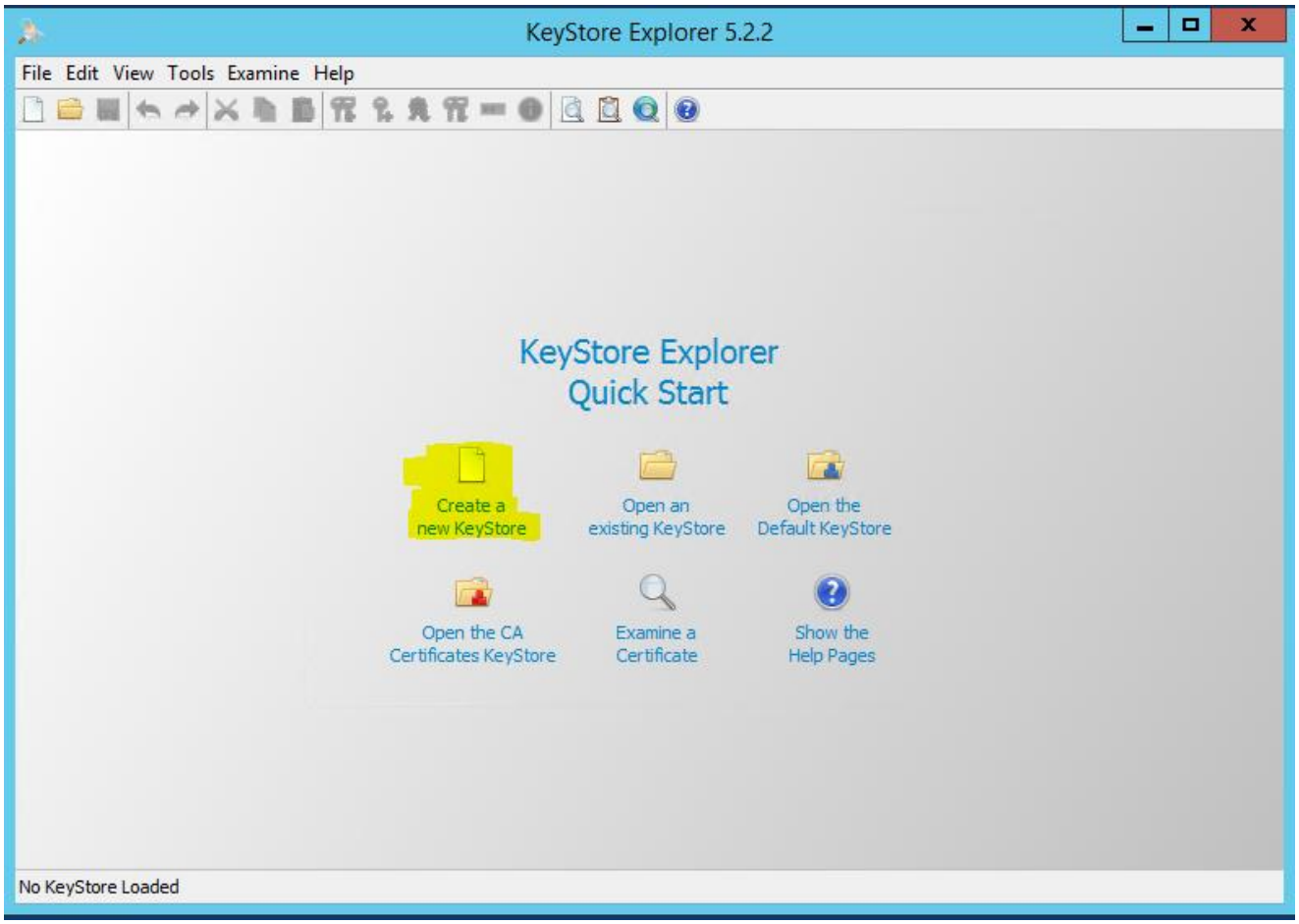


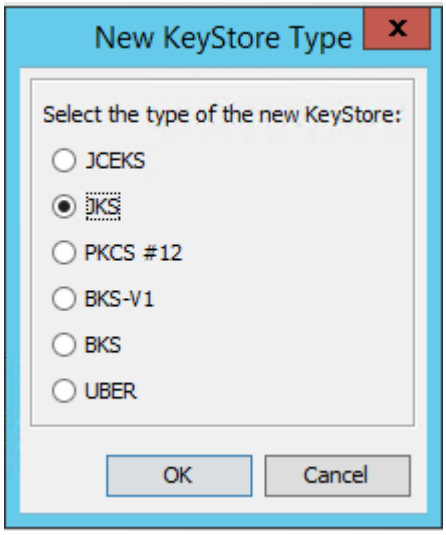
# Keytool Explorer

## Download

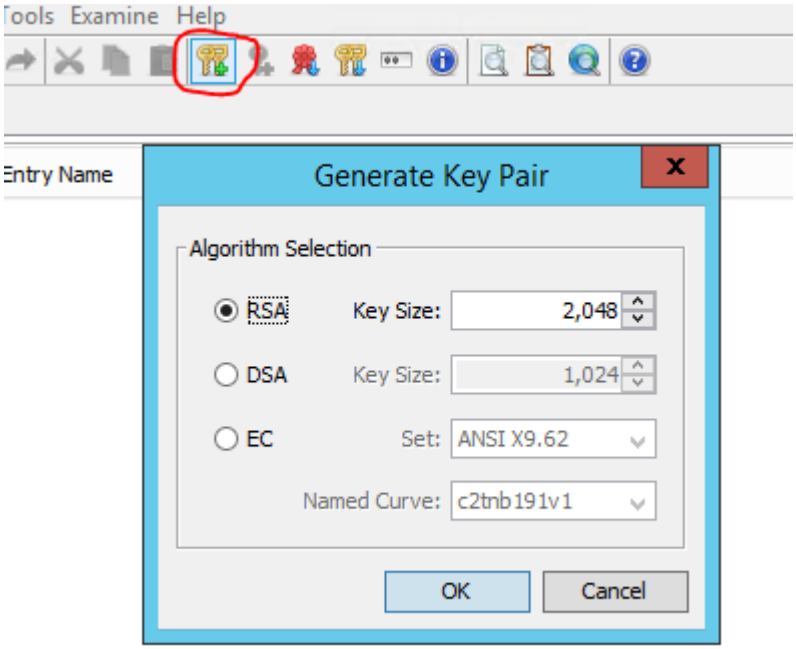
http://keystore-explorer.org/downloads.html

## Generate new keystore with self signed certificate

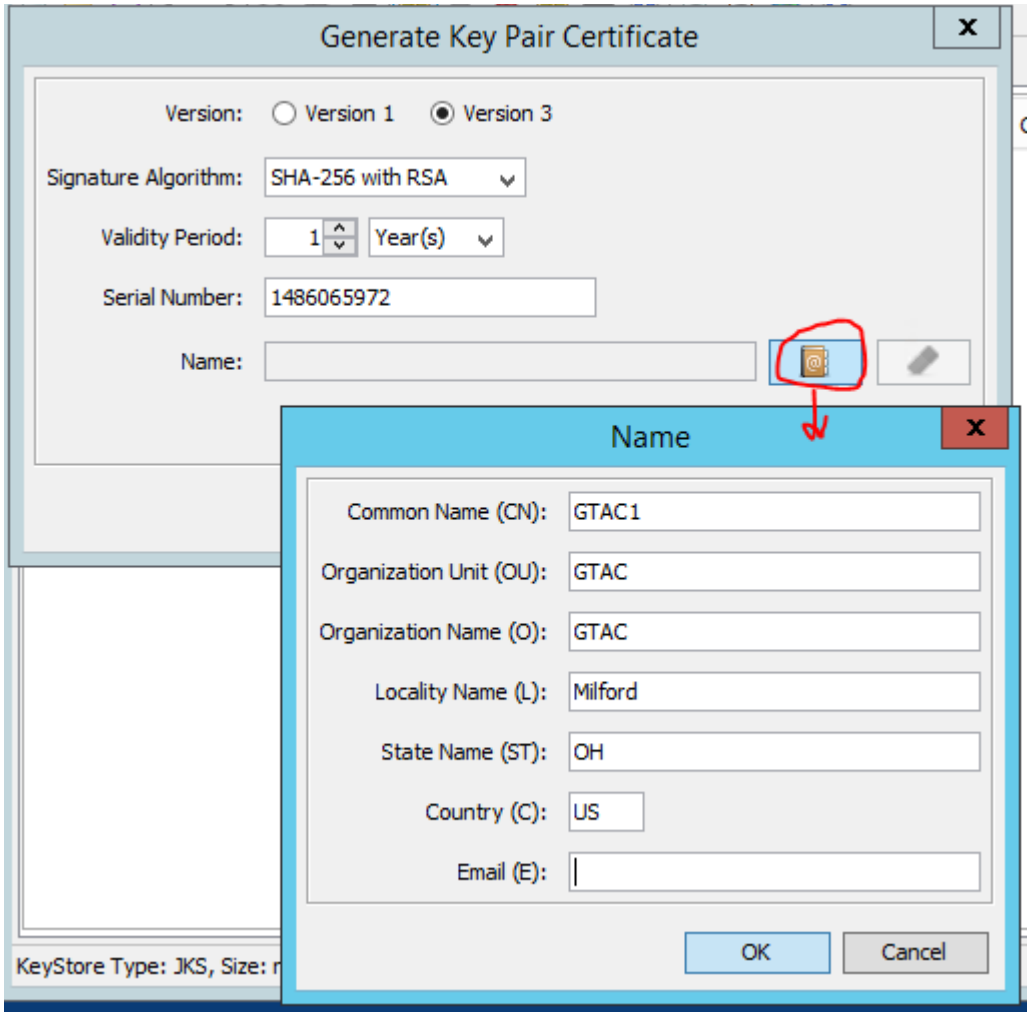




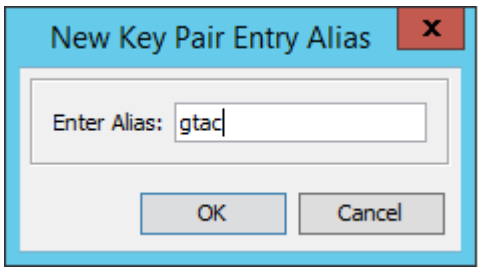

Select the 'Generate Key Pair' icon and accept defaults:



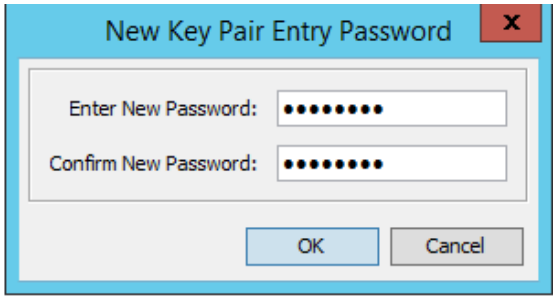
Select 'EDIT NAME' icon and populate the fields. Remember that the 'Common Name (CN)' needs to reflect the host name of the web application server. The CN **MUST** match how that hostname will be provided in the URL. For example: If the URL that end users will use is fully qualified (myserver.us.com) then the CN needs to be 'myserver.us.com'. If the CN name in the certificate does not match the hostname the connection will fail.

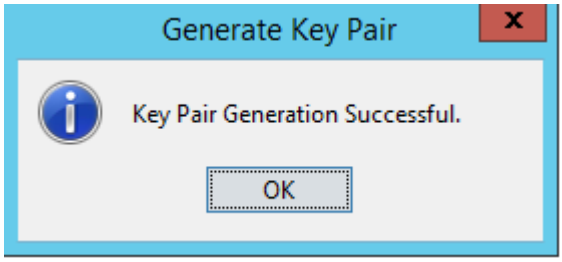



Provide an alias name:

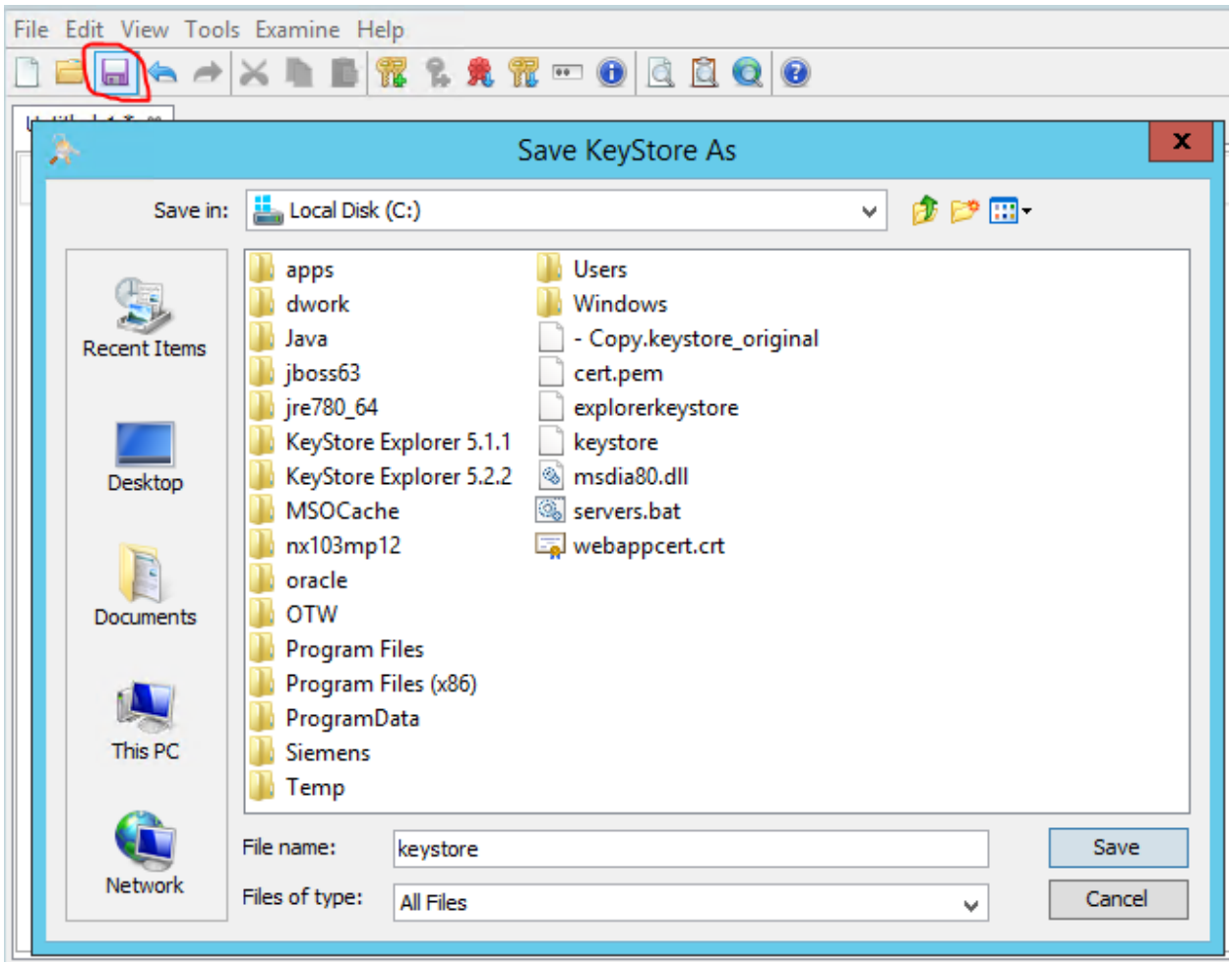


Provide a password.



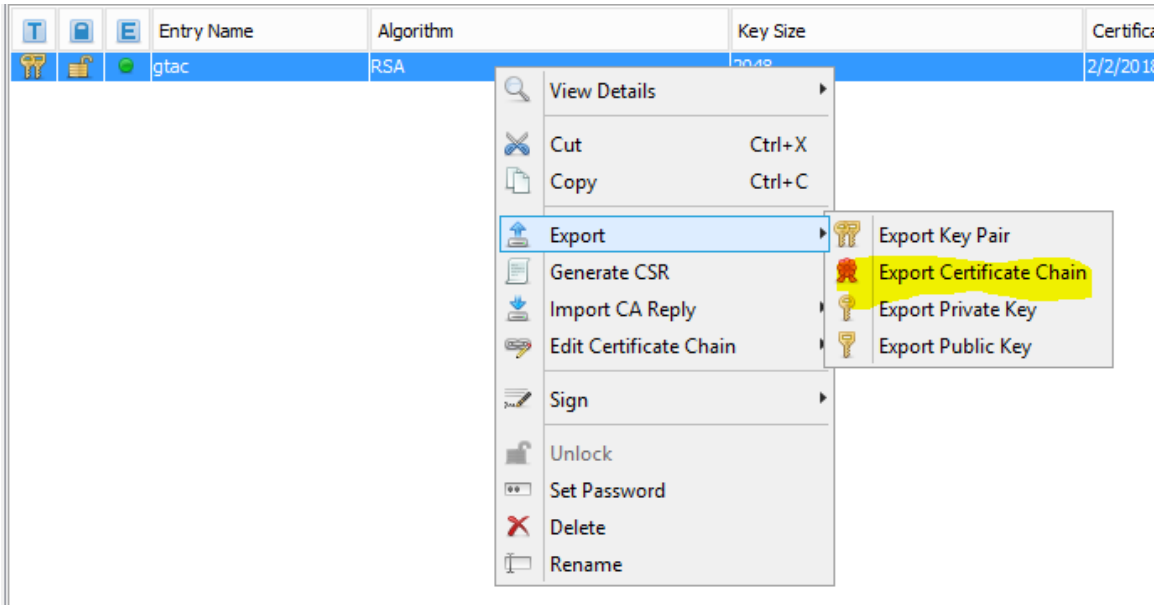




Save the new keystore to a location of your choice:

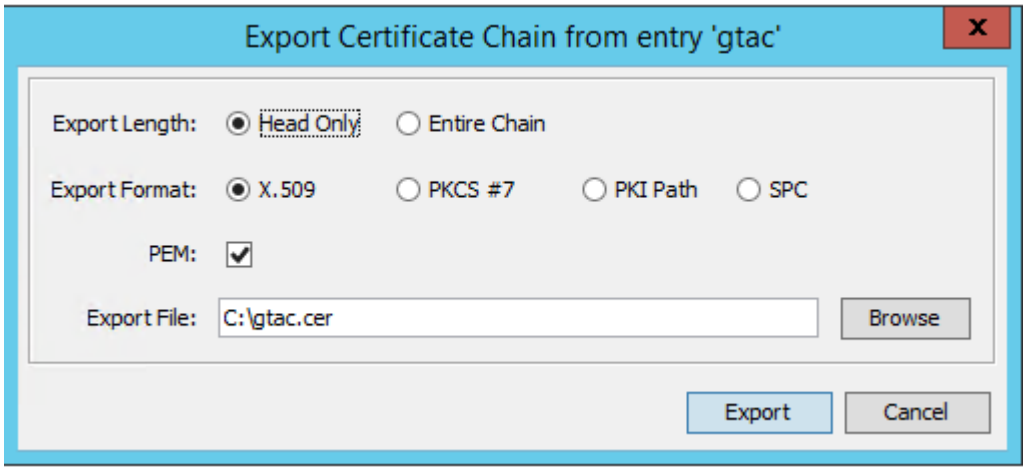


# Import Key Pair into Java Keystore (cacerts file)

Open the keystore that was created in the last step.
Select the key and from the right mouse button menu, select 'Export Certificate Chain':

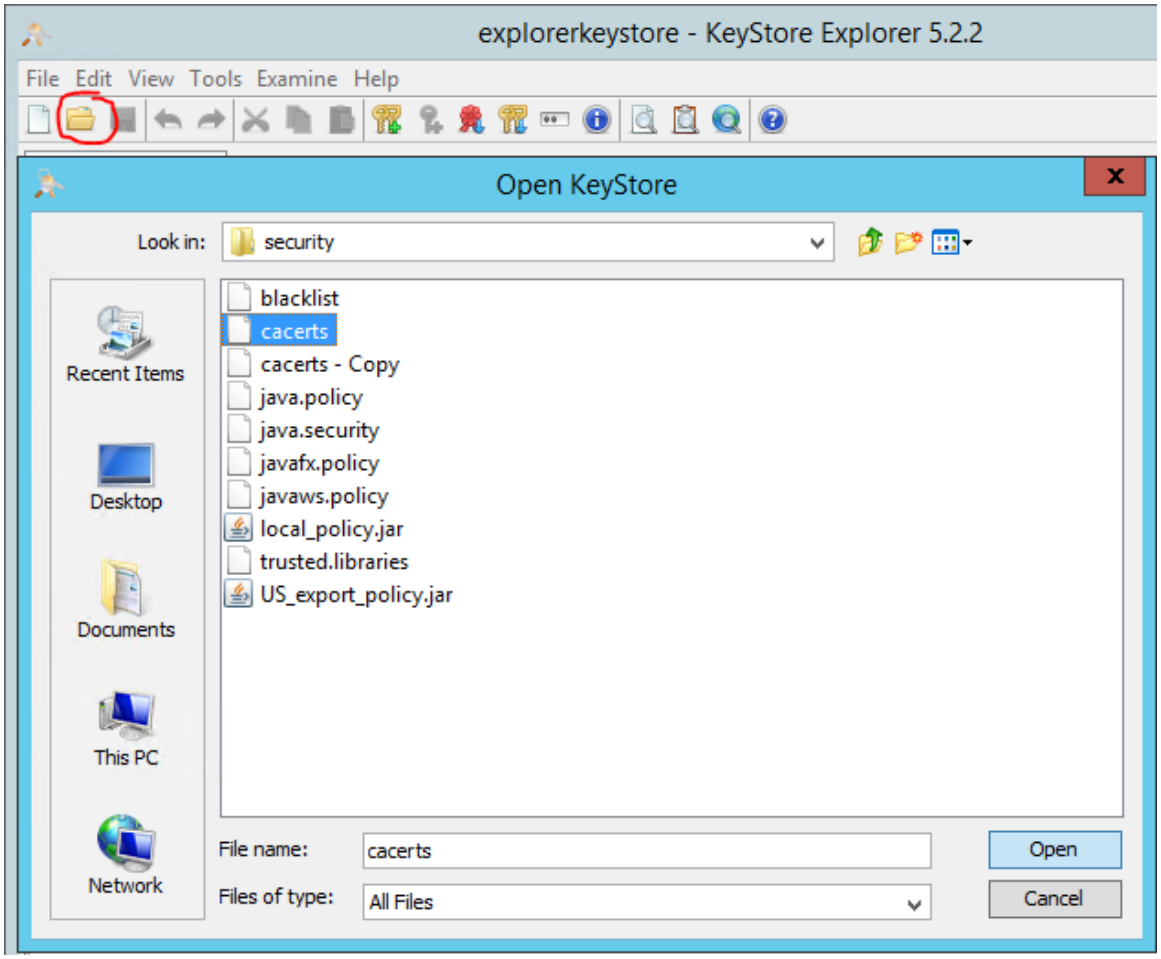



Take default options and enter a path and cert name:



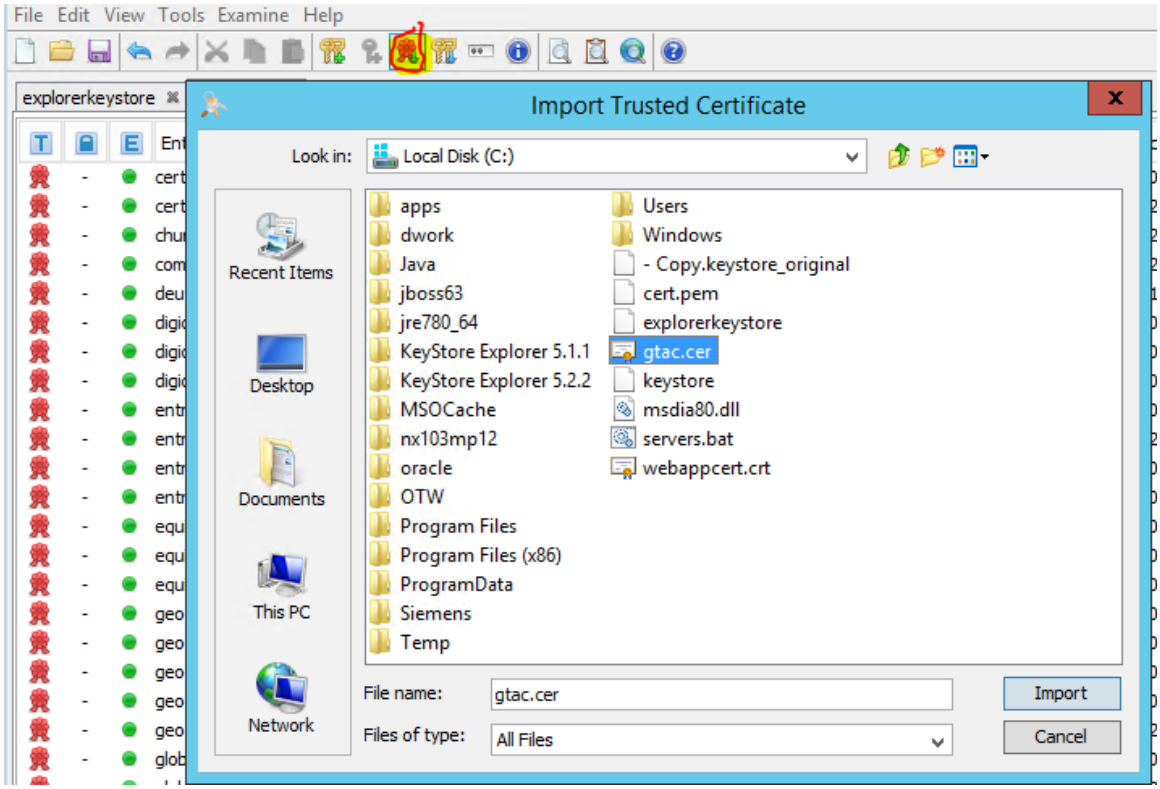

NOTE: The 'PEM' option of this export will not only be a format that can be imported into the Java cacerts file (next step) but will be the PEM file needed for some of the Teamcenter integration configurations to use.

Open the cacerts file from within the Java instance that the Teamcenter RAC client is using:

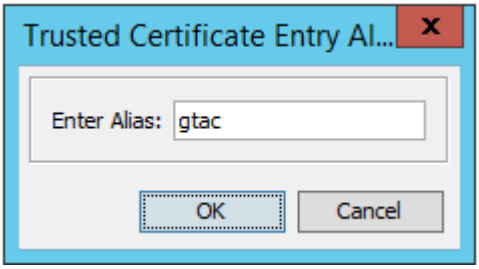



The default password for Java cacerts file is 'changeit'.

Select 'Import Trusted Certificate' option and browse to the PEM cert file created in the last step:




Enter the alias for that cert:




Confirm import:

Be sure to SAVE the cacerts file after the import is completed: