Certificate Properties File Realm

{scrollbar}

This realm type allows you to configure Web applications to authenticate users against it. To get to that point, you will need to first configure Geronimo to use a custom SSL port listener and to get to that point you will need to configure SSL keys and keystore. The following sections describe step-by-step how to configure each of these modules.

- #Create keystore and certificate
- #Create a Certificate Signing Request (CSR) and import CA reply
- #Import trusted certificates
- #Add an HTTPS listener with client authentication

Create keystore and certificate

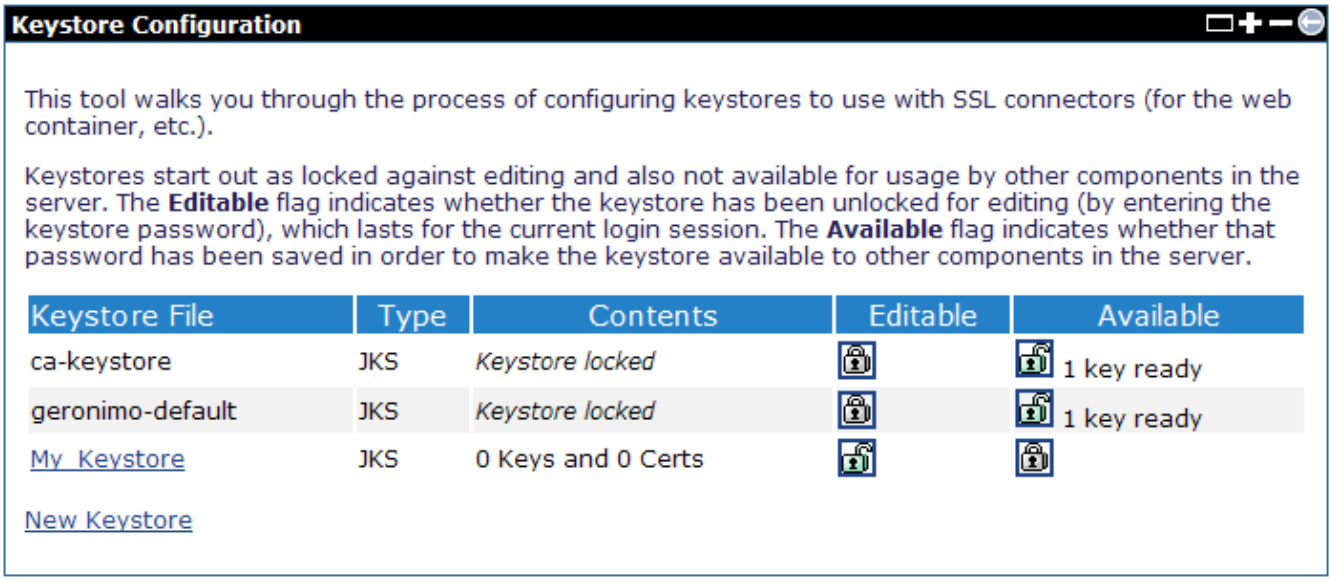
For this configuration we will create a new keystore, a new private key, a CSR and will import the CA reply

We already mentioned in the Administering Certificates section how to create a keystore and a private key, in this section we will complete the picture by generating a CSR and importing the CA's reply.

The keystores in Geronimo are stored in the **<geronimo_home>\var\security\keystores** directory, the default keystore already provided with the installation is **geronimo-default**. For this exercise we will create a new keystore.

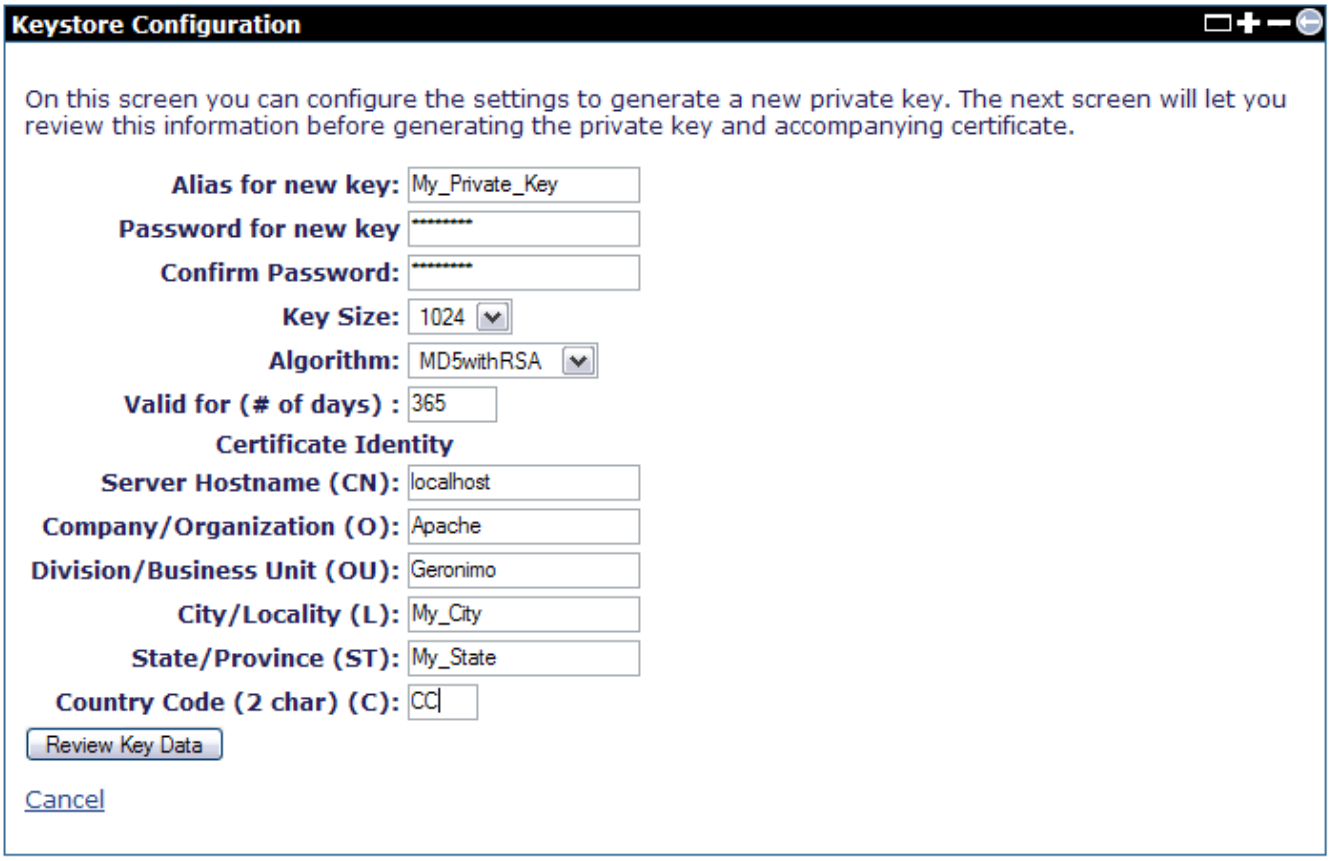
From the Geronimo Administration Console click on **Keystores** to access the **Keystore Configuration** portlet.

Click on **New Keystore**, specify a new keystore name and password and then click on **Create Keystore**. For this example we used `My_Keystore` and `password` respectively.

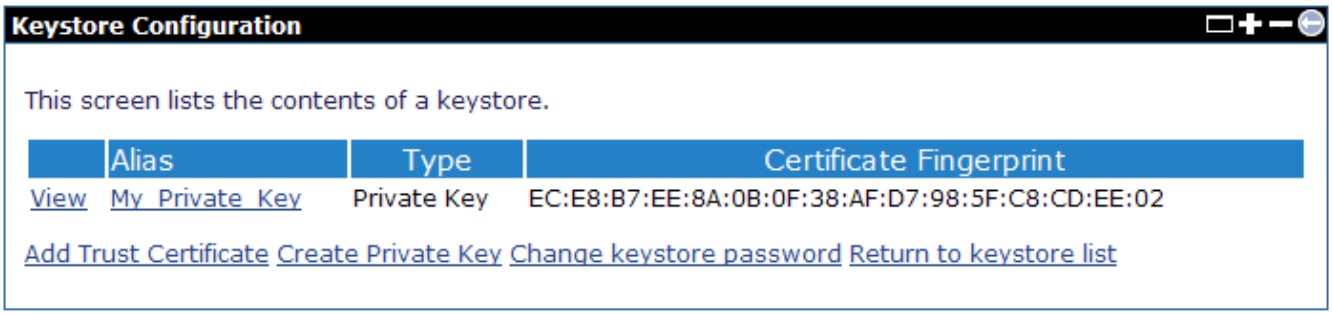


Click on the keystore file you just created, and create a private key by clicking on the appropriate link.

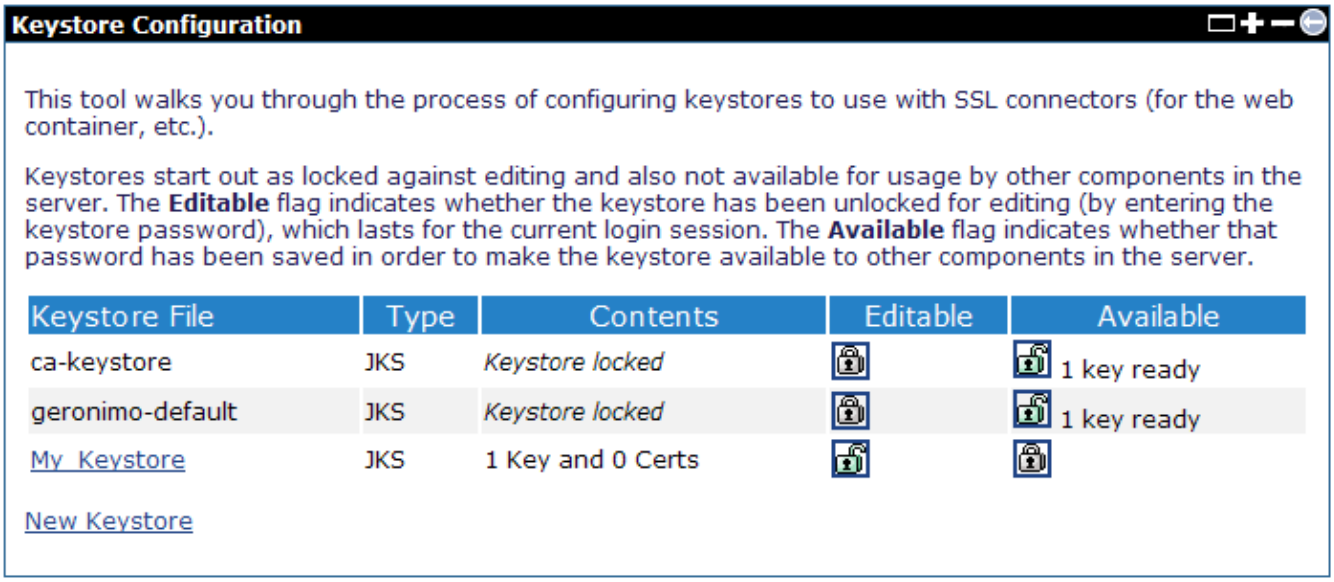
Fill in with the appropriate data and click on **Review Key Data**.



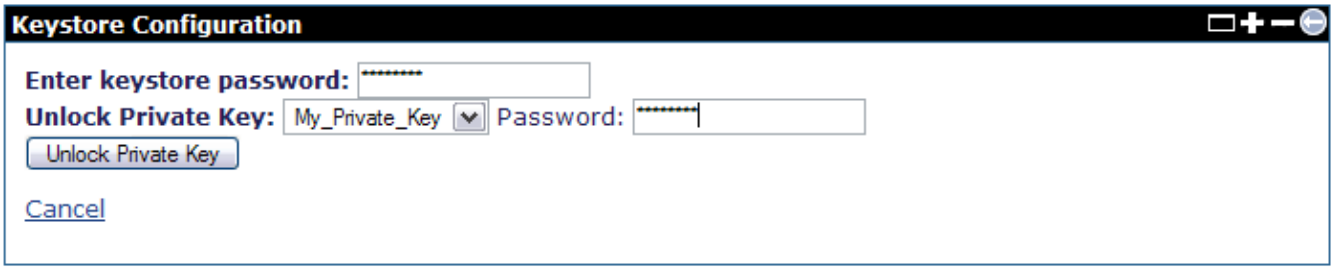
Once you verified the values are correct click on **Generate Key**.



Right after you created a new private key, this key is automatically locked. That means that you can only view it or delete it, to create a Certificate Signing Request (CSR) you will have to unlock the key. To do that click on **Return to keystore list**.



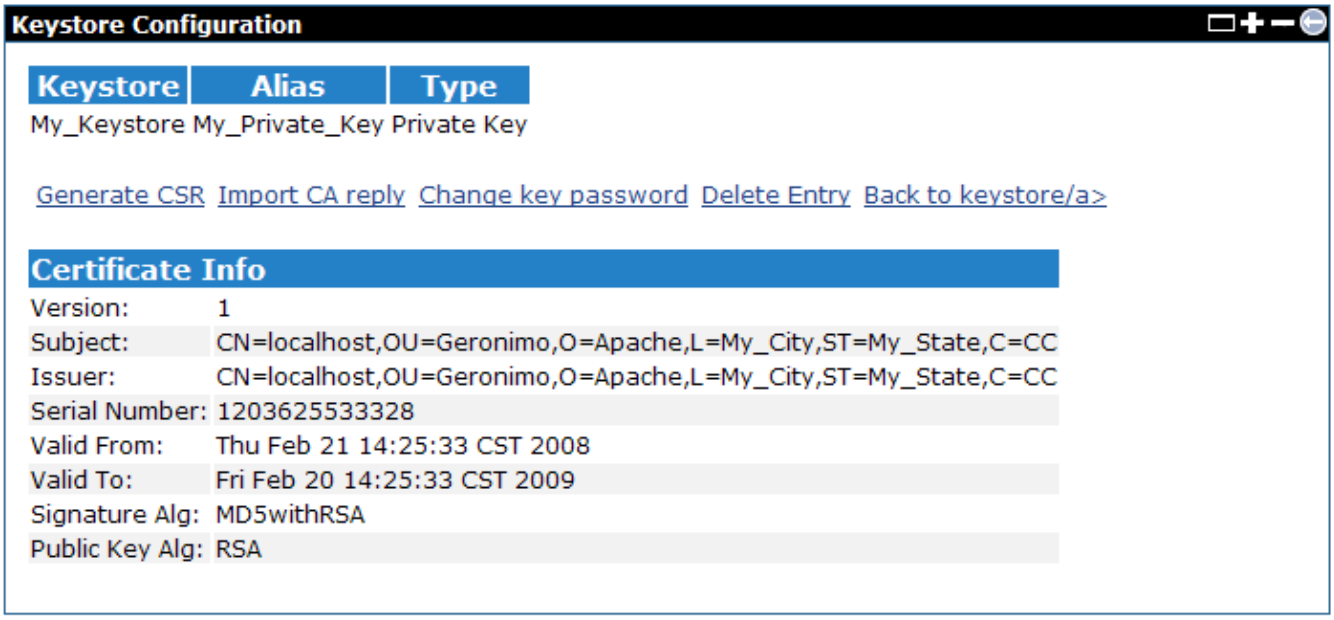
Click on the  to unlock the private key. You will be prompted with the password for the keystore and for the private key.



Click on **Unlock Keystore**.

Create a Certificate Signing Request (CSR) and import CA reply

Now that you have the private key unlocked you may now continue to create a CSR. From the **Keystore Configuration** portlet click on the keystore file you created to display the current content. In this example we only have one private key. Click on either **view** or the alias links for the current private key to display the details and additional actions.



Click on **Generate CSR**, the certificate request should be displayed as illustrated in the following figure.

This is a **PKCS10** certification request, you should copy this text and paste it into a flat txt file so it can be sent to a CA.

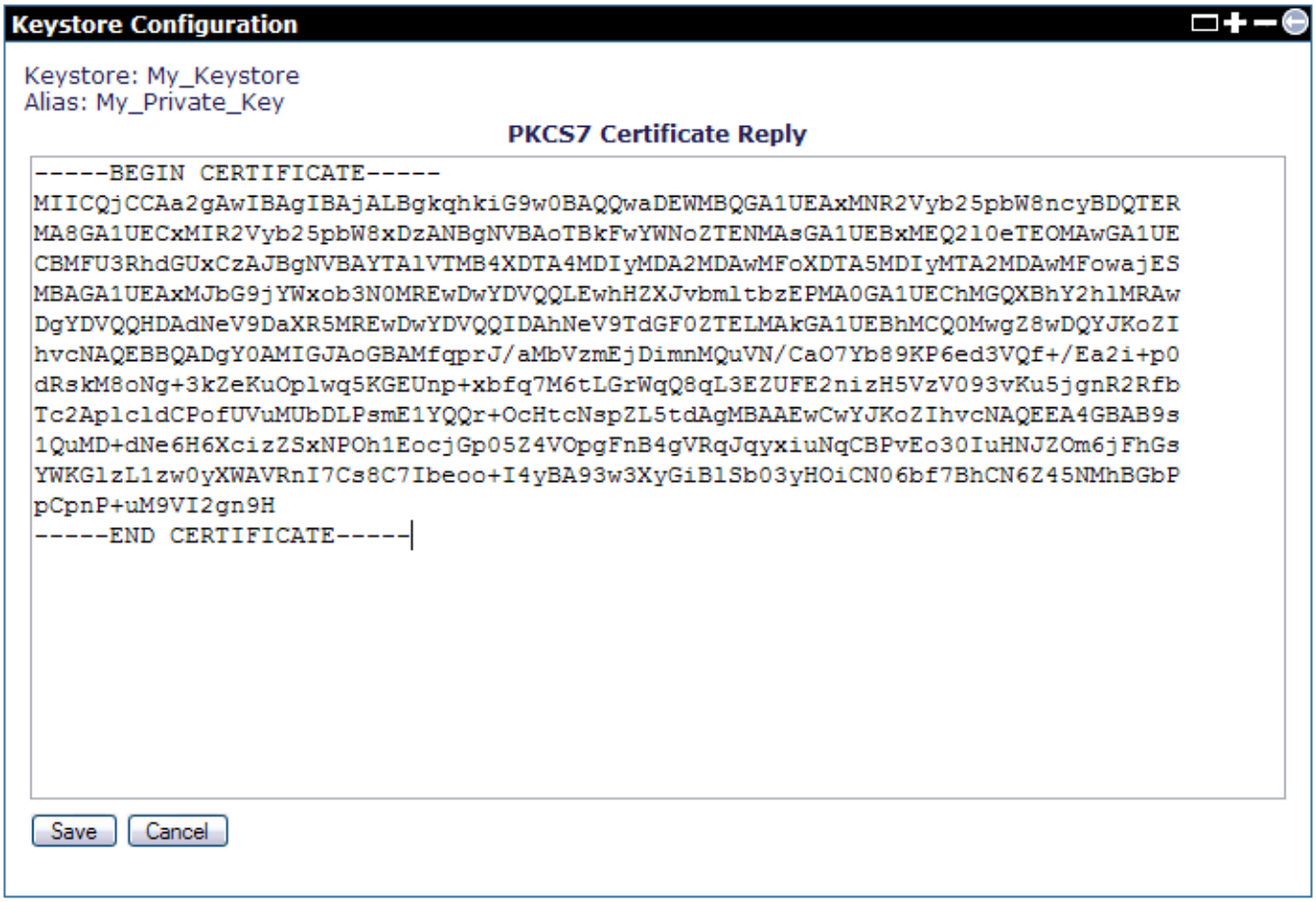
solidcsr.txt -----BEGIN CERTIFICATE REQUEST----- MIIBqDCCARECAQAwajESMBAGA1UEAxMJbG9jYWxob3N0MREwDwYDVQQLEwhHZXJvbmltbz
EPMA0GA1UEChMGQXBhY2hlMRAwDgYDVQQHDAdNeV9DaXR5MREwDwYDVQQIDAhNeV9TdGF0
ZTELMAkGA1UEBhMCQ0MwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMfqprJ/aMbVzm EjDimnMQuVN/CaO7Yb89KP6ed3VQf+
/Ea2i+p0dRskM8oNg+3kZeKuOplwq5KGEUnp+xbf q7M6tLGrWqQ8qL3EZUFE2nizH5VzV093vKu5jgnR2RfbTc2AplcldCPofUVuMUbDLPsmE1
YQQr+OcHtcNspZL5tdAgMBAAEwDQYJKoZIhvcNAQEEBQADgYEAZFuPz0gzKqMZNA0bYLm0
aPFLbR9a19NA0EbgJL2SYzoKnuKyplG2JzMVQ6myaez0J8t+iWtuthz70kBihRzU2vqOWp B4oqh+zbPwn4f87l4l8PjJh3SkiDIYdMcL5U1rxwFNAaIEpfjft
/uJLY/Bv7DZQG7UPsGz +SPdn+DbdBo= -----END CERTIFICATE REQUEST-----

You can now click **Back** to return to the private key details portlet.

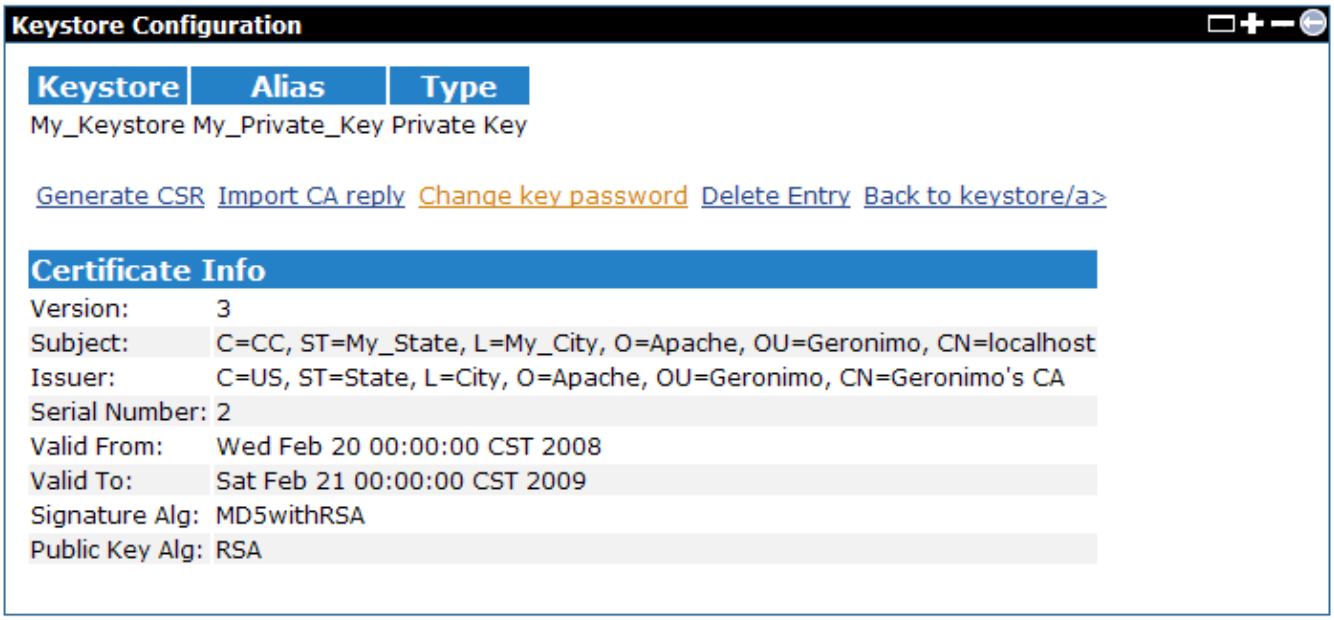
For this example we used a custom, home made CA so we could sign our own certificates for this test without altering the standard procedure. Assuming that you sent you CSR to a CA, the CA should respond back with another similar file containing the CA signed certificate.

solidcsr_ca_reply.txt -----BEGIN CERTIFICATE-----
MIICQjCCAa2gAwIBAgIBAjALBgkqhkiG9w0BAQQwaDEWMBQGA1UEAxMNR2Vyb25pbW8ncyBDQTER
MA8GA1UECxMIR2Vyb25pbW8xDzANBgNVBAoTBkFwYWNoZTENMAsGA1UEBxMEQ2l0eTEOMAwGA1UE
CBMFU3RhdGUxCzAJBgNVBAYTAlVTMB4XDTA4MDIyMDA2MDAwMFoXDTA5MDIyMTA2MDAwMFowajES
MBAGA1UEAxMJbG9jYWxob3N0MREwDwYDVQQLEwhHZXJvbmltbzEPMA0GA1UEChMGQXBhY2hlMRAw
DgYDVQQHDAdNeV9DaXR5MREwDwYDVQQIDAhNeV9TdGF0ZTELMAkGA1UEBhMCQ0MwgZ8wDQYJKoZI
hvcNAQEBBQADgY0AMIGJAoGBAMfqprJ/aMbVzmEjDimnMQuVN/CaO7Yb89KP6ed3VQf+/Ea2i+p0
dRskM8oNg+3kZeKuOplwq5KGEUnp+xbfq7M6tLGrWqQ8qL3EZUFE2nizH5VzV093vKu5jgnR2Rfb
Tc2AplcldCPofUVuMUbDLPsmE1YQQr+OcHtcNspZL5tdAgMBAAEwCwYJKoZIhvcNAQEEA4GBAB9s
1QuMD+dNe6H6XcizZSxNPOh1EocjGp05Z4VOpgFnB4gVRqJqyxiuNqCBPvEo30IuHNJZOm6jFhGs
YWKGlzL1zw0yXWAVRnI7Cs8C7Ibeoo+I4yBA93w3XyGiBlSb03yHOiCN06bf7BhCN6Z45NMhBGbP
pCpnP+uM9VI2gn9H -----END CERTIFICATE-----

From the private key details portlet click on **Import CA reply**. Remove any pre-filled text in the certificate reply window and paste the text from the CA reply file and click on **Save**.



After saving the CA reply you should now notice that the certificate now shows a different **Issuer**. Click on **Back to keystore** and then on **Return to keystore list**.



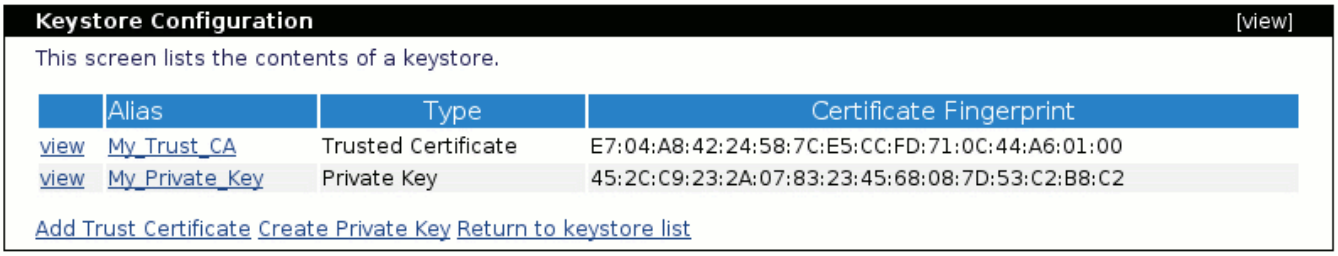
Import trusted certificates

In order to enable client authentication you will need to import the CA who signed your CSR as a trusted certificate, this process has to be only once. The CA should provide along with the signed CSR a separate certificate for the CA itself. For this example we are using our own CA so we generated the following CA certificate.

solidMy_Own_CA_Certificate.txt -----BEGIN CERTIFICATE-----
MIICJTCCAZCgAwIBAgICK2cwCwYJKoZIhvcNAQEEMFoxDTALBgNVBAMTBFRlc3QxDzANBgNVBAsT
BkFwYWNoZTERMA8GA1UEChMIR2Vyb25pbW8xCzAJBgNVBAcTAkxMMQswCQYDVQQIEwJTVDELMAkG
A1UEBhMCTEswHhcNMDcwMjAyMTgwMDAwWhcNMDgwMjAyMTgwMDAwWjBaMQ0wCwYDVQQDEwRUZXN0
MQ8wDQYDVQQLEwZBcGFjaGUxETAPBgNVBAoTCEdlcm9uaW1vMQswCQYDVQQHEwJMTDELMAkGA1UE
CBMCU1QxCzAJBgNVBAYTAkxLMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCUZl1e1eTKLoh0
15vfYqqvhk6Iviva7BWQxZ6mOV9Ye2mii37Btmxajnngz0jKfiwHKqWRQBp6CUzbd9gfZrz2go9g
TwsUBWQwSf6iVypKX1q0Y4WhtTwLcEx78Lx5XN1YCqk34pn4by26SJiHdugs7/ClOiIlcpCt9QVa Q9BH7wIDAQABMAsGCSqGSIb3DQEBBAOBgQAnmoT
/dLvJa7jGstvZJLrsWtMwWQNVJ1ZQmbrDGq9u oFnkAH1mGHIDbaz2avy/wotHJUIysGBlDP0btk5GVskl45EG/feWHLgCVmqwf3NkdRdLl+CznBBJ
KCC5tINbcI6GqXsbO8hhjIrOGweNyV1653WEvZiQVuMYaHTnGNx+RA== -----END CERTIFICATE-----

While in the Keystore Configuration portlet click on the keystore file you created and then click on **Add Trust Certificate**. Delete any pre-filled content from **Trusted Certificate** window and paste the content from the CA certificate and add an alias to this certificate.

Click on **Review Certificate** and then click on **Import Certificate**. You should now see the trusted certificate you just imported.

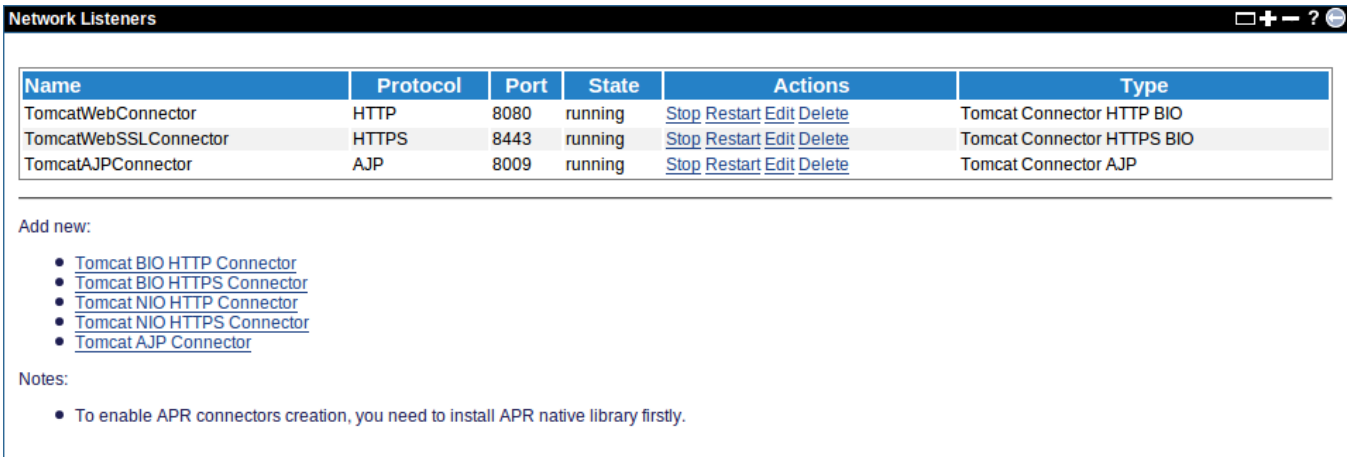


Add an HTTPS listener with client authentication

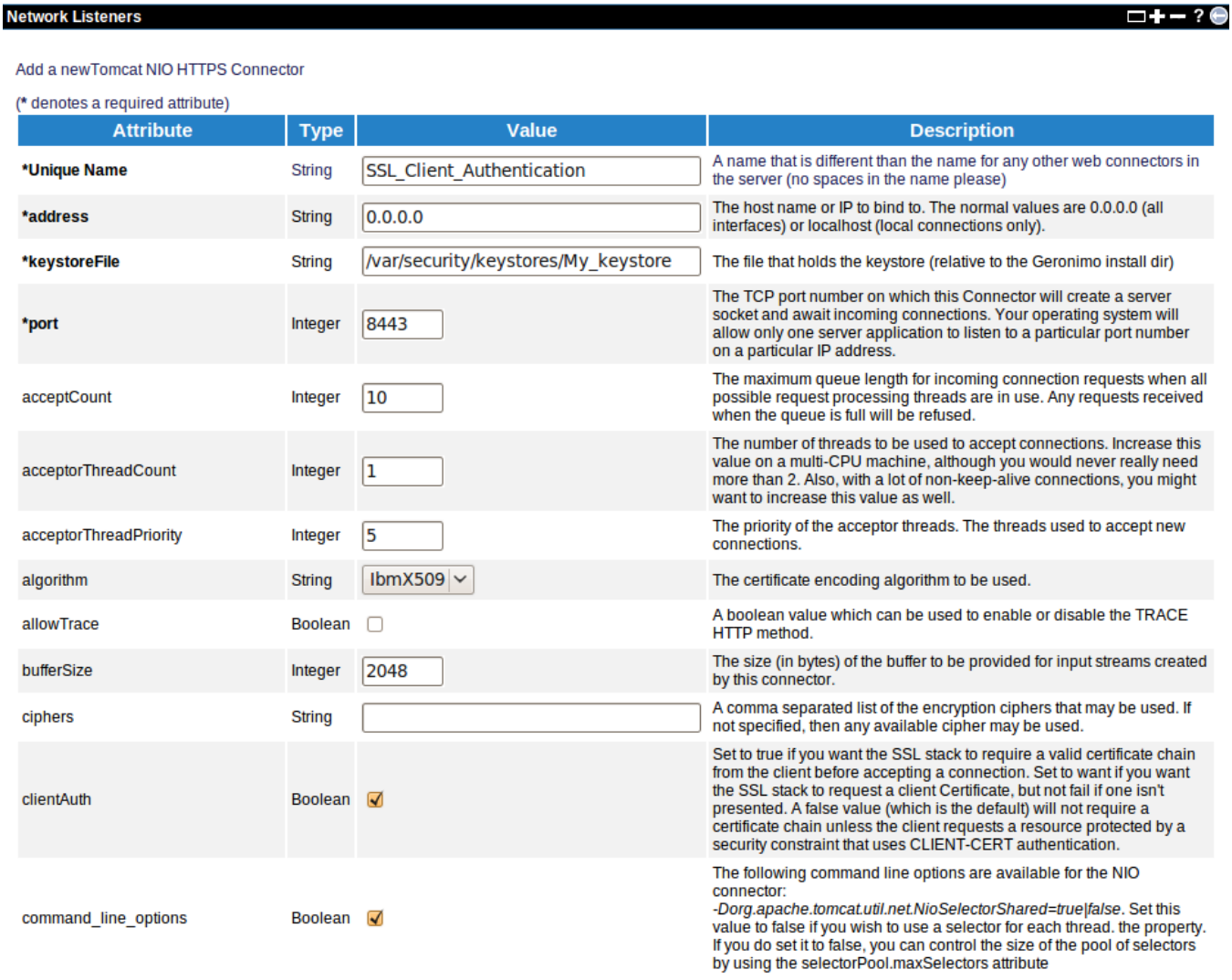
Apache Geronimo comes with a predefined HTTPS listener on port 8443 but this listener is not configured for client authentication. In this example we will add a new HTTPS listener and configure it to request client authentication using the certificates we created and imported in the previous steps.

Note that in this example we are using the Tomcat distribution of Geronimo, although the process is the same some names and links may vary slightly if you are using the Jetty distribution.

From the Geronimo Administration Console click on **Web Server** to access the Network Listener portlet.



From the Network Listener portlet click on **Add new HTTPS listener for Tomcat**



Fill in the fields with the appropriate data and click **Save**. For this example we only specified the keystore and not a trustore. When specifying the keystore file path you should add something similar to **var/security/keystores/<your_keystore>**, this path is relative to Geronimo's installation home directory.

Select the **ClientAuth** check box, this tells the HTTPS listener to only establish an encrypted connection with a client that provides a valid client certificate. The client certificates are verified against the CA certificates stored in any of these locations (in order):

1. The trust store configured above
2. A keystore file specified by the javax.net.ssl.trustStore system property
3. java-home/lib/security/jssecacerts

4. java-home/lib/security/cacerts

Once you saved this HTTPS network listener configuration it will get started automatically as you can see in the status displayed. If you try to access this port with your browser it should fail because at this point you have not configured your client with a valid certificate.