Key Rollover

Note: This page contains general information about key rollover. If you are a deployer wanting to know how to migrate a certificate to and from InCommon Federation metadata, please refer to the Certificate Migration page.

As used here, the term *key rollover* refers to a process whereby one key is systematically replaced by another key in SAML metadata. Since SAML entities (and therefore SAML metadata) are distributed, key rollover must be deliberate, so as not to break the key operations of any relying party.

If metadata is controlled by a 3rd-party federation (which is often the case), key rollover is constrained by the key management capabilities provided by the federation. Moreover, not all implementations fully support the key rollover processes described here. For these reasons, there are numerous ways to get the job done, all of which are covered here. Of course the simplest approach, assuming no interoperability issues, is usually best.

Use Cases

Explicit Signing Key

Case 1:
`<md:KeyDescriptor use="signing">`

Preconditions:

- There is an `<md:KeyDescriptor use="signing">` element in metadata.
- The software is configured to use the corresponding private key as a signing key and/or an TLS key.

Procedure:

1. Add the new `<md:KeyDescriptor use="signing">` element to metadata
2. Wait for the newly updated metadata to propagate
3. Configure the software to use the new key (instead of the old key) as the signing key and/or the TLS key
4. Remove the old `<md:KeyDescriptor use="signing">` element from metadata

Explicit Encryption Key

Case 2:
`<md:KeyDescriptor use="encryption">`

Preconditions:

- There is an `<md:KeyDescriptor use="encryption">` element in metadata.
- The software is configured to use the corresponding private key as a decryption key.

Procedure:

1. Configure the software to use the new decryption key in addition to the old decryption key
2. Replace the old `<md:KeyDescriptor use="encryption">` element with the new `<md:KeyDescriptor use="encryption">` element in metadata
3. Wait for the newly updated metadata to propagate
4. Configure the software to use the new decryption key only (i.e., discontinue use of the old decryption key)

Multipurpose Keys

Case 3a:
`<md:KeyDescriptor use="signing">`

and

`<md:KeyDescriptor use="encryption">`

This case is essentially a concurrent execution of the algorithms in Cases 1 and 2. Apply this sequence of steps when the two key descriptors contain the *same key*.

Note that Cases 3b and 3c can be reduced to Case 3a if the `<md:KeyDescriptor>` element in metadata can be rewritten as an equivalent pair of key descriptors, one with `use="signing"` and one with `use="encryption"`. Although this is desirable for the purposes of key rollover, this option is not always feasible, in which case use Case 3b or 3c as required.

Preconditions:

- There are <md:KeyDescriptor use="signing"> and <md:KeyDescriptor use="encryption"> elements in metadata containing the *same key*.
- The software is configured to use a single private key as a signing key, as an TLS key, and/or as a decryption key.

Procedure:

1. Configure the software to use the new decryption key in addition to the old decryption key
2. Update the metadata as follows:
    a. Add the new <md:KeyDescriptor use="signing"> element to metadata
    b. Add the new <md:KeyDescriptor use="encryption"> element to metadata
    c. Remove the old <md:KeyDescriptor use="encryption"> element from metadata
    d. Leave the old <md:KeyDescriptor use="signing"> element in metadata
3. Wait for the newly updated metadata to propagate
4. Configure the software as follows:
    a. Use the new key (instead of the old key) as the signing key and/or TLS key
    b. Use the new decryption key only (i.e., discontinue use of the old decryption key)
5. Remove the old <md:KeyDescriptor use="signing"> element from metadata

Case 3b: <md:KeyDescriptor>

This case is essentially a concurrent execution of the algorithms in Cases 1 and 2.

Note that Case 3b is very similar to Case 3a. The differences are readily apparent in the pre- and post-conditions, so look at those carefully when choosing between the two.

Case 3b is preferred to Case 3c since 1) it's simpler and 2) only one encryption key is listed in metadata at any one time. If you are unable to perform step 2, presumably because you don't have adequate control over your metadata, skip **all** the steps below and go to Case 3c.

Preconditions:

- There is an <md:KeyDescriptor> element (with no use XML attribute) in metadata.
- The software is configured to use a single private key as a signing key, as an TLS key, and/or as a decryption key.

Procedure:

1. Configure the software to use the new decryption key in addition to the old decryption key
2. Update the metadata as follows:
    a. Add the new <md:KeyDescriptor> element (with no use XML attribute)
    b. Change the old <md:KeyDescriptor> element to an <md:KeyDescriptor use="signing"> element
3. Wait for the newly updated metadata to propagate
4. Configure the software as follows:
    a. Use the new key (instead of the old key) as the signing key and/or TLS key
    b. Use the new decryption key only (i.e., discontinue use of the old decryption key)
5. Remove the old <md:KeyDescriptor use="signing"> element from metadata

Case 3c: <md:KeyDescriptor>

This case is essentially a sequential execution of the algorithms in Cases 1 and 2.

Case 3c is the least desirable situation since 1) it's more complicated, 2) more time consuming, and 3) there are two encryption keys in metadata for a period of time (while the new metadata propagates). Avoid this situation if you can (since some implementations can't handle multiple encryption keys in metadata).

Preconditions:

- There is an <md:KeyDescriptor> element (with no use XML attribute) in metadata.
- The software is configured to use a single private key as a signing key, as an TLS key, and/or as a decryption key.

Procedure:

1. Configure the software as follows:
    a. Continue to use the old key as a signing key, as an TLS key, and/or as a decryption key
    b. Use the new key as a decryption key only
2. Update the metadata as follows:
    a. Add the new <md:KeyDescriptor> element to metadata
    b. Leave the old <md:KeyDescriptor> element in metadata
3. Wait for the newly updated metadata to propagate
4. Configure the software as follows:
    a. Use the old key as a decryption key only (i.e., discontinue the use of the old key as a signing key and/or TLS key)
    b. Use the new key as a signing key, as an TLS key, and/or as a decryption key
5. Remove the old <md:KeyDescriptor> element from metadata
6. Wait for the newly updated metadata to propagate
7. Configure the software as follows:

a. Discontinue the use of the old key as a decryption key
b. Continue to use the new key as a signing key, as an TLS key, and/or as a decryption key

The above procedure is identical to the key rollover process for a Shibboleth 2.x SP fully documented in the Shib wiki.