SAML Metadata 101

Essentials

Federation-as-a-Service Training
21./22. April 2015, Vienna

Lukas Hämmerle
lukas.haemmerle@switch.ch

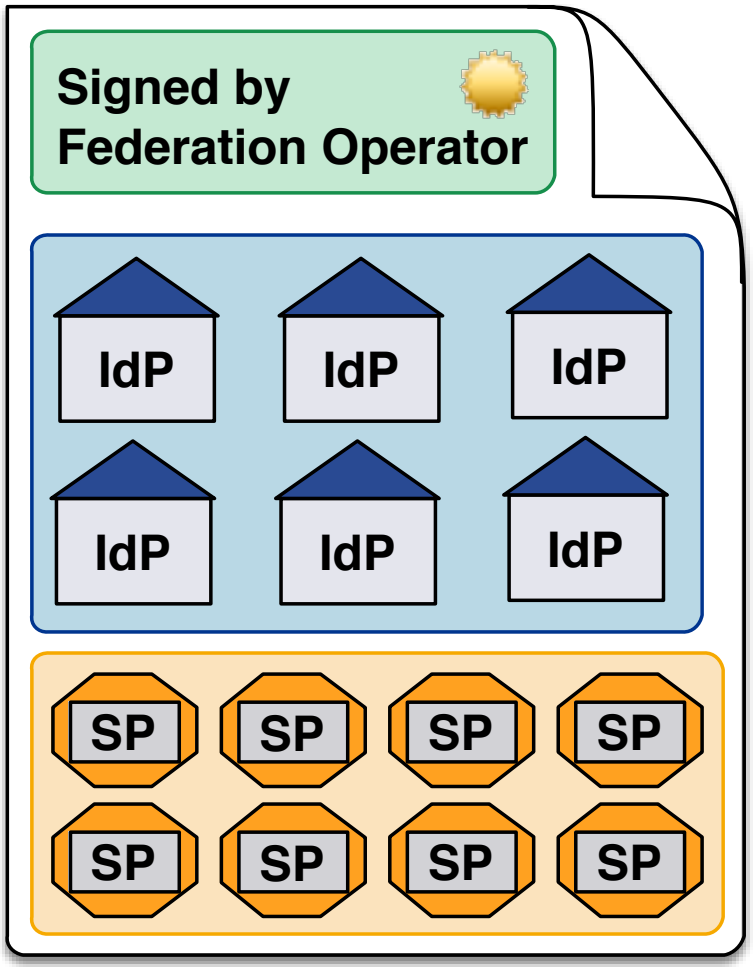
Theory

- Trust between entities
- SAML2 Metadata

Metadata

- Entities in a federation trust each other, but which entities are in federation?
- Entities in a federation must be known to trust them. Therefore, a standard way to list and describe entities is needed.
- (SAML2) Metadata provides such a standard way. Standardized format to describe entities
- Metadata typically is signed by a trusted third party, in our case the federation operator which should be trusted by all participants in the federation

Metadata Structure



To trust the metadata, it should be protected properly.

No defined order of IdPs and SPs.

Other entities could be described too. But mostly IdPs and SPs.

Add Trust to Metadata

- Consumers of metadata must be sure that the metadata was really created by Federation Operator
- Therefore, metadata must be secured
- Two methods to secure metadata:
  A. **Recommended:** Add an **XML signature** on metadata and publish public signing key Metadata can be served via http in this case.
  B. Serve plain metadata via a **secure HTTPS URL**.
  Make web server use a certificate issued by a well-known CA

Demonstration

- Metadata Signature

- Have a look at https://wiki.edugain.org/isFederatedCheck/Federations/ and see how and how often federations sign their metadata.

SAML2 Metadata


```
– <md:EntitiesDescriptor ID="eduGAIN" Name="http://edugain.org/" cacheDuration="PT6H" validUntil="2014-10-21T15:50:04Z">
  – <ds:Signature>
    – <ds:SignedInfo>
        <ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
        <ds:SignatureMethod Algorithm="http://www.w3.org/2001/04/xmldsig-more#rsa-sha256"/>
      – <ds:Reference URI="#eduGAIN">
        – <ds:Transforms>
            <ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
            <ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
          </ds:Transforms>
          <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
          <ds:DigestValue>0OusW5Die9h/Pf9RbWJ05FAjLJjvzwJhuOogpJ1XYzg=</ds:DigestValue>
        </ds:Reference>
      </ds:SignedInfo>
    – <ds:SignatureValue>
        BRvvbDjyPgat5mwH+2v9lnlzvIsvRil+nTKCi5M2DZnQwStIzZ1YKjHUeZUZ6+kvkJbO3zAXvGeC hDb7C4DPFpyT6zBvyRFdn
        BoWbYf9cgKwf9qNQjoILJo4eV8OekXhSSQIhDEttbqag07MBVdzjMRT6uc2i8cP+L1sQlFmqtnXf swToQrqIM01IIMVVlaeshvA
        NdFBNmPqaQmklx8kYycozYQwLR/kxd3ytukjAQ==
      </ds:SignatureValue>
    + <ds:KeyInfo></ds:KeyInfo>
    </ds:Signature>
  – <md:Extensions>
      <mdrpi:PublicationInfo creationInstant="2014-10-17T15:50:04Z" publisher="http://mds.edugain.org/"/>
    </md:Extensions>
```

Root Element of eduGAIN metadata

- **<md:EntitiesDescriptor**
  ID="eduGAIN"
  Name="http://edugain.org/"
  cacheDuration="PT6H"
  validUntil="2014-10-21T15:50:04Z"**>**

- <EntitiesDescriptor/> is always root element in SAML2 metadata
  This element can be nested but is generally not recommended
- Value of "ID" is used to reference what is signed
- "Name" is used mostly for human readability and attribute filtering
- "cacheDuration" tells an entity how often to download the metadata
- "validUntil" contains the expiration date of the metadata

Questions

- Why is metadata needed at all?
- Why does metadata have to be protected (signed)?
- Do you see that there is a scalability issue with the way metadata is handled today?

Theory

- The essential metadata pieces
- entityID
- endpoints
- certificates

EntityID

- Every entity needs a unique identifier: The **entityID**
- Where is entityID used?
  - In transmitted messages, local configuration, metadata, log files, configuration, filtering policies
- EntityID should be: Unique, locally scoped, representative and unchanging
- Convention: Include FQDN of your service
  - `https://`**`myservice.example.org`**`/shibboleth`
  - `https://`**`myapp.nren.net`**`/saml2/sp/metadata`

End points

- End points are URLs that other entities can send messages to

- **For SPs:**
  - **AssertionConsumerService**
  - SingleLogoutService
  - ArtifactResolutionService

- **For IdPs:**
  - **SingleSignOnService:**
  - ArtifactResolutionsService

Endpoint Bindings

- Each endpoint has a binding
- Binding defines how an assertion is transported to an endpoint
- Typical SAML2 Bindings are:
  - HTTP Post Binding (default): Also called Browser-Post binding
  - HTTP Redirect Binding: Assertion transportet via HTTP GET
  - HTTP Artifact Binding: “OneTimeToken” as GET argument
  - SAML SOAP: SP and IdP talk to each other directly via TLS

Endpoint Examples

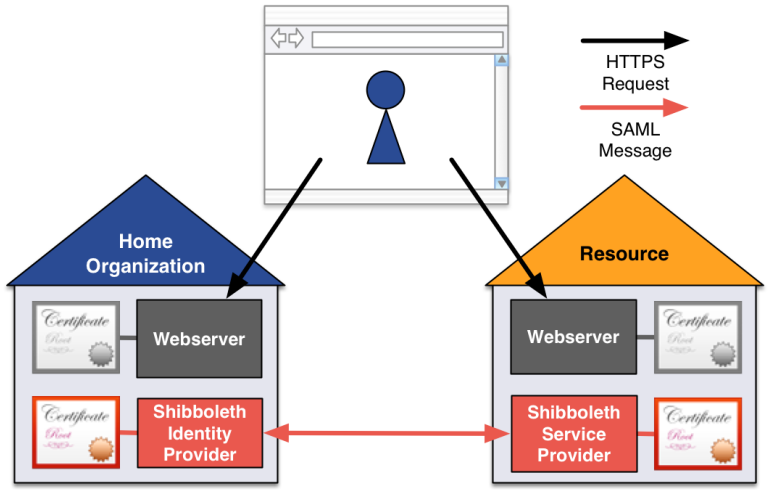
<md:AssertionConsumerService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="https://www.example.org/Shibboleth.sso/SAML2/POST"
index="0"/>

<md:SingleLogoutService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect”
Location="https://www.example.org/Shibboleth.sso/SAML2/Logout"/>

<md:SingleSignOnService
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"
Location="https://sir.rediris.es/ssp/saml2/idp/SSOService.php"/>

Involved X.509 certificates

- Certificate = public key + name + signature
- The two “certificate worlds” of a Shibboleth SP (or IdP):
  - SSL/TLS between the user’s browser and the Web server
  - XML Signature and XML Encryption for SAML messaging between the SP and the IdP



IdP-to-SP Communication

- To secure communications between an IdP and an SP (or vice versa), each entity needs a key pair:
  - For authenticating/signing SAML messages
  - For encrypting SAML messages
- RSA is most popular public-key algorithm today
  - Typically with a key size of 2048 bits
- X.509 is a standard, ubiquitous format for public-key data structures
  - Convenient container for public keys in SAML

Certificates in Metadata

```
<md:KeyDescriptor
  use="signing">
    <ds:KeyInfo>
      <ds:X509Data>
        <ds:X509Certificate>
          MIIEeTC [ Base 64 encoded X.509 certificate ] dYcDH
        </ds:X509Certificate>
      </ds:X509Data>
    </ds:KeyInfo>
</md:KeyDescriptor>
```

If the “use” attribute is ommited, the certificate can be used for signing and encryption.

Demonstration

- Let's dive into eduGAIN metadata
  http://mds.edugain.org

Questions

- Why could a (Shibboleth) IdP throw an error **"Error Message: SAML 2 SSO profile is not configured for relying party http://wiki.edugain.org/shibboleth**"?

- Is it a problem if an endpoint uses http:// instead https:// ?

- Why could an entity have more than one certificate in metadata?