SAML Keys and Certificates

A Primer on SAML Keys and Certificates

Contents

- The KeyDescriptor Element
  - Signing KeyDescriptor Type
  - Encryption KeyDescriptor Type
  - KeyDescriptor Content
- SAML Keys and Certificates
  - Signing Key and Certificate
  - TLS Key and Certificate
  - Encryption Key and Certificate
- Example
  - Creating a SAML Key and Certificate
  - Inspecting the SAML Certificate
  - Binding the SAML Certificate to a KeyDescriptor in Metadata

Terminology

- KeyDescriptor of type "signing": A `<md:KeyDescriptor use="signing">` element in SAML metadata
- KeyDescriptor of type "encryption": A `<md:KeyDescriptor use="encryption">` element in SAML metadata
- Signing certificate: A public key certificate bound to a KeyDescriptor of type "signing" in SAML metadata. A signing certificate is indistinguishable from a back-channel TLS certificate in metadata.
- Back-channel TLS certificate: A public key certificate bound to a KeyDescriptor of type "signing" in SAML metadata. A back-channel TLS certificate is indistinguishable from a signing certificate in metadata.
- Encryption certificate: A public key certificate bound to a KeyDescriptor of type "encryption" in SAML metadata.
- Credential: A private key plus its corresponding public key certificate.
- Signing credential: A key pair used for XML Signature. The public key is bound to a signing certificate in metadata. The private key is securely held by the party that signs the XML message.
- Back-channel TLS credential: A key pair used for back-channel TLS authentication. The public key is bound to a back-channel TLS certificate in metadata. The private key is securely held by the party to be authenticated.
- Encryption credential: A key pair used for XML Encryption. The public key is bound to an encryption certificate in metadata. The private key is securely held by the party that decrypts the XML message.

The KeyDescriptor Element

There are two types of KeyDescriptors in SAML metadata: the signing KeyDescriptor type and the encryption KeyDescriptor type. These KeyDescriptor types correspond to the following metadata elements (resp.):

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

In other words, a KeyDescriptor type is a particular type of `<md:KeyDescriptor>` element in SAML metadata.

The phrase “KeyDescriptor type” is suggested by the SAML Metadata specification. In the SAML Metadata schema, there is an enumerated string type called `md:KeyTypes` with permissible values “signing” and “encryption”.

Signing KeyDescriptor Type
The `signing KeyDescriptor` type corresponds to an `<md:KeyDescriptor use="signing">` element in metadata. The actual certificate contained by this metadata element enables message-level signing (i.e., XML Signature) or back-channel TLS authentication (or both).

**Encryption KeyDescriptor Type**

The `encryption KeyDescriptor` type corresponds an `<md:KeyDescriptor use="encryption">` element in metadata. The actual certificate contained by this metadata element enables message-level encryption (i.e., XML Encryption).

---

The use XML attribute

According to the SAML metadata schema, the `<md:KeyDescriptor/@use>` XML attribute is an optional attribute. A KeyDescriptor with no `use` XML attribute such as

```xml
<md:KeyDescriptor>
    <md:KeyDescriptor use="signing"/>
    <md:KeyDescriptor use="encryption"/>
</md:KeyDescriptor>
```

is merely an optimization for a pair of contiguous elements

**KeyDescriptor Content**

In practice, the content of any given `<md:KeyDescriptor>` element is a public key certificate. There are three types of certificates in metadata:

1. Signing certificate
2. Back-channel TLS certificate
3. Encryption certificate

A KeyDescriptor of type "signing" contains either a signing certificate or a back-channel TLS certificate while a KeyDescriptor of type "encryption" contains an encryption certificate.

---

A KeyDescriptor of type "signing" is ambiguous

A signing certificate is indistinguishable from a back-channel TLS certificate in metadata (and vice versa).

---

Using a signing certificate

The term "signing certificate" is a misnomer. A signing certificate in metadata is actually used for **signature verification**, not signing. The private signing key is held securely by the signing party.

---

Using an encryption certificate

To illustrate how XML Encryption works, an IdP uses the encryption certificate in SP metadata to encrypt a SAML assertion. The private decryption key is held securely by the SP.

---

In practice, the only part of the certificate that matters is the public key. A consumer that conforms to the [OASIS SAML V2.0 Metadata Interoperability Profile](https://docs.oasis-open.org/security/saml/v2.0/profiles.html) will completely ignore all other parts of the certificate except the public key. This defines a trust model called the **Explicit Key Trust Model**.

---

**SAML Keys and Certificates**

**Signing Key and Certificate**

A **signing credential** is a key pair used for XML Signature, which provides authenticity and integrity at the message level. The public key is bound to a signing certificate in metadata. The private key is securely held by the party that signs the XML message.

**TLS Key and Certificate**
A TLS credential is a key pair used for back-channel TLS authentication, which provides authenticity, integrity, and confidentiality at the transport level. The public key is bound to a back-channel TLS certificate in metadata. The private key is securely held by the party to be authenticated.

Encryption Key and Certificate

By definition, an encryption credential is a key pair used for XML Encryption, which provides authenticity and confidentiality at the message level. The public key is bound to the encryption certificate in metadata. The private key is securely held by the party that decrypts the XML message.

Example

Creating a SAML Key and Certificate

This section shows how to use the openssl command-line tool to create a key pair with a long-lived, self-signed certificate.

Security Alert!

If the private key created below is intended for a production system, issue the following command directly on the target system (IdP or SP) only. Otherwise remove the -nodes option to enable encryption of the private key. When the -nodes option is removed, the tool will prompt the user for a decryption password.

Create a private key and certificate

```
# change these values as needed
$ OUT_DIR=/tmp/credentials
$ LIFETIME=3650 # 10 yrs

# normally, only a back-channel TLS certificate needs a subject name
# but it doesn't hurt to add this to a signing or encryption certificate
$ SUBJECT_NAME='/CN=hostname.example.org'

# create a 2048-bit key with long-lived, self-signed certificate
$ /usr/bin/openssl req -new -x509 -nodes
   -newkey rsa:2048 -keyout $OUT_DIR/key.pem
   -days $LIFETIME -subj $SUBJECT_NAME -out $OUT_DIR/cert.pem
```

Inspecting the SAML Certificate

Use the openssl command-line tool to inspect the certificate created in the previous section. Note that the certificate contains a 2048-bit public key.

Inspect the certificate

```
# inspect the certificate
$ /usr/bin/openssl x509 -text -in $OUT_DIR/cert.pem
Certificate:
   Data:
      Version: 3 (0x2)
      Signature Algorithm: sha1WithRSAEncryption
      Issuer: CN=hostname.example.org
      Validity
         Not Before: Jun 4 22:42:33 2016 GMT
         Not After : Jun 2 22:42:33 2026 GMT
      Subject: CN=hostname.example.org
      Subject Public Key Info:
         Public Key Algorithm: rsaEncryption
         RSA Public Key: (2048 bit)
            Modulus (2048 bit):
               00:dd:a9:e8:c0:8a:de:66:29:b1:bf:f1:ff:ae:6f:
               56:b4:1a:3e:cb61:cc:cc:6f:4f:91:64:5c:d8:5f:
               f7:cf:ca:2b:3d:da:ee:d8:3e:cf:03:bb:9e:7d:8a:
               9c:cc:5e:02:cc:5c:41:ac:83:ee:5e:2e:40:8e:dc:80:
```
Note the PEM-encoded certificate appended to the output. In the next section, we will insert that certificate into SAML metadata.

**Binding the SAML Certificate to a KeyDescriptor in Metadata**

Finally, bind the above certificate to an `<md:KeyDescriptor use="signing">` element in SAML metadata:

```xml
<md:KeyDescriptor use="signing">
  "..."
</md:KeyDescriptor>
```
The above certificate is either a signing certificate or a back-channel TLS certificate (or both).