Deployment Profile for the Swedish eID Framework

Version 1.7 - 2021-10-14 - *Draft version*

Registration number: **2019-308** (*previously: ELN-0602*)

Table of Contents

1. Introduction

This profile specifies behaviour and options that deployments of the SAML V2.0 Web Browser SSO Profile, [SAML2Prof], and the SAML V2.0 Holder-of-key Web Browser SSO Profile, [SAML2HokProf], are required or permitted to rely on. The requirements specified in this profile are in addition to the underlying normative requirements of [SAML2Prof] and [SAML2HokProf] (including updates to these specifications provided by [SAML v2.0 Errata 05]).

> Note: The profile is influenced by, but not normatively dependent on, SAML2Int.

Readers should be familiar with all relevant reference documents, and any requirements stated are not repeated unless where deemed necessary to clarify or highlight a certain issue.

Any SAML features specified in referenced SAML documents that are optional are out of scope of this profile, unless explicitly specified by this profile.

1.1. Requirements Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The use of SHOULD, SHOULD NOT, and RECOMMENDED reflects broad consensus on deployment practices intended to foster both interoperability and guarantees of security and confidentiality needed to satisfy the requirements of many organizations that engage in the use of federated identity. Deviating may limit a deployment's ability to technically interoperate without additional negotiation, and should be undertaken with caution.

1.2. References to SAML 2.0 Standards and Profiles

When referring to elements from the SAML 2.0 core specification [SAML2Core], the following syntax is used:

- `<saml2p:Protocolelement>` – for elements from the SAML 2.0 Protocol namespace.
- `<saml2:Assertionelement>` – for elements from the SAML 2.0 Assertion namespace.

When referring to elements from the SAML 2.0 metadata specifications, the following syntax is used:

- `<md:Metadataelement>` – for elements defined in [SAML2Meta].
- `<mdui:Element>` – for elements defined in [SAML2MetaUI].
- `<mdattr:Element>` – for elements defined in [SAML2MetaAttr].
- `<alg:Element>` – for elements defined in [SAML2MetaAlgSupport].

When referring to elements from the SAML V2.0 Holder-of-key Web Browser SSO Profile, [SAML2HokProf], the following syntax is used:

- `<hoksso:ProtocolBinding>` - for elements from the SAML V2.0 Web Browser Holder-of-key namespace.

When referring to elements from the "Identity Provider Discovery Service Protocol and Profile" specification [IdpDisco], the following syntax is used:

- `<idpdisc:DiscoveryResponse>`

When referring to the `Scope` element defined in the "Subject Identifier Attributes Profile" specification, [SAML2SubjIdAttr], the following syntax is used:

- `<shibmd:Scope>`

When referring to elements from the W3C XML Signature namespace (`http://www.w3.org/2000/09/xmldsig\#`) the following syntax is used:

- `<ds:Signature>`

Note: For all references to core SAML specifications, direct or indirect, the [SAML v2.0 Errata 05] MUST always be considered.

2. Metadata and Trust Management

Identity Providers and Service Providers conformant with this profile MUST provide a SAML 2.0 Metadata document representing its entity. The provided metadata MUST conform to "Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0", [SAML2Meta], and SHOULD follow "SAML v2.0 Metadata Profile for Algorithm Support Version 1.0", [SAML2MetaAlgSupport].

Services conforming to this profile MUST be able to consume SAML metadata on an automated and periodic basis using the processing rules defined by "SAML V2.0 Metadata Interoperability Profile Version 1.0" [SAML2MetaIOP]. Automatic consumption of metadata MUST be followed by a successful signature verification of the downloaded metadata before it is trusted and used.

2.1. Requirements for Metadata Content

2.1.1. Generic

2.1.1.1. Display Information

All services that are represented in the Metadata SHALL include a `<md:Organization>` element with mandatory child elements, which includes at least one of each of the elements `<md:OrganizationName>`, `<md:OrganizationDisplayName>` and `<md:OrganizationURL>`.

The `<md:OrganizationName>` element SHALL hold a registered name of the organization, which matches the agreement with the federation operator.

The `<md:OrganizationDisplayName>` element SHALL contain a display name of the organization and SHALL NOT contain a service name that is unrelated to the name of the organization.

Metadata for an entity SHALL contain an `<mdui:UIInfo>` extension, extending the `<md:SPSSODescriptor>` or `<md:IDPSSODescriptor>` element (depending on the type of the entity). This `<mdui:UIInfo>` element SHALL at least contain a `<mdui:DisplayName>` element with the language attribute `sv` (Swedish), representing the entity name that has been approved by the federation operator. The `<mdui:UIInfo>` element SHALL also contain a reference to one, or more, logotype images (`<mdui:Logo>`) as described in section 2.1.5 of [SAML2MetaUI] and SHOULD contain a `<mdui:Description>` element with the language attribute `sv` (Swedish).

It is RECOMMENDED that the above elements represented in Swedish also be represented with the language attribute `en` (English).

A federation operator may impose further requirements regarding the `<mdui:UIInfo>` extension.

2.1.1.2. Certificates in Metadata

Public keys used for signature validation and encryption MUST be expressed via X.509 certificates included in metadata as `<ds:X509Certificate>` child elements to `<md:KeyDescriptor>` elements. These certificates merely acts as containers for the public keys needed for signature validation and encryption and the consumer of a metadata entry cannot expect to be able to execute a successful certificate path validation of such certificates. A metadata consumer MUST NOT reject a metadata entry due to that certificate trust is missing, the certificate is expired, or certificate revocation information is missing.

However, for interoperability reasons it is RECOMMENDED that:

- self-signed certificates are used,
- non-expired certificates (with a long validity time) are used,
- certificates are not signed with MD5- och SHA1-based signature algorithms.

All services represented in metadata SHOULD include at least one `<md:KeyDescriptor>` element holding a certificate to be used for signature validation (with the `use`-attribute set to `signing`), and one `<md:KeyDescriptor>` element containing a certificate to be used for message encryption (with the `use`-attribute set to `encryption`). It is allowed to use the same key and certificate for both usages, but in order to enable a future key rollover it is RECOMMENDED to avoid using only one `<md:KeyDescriptor>` element for both usages (i.e., with an absent `use`-attribute).

In order to facilitate key rollover of a signing key, a service MUST support multiple signing certificates found in peer metadata and MUST support signature validation using a key from any of the available peer signature certificates.

For key rollover of encryption keys, the service that is performing the key rollover MUST only publish the latest key (in a certificate) to its metadata entry. Until all peers have consumed the service's metadata entry containing the new certificate, the service MUST support decrypting messages using both the old and new key.

2.1.1.3. Declaring Algorithm Support

Services MAY declare encryption and signature capabilities as defined in [SAML2MetaAlgSupport]. The declared algorithms MUST meet the algorithm requirements defined by the Swedish eID Framework or by the federation operator . Thus, it is not allowed to declare an algorithm, or key size, that is prohibited from use.

All services conformant with this specification MUST support the mandatory algorithms defined by the Swedish eID Framework, meaning that a service has no possibility to opt-out from a certain mandatory algorithm by excluding it from the declared capabilities. The main purpose of declaring an algorithm using any of the elements `<md:EncryptionMethod>`, `<alg:SigningMethod>` or `<alg:DigestMethod>` is to declare which algorithm the service prefers.

2.1.2. Service Providers

The `<mdattr:EntityAttributes>` element of a Service Provider's entity descriptor SHOULD contain one entity category attribute [EntCat] that holds at least one attribute value representing a service entity category as defined in [EidEntCat], identifying the Service Provider requirements in relation to identity services concerning attribute release and level of assurance.

The example below illustrates how an entity declares the service entity category identifier `http://id.elegnamnden.se/ec/1.0/loa3-pnr` in its metadata.

```
<md:Extensions>
  <mdattr:EntityAttributes xmlns:mdattr="urn:oasis:names:tc:SAML:metadata:attribute">
    <saml2:Attribute Name="http://macedir.org/entity-category"
                     NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
                     xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
      <saml2:AttributeValue xsi:type="xs:string">
        http://id.elegnamnden.se/ec/1.0/loa3-pnr
      </saml2:AttributeValue>
    </saml2:Attribute>
  </mdattr:EntityAttributes>
</md:Extensions>
```

If a Service Provider does not have any attribute requirements other than those implicitly requested by inclusion of one or more service entity categories, the metadata entry of the Service Provider SHOULD NOT include a `<md:AttributeConsumingService>` element (with `<md:RequestedAttribute>` elements).

Any needs for particular attributes from Identify Providers, when present, MUST be expressed through present service entity categories in combination with `<md:RequestedAttribute>` elements under the `<md:AttributeConsumingService>` element in the Service Provider metadata. The `<md:RequestedAttribute>` elements in the Service Provider metadata, when present, hold a list of requested and/or required attributes. This list of attributes MUST be interpreted in the context of present service entity categories defined in [EidEntCat].

> Note: Only the requirements for attributes that are not implicitly requested through a declared service entity category need to be expressed as `<md:RequestedAttribute>` elements.

A Service Provider MAY sign authentication request messages sent to Identity Providers. A Service Provider that signs authentication requests messages MAY also ensure that a receiving Identity Provider will only accept valid signed requests from this Service Provider by assigning the `AuthnRequestsSigned` attribute of the `<md:SPSSODescriptor>` to a value of `true`.

Section E7, “Metadata for Agreeing to Sign Authentication Requests”, of [SAML v2.0 Errata 05] specifies the following concerning the `AuthnRequestsSigned` attribute:

> Optional attribute that indicates whether the `<saml2p:AuthnRequest>` messages sent by this Service Provider will be signed. If omitted, the value is assumed to be false. A value of false (or omission of this attribute) does not imply that the Service Provider will never sign its requests or that a signed request should be considered an error. However, an Identity Provider that receives an unsigned `<saml2p:AuthnRequest>` message from a Service Provider whose metadata contains this attribute with a value of true MUST return a SAML error response and MUST NOT fulfill the request.

Furthermore, a Service Provider MAY require assertions that are issued to it, to be signed. This is done by assigning the `WantAssertionsSigned` attribute of the `<md:SPSSODescriptor>` to a value of `true`.

Note that the response message that carries the assertion will always be signed, so the Service Provider should only require signed assertions in case that it wants to preserve the proof of authenticity of an assertion separate from the response.

A Service Provider wishing to make use of a central discovery service as specified in the "Identity Provider Discovery Service Protocol Profile" [IdPDisco] MUST include at least one `<idpdisc:DiscoveryResponse>` element as an extension under the `<md:SPSSODescriptor>` element.

2.1.2.1. Holder of Key Support

A Service Provider that sends authentication requests using the Holder-of-key Web Browser SSO Profile MUST include a `<md:AssertionConsumerService>` element in its metadata according to section 2.8 of [SAML2HokProf].

If the Service Provider also makes use of the ordinary Web Browser SSO Profile at least one `<md:AssertionConsumerService>` element according to section 2.4.4 of [SAML2Meta] MUST also be included in the Service Provider metadata.

In those cases, where a Service Provider supports both profiles, it is RECOMMENDED that the a `<md:AssertionConsumerService>` element for ordinary Web Browser SSO Profile is marked as default.

```
<md:AssertionConsumerService index="0" isDefault="true"
  xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
  Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
  Location="https://sp.example.org/path1" />

<md:AssertionConsumerService index="1" isDefault="false"
  xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
  xmlns:hoksso="urn:oasis:names:tc:SAML:2.0:profiles:holder-of-key:SSO:browser"
  hoksso:ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
  Binding="urn:oasis:names:tc:SAML:2.0:profiles:holder-of-key:SSO:browser"
  Location="https://sp.example.org/path2" />
```

Example of a Service Provider's declared AssertionConsumerService elements; one for receiving assertions according the Web Browser SSO Profile and one for receiving assertions according to the Holder-of-key Web Browser SSO Profile.

2.1.3. Identity Providers

The `<mdattr:EntityAttributes>` element of an Identity Provider's entity descriptor SHOULD contain one entity category attribute [EntCat] that holds at least one attribute value representing a service entity category as defined in [EidEntCat], defining the Identity Provider ability to deliver assertions (defining the level of assurance and attribute release).

If an Identity Provider has the ability to release attributes, other than those implicitly given by the declared service entity categories, it is RECOMMENDED that those attributes are declared as `<saml:Attribute>` elements under the `<md:IDPSSODescriptor>` element.

The `<mdattr:EntityAttributes>` element of an Identity Provider's metadata SHALL contain an attribute according to [SAML2IAP] with its `Name` attribute set to `urn:oasis:names:tc:SAML:attribute:assurance-certification` and holding at least one attribute value identifying a Level of Assurance (LoA) for which the Identity Provider has been approved. The Swedish eID Framework defines such identifier values in section 3.1.1 of [EidRegistry] and their meanings are defined in [EidTillit].

```
<md:Extensions>
  <mdattr:EntityAttributes xmlns:mdattr="urn:oasis:names:tc:SAML:metadata:attribute">
    <saml:Attribute Name="urn:oasis:names:tc:SAML:attribute:assurance-certification"
                    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
                    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
      <saml:AttributeValue xsi:type="xsd:string">http://id.elegnamnden.se/loa/1.0/loa3</saml:AttributeValue>
    </saml:Attribute>
    ...
  </mdattr:EntityAttributes>
</md:Extensions>
```

Example of how an Identity Provider advertises that the service delivers authentication according to the `http://id.elegnamnden.se/loa/1.0/loa3` *level of assurance.*

An Identity Provider MAY require authentication request messages to be signed. This is indicated by assigning the `WantAuthnRequestsSigned` attribute of the `<md:IDPSSPDescriptor>` element to a value of `true`. See further section E7, "Metadata for Agreeing to Sign Authentication Requests", of [SAML v2.0 Errata 05].

Identity Providers SHALL advertise support for the SAP protocol according to [SigSAP], by including the service property entity category URI `http://id.elegnamnden.se/sprop/1.0/scal2` in its metadata. An Identity Provider that does not advertise support for SAP MAY ignore requests for SAD.

```
<md:Extensions>
  <mdattr:EntityAttributes xmlns:mdattr="urn:oasis:names:tc:SAML:metadata:attribute">
    <saml:Attribute Name="http://macedir.org/entity-category"
                    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
                    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
      <saml:AttributeValue xsi:type="xs:string">http://id.elegnamnden.se/sprop/1.0/scal2</saml:AttributeValue>
    </saml:Attribute>
    ...
  </mdattr:EntityAttributes>
</md:Extensions>
```

Example of how an Identity Provider advertises its support for SCAL2 authentication.

An Identity Provider that wishes to receive the `<psc:PrincipalSelection>` extension in authentication requests SHOULD include the `<psc:RequestedPrincipalSelection>` extension extending the `<md:IDPSSODescriptor>` element. In this extension the Identity Provider declares which attribute value(s) it wants to receive in authentication requests from requestors using the `<psc:PrincipalSelection>` authentication request extension. See section 5.3.3 and [PrincipalSel].

2.1.3.1. Declaring Authorized Scopes

An Identity Provider that releases "scoped attributes", see section 3.1.3 of [EidAttributes], MUST be authorized to do so by the federation operator. Each authorized scope MUST be declared in the Identity Provider metadata entry using a `<shibmd:Scope>` element, see section 3.5.2, "Scope Filtering", in [SAML2SubjIdAttr]. The `<shibmd:Scope>` elements MUST appear within the `<md:Extensions>` element of the `<md:IDPSSODescriptor>`.

```
<md:IDPSSODescriptor ...>
  <md:Extensions>
    <shibmd:Scope regexp="false" xmlns:shibmd="urn:mace:shibboleth:metadata:1.0>2021006883</shibmd:Scope>
    <shibmd:Scope regexp="false" xmlns:shibmd="urn:mace:shibboleth:metadata:1.0>2021006255</shibmd:Scope>
    <shibmd:Scope regexp="false" xmlns:shibmd="urn:mace:shibboleth:metadata:1.0>example.com</shibmd:Scope>
    ...
```

Example of how an Identity Provider declares that it is authorized to deliver scoped attributes for three different scopes; two organizational numbers and one domain.

2.1.3.2. Holder of Key Support

An Identity Provider that supports the Holder-of-key Web Browser SSO Profile MUST declare `<md:SingleSignOnService>` elements[*] in its metadata according to section 2.8 of [SAML2HokProf].

If the Identity Provider also supports authentication according the ordinary Web Browser SSO Profile `<md:SingleSignOnService>` elements[*] according to section 2.4.3 of [SAML2Meta] MUST also be included in the Identity Provider metadata.

```
<md:SingleSignOnService
  xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
  Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"
  Location="https://idp.example.org/path1" />

<md:SingleSignOnService
  xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
  Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
  Location="https://idp.example.org/path2" />

<md:SingleSignOnService
  xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
  xmlns:hoksso="urn:oasis:names:tc:SAML:2.0:profiles:holder-of-key:SSO:browser"
  hoksso:ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect"
  Binding="urn:oasis:names:tc:SAML:2.0:profiles:holder-of-key:SSO:browser"
  Location="https://idp.example.org/path2" />

<md:SingleSignOnService
  xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
  xmlns:hoksso="urn:oasis:names:tc:SAML:2.0:profiles:holder-of-key:SSO:browser"
  hoksso:ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
  Binding="urn:oasis:names:tc:SAML:2.0:profiles:holder-of-key:SSO:browser"
  Location="https://idp.example.org/path2" />
```

Example of an Identity Provider's declared SingleSignOnService elements; two for receiving requests according to the Web Browser SSO Profile and two for receiving requests according to the Holder-of-key Web Browser SSO Profile.

> [*]: This profile requires an Identity Provider to support both the POST and Redirect bindings, see section 5.2 below.

2.1.4. Signature Service

A Signature Service within the Swedish eID Framework is a Service Provider with specific requirements concerning its representation in metadata.

The `<mdattr:EntityAttributes>` element of a Signature Service SP entity descriptor SHALL include the service type entity category identifier `http://id.elegnamnden.se/st/1.0/sigservice` ([EidEntCat]) as a value to the entity category attribute [EntCat].

```
<mdattr:EntityAttributes xmlns:mdattr="urn:oasis:names:tc:SAML:metadata:attribute">`
  <saml2:Attribute Name="http://macedir.org/entity-category"
                   NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
                   xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
    <saml2:AttributeValue xsi:type="xs:string">http://id.elegnamnden.se/ec/1.0/loa3-pnr</saml2:AttributeValue>
    <saml2:AttributeValue xsi:type="xs:string">http://id.elegnamnden.se/st/1.0/sigservice</saml2:AttributeValue>
  </saml2:Attribute>
</mdattr:EntityAttributes>
```

Entity attributes for a Signature Service SP.

A Signature Service MUST assign the `AuthnRequestsSigned` attribute of the `<md:SPSSODescriptor>` element to `true`. This requirement ensures that the Signature Service always signs its authentication requests in order for the request to be accepted by the Identity Provider. The federation operator will enforce that all Service Providers that operate as Signature Services have this attribute set.

3. Name Identifiers

Identity Providers and Service Providers MUST support both the `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent` and the `urn:oasis:names:tc:SAML:2.0:nameid-format:transient` name identifier formats as specified in [SAML2Core].

Identity Providers SHALL default to use the `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent` name identifier format in cases where a Service Provider has not specified the name identifier to use (via the `<md:NameIDFormat>` element of the Service Provider metadata entry, or via the `Format` attribute of the `<saml2p:NameIDPolicy>` element of the authentication request message).

4. Attributes

Attribute specifications for the Swedish eID Framework are defined in [EidAttributes].

The content of `<saml2:AttributeValue>` elements exchanged via any SAML 2.0 messages or assertions SHOULD be limited to a single child text node.

For requirements regarding attribute inclusion in SAML assertions, see section 6.2.1, “Attribute Release and Consuming Rules”, below.

5. Authentication Requests

This profile supports two different SSO profiles; the Web Browser SSO Profile, [SAML2Prof], and the Holder-of-key Web Browser Profile, [SAML2HokProf].

The profile in use is determined by how an authentication process is initiated, meaning to which Identity Provider endpoint an `<saml2p:AuthnRequest>` message is sent, see 2.1.3.2.

5.1. Discovery

This profile does not impose any requirements of how the process of discovery is implemented by Service Providers wishing to display user interfaces for selection of Identity Providers for end users. However, Service Providers making use of a centralized discovery service as defined in [IdpDisco] MUST follow the requirements stated for discovery responses in section 2.1.2, "Service Providers".

5.2. Binding and Security Requirements

The endpoints, at which an Identity Provider receives a `<saml2p:AuthnRequest>` message, and all subsequent exchanges with the user agent, MUST be protected by TLS.

If the Identity Provider supports the Holder-of-key Web Browser SSO Profile, the dedicated endpoints for this profile (see section 2.1.3.2) MUST be configured to require a client X.509 certificate being presented as part of the TLS handshake (mTLS, mutual TLS), see section 2.4 of [SAML2HokProf]. If the Identity Provider receives a request on a Holder-of-key endpoint and is unable to retrieve the client X.509 certificate it MUST respond with an error, see section 2.6.4 of [SAML2HokProf].

A Service Provider sending an `<saml2p:AuthnRequest>` message to an Identity Provider may do so using either the HTTP-Redirect or HTTP-POST binding [SAML2Bind][*], meaning that Identity Providers conformant with this profile MUST support the HTTP-Redirect and the HTTP-POST binding.

An Identity Provider that requires `<saml2p:AuthnRequest>` messages to be signed MUST not accept messages that are not signed, or where the verification of the signature fails. In these cases the Identity Provider MUST respond with an error.

An Identity Provider that itself does not require authentication messages to be signed MUST still accept and verify signed request messages from Service Providers that indicate, in their metadata, that they sign request messages (see 2.1.2 above). If this signature verification fails, the Identity Provider MUST return a SAML error response and MUST NOT fulfil the request.

An Identity Provider that receives a request message that is not signed from a Service Provider that has indicated, in its metadata, that it will only send signed request messages (see 2.1.2 above) MUST respond with an error.

The signature for authentication request messages is applied differently depending on the binding. The HTTP-Redirect binding requires the signature to be applied to the URL-encoded value rather than being placed within the XML-message (see section 3.4.4.1 of [SAML2Bind]). For the HTTP-POST binding the `<saml2p:AuthnRequest>` element MUST be signed using a `<ds:Signature>` element within the `<saml2:AuthnRequest>`.

Before signing the authentication request, the Service Provider SHOULD consult the Identity Provider's metadata (`<alg:SigningMethod>` and `<alg:DigestMethod>` elements) to determine the intersection of algorithms, key sizes and other parameters as defined by particular algorithms that it supports and that the Identity Provider prefers. If the intersection is empty, or if the Identity Provider has not declared any algorithms, the Service Provider MUST use one of the mandatory signing and digest algorithms defined in the Swedish eID Framework during the signature operation.

> [*]: A Service Provider should be aware of web browser, or web server, size limitations of URL:s, and it is RECOMMENDED that a Service Provider use the HTTP-POST binding in cases where the `<saml2p:AuthnRequest>` message becomes very

> large. This may be the case when a `SignMessage` extension containing much data is used, or when a complete certificate chain is attached to the signature of the message.

5.3. Message Content

An `<saml2p:AuthnRequest>` message MUST NOT contain a Document Type Definition (DTD).

The `<saml2p:AuthnRequest>` message SHOULD contain an `AssertionConsumerServiceURL`* attribute identifying the desired response location. The Service Provider MUST NOT use any other values for this attribute than those listed in its metadata record as `<md:AssertionConsumerService>` elements for the HTTP-POST binding (see section 4.1.6 of [SAML2Prof]), and URL canonicalization or normalization MUST NOT be required.

The `Destination` attribute of the `<saml2p:AuthnRequest>` message MUST contain the URL to which the Service Provider has instructed the user agent to deliver the request. This is useful to prevent malicious forwarding of signed requests from being accepted by unintended Identity Providers.

Identity Providers conformant with this profile MUST support the `ForceAuthn` and `IsPassive` attributes received in `<saml2p:AuthnRequest>` messages.

Service Providers SHOULD include the `ForceAuthn` attribute in all `<saml2p:AuthnRequest>` messages and explicitly set its value to `true` or `false`, and not rely on its default value. The reason for this is to avoid accidental SSO.

If the Service Provider has included more than one `<md:AttributeConsumingService>` element in its metadata it is RECOMMENDED that the `<saml2p:AuthnRequest>` message contains the `AttributeConsumingServiceIndex` attribute holding the index of the `<md:AttributeConsumingService>` element that the Identity Provider should consider during attribute release.

> [*]: The `AssertionConsumerServiceURL` attribute is mutually exclusive with the `AssertionConsumerServiceIndex` attribute, meaning that a `<saml2p:AuthnRequest>` message MUST NOT contain both attributes.

5.3.1. Requested Authentication Context

A Service Provider SHOULD explicitly specify a requested authentication context element (`<saml2p:RequestedAuthnContext>`), containing `<saml2:AuthnContextClassRef>` elements each holding a Level of Assurance authentication context URI (see section 3.1.1 of [EidRegistry]) that the Service Provider considers acceptable for the authentication process.

A present `<saml2p:RequestedAuthnContext>` element MUST specify exact matching by means of either an absent `Comparison` attribute or a `Comparison` attribute with the value set to `exact`. This means that the Identity Provider is forced to return an assertion with exactly one of the requested `<saml2:AuthnContextClassRef>` in the request as the declared `<saml2:AuthnContext>`, or return an error response. If the Service Provider requires the Identity Provider to return specifically one out of a selection of acceptable authentication context URIs, then all of these URIs MUST be included in the request.

The requested authentication context SHOULD be consistent with at least one of the service entity categories [EidEntCat] declared in the Service Provider's metadata entry. See further section 5.4.4 below.

```
<saml2p:RequestedAuthnContext Comparison="exact">
  <saml2:AuthnContextClassRef>http://id.elegnamnden.se/loa/1.0/loa3</saml2:AuthnContextClassRef>
</saml2p:RequestedAuthnContext>
```

Example of how an Authentication Context URI identifier representing a requested Level of Assurance is included in an authentication request message.

```
<saml2p:RequestedAuthnContext Comparison="exact">
  <saml2:AuthnContextClassRef>http://id.elegnamnden.se/loa/1.0/loa3</saml2:AuthnContextClassRef>
  <saml2:AuthnContextClassRef>http://id.elegnamnden.se/loa/1.0/eidas-nf-sub</saml2:AuthnContextClassRef>
</saml2p:RequestedAuthnContext>
```

Example of how several Authentication Context URIs are included in an authentication request message. In this case, the Service Provider states that it requests the authentication to be performed according to either the LoA3 URI defined within the Swedish eID Framework or the substantial level for notified eIDs defined within the eIDAS Framework.

5.3.2. Scoping

An Identity Provider that acts as a proxy for other Identity Providers SHOULD support the `<saml2p:Scoping>` element holding one, or more, entries signalling to the Proxy Identity Provider which Identity Provider(s) that may be used to authenticate the user. A Service Provider that sends authentication requests to a Proxy IdP MAY include the `<saml2p:Scoping>` element. See section 3.4.1.5 of [SAML2Core].

```
<saml2p:Scoping>
  <saml2p:IDPList>
    <saml2p:IDPEntry ProviderID="http://idp.example.com" />
  </saml2p:IDPList>
</saml2p:Scoping>
```

Example of how an `AuthnRequest` contains a `Scoping`-element where the requester (Service Provider) signals to the Proxy IdP which Identity Provider that should perform the authentication of the user.

The Swedish eIDAS Connector is a Proxy IdP that proxies requests to foreign eIDAS Proxy Services. Normally, the eIDAS connector presents a country selection dialogue to the user, prompting for the country where the user should be directed to for authentication. However, a Service Provider MAY include an eIDAS Proxy Service alias URI for a specific country's Proxy Service under the `<saml2p:Scoping>` element of the `<saml2p:AuthnRequest>` in order to bypass the country selection dialogue. See section 3.1.9.1 of [EidRegistry].

```
<saml2p:Scoping>
  <saml2p:IDPList>
    <saml2p:IDPEntry ProviderID="http://id.swedenconnect.se/eidas/1.0/proxy-service/no" />
  </saml2p:IDPList>
</saml2p:Scoping>
```

Example of how an `AuthnRequest` sent to the eIDAS Connector requests that the eIDAS Connector should proxy the request to the Norwegian eIDAS Proxy Service for authentication.

> **Note**: A Service Provider may list more than one country in the `<saml2p:IDPList>` of a `<saml2p:Scoping>` element. The eIDAS Connector will then present a country selection dialogue containing only the listed countries.

Note: Another method of bypassing the country selection dialogue is to include the `c` (`urn:oid:2.5.4.6`) attribute with the country code for the requested country as its value in a `<psc:PrincipalSelection>` extension of the `AuthnRequest` message, see section 5.3.3 below.

5.3.3. Principal Selection

The specification "Principal Selection in SAML Authentication Requests", [PrincipalSel], defines the `<psc:PrincipalSelection>` extension that enables a requestor to inform the Identity Provider about known attributes for the principal that is about to be

authenticated.

This may be relevant for Identity Provider services who prompt users for an identifier in order initiate an authentication process with the user's eID. In such cases, this extension can be used to avoid unnecessary prompting when the user's identity is known in advance to the service requesting authentication, such as when a previously authenticated user is requested to sign a document.

An Identity Provider that wishes to receive the `<psc:PrincipalSelection>` extension SHOULD advertise this in its metadata according to section 2.1.3.

A Service Provider that sends an authentication request to an Identity Provider that has declared that it wishes to receive certain attribute values MAY include the `<psc:PrincipalSelection>` extension in the `<saml2p:AuthnRequest>` message, provided that the Service Provider has knowledge about this information.

A Service Provider that is a Signature Service SHOULD include the extension for the case described above.

5.4. Processing Requirements

5.4.1. Validation of Destination

An Identity Provider receiving a `<saml2p:AuthnRequest>` message MUST verify that the `Destination` attribute is present, and that it is consistent with URLs configured in the Identity Provider's metadata.

5.4.2. Validation of Assertion Consumer Addresses

If the `AssertionConsumerServiceURL` attribute is present in the `<saml2p:AuthnRequest>` message, its value MUST be verified to be consistent with one of the `<md:AssertionConsumerService>` elements having the HTTP-POST binding* found in the Service Provider's metadata entry and matching the SSO profile** in use (see section 2.1.2.1). If this is not the case, the request must be rejected.

If the `AssertionConsumerServiceIndex` attribute is present in the `<saml2p:AuthnRequest>` message, a `<md:AssertionConsumerService>` matching this index MUST be present in the Service Provider's metadata. This `<md:AssertionConsumerService>` element MUST also match the binding and SSO profile in use as defined above. If this is not the case, the request must be rejected.

> This means that an Identity Provider that services an authentication request received on an `SingleSignOnService` endpoint dedicated for the Holder-of-key Web Browser Profile MUST NOT allow an `<md:AssertionConsumerService>` element that is not intended for the Holder-of-key profile, see section 2.8 of [SAML2HokProf].

If the attribute is not present in the `<saml2p:AuthnRequest>` message, the Identity Provider MUST obtain the desired response location from the Service Provider's metadata entry. This location is determined by first finding all the `<md:AssertionConsumerService>` elements that matches the SSO profile** in use (see section 2.1.2.1) and the binding* (HTTP-POST), and then selecting the element that is marked as default (has the `isDefault` attribute set), or if no matching element is marked as default, the one with the lowest index value (see section 2.4.4.1 of [SAML2Meta] and section 2.8 of [SAML2HokProf]). If no matching `<md:AssertionConsumerService>` element is found the request must be rejected.

> [*]: For Holder-of-key `<md:AssertionConsumerService>` elements the actual binding is given by the `hoksso:ProtocolBinding` attribute, see section 2.8 of [SAML2HokProf].
>
> [**]: The Web Browser SSO Profile or the Holder-of-key Web Browser SSO Profile are possible profiles.

5.4.3. Identity Provider User Interface

An Identity Provider presenting information related to a Service Provider in its user interface during processing of an `<saml2p:AuthnRequest>` message MUST obtain this information from the `<mdui:UIInfo>` element of the Service Provider's metadata entry. Implementers of this profile MUST be capable of handling display information stored in the `<mdui:DisplayName>`, `<mdui:Logo>` and the `<mdui:Description>` elements.

An Identity Provider displaying logotypes found in a Service Provider's metadata MUST be able to handle embedded in-line images, as well as vector-based images where the height and width given in metadata should be interpreted as image dimensions and not the actual size.

5.4.4. Authentication Context and Level of Assurance Handling

This framework defines a number of authentication context identifiers (URI), where each such identifier specifies a defined Level of Assertion and may define specific requirements on the authentication process. There can be multiple authentication context URIs representing the same Level of Assertion, but one authentication context URI always identifies one defined Level of Assurance.

Identity Providers SHALL exclusively use one of the requested authentication contexts in `<saml2p:AuthnRequest>` in the `<saml2:AuthnContextClassRef>` element under the `<saml2p:RequestedAuthnContext>` element, when present, to determine the requested authentication process and Level of Assurance. The Identity Provider SHALL respond with an error `<saml2p:StatusCode>` with the value `urn:oasis:names:tc:SAML:2.0:status:Requester` [SAML2Core] if no requested authentication context is supported. If no requested authentication context is present in the `<saml2p:AuthnRequest>`, the Identity Provider MAY return the result of a default authentication process that is consistent with the Identity Providers metadata.

Note: The Identity Provider does not have to consider the service entity categories ([EidEntCat]) declared in the Service Provider's metadata entry when determining the requested authentication context under which the authentication should be performed. The purpose of the service entity categories is primarily to support service matching in discovery services and attribute release policies in Identity Providers. Significant Identity Provider products and software are not equipped to use service entity category information to determine the requested authentication context.

5.4.5. Single Sign On Processing

An Identity Provider conformant to this profile MAY issue an assertion relying on a previously established security context (active session) instead of authenticating the user. However, the Identity Provider MUST NOT re-use an already existing security context in the following cases:

- When the security context has expired, i.e., the time elapsed since the security context was established is too long given the SSO-policy stipulated by the federation or Identity Provider itself.
- When the `<saml2p:AuthnRequest>` contains a `ForceAuthn` attribute with the value of `true`.
- If the original authentication process, which led to the establishment of the security context, was performed using another Level of Assurance that what is requested in the current `<saml2p:AuthnRequest>` message.
- If the original authentication was performed according to the Holder-of-key Web Browser SSO Profile, [SAML2HokProf], and no certificate, or a certificate not matching the certificate included in the `<saml2:SubjectConfirmation>` element of the original assertion, is presented in along with the current `<saml2p:AuthnRequest>` message.
- If the original authentication was performed according to the Web Browser SSO Profile, [SAML2Prof] and the current request is made according to the Holder-of-key Web Browser SSO Profile, [SAML2HokProf].

If the Identity Provider user interface contains some sort of user consent, or information, concerning which attributes, or any other information, that is included in an assertion being issued, the Identity Provider SHOULD preserve this functionality if a `<saml2p:AuthnRequest>` message requesting a different set of attributes (or any other information) compared to what was delivered in the assertion at the time of establishing the security context. The Identity Provider may require re-authentication or display a user interface for consent/information in these cases.

6. Authentication Responses

6.1. Security Requirements

The endpoint(s) at which a Service Provider receives a `<saml2p:Response>` message MUST be protected by TLS.

If a Service Provider wants to make use of the Holder-of-key Web Browser SSO Profile it needs to provide a dedicated endpoint for this purpose, see section 2.1.2.1. This endpoint MUST be configured to require a client X.509 certificate being presented as part of the TLS handshake (mTLS, mutual TLS), see section 2.4 of [SAML2HokProf]. If the Service Provider receives a `<saml2p:Response>` on a Holder-of-key endpoint and is unable to retrieve the client X.509 certificate it MUST reject the message and not create a security context for the principal, see section 2.6.6 of [SAML2HokProf].

The `<saml2p:Response>` message issued by the Identity Provider MUST be signed using a `<ds:Signature>` element within the `<saml2p:Response>` element.

The `<saml2:Assertion>` element issued by the Identity Provider MAY be signed using a `<ds:Signature>` element within the `<saml2:Assertion>`. If a Service Provider requires signed assertions, by assigning the `WantAssertionsSigned` attribute of its metadata record (see chapter 2.1.2), the Identity Provider MUST sign assertions issued to this Service Provider (as well as the response message as stated above).

Identity Providers SHALL utilize XML Encryption and return a `<saml2:EncryptedAssertion>` element in the `<saml2p:Response>` message. The elements `<saml2:EncryptedID>` and `<saml2:EncryptedAttribute>` MUST NOT be used; instead the entire assertion MUST be encrypted.

Before performing encryption and signing, the Identity Provider SHOULD consult the Service Provider's metadata (`<md:EncryptionMethod>`, `<alg:SigningMethod>` and `<alg:DigestMethod>` elements) to determine the intersection of algorithms, key sizes and other parameters as defined by particular algorithms that it supports and that the Service Provider prefers. If the intersection is empty, or if the Service Provider has not declared any algorithms, the Identity Provider MUST use algorithms listed as mandatory by the Swedish eID Framework. For encryption, the chosen algorithm MUST also be compatible with the Service Provider's encryption key declared in metadata.

Service Providers SHOULD NOT accept unsolicited `<saml2p:Response>` messages (i.e., responses that are not the result of an earlier `<saml2p:AuthnRequest>` message). Service Providers that do accept unsolicited response messages MUST ensure, by other means, that the security and processing requirements of this profile (section 6.3) can be fully satisfied.

6.2. Message Content

The `<saml2:Response>` message MUST NOT contain a Document Type Definition (DTD).

Successful response messages MUST contain exactly one `<saml2:AuthnStatement>` element and exactly one `<saml2:AttributeStatement>` element.

The `<saml2:Response>` message MUST contain an `<saml2:Issuer>` element containing the unique identifier (entityID) of the issuing Identity Provider.

The `AuthnInstant` attribute of the `<saml2:AuthnStatement>` element MUST be assigned the time when the actual authentication took place. This time may differ from the `IssueInstant` attribute of the assertion itself, which holds the time when the assertion was issued. This is especially important in cases of re-use of already established security contexts at the Identity Provider side (Single Sign On).

Each identity assertion MUST have a `<saml:Subject>` element that specifies the principal that is the subject of all of the statements in the assertion.

The value of the `<saml:NameID>` element under the `<saml:Subject>` element MUST hold a pseudonym identifier of the subject, which SHALL be:

- Unique for the IdP – SP combination being the issuer and recipient for the assertion.
- Constructed in a manner that does not reveal the registered identity of the subject.

The `<saml2:Subject>` element MUST contain one `<saml2:SubjectConfirmation>` element containing a `Method` attribute having the value of `urn:oasis:names:tc:SAML:2.0:cm:bearer` if the Web Browser SSO Profile was used, and `urn:oasis:names:tc:SAML:2.0:cm:holder-of-key` if the Holder-of-key Web Browser SSO Profile was used. This element MUST contain a `<saml2:SubjectConfirmationData>` element that contains at least the following:

- An `InResponseTo` attribute matching the request's ID.
- A `Recipient` attribute containing the Service Provider's assertion consumer service URL (see sections 5.3 and 5.4.1).
- A `NotOnOrAfter` attribute containing a time instant at which the subject no longer can be confirmed.
- An `Address` attribute containing the network address from which an attesting entity (user) can present the assertion.
- If the Holder-of-key Web Browser SSO Profile was used to authenticate the principal a `<ds:KeyInfo>` element identifying the X.509 certificate presented to the Identity Provider. See section 6.2.2 below.

The assertion MUST contain a `<saml2:Conditions>` element containing the following attributes and elements:

- A `<saml2:AudienceRestriction>` element including the requesting Service Provider's unique identifier (entityID) as an `<saml2:Audience>` value.
- A `NotBefore` attribute specifying the earliest time instant at which the assertion is valid.
- A `NotOnOrAfter` attribute specifying the time instant when the assertion expires.

An Identity Provider conformant to this profile MUST, in its issued assertions, include an authentication context URI indicating under which Level of Assurance the assertion was issued. This identifier MUST be placed under the `<saml2:AuthnStatement>` element as the value of an `<saml2:AuthnContextClassRef>` element that is part of the `<saml2:AuthnContext>` element.

```
<saml2:AuthnStatement AuthnInstant="2013-03-15T09:22:00" SessionIndex="b07b804c-7c29-ea16-7300-4f3d6f7928ac">
  <saml2:AuthnContext>
    <saml2:AuthnContextClassRef>http://id.elegnamnden.se/loa/1.0/loa3</saml2:AuthnContextClassRef>
    ...
  </saml2:AuthnContext>
</saml2:AuthnStatement>
```

Example of how an Authentication Context URI identifier representing a Level of Assurance is included in an authentication statement.

An Identity Provider that acts as a proxy for other Identity Providers SHOULD include the `<saml2:AuthenticatingAuthority>` element under the `<saml2:AuthnContext>` element. This element will contain the entityID of the Identity Provider that authenticated the principal.

```
<saml2:AuthnStatement AuthnInstant="2013-03-15T09:22:00" SessionIndex="b07b804c-7c29-ea16-7300-4f3d6f7928ac">
  <saml2:AuthnContext>
    ...
    <saml2:AuthenticatingAuthority>http://idp.example.com/auth</saml2:AuthenticatingAuthority>
  </saml2:AuthnContext>
</saml2:AuthnStatement>
```

Example of how the entityID of an Identity Provider that provided the authentication for the principal is included in an authentication statement.

6.2.1. Attribute Release and Consuming Rules

An Identity Provider determines which attributes to include in the `<saml2:AttributeStatement>` element of an assertion based on the Service Provider requirements and its agreements with the user being authenticated. Service Provider attribute preferences and requirements are specified by the service entity categories [EidEntCat] and requested attributes in the `<md:AttributeConsumingService>` element declared in the Service Provider metadata. A service entity category specifies the attribute set (as defined in [EidAttributes]) that is requested for the attribute release process.

An Identity Provider declares service entity categories in order to publish its ability to deliver attributes according to certain attribute sets. For all declared service entity categories, the Identity Provider MUST possess the ability to deliver the mandatory attributes of the underlying attribute set. See [EidEntCat] and [EidAttributes] for details.

An Identity Provider releasing scoped attributes, see section 3.1.3 of [EidAttributes], MUST be authorized to release attributes with given scopes by the federation operator. An Identity Provider's authorized scopes are published in its metadata according to section 2.1.3.1, "Declaring Authorized Scopes".

Consequently, a Service Provider consuming a scoped attribute SHOULD verify that the issuing Identity Provider has been authorized to release attributes for the given scope by asserting its presence in the Identity Provider's metadata, see section 2.1.3.1 and section 3.5.2 of [SAML2SubjIdAttr]. A scoped attribute that fails this check MUST NOT be accepted by the Service Provider.

The Service Provider is responsible for checking that an Identity Provider is capable of providing necessary attributes before sending a request and to verify that it received all attributes necessary for providing a requested service. Checks whether an Identity Provider is capable of fulfilling the needs of a Service Provider can be done either by relying on a discovery process to filter out non-conformant Identity Providers, and/or by examining the metadata of Identity providers.

An Identity Provider receiving a request for more attributes than it can provide SHOULD return an assertion with the attributes it can provide according to its defined attribute release policy if the user has been successfully authenticated, leaving it up to the Service Provider to decide how to proceed, e.g., by denying service to the authenticated user, provide limited services or to use other resources to collect necessary attributes. However, if a Service Provider has expressed a specific attribute requirement using the `<md:RequestedAttribute>` element of a matching `<md:AttributeConsumingService>` element in its metadata and assigned the `isRequired`-attribute to `true`, and the Identity Provider knows that it will not be able to provide this attribute, then the Identity Provider SHOULD reject any request for authentication from that Service Provider and respond with an error.

For privacy reasons, an Identity Provider SHOULD NOT release identity attributes[*] that are not requested by the Service Provider via its metadata declarations (service entity categories or `<md:RequestedAttribute>` elements).

> [*]: This requirement does not apply to attributes that are not "identity" attributes, for example transaction identifiers or any other attribute that does not directly belong to a user's identity.

6.2.2. Message Content Requirements for Holder-of-key

When the Holder-of-key Web Browser SSO Profile has been used during the authentication of the user, information about the X.509 certificate presented by the user MUST be included in a `<ds:KeyInfo>` element placed in the `<saml2:SubjectConfirmationData>` element. Section 2.4 of [SAML2HokAP] defines the requirements for this.

This profile adds the following requirements to what is specified in [SAML2HokAP]:

The user X.509 certificate SHOULD be represented using a `<ds:X509Certificate>` element in all cases, except for the cases when this certificate contains identity information about the subject that is not represented as SAML attributes in the `<saml2:AttributeStatement>` element. The reason for this is to ensure the privacy of the user and not release identity information that is not requested by the Service Provider (see section 6.2.1 above).

In these cases the `<ds:X509Certificate>` MUST NOT be used, and the Identity Provider SHOULD include a `<ds:KeyValue>` representing the public key of the user certificate.

6.3. Processing Requirements

This profile mandates a correct processing of a `<saml2p:Response>` message in order to ensure proper protection from the security threats described in [SAML2Sec]. Processing requirements are listed in [SAML2Core], [SAML2Prof] and [SAML2Sec]. This document will list the necessary requirements that apply to this profile.

After the Service Provider has encrypted the assertion from the received response message the following requirements apply. Any verification that fails MUST lead to that the Service Provider rejects the response message and does not use the assertion.

Some of the processing requirements below are defined in order to protect from MITM- or MITB-attacks[*] were unsigned authentication requests may be changed before being sent to the Identity Provider. However, a Service Provider MUST implement all of the specified processing requirements even if it sends signed authentication request messages.

> [*]: MITM stands for ”man in the middle” and MITB stands for ”man in the browser”.

6.3.1. Signature Validation

The signature present on the `<saml2p:Response>` message, and optionally on the `<saml2:Assertion>`, MUST be successfully verified.

The public key being used to verify the signature MUST appear in the issuing Identity Provider’s metadata record (as a `<ds:X509Certificate>` under the `<ds:KeyInfo>` element).

6.3.2. Subject Confirmation

Based on the `InResponseTo` attribute of the `<saml2:SubjectConfirmationData>` the Service Provider MUST be able to obtain the corresponding `<saml2p:AuthnRequest>` message, or a secure context containing corresponding information from the request (for future processing of the assertion).

The `Recipient` attribute from the `<saml2:SubjectConfirmationData>` element MUST match the location to which the `<saml2p:Response>` message was delivered **and** match the value the `AssertionConsumerServiceURL` attribute included in the request message, or if this attribute was not provided in the request message, the default response location specified in the Service Provider’s metadata entry, as described in section 5.4.2.

The time from the `NotOnOrAfter` attribute from the `<saml2:SubjectConfirmationData>` MUST NOT have passed compared with the time instant at which the subject is confirmed (i.e., when the assertion is validated). A reasonable allowable clock skew between the providers should be taken in account.

If the `Address` attribute is assigned to the `<saml2:SubjectConfirmationData>` element, the Service Provider MAY choose to check the user agent’s client address against it. Practical issues regarding the Service Provider’s network setup and the risk of introducing false negatives makes this an optional step in the validation phase.

If the `<saml2:SubjectConfirmation>` element has a `Method` set to `urn:oasis:names:tc:SAML:2.0:cm:holder-of-key` its containing `<saml2:SubjectConfirmationData>` element MUST be processed according to section 2.5 of [SAML2HokAP]. The Service Provider MUST also be able to verify the holder-of-key assertion if the `<saml2:SubjectConfirmationData>` element contains a `<ds:KeyValue>` element, see 6.2.2.

6.3.3. Conditions

The Service Provider MUST assert that the value of the `<saml2:Audience>` element under the `<saml2:AudienceRestriction>` element matches the unique entityID of the Service Provider.

The Service Provider MUST verify that the time instant at which the assertion is validated is within the range given by the `NotBefore` and `NotOnOrAfter` attributes of the `<saml2:Conditions>` element (allowing for a reasonable clock skew). See also the processing of the `NotOnOrAfter` attribute in section 6.3.2.

6.3.4. The Authentication Statement

The Service Provider MUST assert that the `<saml2:AuthnStatement>` contains a `<saml2:AuthnContext>` element that holds a `<saml2:AuthnContextClassRef>` element having as its value the authentication context URI indicating under which Level of Assurance the authentication was performed. If the Service Provider declared one, or more, `<saml2:AuthnContextClassRef>` elements under the `<saml2p:RequestedAuthnContext>` element of the authentication request (see section 5.4), the received authentication context URI MUST match one of the declared authentication context URI:s from the request. If not, the Service Provider MUST reject the assertion*.

> [*]: If the Service Provider does not declare an authentication context URI in the authentication request it should be prepared to receive any of the authentication context URI:s declared by the Identity Provider in its metadata record (see section 2.1.3).

6.3.5. General Security Validation

In order to protect itself from replay attacks, the Service Provider MUST ensure that the same assertion is not processed more than once within the time it is valid (with respect to the `NotOnOrAfter` attribute of the `<saml2:Conditions>` element).

In order to prevent stolen assertions and user impersonation, the Service Provider SHOULD implement a validation that rejects an assertion if the time given it its `IssueInstant` attribute compared to the time when the response message is received is too great. This time is typically on the order of seconds, and limits the time window when a stolen assertion could be used.

If the Service Provider included the attribute `ForceAuthn` with a value of `true` in the authentication request, the Service Provider SHOULD ensure that the `AuthnInstant` attribute of the `<saml2:AuthnStatement>` element is greater than the time when the request was sent (allowing for a reasonable clock skew).

A reasonable clock skew according to this profile is between 3 and 5 minutes in either direction.

6.4. Error Responses

If the Identity Provider returns an error, it MUST NOT include any assertions in the `<saml2p:Response>` message.

An Identity Provider conformant with this profile SHOULD NOT make use of any other `<saml2p:StatusCode>` values than those specified in section 3.2.2.2 of [SAML2Core] or in section 3.1.4 of [EidRegistry]. The top-level `<saml2p:StatusCode>` value may only be one of the following error identifiers:

- `urn:oasis:names:tc:SAML:2.0:status:Requester` – The request could not be performed due to an error on the part of the Service Provider.
- `urn:oasis:names:tc:SAML:2.0:status:Responder` – The request could not be performed due to an error on the part of the Identity Provider.
- `urn:oasis:names:tc:SAML:2.0:status:VersionMismatch` – The Identity Provider could not process the request because the version of the request message was incorrect.

If the user cancels an authentication process the Identity Provider SHOULD indicate this by assigning the second-level status code to `http://id.elegnamnden.se/status/1.0/cancel`.

If an Identity Provider displays information describing an error in its user interface it MUST also offer ways for the end user to confirm this information (for example, by including an OK-button). When the end user acknowledges taking part of the

information (i.e., clicks on the OK-button), the `<saml2p:Response>` message is posted back to the Service Provider according to the HTTP POST binding [SAML2Bind].

If an Identity Provider detects suspicious fraudulent behaviour or if any of its security checks alerts a (possible) fraud, the Identity Provider MUST NOT issue an assertion but instead display an error message. After the end user confirms this error message, the error message posted back to the Service Provider SHOULD contain a second-level status code set to `http://id.elegnamnden.se/status/1.0/fraud` or `http://id.elegnamnden.se/status/1.0/possibleFraud` (depending on whether the Identity Provider aborted the authentication due to a determined or suspected fraud).

7. Authentication for Signature

“DSS Extension for Federated Central Signing Services”, [EidDSS], defines an extension to the OASIS DSS protocol for providing centralized Signature Services within the Swedish eID Framework. This specification defines the communication between a *Signature Requestor*[*] and a Signature Service, but does not cover SAML specific requirements regarding the user authentication phase that is part of the signature process.

This section defines requirements on the SAML authentication process when authentication is requested by a Signature Service, acting as a SAML Service Provider. All requirements regarding user authentication specified earlier in this profile are still valid. This section extends these requirements for the “authentication for signature” process.

> [*]: A Signature Requestor is a Service Provider within the federation to which the user previously has logged in to and from where the user initiates a signature operation.

7.1. Authentication Requests

Authentication requests from a Signature Service SHALL meet the following requirements:

- The `ForceAuthn` attribute of the `<saml2p:AuthnRequest>` element MUST be set to `true`.
- The `<saml2p:AuthnRequest>` element MUST be signed. This MUST also be indicated in the Signature Service metadata record using the `AuthnRequestsSigned` attribute (see section 2.1.4).

If the receiving Identity Provider has declared that it wishes to receive principal selection information about the signer (see section 5.3.3), and the Signature Service has received any of the requested attribute values in the `Signer` element of the `SignRequest` ([EidDSS]), the Signature Service SHOULD include those attribute values in a `<psc:PrincipalSelection>` extension of the authentication request.

It is RECOMMENDED that the `<saml2p:Scoping>` element containing a `<saml2p:RequesterID>` element holding the entityID of the Service Requestor is included in `<saml2p:AuthnRequest>` messages generated by a Signature Service.

```
<saml2p:Scoping>
  <saml2p:RequesterID>http://www.origsp.com/sp</saml2:RequesterID>
</saml2p:Scoping>
```

Example when the `<saml2p:RequesterID>` element is used to inform the Identity Provider about which Service Provider that requested the signature associated with this request for authentication.

> The reason for this recommendation is that an Identity Provider may adapt user interfaces or authentication procedures to different Service Providers based on either static configuration or on information found in the Service Provider's metadata. It can therefore be useful for an Identity Provider to know which Service Provider that requested the signature (*Signature Requestor*) that caused a Signature Service to request authentication. The Identity Provider may use this information to maintain the same user experience and procedures regardless of whether authentication is requested directly by the Service Provider, or by a Signature Service as a result of a signature request from the same Service Provider.

An Identity Provider that accepts an `<saml2p:AuthnRequest>` message from a Service Provider that has indicated that it is a Signature Service[*] MUST provide a user interface that is indicating that the end user is performing a signature.

> [*]: An Identity Provider identifies a Service Provider as a Signature Service if it declares the `http://id.elegnamnden.se/st/1.0/sigservice` URI as a service type entity category in its metadata (see 2.1.4).

7.1.1. Requesting Display of Signature Message

[EidDSS_Profile] specifies that a Signature Requestor may include a `SignMessage` element (as defined by [EidDSS]) in a signature request. This element holds a message that the Identity Provider, which is responsible for “authentication for signature”, should present to the user that is performing the signature.

A Signature Service MAY request the Identity Provider to show a sign message to the user by including the `SignMessage` element from the signature request as a child element to an `<saml2p:Extensions>` element in the `<saml2p:AuthnRequest>` message (see section 3.2.1 of [SAML2Core]).

All Identity Providers compliant with this profile MUST be able to parse and understand the `SignMessage` extension.

If the `Message`-element of the `SignMessage` is to be encrypted, the Service Provider SHOULD consult the Identity Provider's metadata (`<md:EncryptionMethod>` elements) to determine the intersection of algorithms, key sizes and other parameters as defined by particular algorithms that it supports and that the Identity Provider prefers. If the intersection is empty, or if the Identity Provider has not declared any algorithms, the Service Provider MUST use one of the mandatory algorithms defined in the Swedish eID Framework during the encryption operation, which is compatible with the Identity Provider's encryption key declared in metadata.

If an Identity Provider that has ways of determining the principal's identity before displaying a sign message receives a `<psc:PrincipalSelection>` extension in the authentication request it SHOULD ensure that the contents of the `<psc:PrincipalSelection>` extension matches the principal identity. If there is a mismatch, the Identity Provider MUST NOT display the sign message, and if the request states that the sign message must be displayed, fail with an error response where the second-level status code is set to `urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal` ([SAML2Core]).

> Typical scenarios where the above requirement apply is for the Swedish eIDAS connector that first authenticates a principal and then displays the sign message, and for Identity Providers that prompt the user for its user identity as part of its authentication scheme. In those scenarios the Identity Providers must compare the principal identity with the contents of `<psc:PrincipalSelection>`, if received in the authentication request.

Identity Providers processing a request containing a `SignMessage` extension SHALL meet the following requirements (in addition to other general requirements associated with requests from signature services):

- If the `MustShow` attribute of the `SignMessage` extension is set, the Identity Provider MUST fail with an error response if the message, for some reason, can not be displayed to the user.
- The Identity Provider MUST display the sign message to the user in a manner that is consistent with the data format of the sign message. If necessary, the Identity Provider MUST process defined filtering rules on the message. If the present message format is not supported or the sign message for any reason cannot be displayed in a proper manner, the Identity Provider must return an error response.
- If authentication and sign message confirmation by the user was successful, the Identity Provider MUST include the `signMessageDigest` attribute (see section 3.2.4 of [EidAttributes]) in the assertion that is issued.
- The Identity Provider MUST NOT return an assertion without performing an authentication process consistent with the requested authentication context which includes display of a sign message, even if the request has no present `ForceAuthn` attribute or includes a `ForceAuthn` attribute set to the value `false`.

7.1.2. Requesting SCAL2 Signature Activation Data

The type of signature requested in a signature request is, according to [EidDSS_Profile], specified by the `CertType` attribute of the `<CertRequestProperties>` element. When the value of this attribute is set to `QC/SSCD`, the requested signature is a Qualified Signature created in a Qualified Signature Creation Device (QSCD). To achieve this level of signature the Authentication Request MUST include a request for Signature Activation Data (SAD) for Sole Control Assurance Level 2 (SCAL2) in accordance with the "Signature Activation Protocol for Federated Signing" [SigSAP]. An authentication request message that includes this `SADRequest` extension MUST also include the `SignMessage` extension (as described above).

As pointed out in section 2.1.3 an Identity Provider that supports processing of SAD requests and issuance of SAD-attributes SHALL advertise this by declaring the service property entity category `http://id.elegnamnden.se/sprop/1.0/scal2` in its metadata. An Identity Provider that has declared this entity category MUST return a SAD attribute in an issued assertion if the corresponding `<saml2p:AuthnRequest>` message contains the `<sap:SADRequest>` extension.

A SAD returned from the Identity Provider MUST have a signature which can be verified using a certificate from the Identity Provider's metadata entry. The signature algorithm used to sign the SAD MUST be equivalent to the algorithm used to sign the responses and assertions from the Identity Provider.

Verification of a received SAD-attribute MUST follow the verification rules specified in section 3.2.3 of [SigSAP].

7.2. Authentication Responses

By including the `signMessageDigest` attribute (see section 3.2.4 of [EidAttributes]) in the SAML assertion, the Identity Provider asserts that it has successfully displayed the sign message received in the request for the user and that the user has accepted to sign under the context of this sign message. This attribute MUST be included in the issued assertion if a sign message was displayed for, and accepted by, the user.

8. Cryptographic Algorithms

This section lists the requirements for crypto algorithm support for being compliant with this profile.

For signature and encryption keys the following requirements apply:

- RSA public keys MUST be at least 2048 bits in length. 3072 bits or more is RECOMMENDED.
- EC public keys MUST be at least 256 bits in length (signature only).
  - The curves NIST Curve P-256, NIST Curve P-384 and NIST Curve P-521 MUST be supported ([RFC5480]).

Services conforming to this profile MUST support the mandatory algorithms below, and SHOULD support the algorithms listed as optional.

The sender of a secure message MUST NOT use an algorithm that is not listed as mandatory in the sections below, unless it is explicitly declared by the peer in its metadata (see [SAML2MetaAlgSupport]).

A service processing a message in which an algorithm not listed below has been used MUST refuse to accept the message and respond with an error, unless this algorithm has been declared as preferred or supported by the service in its metadata entry.

> This profile does not specify a complete list of blacklisted algorithms. However, there is a need to explicitly point out that the commonly used algorithms SHA-1 for digests and RSA PKCS#1 v1.5 for key transport are considered broken and SHOULD not be used or accepted.

The algorithms below are defined in [XMLEnc], [XMLDSig] and [RFC4051].

8.1. Digest Algorithms

Mandatory:

- SHA-256, `http://www.w3.org/2001/04/xmlenc#sha256`

Optional:

- SHA-384, `http://www.w3.org/2001/04/xmldsig-more#sha384`
- SHA-512, `http://www.w3.org/2001/04/xmlenc#sha512`

8.2. Signature Algorithms

Mandatory:

- RSA-SHA256, `http://www.w3.org/2001/04/xmldsig-more#rsa-sha256`
- ECDSA-SHA256, `http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256`

Optional:

- RSA-SHA384, `http://www.w3.org/2001/04/xmldsig-more#rsa-sha384`
- RSA-SHA512, `http://www.w3.org/2001/04/xmldsig-more#rsa-sha512`
- ECDSA-SHA384, `http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha384`
- ECDSA-SHA512, `http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha512`

8.3. Block Encryption Algorithms

Mandatory:

- AES128-CBC, `http://www.w3.org/2001/04/xmlenc#aes128-cbc`
- AES192-CBC, `http://www.w3.org/2001/04/xmlenc#aes192-cbc`
- AES256-CBC, `http://www.w3.org/2001/04/xmlenc#aes256-cbc`

Optional:

- AES128-GCM, `http://www.w3.org/2009/xmlenc11#aes128-gcm`
- AES192-GCM, `http://www.w3.org/2009/xmlenc11#aes192-gcm`
- AES256-GCM, `http://www.w3.org/2009/xmlenc11#aes256-gcm`

Note: The AES-CBC algorithm has been reported to have vulnerabilities, but these weaknesses can not be exploited when using signed SAML messages. Also, the support for AES-GCM in SAML software is not sufficiently spread at the time of writing. Therefore, the current version of this profile specifies AES-CBC as the default block encryption algorithm.

Services supporting the AES-GCM algorithm SHOULD indicate this in its metadata.

```
...
<md:KeyDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" use="encryption">
  <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    <ds:X509Data>
      <ds:X509Certificate>MIIE+DC...SpWg==</ds:X509Certificate>
    </ds:X509Data>
  </ds:KeyInfo>
  <md:EncryptionMethod Algorithm="http://www.w3.org/2009/xmlenc11#aes256-gcm"/>
  <md:EncryptionMethod Algorithm="http://www.w3.org/2009/xmlenc11#aes128-gcm"/>
  ...
</md:KeyDescriptor>
```

Example of how a service announces that it has support for AES-GCM and wishes peers to use these algorithms when encrypting for the service.

8.4. Key Transport Algorithms

Mandatory:

- RSA-OAEP-MGF1P, `http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p`
  - Key transport digest: SHA-1 (`http://www.w3.org/2000/09/xmldsig#sha1`)

For interoperability reasons the current version of this profile specifies the RSA-OAEP-MGF1P algorithm as the default key transport algorithms. This algorithm's padding method defaults to the use of SHA-1 as a digest algorithm and also uses the "MGF1 with SHA-1" mask generation function*.

Optional:

- RSA-OAEP-MGF1P, `http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p`
  - Key transport digest: Any of the digest methods listed as mandatory or optional in section 8.1.

A service wishing to receive encrypted messages where SHA-1 is not used as the key transport digest SHOULD indicate this in its metadata using the `<md:EncryptionMethod>` element as the example below.

```
...
<md:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p">
  <ds:DigestMethod Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"/>
</md:EncryptionMethod>
...
```

Example of how a service announces that it wishes that the peer uses SHA-256 as the key transport digest when encrypting using RSA-OAEP-MGF1P.

> [*]: Note that the use if SHA-1 in this context (both as digest algoritm and as mask generation function) is limited to providing randomness of padding data and as a hash over optional OAEP parameter data which typically is an empty string. It is not used as a hash function to assert the integrity of the encrypted data. No weaknesses of SHA-1 is is known to be relevant to its use in this context.

9. Normative References

[RFC2119]

> Bradner, S., Key words for use in RFCs to Indicate Requirement Levels, March 1997.

[XMLEnc]

> D. Eastlake et al. XML Encryption Syntax and Processing. W3C Recommendation, April 2013.

[XMLDSig]

> D. Eastlake et al. XML-Signature Syntax and Processing, Version 1.1. W3C Recommendation, April 2013.

[SAML2Core]

> OASIS Standard, Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0, March 2005.

[SAML v2.0 Errata 05]

> SAML Version 2.0 Errata 05. 01 May 2012. OASIS Approved Errata.

[SAML2Bind]

> OASIS Standard, Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0, March 2005.

[SAML2Prof]

> OASIS Standard, Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0, March 2005.

[SAML2Meta]

> OASIS Standard, Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0, March 2005.

[SAML2Sec]

> Security and Privacy Considerations for the OASIS Security Assertion Markup Language (SAML) V2.0, March 2005.

[SAML2IAP]

> SAML V2.0 Identity Assurance Profiles Version 1.0, 05 November 2010.

[SAML2MetaIOP]

> OASIS Standard, SAML V2.0 Metadata Interoperability Profile Version 1.0, October 2019.

[SAML2MetaUI]

> OASIS Draft, SAML V2.0 Metadata Extensions for Login and Discovery User Interface Version 1.0, April 2012

[SAML2MetaAttr]

> OASIS Committee Specification, SAML V2.0 Metadata Extension for Entity Attributes Version 1.0, August 2009.

[SAML2MetaAlgSupport]

> SAML v2.0 Metadata Profile for Algorithm Support Version 1.0, 21 February 2011.

[EntCat]

> RFC8409 - The Entity Category Security Assertion Markup Language (SAML) Attribute Types.

[IdpDisco]

> OASIS Committee Specification, Identity Provider Discovery Service Protocol and Profile, March 2008.

[SAML2SubjIdAttr]

> OASIS Committee Specification, SAML V2.0 Subject Identifier Attributes Profile Version 1.0, January 2019.

[SAML2HokProf]

> OASIS Committee Specification, SAML V2.0 Holder-of-Key Web Browser SSO Profile Version 1.0, August 2010.

[SAML2HokAP]

> OASIS Committee Specification, SAML V2.0 Holder-of-Key Assertion Profile Version 1.0, January 2010.

[EidTillit]

> Tillitsramverk för Svensk e-legitimation - 2018-158

[EidRegistry]

> Swedish eID Framework - Registry for identifiers.

[EidAttributes]

> Attribute Specification for the Swedish eID Framework.

[EidEntCat]

> Entity Categories for the Swedish eID Framework.

[EidDSS]

> DSS Extension for Federated Central Signing Services.

[EidDSS_Profile]

> Implementation Profile for Using OASIS DSS in Central Signing Services.

[SigSAP]

> Signature Activation Protocol for Federated Signing.

[PrincipalSel]

> Principal Selection in SAML Authentication Requests.

[RFC4051]

> IETF RFC 4051, Additional XML Security Uniform Resource Identifiers, April 2005.

[RFC5480]

> IETF RFC 5480, Elliptic Curve Cryptography Subject Public Key Information, March 2009.

10. Changes between versions

Changes between version 1.6 and 1.7:

- Sections 2.1.2 and 2.1.3 were updated with clarifications for how a Service Provider declares requested attributes and how an Identity Provider declares its ability to deliver attributes.
- Section 6.2.1, "Attribute Release and Consuming Rules", was updated with a privacy requirement that tells that an Identity Provider must not release identity attributes not requested by the Service Provider.
- Section 2.1.3.1, "Declaring Authorized Scopes", was introduced in order to define how an Identity Provider declares authorized scopes. Section 6.2.1, "Attribute Release and Consuming Rules", was updated with release and consumption rules for scoped attributes.
- Support for the Holder-of-key Web Browser SSO Profile has been added.
- Removed text in section 7, "Authentication for Signature", regarding the use of the deprecated sigmessage URI:s.

Changes between version 1.5 and 1.6:

- In order to facilitate algorithm interoperability between peers additions concerning "Metadata Profile for Algorithm Support" [SAML2MetaAlgSupport] was added. Section 2.1.1 was updated with a section defining how preferred algorithms are declared in metadata, and sections 5.2, 6.1 and 7.2.1 was updated with requirements for algorithm selection during signing and encryption.
- Section 5.3, "Message Content", was re-structured with sub-chapters for requested authentication contexts, scoping and principal selection.
- The `PrincipalSelection` and `RequestedPrincipalSelection` extensions were introduced to sections 2.1.3, 5.3.3 and 7.2.
- The link for the "Tillitsramverk för Svensk e-legitimation" specification was updated.
- This profile is no longer normatively dependent upon SAML2Int. Therefore, the profile has been updated with requirements that previously was implicit (due to the normative dependency to SAML2Int).
- Section 8, "Cryptographic Algorithms", was introduced in order to clearly define the algorithm requirements for services that are conformant to this profile.
- Section 2.1.1 was updated with elaborations concerning certificates in metadata.
- Section 7.2.1 was updated with a requirement that ensures that a sign message is only displayed to the intended user.
- Section 7, "Authentication for Signature", was updated where "Sign Message Authentication Context URI" were deprecated and the use of the `signMessageDigest` attribute was introduced.

Changes between version 1.4 and 1.5:

- Section 2.1.3 "Identity Providers", was extended to include requirements on metadata to signal support for the SAP (Signature Activation Protocol).
- Section 7.2, "Authentication Requests", was extended to recommend the usage of the `<saml2p:RequesterID>` element within `<saml2p:Scoping>`. The reason for this recommendation is that Identity Providers may need information about the "Signature Requestor", i.e., the Service Provider that requested the signature that caused a Signature Service to request authentication.
- Section 6.3.4, "The Authentication Statement", contained a requirement about how to process a received authentication context URI that was incorrect. This has been corrected.
- A new section, 7.2.2, "Requesting SCAL2 Signature Activation Data", was added. This amendment describes how and when to request Signature Activation Data from an Identity Provider in order to enable a signature service to operate as a Qualified Signature Creation Device (QSCD).
- Attribute release rules have been clarified in sections 2.1.2, "Service Providers" and 6.2.1, "Attribute Release Rules".
- Section 2.1.2, "Service Providers", was updated to include a requirement for Service Providers communicating with a centralized discovery service.
- Section 5.3, "Message Content", was updated to describe the use of `<saml2p:IDPEntry>` elements under the `<samp2p:Scoping>` elements for requests sent to Identity Providers acting as proxies for other Identity Providers.

- The reference to [SAML2MetaUI] was updated to the latest official version: http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-metadata-ui/v1.0/sstc-saml-metadata-ui-v1.0.html.
- The reference to [EntCat] was updated to the latest version.

Changes between version 1.3 and version 1.4:

- Version 1.3 of this profile stated that a `<saml2p:AuthnRequest>` message MUST contain an `AssertionConsumerServiceURL` attribute identifying the desired response location. It has shown that this requirement aggravates interoperability since some of the major providers of Service Provider software do not fully support this attribute. Furthermore, the requirement does increase security since an Identity Provider may only post response messages to locations registered in the `<md:AssertionConsumerService>` elements of the Service Provider metadata entry. Therefore, chapter 5.3, “Message Content”, has been changed to state that the `<saml2p:AuthnRequest>` message SHOULD contain an `AssertionConsumerServiceURL` attribute. Changes have also been made to sections 5.4.2 and 6.3.2 where processing requirements were updated.
- In section 5.3, a clarification regarding specifying more than one authentication context URI was made.
- In section 7.1, a set of authentication context URIs for the eIDAS Framework was added.
- In section 6.4, the requirement to use the sub-level status code `http://id.elegnamnden.se/status/1.0/cancel` was added. This status should be used to indicate a cancelled operation.
- In section 6.4, the status codes `http://id.elegnamnden.se/status/1.0/fraud` and `http://id.elegnamnden.se/status/1.0/possibleFraud` were introduced. Their purpose is to alert (suspected) fraudulent behaviour.
- The specification for “Discovery within the Swedish eID Framework” has been deprecated and requirements referring to this document have been updated.
- A clarification to section 5.2 was made stating that conformant Identity Providers MUST support the HTTP-POST binding.
- Section 6.2 was updated with requirements for proxy-IdP:s that are expected to include the `<saml2:AuthenticatingAuthority>` element holding the entityID of the Identity Provider that provided the authentication of the principal.

Changes between version 1.2 and version 1.3:

- This profile now extends a newer version of the SAML2Int Deployment Profile (see http://saml2int.org/profile/current/).
- Clarifications on how entity categories are represented in metadata were made to chapters: 2.1.2, 2.1.3, and 2.1.4.
- Changes were made to chapter 6.1, “Security Requirements”, where the profile now requires the entire `<saml2p:Response>` message to be signed, as compared to the previous version where the signature requirement was put on `<saml2:Assertion>` elements.
- In chapter 6.2, it is now specified that an Address attribute MUST be part of the `<saml2:SubjectConfirmationData>` element. The previous version stated SHOULD.
- Chapter 6.2.1, “Attribute Release Rules”, was introduced to clarify how the attribute release process should be handled by an issuing entity.
- A Service Provider is now obliged to explicitly specify the required Level of Assurance under which a specific authentication should be performed. This is specified in chapter 5.3, “Message Content”, and 5.4.4, “Authentication Context and Level of Assurance Handling”.
- The specification “Authentication Context Classes for Levels of Assurance for the Swedish eID Framework” has been removed from the Swedish eID Framework. The reason for this is that it was proven difficult to make use of the `<saml2:AuthnContextDecl>` element to store authentication context parameters, and that no commercial, or open source, Identity Provider software had support for this feature. [EidAttributes] now describe how the `authContextParams` attribute may be used for the same purpose, and the examples where this information was stored under the `<saml2:AuthnContextDecl>` element was removed from chapter 6.2, “Message Content”.
- Chapter 7, “Authentication for Signature”, was introduced to specify requirements regarding the process of “authentication for signature” where a *Signature Service* requests that a user performing a signature authenticates.

Changes between version 1.1 and version 1.2:

- This profile now explicitly defines requirements for the use of signed authentication request messages, see sections 2.1and 5.2.
- This profile now allows the HTTP-POST binding to be used for sending authentication request messages (see chapter 5.2, “Binding and Security Requirements”). The main reason for this is to facilitate the use of signed authentication request messages.
- In chapter 5.4, additional processing requirements for received authentication requests were added or changed. These include:
- Validation of assertion consumer addresses (5.4.1).
- Clarifications to chapter 5.4.4.
- Single Sign On processing (5.4.5).
- This profile now states that “Unsolicited response” messages are not accepted by Service Providers due to security reasons, see chapter 6.1, “Security Requirements”.
- Changes and additions in chapter 6.2, “Message Content”, for responses including:
- Clarifications about the usage of the `AuthnInstant` attribute of the `<saml2:AuthnStatement>` element.
- Specifications of the use of `<saml2:SubjectConfirmation>` in assertions.
- Clarifications on the use of audience restrictions and assertion validity.
- Chapter 6.3, “Processing Requirements”, was added. This chapter contains specifications and requirements of how a response message should be processed in order to maintain security.

Changes between version 1.0 and version 1.1:

- In chapter 5.1, “Discovery”, a reference to the specification “Discovery within the Swedish eID Framework” [Eid2Disco] was added.
- In chapter 5.4.4, a note was added that informs about the need to ensure IdP-capabilities regarding level of assurance before issuing a request.
- In chapter 6.2, “Message Content”, an example of how an Identity Provider may include an authentication context class declaration was provided.
- Some faulty references were corrected.