saml-metadata-cryptographic-keys

Jump to:

How do the keys work? | What about x.509 certificates? | | The technical details

Managing signing and encryption keys using Federation Manager

Log into the Federation Manager as a Site Administrator(SA).

Click on the entity you wish to update to bring up the View/Edit page.

On the left navigation, click "Signing/Encryption Keys" to bring up the IdP SSO Keys (or SP SSO Keys) section.

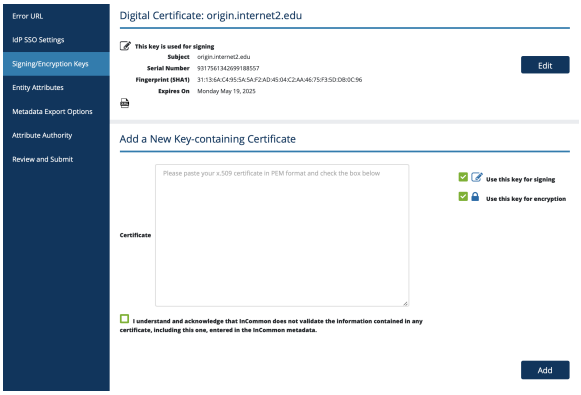
The "Add a New Key-containing Certificate" form allows you to upload a new certificate in PEM format. Use the checkboxes to the right to designate whether it is used for signing, encryption, or both.

IMPORTANT: When registering a service provider, make sure to read Signaling Encryption Method Support for a Service Provider first to make sure you are choosing the right encryption algorithm options when uploading encryption keys.

If you have previously uploaded other keys, they will appear first. The edit button lets you change the usage (signing or encryption) of these previously uploaded keys.

Important Notes

- An IdP metadata must contain at least one signing key. If you only have one signing key, you cannot delete that key (or uncheck "this key is used for signing") until you have uploaded another.
- An SP metadata must contain at least one encryption key. If you only have one encryption key, you cannot delete that key (or uncheck "this key is used for encryption") until you have uploaded another.
- You may designate multiple keys for signing and encryption. In fact, this is a preferred practice when performing key rollovers in order to avoid service interruption. However, be aware that some commercial SAML implementations do not function properly when multiple keys are designated for the same use within a single entity.
- Consult The technical details section below for additional information.



Remember: your metadata is not published to the InCommon metadata until you submit it for publishing using the "Submit This Entity for Publishing" button in the Review and Submit section. When you are ready to publish your metadata, don't forget to press that button.

How do the keys work?

Message signing and encryption in SAML rely on public key cryptography. Both Identity Providers (IdP) and Service Providers (SP) include in their respective metadata the **public keys** they intend to use for signing and/or encryption. Do not upload your private keys.

Each key uploaded to a SAML metadata is packaged in a digital certificate and encoded in the Privacy-Enhanced Mail (PEM) format, a de facto format used to encode digital certificates.

Working with SAML metadata

- Manage metadata export options
- Qualifications and Capabilities (Entity Attributes, etc.)
- SAML Representation of InCommon Metadata
- Requested Attributes
- Entity ID
- Scope
- Contacts information
- IdP SSO Settings (IDPSSODescriptor)
- SP SSO Settings (SPSSODescriptor)
- Signaling Encryption Method Support for a Service Provider

Related content

- Requirements to use Federation Manager
- What's New in Federation Manager
- Review and submit metadata
- Understanding the Endpoint Encryption Score
- Reset your Federation Manager user password
- Federation Manager
- Assign metadata management to a Delegated Administrator
- Prepare for Delegated Administration assignment
- Assign access to a Delegated Administrator
- Add an identity provider

Get help

Can't find what you are looking for?

help Ask the community

A signed SAML message lets the message recipient verify that the message delivered is sent by a trusted sender and has not been altered while in transit. A encrypted SAML message ensures that only the recipient of the SAML message can access its content.

InCommon Federation supports both signing and encryption of SAML messages. To enable message signing and/or encryption for your entity, use Federation Manager to add signing and encryption keys in your entity's metadata.

NOTE: 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.

Keys published by an Identity Provider

When an IdP designates a key as a signing key, the key is used by interoperating SP's to verify a signed message from the IdP: the IDP signs a SAML message using its private key; the receiving SP verifies the signature using the IdP's published public "signing" key in metadata.

When an IdP designates a key as a encryption key, the key is used by interoperating SP's to encrypt a SAML message destined for the IdP: the SP encrypts the SAML message using the IdP's published public "encryption" key in metadata; the IdP decrypts the message using its matching private key.

Keys published by an Service Provider

When an SP designates a key as a signing key, the key is used by interoperating IdP's to verify a signed message from the SP: the SP signs a SAML message using its private key; the receiving IdP verifies the signature using the SP's published public "signing" key in metadata.

When an SP designates a key as a encryption key, the key is used by interoperating IdP's to encrypt a SAML message destined for the SP: the IdP encrypts the SAML message using the SP's published public "encryption" key in metadata; the SP decrypts the message using its matching private key.

Additional Information

- IdP Key Handling
- Key Usage

What about x.509 certificates?

A cryptographic key published in SAML metadata is by convention packaged in a x.509 digital certificate and uploaded in the Privacy-Enhanced Mail (PEM) format. PEM is a de facto format used to encode digital certificates. This packaging convention often leads to implementors using the term "x.509 certificate" when discussing signing and encryption keys in SAML metadata.

In fact, keys in SAML metadata, at least when applied in the research and education federation model (ala the InCommon model), does not rely on the certificate authority trust model typically associated with x.509 certificates. When you publish your metadata in InCommon, the Incommon Federation is vouching that you in fact issued the keys (along with other data elements) embedded in your metadata. It does not matter who signed the certificate used to package the key. In fact, packing your keys in long-lived, self-signed certificates is the preferred way to go. It is easy to create a self-signed certificate with the OpenSSL command-line tool.

The technical details

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.

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.
- 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:

- While a self-signed certificate is preferred and safe to use when exchanging cryptographic keys in the Incommon Federation, a potential federation partner (especially a partner not using the Shibboleth software) may question the use of self-signed certificates due to common perceptions of how digital certificate works.
- 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 Federation Manager, 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.
- InCommon allows participants to use the same key/certificate for multiple SPs within their organization. That is, **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