Managing trust in keys used for metadata

Jump to:

There are two metadata trust models in wide use today, the *Explicit Key Trust Model* and the *PKIX Trust Model*. The latter aligns with common use of the term "public key infrastructure" (PKI), more appropriately called an "X.509 PKI." In contrast, the Explicit Key Trust Model is sometimes referred to as a "SAML PKI." This document concentrates on the Explicit Key Trust Model, the metadata trust model employed by the InCommon Federation.

A brief public key infrastructure primer

Regardless of the trust model, there are three keys of interest (listed below in increasing order of importance):

1. The SP decryption key
2. The IdP signing key
3. The FedOp signing key

The SP decryption key is used by the SP software to decrypt the SAML assertions obtained from the IdP. To encrypt the SAML assertions in the first place, the IdP uses the public key bound to the encryption certificate in trusted SP metadata.

The IdP uses its signing key to sign the SAML assertions transmitted to the SP. The SP verifies the signature on the assertion using the public key bound to the signing certificate in trusted IdP metadata.

The signing key controlled by the federation operator (FedOp) is used in conjunction with the Metadata signing process. An entity (IdP or SP) verifies the signature on the metadata immediately after retrieving the metadata file. To verify the signature, the entity uses the public key bound to the FedOp's Metadata signing certificate, which was previously obtained by some secure out-of-band process.

> ⚠ **Signing Certificate or Verification Certificate?**
>
> A *signing key* is a private key, but the corresponding public key bound to a certificate (usually referred to as the "signing certificate") is actually used for signature verification, so technically the certificate should be called a *verification certificate*. Although the term *signing certificate* is a misnomer, it is entrenched and so we use it here.

The Explicit Key Trust Model

Under the *Explicit Key Trust Model*, the public keys bound to certificates in metadata are trusted, not the certificates themselves. A certificate is merely a convenient wrapper for a trusted public key. Consequently, entities are encouraged to use long-lived, self-signed certificates, which simplifies key maintenance.

Likewise the public key corresponding to the federation operator's Metadata Signing Key is trusted. The content of the certificate containing the public key is completely ignored. This has important consequences on how entities securely obtain and consume metadata.

Changing the IdP signing key

Before changing the IdP's signing key, an additional public key certificate is inserted into IdP metadata, which is then distributed to SP partners so that SPs can subsequently verify the signature on an assertion issued by the IdP under the new signing key. Once the new certificate has propagated to all SP partners, the new signing key can be configured in the IdP software. Under normal circumstances, moving certificates in and out of IdP metadata requires an orderly migration process to avoid disruption of partner services.

However, if the IdP's signing key is believed to be compromised, both the private key and corresponding public key certificate in metadata should be replaced **immediately**. This will break interoperability with SPs until such time as they have refreshed metadata, so replace the IdP's signing key only under the most serious circumstances.

Changing the SP decryption key

Related content

- Using the fallback aggregate
- Download InCommon Metadata
- Metadata signing certificate
- Configure SimpleSAMLphp to consume InCommon metadata
- Best practices when consuming InCommon metadata
- Configure ADFS to consume InCommon metadata
- Configure Shibboleth to consume InCommon metadata

Get help

Can't find what you are looking for?

help Ask the community

Systematically changing the SP's decryption key is a more complicated operation. It requires two decryption keys to be configured in the SP software at one time. Once this is done, the old public key certificate is replaced by the new public key certificate, which is then distributed to all IdPs. Once the new certificate has propagated to all IdP partners, the old decryption key can be removed from the SP software configuration. For procedural details see the wiki topic on SP certificate migration elsewhere in this wiki.

If the SP's decryption key is believed to be compromised, both the private key and the corresponding public key certificate in metadata should be replaced **immediately**. Again, this will break interoperability with IdPs until such time as they have refreshed metadata. The danger of IdPs possibly releasing sensitive information to a rogue SP totally depends on the IdP's metadata refresh process.

Changing the metadata signing certificate

We now turn our attention to the federation operator's Metadata Signing Key, a critical component of the Metadata signing process. When the federation operator signs metadata, the corresponding public key certificate (i.e., the Metadata signing certificate) is inserted into metadata as part of the XML signature. A relying party verifies the signature and accepts the metadata if and only if it trusts the public key bound to the signing certificate in metadata.

To bootstrap the trust fabric of the federation, each relying party obtains and configures an authentic copy of the federation operator's Metadata signing certificate into its metadata refresh process. Note that the certificate must be obtained securely in the first place since all subsequent operations depend on it.

Assuming the metadata refresh process depends on the public key only (not the certificate), the federation operator can change the wrapper on the public key (i.e., the certificate) without affecting metadata refresh. If, on the other hand, the metadata refresh process depends on the certificate itself, changing the Metadata signing certificate will break metadata refresh even if the public key bound to the certificate does not change.

> Shibboleth is the only known SAML implementation whose metadata refresh capability depends on the public key only (not the signing certificate).

Please keep this in mind as you choose SAML software to deploy in the InCommon Federation.

Changing the metadata signing key

A new Metadata Signing Key is the most disruptive event under the Explicit Key Trust Model. In this case, **every** entity in the federation must (securely) obtain a new Metadata signing certificate (since a new private key implies a new public key). In effect, a key change "reboots" the trust fabric of the federation.

If a metadata client supports certificate chaining, a somewhat orderly migration to a new Metadata Signing Key is possible since two signing certificates can be configured in advance of the announced "Flag Day." Once the federation operator begins signing with the new key, the old signing certificate can be removed from each configuration. In a large federation, such a process may take many months to complete. Thus a key change is a painful process, one to be avoided if at all possible.

In the unlikely event that the private Metadata Signing Key is compromised, every entity in the federation is **immediately** vulnerable to compromise, precluding the possibility of an orderly migration. This is why the InCommon Metadata Signing Key is an offline key with strict handling procedures—to limit its exposure and therefore minimize the likelihood of compromise.

Refreshing the metadata

An SP that trusts an IdP consumes IdP metadata so that it can verify the signature on SAML assertions issued by the IdP. Likewise an IdP that trusts an SP consumes SP metadata so that it can encrypt SAML assertions transmitted to that SP. Clearly the trust model is predicated on the consumption of trusted metadata. Moreover, in the event of a key compromise, the trust fabric is disrupted until such time as all entities that trusted that key have refreshed metadata.

> The time it takes for the federation as a whole to recover from a disruptive key compromise is a function of the metadata refresh behavior of federation entities.

For this reason, all SAML entities are strongly encouraged to refresh metadata often.

More Information

- X.509 Certificates in Metadata

- Best practices when consuming InCommon metadata
- https://wiki.shibboleth.net/confluence/display/SHIB2/TrustManagement
- https://wiki.shibboleth.net/confluence/display/SHIB2/IdPTrustEngine