Certificate Migration

This is one of a series of documents regarding the use of X.509 certificates in metadata. The following instructions are intended for InCommon Federation participants wishing to replace an old certificate with a new certificate in metadata. Such a migration process (or key rollover, as it is sometimes called) is required, for example, in the case of expired certificates.

Contents

Getting Started

Start by reading the Key Usage topic and the sections below on Metadata Propagation and Implementation Support, which set the stage for certificate migration. If you're an IdP site administrator, continue by reading the article on Migrating a Certificate in IdP Metadata. SP site administrators, on the other hand, should read the article on Migrating a Certificate in SP Metadata instead.

- IdP Certificate Migration
- SP Certificate Migration

For general information about certificate migration strategies, see the comprehensive document on key rollover.

Multiple Keys for One Entity

Some of the procedures documented on the child pages of this wiki page lead to a situation where there are multiple key descriptors per role descriptor in metadata. In particular, to migrate a signing key, two signing keys will appear in metadata for a time. This is required to avoid interoperability problems.

Assuming an entity signs messages with one and only one key, there will be one or two signing keys in metadata at any given point in time. Unless the entity is undergoing key migration, there will be only one signing key in metadata. During key migration, there will be two signing keys in metadata at the same time.

It's actually more complicated than that. As noted on the Key Usage page, an `<md:KeyDescriptor use="signing">` element designates either a signing key or a back-channel TLS key (or both). If the software supports it, there might be separate keys configured for signing and back-channel TLS, in which case there would be two such key descriptors in metadata under normal circumstances and as many as four key descriptors during key migration.

Encryption keys are more straightforward. For any given encryption method, only one encryption key should be called out in metadata. Multiple encryption keys forces the relying party (usually the IdP) to make an unnecessary (and arbitrary) choice.

Site administrators are asked not to leave migrating keys in metadata indefinitely but rather limit the period of migration as discussed in the next section.

Metadata Propagation

The certificate migration processes outlined below require all changes to metadata to be propagated to your federation partners before the next step in the process can safely occur. How this happens depends on who your federation partners actually are. If your partners are members of the InCommon Federation, metadata propagation is fairly well understood (as long as your partners follow InCommon recommendations with respect to metadata consumption) but if you have partners that are not InCommon participants, then it is up to you to make sure those partners receive critical metadata updates.

For InCommon participants, metadata is refreshed at various times so the actual time required to propagate new metadata throughout the Federation is variable. Deployers are encouraged to communicate directly with Federation partners to ensure that critical metadata updates are received in a timely manner.

Since InCommon participants are strongly encouraged to update their metadata daily, you should *wait at least a day for new metadata to propagate*. You may want to wait longer, however. In any case, *wait no more than two (2) weeks for new metadata to propagate* since any given Federation metadata file has an explicit 2-week lifetime.

Wait time for metadata propagation

Wait 1–14 days for new metadata to propagate throughout the InCommon Federation.

Implementation Support

In general, SAML implementations have varying degrees of support for X.509 certificates in metadata, which leads to known interoperability issues. Thus software limitations need to be factored into the certificate migration decision-making process.

To fully support key rollover, implementations MUST support the following features:

1. An implementation MUST be able to consume and utilize two or more signing keys bound to a single role descriptor. There are two cases that MUST be supported:
    a. `<md:KeyDescriptor use="signing">` and `<md:KeyDescriptor use="signing">` elements in a single role descriptor
    b. `<md:KeyDescriptor use="signing">` and `<md:KeyDescriptor>` elements in a single role descriptor
2. If an implementation supports inbound encryption, it MUST be configurable with up to two decryption keys.
3. An implementation MAY support (i.e., consume and utilize) multiple encryption keys per role descriptor in metadata. In particular, the implementation MAY support two `<md:KeyDescriptor>` elements (with no `use` XML attribute) in a single role descriptor, but this is not strictly required since it can usually be avoided in practice.

It is known, for instance, that some IdP implementations will not consume SP metadata containing more than one encryption key descriptor. Depending on how you do key rollover, there may be a period of time where the migration of an encryption certificate in SP metadata will "break" interoperability with IdP deployments based on that implementation. In such a situation, the wait time for propagating metadata (e.g.) becomes a strategic decision left to the deployer's discretion.

Consult your software documentation to better understand its capabilities. Indeed, evaluate software capabilities with respect to certificate handling *before* you deploy, if at all possible.