

Krestfield PKCloud

PKI Channels

version 2.0
Copyright Krestfield 2020

Configuration

Creating the Channel

Select the **Channels** menu item and click **New PKI Channel**

In the form enter the following details

- **Enabled**
 - Check this to enable the channel. If unchecked the channel will be disabled and any calls referencing this channel will fail
- **Channel Name**
 - Enter a name (do not include spaces) for the channel. This name is case sensitive and must be passed by client applications in the API calls to reference this specific channel
- **Key Store**
 - From the drop down, select the Key Store to use for this channel. Refer to the Key Stores documentation for details on Key Store setup. A Key Store must have already been configured before the channel configuration can be completed. If this has not been done, go back and setup the Key Store first

Signature Key Options

These options determine the algorithm that will be used to generate keys in the channel. When a CSR is generated a key of the type specified here will be generated

The available options are:

- **Signature Algorithm**
 - From the drop down choose either **RSA** or **ECDSA**
 - If **RSA** is chosen the **Key Size** option is made available, Choose the key size required
 - If **ECDSA** is chosen the **Curve** option is made available. Choose the curve required from the list of SECG curves (including NIST approved). If another curve is required, select **Other...** and type the name in the **Curve Name** box. This can be any other curve

name such as brainpoolp256r1 but you must ensure that the curve is supported by the key store/HSM

Signature Generation Options

These options determine how signatures will be created

The available options are:

- **Signature Type**

- From the drop down choose either **PKCS#7** or **Raw/PKCS#1**

The **PKCS#7** option will generate signatures following the PKCS#7 and CMS (Cryptographic Message Syntax) standards as described in RFC 2315 and RFC 3852. Note: Regarding the signatures generated in this instance there are no changes between these RFCs

The **Raw/PKCS#1** option will generate signatures in line with RFC 2437 using the RSAES-PKCS1-v1_5 scheme (for RSA) and in line with RFC5480 for ECDSA

- If the **PKCS#7** option is chosen the following options are also available:
 - **Include Content with Signature**
 - If checked, the original content sent for signing (whether previously hashed or not) will be included within the PKCS#7 structure
 - **Include Signed Attributes**
 - If checked the signed attributes – Signing Time, Message Digest and Content Type will be included in the PKCS#7 structure
 - **Certificates to Include**
 - From the drop down choose whether to include only the signer certificate (**Signer Only**), all certificates except the root (**All Except the Root**) or all certificates in the path (**All**) within the PKCS#7 structure
- **Hash Algorithm**
 - Choose the algorithm that will be used to hash the provided data (if not already hashed). This option is available for both types of signature

Signature Verification Options

These options determine what additional checks will be performed when signatures are verified

The available options are:

- **Additional Checks**

Checking any of the following will result in the check being performed on the signature being verified

- **The Signature Must Contain 'Signed Attributes'**
 - The signature must include these attributes otherwise verification will fail
- **The Signature Must be Generated with a Strong Hash Algorithm (SHA-2 and stronger)**
 - If the signature hash algorithm is one other than SHA-2/SHA-3 (e.g. SHA-1, MD5), verification will fail
- **All Certificates in the Signature Must be Generated With Strong Hash Algorithms (SHA-2 and stronger)**
 - As above but this setting relates to the certificate hash algorithms (rather than the data signature algorithm)
- **The Signing Certificate Must Include The Non Repudiation Key Usage**
 - If Non-Repudiation is required (rather than just the Digital Signature key usage) check this option
- **CA Certificates Must Contain The Basic Constraints Key Usage**
 - CA certificates should contain this extension, but some older certificates do not. Uncheck this option to allow these older non-compliant certificates. RFC 5280 mentions that conforming CAs MUST include this option
- **Relax Root Certificate Extension Checks**
 - Some root CA certificates are non-compliant regarding hash algorithms, key usage etc. but are still trusted. Check this option to allow these legacy root certificates to pass verification
- **Relax all certificate extension checks**
 - Check this option to permit all certificates to violate the usual hash algorithm and key usage rules. Checking this option overrides any other setting above
- **Allow Expired Certificates**
 - Expired certificates should fail verification. In extreme cases however, the risk to allow an expired certificate (for a number of days only) is less than the risk to operations or the business. Check this option to permit expired certificates
 - If the above option is checked you may specify the number of days an expired certificate will be permitted in the **Allow Expired Certs for (days)** field. For example, if a certificate has just expired and you wish to allow this for 5 days whilst a replacement is obtained, set this value to 5
- **Minimum Key Size**
 - The minimum key size that will be permitted. For example, if you wish to disallow certificates generated with keys smaller than 2048 bits, set this to 2048. Signatures generated with certificates whose key sizes are less than 2048 bits (e.g. 512bits and 1024bits) will then fail verification

- **Maximum Key Size**
 - The maximum key size that will be permitted. Signatures produced with large key sizes can become computationally expensive to process. Set this value to limit the maximum key size
- **Path Check Class**
 - If you have developed a custom path check class, specify the full classpath here. For more details on custom path checking see the **EzSign Installation and Configuration Guide**, section titled: **Custom Certificate Path Checking**

Revocation Checking Options

There are three options available for the Revocation Checker:

- **NONE**
 - Select this option if no revocation checking is required. If this option is selected no further configuration is required
- **OCSP**
 - Select this option if you wish to check revocation using OCSP (Online Certificate Status Protocol - see RFC 6960 for details). If this option is selected several other configuration options become available (see below)
- **CRL**
 - Select this option if you wish to check revocation using CRLs (Certificate Revocation Lists). If this option is selected several other configuration options become available (see below)

OCSP Configuration

The following options are available when OCSP revocation checking is configured

- **Connect Timeout (secs)**
 - The time in seconds the server will wait for a connection to be made to the OCSP server. If the OCSP server is busy or down, revocation checking will fail when this timeout is exceeded
- **Read Timeout (secs)**
 - The time in seconds the server will wait for a response from the OCSP server after a connection has been made and the request sent. If the OCSP server is busy or down, revocation checking will fail when this timeout is exceeded
- **Use Default OCSP URL**
 - If this option is checked, enter the OCSP URL in the **Default OCSP URL** box. All OCSP requests (including those for the end-entity, intermediate certificates and OCSP signer certificates) will then be sent to this same URL. This is required for IdenTrust compliance

- If this option is not checked then OCSP requests will be sent to the URL specified in the certificate's AIA (Authority Information Access) extension
- **Sign Request**
 - If checked, every OCSP request will be signed. The certificate that will perform the signing is the configured signing certificate for that channel. This is required for IdenTrust compliance. If this option is checked you must also choose the **Signature Hash Algorithm** which determines the algorithm that will be used to generate the signature over the OCSP request
- **Max Time Since Response Produced (minutes)**
 - This value determines how fresh OCSP responses must be. The OCSP Response *Produced At* value is examined and if further in the past than the value specified here, revocation checking will fail
- **Enable Response Caching**
 - If checked OCSP responses are cached for the number of seconds specified in the **Cache Response for (secs)** value. If a request for the same certificate is made within the cache period, the cached response will be used
 - If only the OCSP responses for CA certificates should be cached, check the **Cache CA Certs Only** option. This will result in an OCSP request being made for every end-entity certificate request, but requests for CA certificates will be cached
- **Check OCSP Response Signer Key Usage**
 - If checked, the certificate which signed the OCSP response must have the *OCSP Signing* enhanced key usage set
- **Ignore SSL/TLS Errors**
 - If TLS is used to secure the OCSP URL, the TLS root certificate (and potentially other certificates in the chain) must be present (and trusted) in the server's Java runtime (e.g. added to the cacerts file). If there are reasons why this cannot be performed and the OCSP URL is otherwise trusted, TLS errors can be ignored by checking this option
- **Use Proxy**
 - If access to the OCSP URL must go via a proxy, check this box. Ensure the proxy has already been setup in the **Proxy** section

CRL Configuration

The following options are available when CRL revocation checking is configured

- **Force CRL Download Every Time**
 - If checked, every time a revocation check is performed the CRL will be downloaded. This will ensure that the latest produced CRL is always retrieved, but is not recommended for high volume applications
 - If this is not checked a CRL will only be downloaded when the existing CRL expires
- **Allow Expired CRLs**

- An expired CRL should cause a revocation failure (if a fresh CRL cannot be downloaded). In extreme cases however, the risk to allow an expired CRL (for a number of days only) is less than the risk to operations or the business. Check this option to permit expired CRLs
- If this option is checked you may specify the number of days an expired CRL may be used in the **Allow Expired CRLs for (days)** field. For example, if a CRL has just expired and you wish to allow this for 5 days whilst the issue is resolved, set this value to 5
- **Ignore SSL/TLS Errors**
 - If TLS is used to secure the CRL download URL, the TLS root certificate (and potentially other certificates in the chain) must be present (and trusted) in the server's Java runtime (e.g. added to the cacerts file). If there are reasons why this cannot be performed and the CRL download URL is otherwise trusted, TLS errors can be ignored by checking this option
- **Use Proxy**
 - If access to the CRL download URL must go via a proxy, check this box. Ensure the proxy has already been setup in the **Proxy** section

Managing Certificates

From the **Channels** menu, click **Manage Certificates** from the **Action** drop down for the PKI channel required

Manage Certificates

Manage PKI keys and certificates

Channel Name

Keystore

Generate CSR
Import Certificates
Import From HSM

Current Signing Certificates

There are no signing certificates

Other Certificates

There are no other certificates

Cancel

CSR Generation

To generate a new CSR (Certificate Signing Request) click the **Generate CSR** button

CSR Generation
Generate a PKCS#10 Certificate Signing Request

Channel Name: SigningChannel

Distinguished Name: CN=Test,O=Krestfield,C=GB

Key Size: 2048

Generate Cancel

Enter the required DN (**Distinguished Name**) for the request and select the **Key Size** from the drop down. Click **Generate**

CSR Generation was successful

CSR details are below

Channel Name: PKIHSM

CSR Data

```
-----BEGIN CERTIFICATE REQUEST-----  
MIICcDCCAygCAQAwLTELMAkGA1UEBhMCR0IxDALBgNVBAoMBFRlc3QxZDZANBgNVBAMMBk15Q2Vy  
dDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBALd/JAYxygRqp3R3PBLb8C1vIv//16xy  
fTrbu+Fpq/XDlhv2LZ2s1U0f8i8dFKJCQv5f9gZW87D1IXX9ydyfLEbAYf2bTJoQlvXNzAdUNOjJ  
u80+KvVLgA72eSc4KJ302w/uLNKe65L5jnr0Acui0zKxm+eTTPMKhE1I8X33GCAnrLnH0wsGohY  
Wyr+znbBG5n0AbpC2E+DCiA8DWhXE3U34Z6TLsV83lXXqJt03lBocE1TPzuJJ3CQy1mMdvGiuVTI  
EOyUFVUMUX6j4SRc0fPQbTatlLEMCD17SaDf/Xn+jerZ7uHAF0FNjR8eunTJ+zujwEAFp0d/ZZ9+  
pZhM5E8CAwEAATANBgkqhkiG9w0BAQsFAAOCAQEAJ9FFm6W0e0Wx7zouJ8eJHVexyyz61CtDZ2Rq  
shq9y7fwsoS2bbatrYU69g/mZpvbIjIZnXBcdmDTms8f6xm26ZsV1QBpZACrAm2v1q/MHAz0UuDu  
4ddBbSNI6MQn4XaLxQlMwMUeGrIrCFTju6BXLVc/N7fwtWNXERFF0MG5bJrjFRCCVKPnY+Dzf9b  
SS6yyDzdpZxj2fpYaViN7MB/iNLhvrZ/uwSbmXyFpH/xyS2gpthG65FTXyMGp8kDrnlGC4r1r0Th  
NsGEPZ0fhPPkbrjQ/QqZbrpuke0bTttTQfzAUn1J7AG0RTd+eECKSOAQexhth6+wM70Tgv/0yzx0  
OQ==  
-----END CERTIFICATE REQUEST-----
```

Download Cancel

The CSR will be generated and can then be downloaded for submission to the CA (Certificate Authority) for signing

Importing Certificates

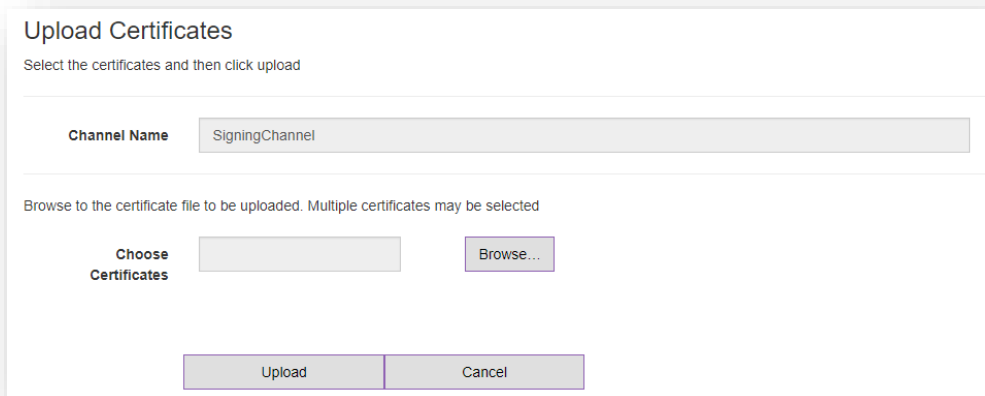
When the CSR (above) has been processed, the following must then be imported:

- The signing certificate issued from the CSR
- The issuing CA certificate
- The root CA certificate (if different to the issuing CA)
- Any other intermediate CA certificates in the chain

Such that the full certificate chain is imported

If the channel is to be used for verification purposes only (and not signing) at a minimum only the root CA certificate needs to be imported. However, if the remaining certificates in the chain (intermediate CA certificates) are not included in signatures or provided by the client when making a verify signature call, these must also be uploaded

To import a certificate, from the *Manage Certificates* menu click **Import Certificates**



Upload Certificates

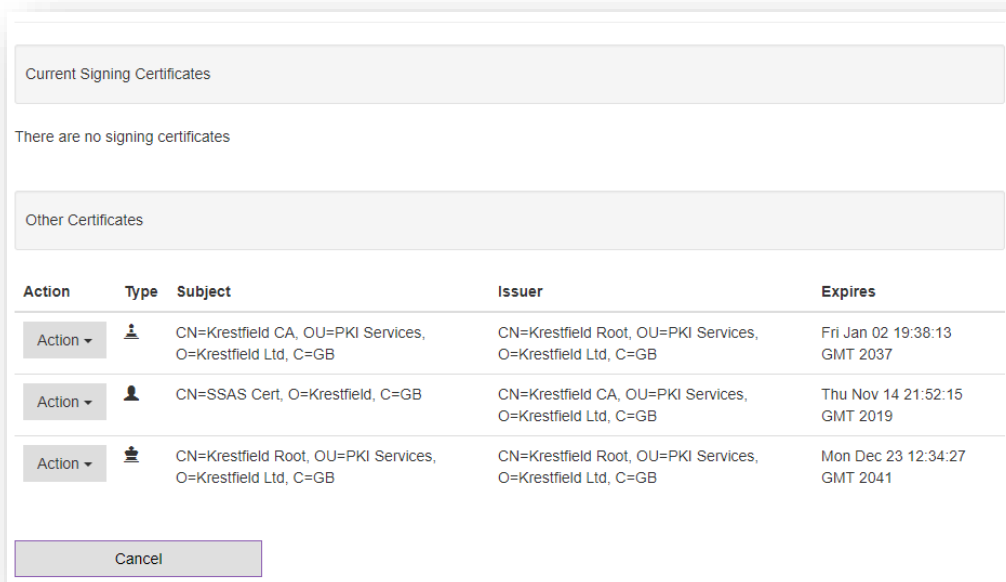
Select the certificates and then click upload

Channel Name

Browse to the certificate file to be uploaded. Multiple certificates may be selected

Choose Certificates

Click **Browse...** and navigate to the certificate files. These may include the end-entity (issued from the CSR), intermediate and root certificates. Multiple certificates can be selected including multiple end-entity and CA certificates. Files must be .cer, .crt files and may be binary (DER encoded) or SMIME base64 encoded files



The certificates will be imported

The Type symbol displayed indicates the certificate type, as follows:

- End Entity Certificate
- Intermediate CA Certificate
- Root CA Certificate

The options available under the **Action** button for each certificate are:

- **View Details**
 - View the certificate details
- **Download**
 - Download the certificate to the local machine
- **Delete**
 - Delete the certificate from the Key Store

For end entity certificates, if an associated private key exists (i.e. the certificate has been issued from a previously generated CSR), the additional option of **Set as the Signing Certificate** will also be present under the **Action** button. Note that the full certificate chain must be imported before this option becomes available

If this option is selected the certificate will be set as the signing certificate and sign data calls to the channel will use this certificate to sign the data

Once the signing certificate has been set, the full certificate path will be displayed under the **Current Signing Certificates** heading. Note that the full path will be displayed in the order: End Entity, Intermediate CA Certificates, Root Certificate

Certificates not involved in the signing path will be displayed under the **Other Certificates** heading

Current Signing Certificates				
Action	Type	Subject	Issuer	Expires
Action ▾	👤	CN=SSAS Cert, O=Krestfield, C=GB	CN=Krestfield CA, OU=PKI Services, O=Krestfield Ltd, C=GB	Thu Nov 14 21:52:15 GMT 2019
Action ▾	👤	CN=Krestfield CA, OU=PKI Services, O=Krestfield Ltd, C=GB	CN=Krestfield Root, OU=PKI Services, O=Krestfield Ltd, C=GB	Fri Jan 02 19:38:13 GMT 2037
Action ▾	👤	CN=Krestfield Root, OU=PKI Services, O=Krestfield Ltd, C=GB	CN=Krestfield Root, OU=PKI Services, O=Krestfield Ltd, C=GB	Mon Dec 23 12:34:27 GMT 2041
Other Certificates				
Action	Type	Subject	Issuer	Expires
Action ▾	👤	CN=VeriSign Class 3 Extended Validation SSL SGC CA, OU=Terms of use at https://www.verisign.com/rpa (c)06, OU=VeriSign Trust Network, O="VeriSign, Inc.", C=US	CN=VeriSign Class 3 Public Primary Certification Authority - G5, OU="(c) 2006 VeriSign, Inc. - For authorized use only", OU=VeriSign Trust Network, O="VeriSign, Inc.", C=US	Mon Nov 07 23:59:59 GMT 2016
Action ▾	👤	CN="*.tools.ietf.org, OU=Domain Control Validated, O="*.tools.ietf.org	SERIALNUMBER=10688435, CN=Starfield Secure Certification Authority, OU=http://certificates.starfieldtech.com/repository, O="Starfield Technologies, Inc.", L=Scottsdale, ST=Arizona, C=US	Sat Nov 30 23:34:19 GMT 2013
Action ▾	👤	C=GB, O=Krestfield Ltd, CN=Krestfield Email CA	C=GB, O=Krestfield Ltd, CN=Krestfield LA Root CA	Sat Mar 11 11:12:13 GMT 2028

Expired certificates are displayed in red

Importing From HSM

If the channel is using a PKCS#11 Key Store then the additional option **Import From HSM** will also be available

This option will import all keys and certificates that are already stored on the HSM. For example, if the HSM was used by another signing solution, the keys and certificates used by that system may be imported in to this channel

To perform this operation simply click the **Import From HSM** button. A confirmation message will be displayed once all objects have been imported. Use the **Set as the Signing Certificate** and **Delete** options to set the desired signing key and delete any unwanted certificates

Note that for any certificate changes to take effect, any running server instances must be restarted