

# Configure HTTPS/SSL for Documentum REST Services

---

This document walks through the typical HTTPS/SSL configuration for Documentum REST Services.

## Environment

REST Server: Windows 2008 R2, Java 1.7.0\_55, Tomcat 7.0

REST Client: Mac OS X, Java 1.7.0\_60, Maven 3.2.1, [Documentum REST Sample Java Client](#)

## Prepare the certificate for Tomcat

Following [Apache Tomcat 7 \(7.0.57\) - SSL Configuration HOW-TO](#) to setup the SSL for the Tomcat server. For the development environment, you can use [Java Keytool](#) or [OpenSSL](#) to generate the certificate and key. [Portecle](#) provides a GUI tool to generate and export the certificate.

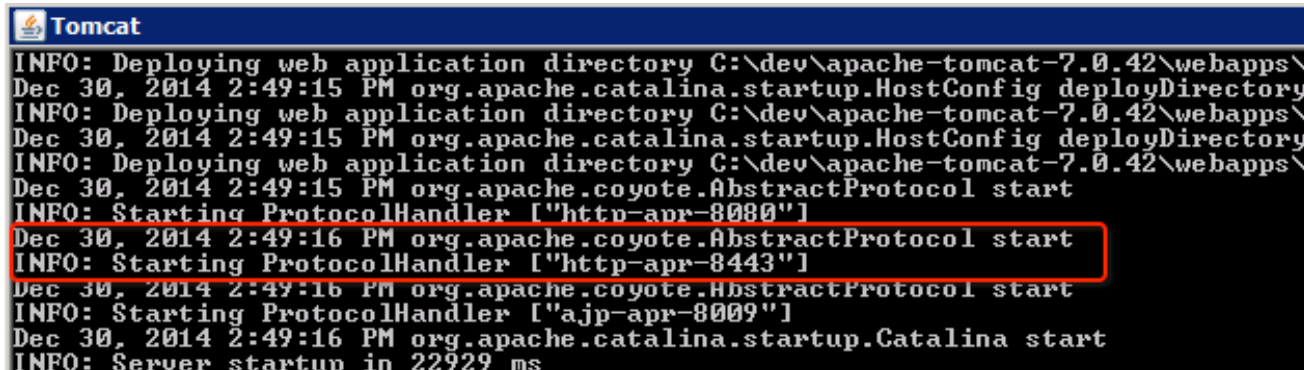
The alias of the certificate must be identical to the host name of the REST server. Assumed that we have generated the certificate and key files: ***williamappsvr.cer*** and ***williamappsvr.pem***, where the certificate alias (as well as the host name) is "***williamappsvr***".

Navigate to ***<Tomcat>/conf/server.xml*** and add a connector for port ***8443***.

```
<Connector port="8443" protocol="HTTP/1.1"    SSLEnabled="true"  maxThreads="150"    scheme="https" secure="true" sslProtoc
```

This is the SSL configuration for ARP style (default). If ARP is turned off, follow [Apache Tomcat 7 \(7.0.57\) - SSL Configuration HOW-TO](#) to configure the SSL connector.

Deploy the **dctm-rest.war** and start up the Tomcat server. The port 8443 should be setup successfully.



```
Tomcat
INFO: Deploying web application directory C:\dev\apache-tomcat-7.0.42\webapps\
Dec 30, 2014 2:49:15 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory C:\dev\apache-tomcat-7.0.42\webapps\
Dec 30, 2014 2:49:15 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory C:\dev\apache-tomcat-7.0.42\webapps\
Dec 30, 2014 2:49:15 PM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler [\"http-apr-8080\"]
Dec 30, 2014 2:49:16 PM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler [\"http-apr-8443\"]
Dec 30, 2014 2:49:16 PM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler [\"ajp-apr-8009\"]
Dec 30, 2014 2:49:16 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 22929 ms
```

## Prepare the client

If the REST server host name "**williamappsrv**" is not known by your client network, add the host IP map into your local host. Here is a sample for Mac OS X client.

Edit the host file (for Windows OS, it is commonly located at: C:\Windows\System32\drivers\etc\host).

```
sudo vi /etc/hosts
```

Add the host IP map to the end line.

```
192.168.1.5 williamappsrv
```

Next, you can test the REST server in the web browser. Input the address "<https://williamappsrv:8443/dctm-rest/services.json>" in the web browser. Proceed and ignore the certificate warning as this sample certificate is not authorized by CA. You should be able to get the home document resource.



Now the REST server for SSL has been working. We need to import the certificate to the client JRE to make the Java client succeed to access to the REST server.

Copy the certificate file ***williamappsrvr.cer*** from the REST server to the client machine. The other option is to export it from the web browser. We assume that the file is copied to ***~/Develop/williamappsrvr.cer***.

Run below command in the terminal to add the certificate to the JRE.

```
sudo keytool -import -trustcacerts -alias "williamappsrvr" -file ~/Develop/williamappsrvr.cer -keystore /Library/Java/JavaVirtualMachines
```

Here is the screenshot for this command.

```
c ~ $ sudo keytool -import -trustcacerts -alias "williamappsvr" -file ~/Develop/williamappsvr.cer -keystore /Library/Java/JavaVirtualMachines/jdk1.7.0_60.jdk/Contents/Home/jre/lib/security/cacerts -storepass "changeit"
Password:
Owner: CN=WILLIAMAPPSVR, OU=IIG, O=EMC, L=EN, ST=CA, C=US, EMAILADDRESS=admin@sample.com
Issuer: CN=WILLIAMAPPSVR, OU=IIG, O=EMC, L=EN, ST=CA, C=US, EMAILADDRESS=admin@sample.com
Serial number: 54a24439
Valid from: Tue Dec 30 14:20:41 CST 2014 until: Sun Sep 24 14:20:41 CST 2017
Certificate fingerprints:
    MD5: E1:6B:51:EB:F7:1E:54:9E:70:4E:AD:D1:A2:CB:92:87
    SHA1: DF:86:8E:95:A4:B5:4F:58:8C:C9:B8:5B:71:92:30:7F:C0:6E:60:52
    SHA256: 9A:CF:B8:82:AE:BE:A7:75:D7:A3:4C:AB:28:15:78:24:F1:8C:E1:B3:12:7E:D1:A2:B6:A8:AD:76:9D:0C:64:8A
    Signature algorithm name: SHA1withRSA
    Version: 1
Trust this certificate? [no]: y
Certificate was added to keystore
c ~ $
```

With above steps, the development environment for the client side has been setup successfully.

## Run the REST service

Now we can test the REST services from the Java client. For the generic Apache HTTP client, follow [HttpClient - HttpClient SSL Guide](#) to setup the SSL connection. For the Spring REST Template, the SSL is configured by default. So we will use the [Documentum REST Sample Java Client](#) to test our SSL connection.

Firstly, download the sample Java client from EDN: [Documentum REST Sample Java Client](#).

Then, navigate the project home directory, and run maven command:

```
mvn clean install
```

After the project is built, navigate to the **target** folder, and run below command:

```
java -jar rest-api-client-sample-7.1.jar
```

Then input the test parameters as promoted:

```

:target $ java -jar rest-api-client-sample-7.1.jar
Please input the client representation type: XML JSON [default XML]

Please input the REST context path: [default http://localhost:8080/dctm-rest]
https://williamappsvr:8443/dctm-rest
Please input the repository name:
REPO
Please input the username:
dmdadmin
Please input the password:
password
Please input the whether add format extension .xml or .json for URI: [default false]

Please input whether print debug information: [default false]
true
[debug] Build DCTMRestClient with JAXB implementation
[debug] Context Root=https://williamappsvr:8443/dctm-rest
[debug] Repository=REPO
[debug] User Name=dmdadmin
[debug] Password=password
[debug] Use Format Extension=false

```

Last, choose the test suites to run. In below screenshot, the navigation is tested.

```
Please input the number of the sample operation which need be executed:
0 Exit
1 Navigation
2 Folder Create/Update/Delete
3 Sysobject Create/Update/Delete
4 Document Create/Update/Delete
5 Content Management
6 Version Management
7 DQL Query
1
start Navigation sample
-----get home document resource
[debug] Resource URI: https://williamappsvr:8443/dctm-rest/services
[debug] HTTP method: GET
[debug] Request headers: {Accept=[application/home+xml]}
[debug] Expected response body class: JaxbHomeDocument
[debug] Request parameters:
[debug] Sending GET request to https://williamappsvr:8443/dctm-rest/services
[debug] Response status: 200
[debug] Response headers: {Server=[Apache-Coyote/1.1], Content-Type=[applicati
4 07:13:29 GMT]}
http://identifiers.emc.com/linkrel/repositories -> https://williamappsvr:8443/dc
about -> https://williamappsvr:8443/dctm-rest/product-info

-----get repositories collection
[debug] Resource URI: https://williamappsvr:8443/dctm-rest/repositories
[debug] HTTP method: GET
[debug] Request headers: {Accept=[application/atom+xml]}
[debug] Expected response body class: JaxbFeed
[debug] Request parameters:
[debug] Sending GET request to https://williamappsvr:8443/dctm-rest/repositori
[debug] Response status: 200
[debug] Response headers: {Server=[Apache-Coyote/1.1], Content-Type=[applicati
4 07:13:29 GMT]}
There are 1 repositories in total.
REPO -> https://williamappsvr:8443/dctm-rest/repositories/REPO
```

[Learn more about Documentum REST Services >>](#)