Generate CSR for Tomcat How to Guide

Contents

Overview

This article details the steps required to generate a certificate signing request from a software only PINsafe installation within Apache Tomcat. It is not suitable for use with the PINsafe Appliance as the appliance has built in functionality to automate this process.

If you're looking to generate a CSR and you have a PINsafe appliance, please see SSL Certificate for Appliance How to Guide

Prerequisites

- PINsafe software installation
- Apache Tomcat

Solution

Locate your keystore

You will need to generate a CSR from Tomcat.

The first thing you need to do is to establish where your Tomcat certificate store (keystore) actually is. To do this, look at the Tomcat configuration file under <TOMCAT_ROOT>\conf\server.xml. Look for a line starting with '<Connector port="8080"'. Somewhere in that line you should see an attribute named "keystoreFile", which gives the full path to the keystore. Also make a note of the keystorePass attribute as well: this is the password for the keystore, which will be required to access it.

NOTE: if you are using this article to generate a CSR for an appliance, because your appliance does not have the relevant CMI menu options, the keystore file will be:

```
/home/swivel/.keystore
```

List the keystore

Now you have located the keystore, make sure the file exists. To list the contents of the keystore, open a command prompt and type the following command:

```
keytool -list -v -keystore <keystoreFile>
```

where <keystoreFile> is the name of the keystore file (it's probably simplest to change to the correct directory first). Enter the keystore password when requested. The -v option requests a verbose listing, so there will be a lot of information: it's probably best either to list it to a file, or use the "| more" option. Hopefully somewhere among the listed certificates you will find the one you need to update. It will have a type of "privateKeyEntry" (simply "keyEntry" on older versions of PINsafe), with the correct subject and expiry date. Make a note of the corresponding certificate alias.

Generate a new Self-Signed Certificate

If you do not see a certificate of the correct type, you will have to create one. To do so, use the following command:

```
keytool -genkey -keystore <keystoreFile> -alias <certificateAlias>
```

You will be prompted for the certificate details. **IMPORTANT:** the first thing you will be asked for is your first and last name. What it is *actually* asking for is the subject of the certificate; in other words, the fully-qualified host name that will be used to reference this server. Make sure you get this correct. Another important point is that when you are asked for a password for the certificate, you **MUST** press <Enter> to use the same password as the keystore. Tomcat does not have the ability to use different passwords for the keystore and the certificate.

If your certificate authority requires 2048-bit certificates, which many now do, use the following command:

```
keytool -genkey -keystore <keystoreFile> -alias <certificateAlias> -keyalg RSA -keysize 2048
```

Request a new certificate

To request a new certificate, use the following command:

```
keytool -certreq -keystore <keystoreFile> -alias <certificateAlias> -file <CSRFileName>
```

Replace the names in brackets with the appropriate values (<CSRFileName> is the name of the output file for the CSR request).

You can now send off the generated CSR to your certificate authority. Make sure that you request a Tomcat or Java-compatible format.

Import the response

When you get the response back, use the following command to import the response. I recommend that you make a copy of the keystore first. If there are any intermediate certificates provided, and they are not already in the keystore, you need to import them into the store before you import the response file. You can use the same command to import the intermediate certificates as for the CSR response. The important thing to remember is when importing an intermediate certificate, use an alias that does not already exist in the file (you can use any name, but it must be unique). When importing the CSR response, you MUST use the same alias that you used to generate the CSR.

```
 keytool -importcert -keystore <keystoreFile> -alias <certificateAlias> -file <responseFile> -trustcacerts
```

If you get a warning that you need to delete the existing alias first, then there is a problem with the certificate, and it is not a proper response to the existing certificate. Otherwise, it should indicate a successful import. Run the keytool -list command again to check that all is correct: in particular, that the type for the CSR response is privateKeyEntry, not trustedCertEntry. The intermediate certificates should have a type of trustedCertEntry.

Restart Tomcat

Tomcat only reads the keystore file on startup, so you can carry out operations on the file without affecting the running application. Changes to the certificate store do not take effect until Tomcat is restarted, so restart Tomcat to apply the changes.

Testing on non-production server

Note that, if you want to test the procedure, there is no problem with copying the keystore to another machine, although obviously, you need to test with the correct host name, by editing your local hosts file, for example.

Deleting an old or self-signed certificate

When Tomcat receives an HTTPS request, it will look for and use the first PrivateKeyEntry certificate it finds, irrespective of whether the host name matches the request host name, or whether the certificate has expired. Therefore, if you have an old certificate, or the self-signed certificate provided with a PINsafe appliance, still in the keystore, Tomcat may will use that in preference to the correct one.

Therefore, you need to make sure that you delete any alias that has a type of PrivateKeyEntry except for the one you want to use. To delete a certificate, use the following command:

```
 keytool -delete -keystore <keystoreFile> -alias <certificateAlias>
```

It is recommended that you make a backup copy of the keystore before doing this, and remember that you need to restart Tomcat in order to register any changes you make to the keystore.

Troubleshooting

If you get a warning that you need to delete the existing alias first, then there is a problem with the certificate and it is not a proper response to the existing certificate. Otherwise, it should indicate a successful import.

- Run the keytool -list command again to check that all is correct: in particular, that the type for the CSR response is privateKeyEntry, not trustedCertEntry.
- The intermediate certificates should have a type of trustedCertEntry.

Use a unique alias name, "swivel" already used

seen in CMI Certs.: 0.8

This can be caused by a bug in the CMI, to continue the process from the command line:

keytool -importcert -keystore /home/swivel/.keystore -alias swivel -file <certresponse>

Where <certresponse> is the name of the certificate response file that you were attempting to import via the menu.