Understanding X.509 certificates in metadata

Jump to:

Background | Requirements | Interoperability | References

This article discusses the use of X.509 certificates in Federation metadata. It has security implications so please read it carefully.

A SAML entity uses public key cryptography to secure the data transmitted to trusted partners. Public keys are published in the form of X.509 certificates in metadata whereas the corresponding private keys are held securely by the entity. These keys are used for message-level signing and encryption, and to create secure back channels for transporting SAML messages over TLS. They are **not** used for browser-facing TLS transactions on port 443. See the Key Usage topic for more information.

Uses of Certificates in Metadata

Certificates in metadata are used for message-level signing and encryption, **not** browser-facing TLS transactions on port 443.

The InCommon Federation is based on the *Explicit Key Trust Model*, one of several possible metadata trust models. Consequently, the use of **long-lived, self-signed certificates** in metadata is strongly recommended. Certificates signed by a Certification Authority (CA) are allowed, and in most situations will work just fine, but the use of certificates other than self-signed certificates is discouraged. See the Background information and the Interoperability notes below for further discussion.

Trust the Key, Not the Certificate

From a security perspective, the only element of a certificate in metadata that matters is **the public key**. Conforming software will ignore all other certificate content.

Any certificates you want to use with your SAML software are uploaded via the Federation Manager. Typically only one certificate is required per entity but multiple certificates may be uploaded and used as needed. In particular, multiple certificates may be used to facilitate the controlled rollover of expired certificates or compromised keys. To avoid interoperability problems, refer to the Certificate Migration topic for recommended guidelines regarding the rollover process.

It is easy to create a self-signed certificate with the OpenSSL command-line tool. Before you do this, however, take a moment to consider how best to handle the all-important private key.

Prepare to Generate a New Private Signing Key!

Before generating a new private signing key for your IdP, read the IdP Key Handling topic.

Background

In the base SAML metadata specification [1], a certificate signing authority (CA) has no assumed relevance to the trust model that secures the interactions among a federation's participants. In fact, certificates signed by a CA are discouraged since they can create interoperability issues in certain situations and lead to configurations that mistakenly establish trust based on the certificate signer. Allowing self-signed certificates simplifies the work of participants who may be required to join multiple federations, or who support local systems that are not registered in the Federation.

InCommon conforms to the *SAML V2.0 Metadata Interoperability Profile* [2] from OASIS. Participant site administrators securely transmit X.509 certificates and metadata to InCommon via the administrative web interface. InCommon signs the entire metadata file, securing the keys of its participants whether those keys are bound to self-signed certificates or certificates signed by a CA. The critical element in the certificate is the public key, which is associated with an entity via its entity ID. Theoretically, if all the relevant software systems could accept a public key without a certificate wrapper, InCommon would only need to include the public key of each entity. As it is, the certificate is a convenient container for the public key, the critical element being that the key is bound to a particular entity in the metadata.

Related content

- Download InCommon Metadata
- Using the fallback aggregate
- Metadata signing certificate
- Signaling Encryption Method Support for a Service Provider
- Signing and Encryption Keys
- Best practices when consuming InCommon metadata
- Configure SimpleSAMLphp to consume InCommon metadata
- Working with SAML metadata
- Introduction to Federation Manager
- Tagging an entity with Research and Scholarship entity attribute

Get help

Can't find what you are looking for?

help Ask the community

Requirements

InCommon sets the following security and trust requirements around certificates included in Federation metadata:

- The use of **long-lived, self-signed certificates** in Federation metadata is strongly RECOMMENDED.
    - Certificates with lifetimes of at least 10 years are RECOMMENDED to avoid unnecessary technically-imposed deadlines on key rollover.
    - Certificates SHOULD expire before 2038 to avoid the so-called Year 2038 problem.
- RSA keys with a **minimum size of 2048 bits** MUST be used for all new certificates introduced into Federation metadata.
    - New certificates with **key sizes less than 2048 bits are not allowed** in Federation metadata.
    - Certificates with keys greater than 2048 bits are NOT RECOMMENDED since such keys force relying parties to perform unnecessary computation.
- **Expired certificates** SHOULD NOT be introduced into Federation metadata. An **expired certificate** in metadata SHOULD be removed once a certificate migration process to a new certificate has been completed.
    - A certificate's expiration date has nothing to do with the security of the corresponding private key, which is an ongoing concern.
- If a private key is lost or stolen, immediate steps MUST be taken to configure **a new private key** and to introduce the corresponding public key certificate into metadata. Since there are no other known attacks on RSA 2048-bit keys, generating **a new private key** for any other purpose is NOT RECOMMENDED.
- Service providers MUST include an **encryption key** in SP metadata.
    - The encryption key is used by IdPs to encrypt SAML V2.0 assertions transmitted to the SP.
- InCommon does not validate Subject information in self-signed certificates because this information is irrelevant from a security perspective. However, at its discretion, **InCommon will reject metadata submissions if that submission contains a certificate with fields that contain egregiously misrepresented Subject information** as decided by InCommon on a case-by-case basis. Generally, Subject information should express a somewhat reasonable relationship between the certificate and the organization.

Interoperability

Consider the following interoperability issues as you set up and maintain your deployment:

- A potential federation partner (especially a partner not using the Shibboleth software) may question the use of self-signed certificates. As discussed in the Background section, there are, in fact, fewer interoperability issues with self-signed certificates compared to CA-signed certificates.
- The Shibboleth software does not check the expiration dates of certificates [4], but **expired certificates often cause interoperability issues** with other software (such as AD FS 2.0 and the OIOSAML Java SP) and with older versions of Apache used to deploy the Shibboleth IdP. InCommon recommends that you plan ahead and migrate to an unexpired certificate well ahead of your certificate's expiration date.
- For key management purposes, **InCommon allows multiple certificates per role descriptor** at any time. (You can log into the administrative interface, select a particular role, and associate more than one certificate with that role for the purposes of migrating from one certificate to another.) Bear in mind, however, that some SAML **implementations do not support multiple keys properly** and you may want to test this capability with your non-Shibboleth partners. For example:
    - EZProxy is known to ignore additional keys beyond the first.
    - AD FS 2.0 will not consume an `<md:EntityDescriptor>` element containing more than one encryption key.
- At the deployer's convenience, **a single certificate may be bound to multiple SPs** in InCommon metadata. However, some implementations (e.g., AD FS 2.0) do not allow the same certificate to be used by two distinct entities.
- If the certificate will be used for TLS server authentication, **the certificate's CN (and/or subjectAltName) value should match the server's hostname**. This is especially true for IdPs but may also be true in certain advanced scenarios where the SP acts as a SOAP responder.
- Avoid **certificates with special certificate extensions**, since some implementations will actually try to use them. For example, AD FS 2.0 will attempt to access the CRL at the location given in the CRL Distribution Point certificate extension.

References

[1] *Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0* http://saml.xml.org/saml-specifications
[2] *SAML V2.0 Metadata Interoperability Profile* http://wiki.oasis-open.org/security/SAML2MetadataIOP
[3] X.509 Certificates in the Federation Metadata: A technical webinar presented by the *InCommon Technical Advisory Committee* (October 22, 2009)
[4] *The Shibboleth ExplicitKey Trust Engine* https://wiki.shibboleth.net/confluence/display/SHIB2/ExplicitKeyTrustEngine
]5] *Shibboleth Security and Networking* https://wiki.shibboleth.net/confluence/x/VoEOAQ