Authorize a self signed certificate by a trusted 3rd party Certificate Authority

Creating a certificate signed by a trusted third party

To create a certificate signed by a trusted third party, first follow the steps under #Creating a self signed certificate and then continue with the steps below:

Create Certificate Signing Request
After creation of the Keystore, a "Certificate Signing Request" (CSR) needs to be created, using the following command:

```
keytool -certreq -alias MyPlugins -keyalg RSA -file mykeystore.csr -keystore mykeystore.ks
```

> ⚠ **Use the same values**
>
> The argument values used for the options of the creation of the CSR must be the same ones as used for the creation of the self signed certificate.

> ⛔ Make a backup copy of the keystore and CSR files. If lost lost or accidentally destroyed, the signed certificate created by the next step will be useless and money paid for it is lost.

Authenticate the CSR
Next step is to authenticate the CSR. This can be done with any third-party. This example uses http://www.instantssl.com

1. Browse to http://www.instantssl.com
2. Click on "Code Signing Certificate"
    a. Select your duration (1, 2, or 3 years).

    > ⚠ Make sure the validity is shorter than the validity you chose when creating the keystore, otherwise you are paying for more that you will get!!!

3. Copy the contents of the mykeystore.csr file into the form where it says to copy the CSR to.
4. Select OTHER for the software used to generate it.
5. Deselect all the checks of step 4 on the web page (newsletters and other stuff).
6. Fill in the rest and follow the instructions (corporate details, etc.).

After completing all the steps, InstantSSL will email all the required info, including a signed .crt file to the specified email address.

Import the signed certificate
The signed certificate (+ any required intermediate certificate) received from the SSL supplier need to be imported into the keystore.

The signed .crt file can be imported into the keystore using the following command:

```
keytool -import -alias MyPlugins -keystore mykeystore.ks -trustcacerts -file mykeystore.crt
```

> ⚠ **Use the same values**
>
> The argument values used for the options of the import of the .crt file must be the same ones as used for the creation of the self signed certificate/certificate signing request.

When the command is run, it will ask for the password in the console:

1. Provide the password (the same password as set in the step "Create keystore") and trust the certificate.

After this command, a message error can be thrown sometimes by Keytool, with a message that says that a chain of trust could not be established for the given certificate reply. This error appears when the reply is in form of a chain of certificates (PKCS#7 format). The keytool needs to "read" this chain from the first parent of the certificate reply up to the CA root (which will be trusted or not, depending on the presence of this CA root into the cacerts file or the presence of the "-trustcacerts" argument). A fix that always works:

1. Note all the "parents" of the response certificate in order (from the first parent to the CA root) - you can see them in windows by opening the .crt file (e.g. servoy.crt) and see the certification path (a "tree" with all the "parents").
2. Open the certificate in an editor - let's call it target certificate.
3. For each parent in the parent's list (taken from the "first" parent up to the CA root)
    a. add its content into the target certificate's file at the *beginning* of the file; the content is from "~~-BEGIN CERTIFICATE-~~" to "~~-END CERTIFICATE-~~" (new lines are allowed between certificates' content)
4. run the command again: keytool -import -alias MyPlugins -keystore mykeystore.ks -trustcacerts -file mykeystore.crt

Each certificate is viewable as text encoded Base64.

This is needed in order to make the certificate file conform to the PKCS#7 format; this can be done also in other ways, by adding the CA root into the keystore with the keytool command, but it will not work all the times, and if it works, the result is the same as using the above described solution.

Security

The Keystore contains very sensitive information. If this information is compromised people can do all kinds of nasty stuff such as listen in on and modify the supposedly encrypted information. When this happens and is detected (which is a best case scenario) the certificate has to be revoked and a new one needs to be purchased and installed. If the Keystore is lost or accidentally destroyed, a new certificate will have to be purchased. Make proper backups at all times and make sure only authorized personnel has access to these backups.