Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0

OASIS Draft, 136 June 2004

Document identifier: sstc-saml-profiles-2.0-draft-1


Editors: Frederick Hirsch, Nokia

Contributors: TBD

Abstract: This specification defines profiles for the use of SAML assertions and request-response messages in communications protocols and frameworks as well as attribute syntaxes for use in attribute statements.

Status: This is a Draft.

Committee members should submit comments and potential errata to the security-services@lists.oasis-open.org list. Others should submit them to the security-services-comment@lists.oasis-open.org list (to post, you must subscribe; to subscribe, send a message to security-services-comment-request@lists.oasis-open.org with "subscribe" in the body) or use other OASIS-supported means of submitting comments. The committee will publish vetted errata on the Security Services TC web page (http://www.oasis-open.org/committees/security/).

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights web page for the Security Services TC (http://www.oasis-open.org/committees/security/ipr.php).
Table of Contents

1 Introduction ........................................................................................................................................... 5
  1.1 Profile Concepts ............................................................................................................................... 5
  1.2 Notation ........................................................................................................................................... 6
2 Specification of Additional Profiles ..................................................................................................... 7
  2.1 Guidelines for Specifying Profiles .................................................................................................... 7
  2.2 Process Framework for Describing and Registering Profiles .......................................................... 7
3 Confirmation Method Identifiers .......................................................................................................... 8
  3.1 Holder of Key ................................................................................................................................... 8
  3.2 Sender Vouches ............................................................................................................................... 8
  3.3 Bearer ............................................................................................................................................. 9
4 SSO Profiles of SAML .......................................................................................................................... 10
  4.1 Web Browser SSO Profile ............................................................................................................... 10
    4.1.1 Required Information ................................................................................................................. 10
    4.1.2 Profile Overview ....................................................................................................................... 10
    4.1.3 Profile Description ..................................................................................................................... 11
      4.1.3.1 HTTP Request to Service Provider ..................................................................................... 11
      4.1.3.2 Service Provider Determines Identity Provider ............................................................... 11
      4.1.3.3 <AuthnRequest> issued by Service Provider to Identity Provider .................................... 12
      4.1.3.4 Identity Provider identifies Principal ............................................................................... 12
      4.1.3.5 Identity Provider issues Response to Service Provider ................................................... 12
      4.1.3.6 Service Provider grants or denies access to User Agent .................................................... 13
    4.1.4 Use of Authentication Request Protocol ..................................................................................... 13
      4.1.4.1 <AuthnRequest> Usage ....................................................................................................... 13
      4.1.4.2 <Response> Usage ............................................................................................................. 13
      4.1.4.3 <Response> Message Processing Rules .............................................................................. 14
      4.1.4.4 Artifact-Specific <Response> Message Processing Rules ............................................... 14
      4.1.4.5 POST-Specific Processing Rules ....................................................................................... 14
    4.1.5 Unsolicited Responses ................................................................................................................. 15
    4.1.6 Use of Metadata .......................................................................................................................... 15
  4.2 Enhanced Client and Proxy (ECP) Profile .......................................................................................... 15
    4.2.1 Required Information ................................................................................................................. 16
    4.2.2 Preliminaries .............................................................................................................................. 16
    4.2.3 Step 1: Accessing the Service Provider: ECP>SP .................................................................... 17
    4.2.4 Steps 2,3: SOAP Message containing <AuthnRequest>: SP>ECP>IDP .................................... 17
1 Introduction

This document specifies profiles for the use of SAML assertions and request-response messages in communications protocols and frameworks.

A separate specification [SAMLCore] defines the SAML assertions and request-response messages themselves and another [SAMLBind] defines protocol bindings.

The following sections define profiles of SAML that are sanctioned by the OASIS Security Services Technical Committee.

A web browser-based profile of the Authentication Request protocol is defined to support web single sign-on (SSO), supporting Scenario 1-1 of the SAML requirements document [SAMLReqs]:

- The browser profile of SAML

An additional web SSO profile is defined to support enhanced clients:

- The ECP profile of SAML

Binding-specific profiles of protocols defined in [SAMLCore] are also defined:

- Single Logout Profiles
- NameID Management Profiles
- NameID Mapping Profile
- Artifact Request Profile

<saml:Attribute> and <samlp:AttributeQuery> elements provide a flexible means of specifying attributes and attribute queries within SAML 2.0. However [SAMLCore], defines only a limited set of attribute names and syntaxes. It is further suggested that additional attribute names and syntaxes may be defined by attribute profiles.

Section XYZ includes guidelines for defining such profiles. It also describes three specific attribute profiles:

(1) Basic attribute profile (XML schema Names with values drawn from Section 3 of [XML-Schema-Part2])
(2) X.500/LDAP,
(3) UUID.

Some additional profiles that have been published outside the Security Services Technical Committee are:

- The OASIS Web Services Security Technical Committee has produced a draft “SAML token profile” of the WSS specification [WSS-SAML], which describes how to use SAML assertions to secure a SOAP message.
- The Liberty Alliance Project [Liberty] has produced a set of additional profiles for the use of SAML to secure web services.

1.1 Profile Concepts

One type of SAML profile defines a set of constraints on the use of a general SAML protocol or assertion capability for a particular environment or context of use. Profiles of this nature may constrain optionality, require the use of specific SAML functionality (e.g. Conditions, bindings), and in other respects define the processing rules to be followed by profile actors.
Another type of SAML profile outlines a set of rules describing how to embed SAML assertions into and extract them from a framework or protocol. Such a profile describes how SAML assertions are embedded in or combined with other objects (for example, files of various types, or protocol data units of communication protocols) by an originating party, communicated from the originating party to a receiving party, and subsequently processed at the destination. A particular set of rules for embedding SAML assertions into and extracting them from a specific class of <FOO> objects is termed a <FOO> profile of SAML.

For example, a SOAP profile of SAML describes how SAML assertions can be added to SOAP messages, how SOAP headers are affected by SAML assertions, and how SAML-related error states should be reflected in SOAP messages.

The intent of this specification is to specify a selected set of profiles in sufficient detail to ensure that independently implemented products will interoperate.

For other terms and concepts that are specific to SAML, refer to the SAML glossary [SAMLGloss].

1.2 Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in IETF RFC 2119 [RFC2119].

Lists of productions or other normative code appear like this.

Example code listings appear like this.

Note: Non-normative notes and explanations appear like this.

Conventional XML namespace prefixes are used throughout this specification to stand for their respective namespaces as follows, whether or not a namespace declaration is present in the example:

- The prefix saml: stands for the SAML assertion namespace [SAMLCore].
- The prefix samlp: stands for the SAML request-response protocol namespace [SAMLCore].
- The prefix md: stands for the SAML metadata namespace [XX].
- The prefix ds: stands for the W3C XML Signature namespace, http://www.w3.org/2000/09/xmldsig# [XMLSig].

This specification uses the following typographical conventions in text: <SAMLElement>, <ns:ForeignElement>, Attribute, Datatype, OtherCode. In some cases, angle brackets are used to indicate non-terminals, rather than XML elements; the intent will be clear from the context.
2 Specification of Additional Profiles

This specification defines a selected set of profiles, but others will possibly be developed in the future. It is not possible for the OASIS Security Services Technical Committee to standardize all of these additional profiles for two reasons: it has limited resources and it does not own the standardization process for all of the technologies used. The following sections offer guidelines for specifying profiles and a process framework for describing and registering them.

2.1 Guidelines for Specifying Profiles

This section provides a checklist of issues that MUST be addressed by each profile.

1. Describe the set of interactions between parties involved in the profile. Any restrictions on applications used by each party and the protocols involved in each interaction must be explicitly called out.

2. Identify the parties involved in each interaction, including how many parties are involved and whether intermediaries may be involved.

3. Specify the method of authentication of parties involved in each interaction, including whether authentication is required and acceptable authentication types.

4. Identify the level of support for message integrity, including the mechanisms used to ensure message integrity.

5. Identify the level of support for confidentiality, including whether a third party may view the contents of SAML messages and assertions, whether the profile requires confidentiality, and the mechanisms recommended for achieving confidentiality.

6. Identify the error states, including the error states at each participant, especially those that receive and process SAML assertions or messages.

7. Identify security considerations, including analysis of threats and description of countermeasures.

8. Identify SAML confirmation method identifiers defined and/or utilized by the profile.

9. Identify relevant SAML metadata defined and/or utilized by the profile.

2.2 Process Framework for Describing and Registering Profiles

For any new profile to be interoperable, it needs to be openly specified. The OASIS Security Services Technical Committee will maintain a registry and repository of submitted profiles titled “Additional Bindings and Profiles” at the SAML website [SAMLWeb] in order to keep the SAML community informed. The committee will also provide instructions for submission of profiles by OASIS members.

When a profile is registered, the following information MUST be supplied:

1. Identification: Specify a URI that uniquely identifies this profile.

2. Contact information: Specify the postal or electronic contact information for the author of the profile.

3. Description: Provide a text description of the profile. The description SHOULD follow the guidelines described in Section 2.1.

4. Updates: Provide references to previously registered profiles that the current entry improves or obsoletes.
3 Confirmation Method Identifiers

The SAML assertion and protocol specification [SAMLCore] defines the <SubjectConfirmation> element as a Method plus optional <SubjectConfirmationData>. The <SubjectConfirmation> element SHOULD be used by the relying party to confirm that the request or message came from a System Entity that corresponds to the subject of the assertion.

The Method attribute indicates the specific method that the relying party should use to make this determination. This may or may not have any relationship to an authentication that was performed previously. Unlike the authentication context, the subject confirmation method will often be accompanied by additional information, such as a certificate or key, in the <SubjectConfirmationData> element that will allow the relying party to perform the necessary verification. A common set of attributes are also defined and MAY be used to constrain the conditions under which the verification can take place.

It is anticipated that profiles will define and use several different values for <ConfirmationMethod>, each corresponding to a different SAML usage scenario. The following methods are defined for use by profiles defined within this specification and other profiles that find them useful.

3.1 Holder of Key

URI: urn:oasis:names:tc:SAML:2.0:cm:holder-of-key

One or more <ds:KeyInfo> elements MUST be present within the <SubjectConfirmationData> element. An xsi:type attribute MAY be present in the <SubjectConfirmationData> element and MUST be set to saml:KeyInfoConfirmationDataType (the Qname prefix is arbitrary but must reference the SAML assertion namespace).

As described in [XMLSig], each <ds:KeyInfo> element holds a key or information that enables an application to obtain a key. The holder of a specified key is considered to be the subject of the assertion by the asserting party.

Note that in accordance with [XMLSig], each <ds:KeyInfo> element MUST identify a single cryptographic key. Multiple keys MAY be identified with separate <ds:KeyInfo> elements.

Example: The holder of the key named "By-Tor" or the holder of the key named "Snow Dog" can confirm itself as the subject.

```xml
<SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:holder-of-key">
  <SubjectConfirmationData xsi:type="saml:KeyInfoConfirmationDataType">
    <ds:KeyInfo>
      <ds:KeyName>By-Tor</ds:KeyName>
    </ds:KeyInfo>
    <ds:KeyInfo>
      <ds:KeyName>Snow Dog</ds:KeyName>
    </ds:KeyInfo>
  </SubjectConfirmationData>
</SubjectConfirmation>
```

3.2 Sender Vouches

URI: urn:oasis:names:tc:SAML:2.0:cm:sender-vouches

Indicates that no other information is available about the context of use of the assertion. The relying party SHOULD utilize other means to determine if it should process the assertion further, subject to optional constraints on confirmation using the attributes that MAY be present in the <SubjectConfirmationData> element, as defined by [SAMLCore].
3.3 Bearer

**URI:** urn:oasis:names:tc:SAML:2.0:cm:bearer

The subject of the assertion is the bearer of the assertion, subject to optional constraints on confirmation using the attributes that MAY be present in the `<SubjectConfirmationData>` element, as defined by [SAMLCore].

**Example:** The bearer of the assertion can confirm itself as the subject, provided the assertion is delivered in a message sent to "https://www.serviceprovider.com/saml/consumer" before 1:37 PM GMT on March 19th, 2004, in response to a request with ID "_1234567890".

```xml
<SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
  <SubjectConfirmationData InResponseTo="_1234567890"
                          Recipient="https://www.serviceprovider.com/saml/consumer"
                          NotOnOrAfter="2004-03-19T13:27:00Z"
  />
</SubjectConfirmation>
```
4 SSO Profiles of SAML

4.1 Web Browser SSO Profile

In the scenario supported by the web browser SSO profile, a web user either accesses a resource at a
service provider, or accesses an identity provider such that the service provider and desired resource are
understood or implicit. The web user authenticates (or has already authenticated) to the identity provider,
which then produces an authentication assertion (possibly with input from the service provider) and the
service provider consumes the assertion to establish a security context for the web user. During this
process, a name identifier might also be established between the providers for the principal, subject to the
parameters of the interaction and the consent of the parties.

To implement this scenario, a profile of the SAML Authentication Request protocol is used, in conjunction
with the HTTP Redirect, HTTP POST and HTTP Artifact bindings.

It is assumed that the user is using a standard commercial browser and can authenticate to the identity
provider by some means outside the scope of SAML.

4.1.1 Required Information


Contact information: security-services-comment@lists.oasis-open.org

SAML Confirmation Method Identifiers: The SAML 2.0 "bearer" confirmation method identifier is used
by this profile. The following RECOMMENDED identifier has been assigned to this confirmation method:

urn:oasis:names:tc:SAML:2.0:cm:bearer

Description: Given below.

Updates: SAML 1.1 browser artifact and POST profiles and bearer confirmation method.

4.1.2 Profile Overview

The following figure illustrates the basic template for achieving SSO:

<need figure>

The following steps are described by the profile. Within an individual step, there may be one or more
actual message exchanges depending on the binding used for that step and other implementation-
dependent behavior.

1. HTTP Request to Service Provider

In step 1, the principal, via an HTTP User Agent, makes an HTTP request for a secured resource
at the service provider without a security context.

2. Service Provider Determines Identity Provider

In step 2, the service provider obtains the location of an endpoint at an identity provider for the
authentication request protocol that supports its preferred binding. The means by which this is
accomplished is implementation-dependent. The service provider MAY use the SAML identity
provider discovery profile described in 4.3.

3. <AuthnRequest> issued by Service Provider to Identity Provider
In step 3, the service provider issues an `<AuthnRequest>` message to be delivered by the user agent to the identity provider. Either the HTTP Redirect, HTTP POST, or HTTP Artifact binding can be used to transfer the message to the identity provider through the user agent.

4. Identity Provider identifies Principal

In step 4, the principal is identified by the identity provider by some means outside the scope of this profile. This may require a new act of authentication, or it may reuse an existing authenticated session.

5. Identity Provider issues Response to Service Provider

In step 5, the identity provider issues a `<Response>` message to be delivered by the user agent to the service provider. Either the HTTP POST, or HTTP Artifact binding can be used to transfer the message to the service provider through the user agent. The message may indicate an error, or will include (at least) an authentication assertion. The HTTP Redirect binding MUST NOT be used, as the response will typically exceed the URL length permitted by most user agents.

6. Service Provider grants or denies access to Principal

In step 6, having received the response from the identity provider, the service provider can respond to the principal's user agent with its own error, or can establish its own security context for the principal and return the requested resource.

Note that an identity provider can initiate this profile at step 5 and issue a `<Response>` message to a service provider without the preceding steps.

4.1.3 Profile Description

If the profile is initiated by the service provider, start with section 4.1.3.1. If initiated by the identity provider, start with section 4.1.3.5. In the descriptions below, the following are referred to:

Intersite Transfer Service

This is an endpoint at either the identity provider or service provider which responds to an unspecified HTTP request from a user agent with a SAML protocol message using either the HTTP Redirect, HTTP POST, or HTTP Artifact binding.

Single Sign-On Service

This is the authentication request protocol endpoint at the identity provider to which the `<AuthnRequest>` message (or artifact representing it) is delivered by the user agent.

Assertion Consumer Service

This is the authentication request protocol endpoint at the service provider to which the `<Response>` message (or artifact representing it) is delivered by the user agent.

4.1.3.1 HTTP Request to Service Provider

If the first access is to the service provider, an arbitrary request for a resource can initiate the profile. There are no restrictions on the form of the request. The service provider is free to use any means it wishes to associate the subsequent interactions with the original request. Each of the -bindings provide a RelayState mechanism that the service provider MAY use to associate the profile exchange with the original request. The service provider SHOULD reveal as little of the request as possible in the RelayState value unless the use of the profile does not require such privacy measures.

4.1.3.2 Service Provider Determines Identity Provider

This step is implementation-dependent. The service provider MAY use the SAML identity provider discovery profile, described in section 4.3. The service provider MAY also choose to redirect the user.
agent to another service that is able to determine an appropriate identity provider. In such a case, the service provider may issue an `<AuthnRequest>` (as in the next step) to this service to be relayed to the identity provider, or it may rely on the intermediary service to issue an `<AuthnRequest>` message on its behalf.

4.1.3.3 `<AuthnRequest>` issued by Service Provider to Identity Provider

Once an identity provider is selected, the location of its single sign-on service is determined, based on the SAML binding chosen by the service provider for sending the `<AuthnRequest>`. Metadata MAY be used for this purpose. The user agent is directed to send an HTTP request to the service provider's intersite transfer service. The HTTP response to that request MUST contain an `<AuthnRequest>` message or an artifact, depending on the SAML binding used, to be delivered to the identity provider's single sign-on service. The exact format of this HTTP response and the subsequent HTTP request is defined by the SAML binding used. Profile-specific rules for the contents of the `<AuthnRequest>` message are included in section 4.1.4.1.

It is RECOMMENDED that the HTTP exchanges in this step be made over either SSL 3.0 ([SSL3]) or TLS 1.0 ([RFC2246]) to maintain confidentiality and message integrity. The `<AuthnRequest>` message MAY be signed, if authentication of the request issuer is required. The HTTP Artifact binding also provides for an alternate means of authenticating the request issuer when the artifact is dereferenced.

The identity provider MUST process the `<AuthnRequest>` message as described in [SAMLCore]. This may constrain the subsequent interactions with the user agent, for example if the IsPassive attribute is included.

4.1.3.4 Identity Provider identifies Principal

At any time during the previous step or subsequent to it, the identity provider MUST establish the identity of the principal (unless it returns an error to the service provider). The ForceAuthn `<AuthnRequest>` attribute, if present with a value of true, obligates the identity provider to freshly establish this identity, rather than relying on an existing session it may have with the principal. Otherwise, and in all other respects, the identity provider may use any means to authenticate the user agent, subject to any requirements included in the `<AuthnRequest>` in the form of `<RequestedAuthnContext>`.

4.1.3.5 Identity Provider issues Response to Service Provider

Regardless of the success or failure of the `<AuthnRequest>`, the identity provider SHOULD direct the user agent to send an HTTP request to its intersite transfer service. The HTTP response to this request MUST contain a `<Response>` message or an artifact, depending on the SAML binding used, to be delivered to the service provider's assertion consumer service. The exact format of this HTTP response and the subsequent HTTP request to the assertion consumer service is defined by the SAML binding used. Profile-specific rules on the contents of the `<Response>` are included in section 4.1.4.2.

The location of the assertion consumer service MAY be determined using metadata. The identity provider MUST have some means to establish that this location is in fact controlled by the service provider. A service provider MAY indicate the SAML binding and the specific assertion consumer service to use in its `<AuthnRequest>` and the identity provider MUST honor them if it can.

It is RECOMMENDED that the HTTP requests in this step be made over either SSL 3.0 ([SSL3]) or TLS 1.0 ([RFC2246]) to maintain confidentiality and message integrity. The `<Assertion>` element(s) in the `<Response>` MUST be signed, if the HTTP POST binding is used, and MAY be signed if the HTTP-Artifact binding is used.

The service provider MUST process the `<Response>` message and any enclosed `<Assertion>` elements as described in [SAMLCore].
4.1.3.6 Service Provider grants or denies access to User Agent

To complete the profile, the service provider processes the <Response> and <Assertion>(s) and grants or denies access to the resource. The service provider MAY establish a security context with the user agent using any session mechanism it chooses. Any subsequent use of the <Assertion>(s) provided are at the discretion of the service provider and other relying parties, subject to any restrictions on use contained within them.

4.1.4 Use of Authentication Request Protocol

This profile is based on the Authentication Request protocol defined in [SAMLCore]. In the nomenclature of actors enumerated in section 3.4 of that document, the service provider is the request issuer and the relying party, and the principal is the presenter, requested subject, and confirming subject. There may be additional relying parties or confirming subjects at the discretion of the identity provider (see below).

4.1.4.1 <AuthnRequest> Usage

A service provider MAY include any message content described in [SAMLCore], section 3.4.1. All processing rules are as defined in [SAMLCore]. If the identity provider cannot or will not satisfy the request, it MUST respond with a <Response> message containing an appropriate error status code or codes.

Note that the service provider MAY include a <Subject> element in the request that names the actual identity about which it wishes to receive an assertion. This element MUST NOT contain any <SubjectConfirmation> elements. If the identity provider does not recognize the principal as that identity, then it MUST respond with a <Response> message containing an error status and no assertions.

The <AuthnRequest> message MAY be signed (as directed by the SAML binding used). If the HTTP Artifact binding is used, authentication of the parties is OPTIONAL and any mechanism permitted by the binding MAY be used.

Note that if the <AuthnRequest> is not authenticated and/or integrity protected, the information in it MUST NOT be trusted except as advisory. The identity provider MUST insure that any <AssertionConsumerServiceURL> or <AssertionConsumerServiceID> elements in the request are verified as belonging to the service provider to whom the response will be sent. Failure to do so can result in a man-in-the-middle attack.

4.1.4.2 <Response> Usage

If the identity provider wishes to return an error, it MUST NOT include any assertions in the <Response> message. Otherwise, if the request is successful (or if the response is not associated with a request), the <Response> element MUST conform to the following:

- It MUST contain at least one <Assertion>.
- The set of one or more assertions MUST contain at least one <AuthnStatement> that reflects the authentication of the principal to the identity provider.
- At least one assertion containing an <AuthnStatement> MUST contain a <Subject> element with at least one <SubjectConfirmation> element containing a Method of urn:oasis:names:tc:SAML:2.0:cm:bearer.
- Any bearer <SubjectConfirmationData> elements MUST contain a Recipient attribute containing the service provider's assertion consumer service URL and a NotOnOrAfter attribute that limits the window during which the assertion can be delivered. It MAY contain an IPAddress attribute limiting the client address from which the assertion can be delivered. It MUST NOT contain a NotBefore attribute. If the containing message is in response to an...
<AuthnRequest>, then the InResponseTo attribute MUST match the request's ID.

- Other statements and confirmation methods MAY be included in the assertion(s) at the
discretion of the identity provider.

- The assertion(s) containing a bearer subject confirmation MUST contain an
  <AudienceRestrictionCondition> including the service provider's unique identifier as
  an <Audience>.

- Other conditions (and other <Audience> elements) MAY be included as requested by the
  service provider or at the discretion of the identity provider. (Of course, any such conditions
  MUST be understood by and accepted by the service provider in order for the assertion to be
  considered valid.)

4.1.4.3 <Response> Message Processing Rules

Regardless of the SAML binding used, the service provider MUST:

- verify any signatures present on the assertion(s)
- verify that the Recipient attribute in any bearer <SubjectConfirmationData> matches
  the assertion consumer service URL to which the <Response> or artifact was delivered
- verify that the NotOnOrAfter attribute in any bearer <SubjectConfirmationData> has
  not passed, subject to allowable clock skew between the providers
- verify that the InResponseTo attribute in the bearer <SubjectConfirmationData> equals
  the ID of its original <AuthnRequest> message, unless the response is unsolicited (see
  section 4.5) in which case the attribute MUST NOT be present
- verify that any assertions relied upon are valid in other respects

If any bearer <SubjectConfirmationData> includes an IPAddress attribute, the service provider
MAY check the user agent's IP address against it.

Any assertion which is not valid, or whose subject confirmation requirements cannot be met SHOULD be
discarded and SHOULD NOT be used to establish a security context for the principal.

- If an <AuthnStatement> used to establish a security context for the principal contains a
  SessionNotOnOrAfter attribute, the security context SHOULD be discarded once this time is reached,
  unless the service provider reestablishes the principal's identity by repeating the use of this profile.

4.1.4.4 Artifact-Specific <Response> Message Processing Rules

If the HTTP Artifact binding is used to deliver the <Response>, the dereferencing of the artifact using the
Artifact protocol MUST be mutually authenticated, integrity protected, and confidential.

The identity provider MUST ensure that only the service provider to whom the <Response> message has
been issued is given the message as the result of an <ArtifactRequest>.

Either the SAML binding used to dereference the artifact or message signatures can be used to
authenticate the parties and protect the messages.

4.1.4.5 POST-Specific Processing Rules

If the HTTP POST binding is used to deliver the <Response>, the enclosed assertion(s) MUST be
signed.

The service provider MUST ensure that bearer assertions are not replayed, by maintaining the set of used
ID values for the length of time for which the assertion would be considered valid based on the
4.1.5 Unsolicited Responses

An identity provider may initiate this profile by delivering an unsolicited `<Response>` message to a service provider.

An unsolicited `<Response>` MUST NOT contain an `InResponseTo` attribute, nor should any bearer `<SubjectConfirmationData>` elements. If metadata is used, the `<Response>` or artifact SHOULD be delivered to the `<md:AssertionConsumerService>` endpoint of the service provider labeled with the `isDefault` attribute.

Of special mention is that the identity provider SHOULD include a binding-specific "RelayState" parameter that indicates, based on mutual agreement with the service provider, how to handle subsequent interactions with the user agent. This MAY be the URL of a resource at the service provider.

4.1.6 Use of Metadata

TODO[SAMLMeta] defines an endpoint element, `<md:SingleSignOnService>`, to describe the location(s) to which a service provider may send requests to an identity provider using this profile. The `Binding` attribute of the element MUST contain the identifier of the protocol binding supported by that endpoint. The `ResponseLocation` attribute MUST be omitted from endpoints that support this profile.

The `<md:IDPDescriptor>` element's `WantAuthnRequestsSigned` attribute MAY be used by an identity provider to document a requirement that requests be signed. The `<md:SPDescriptor>` element's `AuthnRequestsSigned` attribute MAY be used by a service provider to document the intention to sign all of its requests.

The service provider MAY document the key(s) used to sign its requests with `<md:KeyDescriptor>` element(s) with a `use` attribute of `sign`.

The endpoint element `<md:AssertionConsumerService>` is used to describe the location(s) to which an identity provider may send responses to a service provider using this profile. The `Binding` attribute of the element MUST contain the identifier of the protocol binding supported by that endpoint. The `ResponseLocation` attribute MUST be omitted from endpoints that support this profile (as its `Location` attribute is intended as a response location). The `ID` attribute MUST be specified to distinguish the possible endpoints that MAY be specified by reference in the `<AuthnRequest>` message. The `isDefault` attribute MAY be used to specify the endpoint to use if not specified in a request.

The `<md:SPDescriptor>` element's `WantAssertionsSigned` attribute MAY be used by a service provider to document a requirement that assertions delivered with this profile be signed. This is in addition to any requirements for signing imposed by the use of a particular binding.

If the request or response message is delivered using the HTTP Artifact binding, the artifact issuer MUST provide at least one `<md:ArtifactLookupService>` endpoint element in its metadata. The `Binding` attribute of the element MUST contain the identifier of the protocol binding supported by that endpoint. The `ResponseLocation` attribute MUST be omitted from these endpoints.

The `<md:AttributeRequesterDescriptor>` element MAY be used to document the service provider's need or desire for SAML attributes to be delivered along with authentication information. The actual inclusion of attributes is of course at the discretion of the identity provider.

4.2 Enhanced Client and Proxy (ECP) Profile

In the scenario supported by the enhanced client and proxy profile, a user of an enhanced client or proxy either accesses a resource at a service provider, or accesses an identity provider such that the service provider and desired resource are understood or implicit. The user authenticates (or has already...
authenticated) to the identity provider, which then produces an authentication assertion (possibly with input
from the service provider) and the service provider consumes the assertion to establish a security context
for the user. During this process, a name identifier might also be established between the providers for the
principal, subject to the parameters of the interaction and the consent of the parties.

To implement this scenario, a profile of the SAML Authentication Request protocol is used, in conjunction
with the Reverse-SOAP binding.

It is assumed that the user is using an enhanced client or proxy (see below) and can authenticate to the
identity provider by some means outside the scope of SAML.

4.2.1 Required Information

**Identification:** urn:oasis:names:tc:SAML:2.0:profiles:ecp

**Contact information:** security-services-comment@lists.oasis-open.org

**SAML Confirmation Method Identifiers:** The SAML 2.0 "bearer" confirmation method identifier is used
by this profile. The following RECOMMENDED identifier has been assigned to this confirmation method:

urn:oasis:names:tc:SAML:2.0:cm:bearer

**Description:** Given below.

**Updates:** None.

4.2.2 Preliminaries

The Enhanced Client and Proxy (ECP) profile specifies interactions between enhanced clients and/or
proxies, service providers, and identity providers. An enhanced client or proxy (ECP) is a client or proxy
that:

1. Has, or knows how to obtain, knowledge about the identity provider that the principal associated
   with the client wishes to use with the service provider.
   - This allows a service provider to make an authentication request to such a client without the
     need to know or discover the appropriate identity provider.
2. Is able to use a reverse SOAP (PAOS) binding as profiled here for an authentication request and
   response.
   - This enables a service provider to obtain an authentication assertion from a client that is not
     necessarily directly addressable and not necessarily continuously available.
   - It leverages the benefits of SOAP while using a well-defined exchange pattern and profile to
     enable interoperability.
   - The enhanced client may be viewed as a SOAP intermediary between the service provider and
     the identity provider.

The enhanced client may be a browser or some other user agent that supports the functionality described
in this profile. An enhanced proxy is an HTTP proxy (typically a WAP gateway) that emulates an enhanced
client. Unless stated otherwise, all statements referring to enhanced clients are to be understood as
statements about both enhanced clients as well as enhanced client proxies.

Since the enhanced client sends and receives messages in the body of HTTP requests and responses, it
has no arbitrary restrictions on the size of the protocol messages.

This profile leverages the Reverse SOAP binding [SAMLBind]. Implementers of this profile MUST follow
the rules for HTTP indications of PAOS support specified in that binding, in addition to those specified in
this profile. This specification profiles a PAOS SOAP header block conveyed between the HTTP
responder and the ECP but does not define PAOS. The PAOS specification is normative in case of
question regarding PAOS [PAOS].

This profile defines SOAP header blocks that accompany the SAML requests and responses. These
header blocks may be composed with other SOAP header blocks as necessary, for example with the SOAP Message Security [WSS] header block to add security features if needed, for example encryption of the authentication request.

Two header blocks are defined: one a PAOS header block for generic PAOS information and a second ECP header block to convey information specific to ECP profile functionality.

The following diagram shows the processing flow in the ECP profile:

4.2.3 Step 1: Accessing the Service Provider: ECP>SP

In step 1, the ECP accesses the service provider with an HTTP request. This HTTP request MUST conform to the PAOS binding, which means it must include the following HTTP header fields:

1. The HTTP Accept Header field indicating the ability to accept the MIME type "application/vnd.paos+xml"
2. The HTTP PAOS Header field specifying the PAOS version with urn:liberty:paos:2003-08 at minimum.
3. This profile MUST be specified in the HTTP PAOS Header field as a service value, with the value urn:oasis:names:tc:SAML:2.0:profiles:ecp. This value should correspond to the service attribute in the PAOS Request SOAP header block.

To give an example, a user-agent may request a page from the SP as follows:

GET /index HTTP/1.1
Host: identity-service.example.com
Accept: text/html; application/vnd.paos+xml

4.2.4 Steps 2,3: SOAP Message containing <AuthnRequest>: SP>ECP>IDP

When the service provider requires a security context for the principal before providing a service or data, it can respond to the HTTP request with an <AuthnRequest> message in the HTTP response. The service provider will issue an HTTP 200 OK response to the ECP containing a single SOAP envelope.
The SOAP envelope MUST contain:

1. An `<AuthnRequest>` element in the SOAP body, targeted at the ultimate SOAP receiver, the identity provider.

2. A PAOS SOAP header block targeted at the ECP using the SOAP role “next”, the ECP. This header block provides control information such as the URL to which to send the response in this solicit-response message exchange pattern.

3. An ECP Request SOAP header block targeted at the ECP using the SOAP role “next”. The ECP Request header block defines information related to the authentication request that the ECP may need to process it, such as a list of identity providers acceptable to the service provider, whether the ECP may interact with the principal, and a service provider human-readable name that may be displayed to the principal.

The ECP will determine which identity provider is appropriate and route the SOAP message appropriately.

The ECP MUST remove the PAOS and ECP header blocks before passing the SOAP request on to the identity provider.

Note that the `<AuthnRequest>` element may itself be signed by the service provider.

### 4.2.4.1 PAOS Request Header Block: SP>ECP

The PAOS header block signals the use of PAOS processing and includes the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Meaning</th>
<th>Usage</th>
</tr>
</thead>
</table>
| responseConsumerURL | Specifies where the ECP is to send an error response. Also used to verify the correctness of the identity provider’s response, since it MUST match the assertionServiceConsumerURL. | This value MUST be present and MUST be the same as the AssertionServiceConsumerURL (or the URL referenced in metadata) conveyed in the `<AuthnRequest>`.
| service             | This indicates that the ECP authentication service is used as outlined in this profile.     | Required.                                                                                                     |
|                     | The value is defined in this profile as: “urn:oasis:names:tc:SAML:2.0:profiles:ecp”          |                                                                                                               |
| [messageID]         | Allow optional response correlation.                                                       | This attribute MAY be used in this profile. This is NOT required when using this profile since this functionality is provided by the SAML protocol layer, via the `ID` attribute in the `<AuthnRequest>` and the `InResponseTo` attribute in the `<Response>`.
| mustUnderstand      | A Fault must be generated if the PAOS header block is not understood.                      | Required, value MUST be set to true.                                                                         |
| actor/role          | Targeted SOAP node                                                                         |                                                                                                               |
The PAOS SOAP request header block has no element content.

### 4.2.4.2 ECP Request Header Block: SP > ECP

The SOAP ECP Request header block is used to convey information needed by the ECP to process the authentication request. It is mandatory and its presence signals the use of this profile.

It has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>mustUnderstand</td>
<td>1</td>
</tr>
<tr>
<td>actor/role</td>
<td>next</td>
</tr>
</tbody>
</table>

The element content of the ECP header block conveys information needed by an ECP to process a service provider’s authentication request and is defined to contain a sequence of the following elements when sent from a service provider to an ECP:

<table>
<thead>
<tr>
<th>Element</th>
<th>Use</th>
<th>Definition</th>
<th>XML Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProviderId</td>
<td>Required</td>
<td>Identification of service provider</td>
<td><code>&lt;element name=&quot;ProviderId&quot; type='anyURI' minOccurs='0' maxOccurs='1'/&gt;</code></td>
</tr>
<tr>
<td>ProviderName</td>
<td>Optional</td>
<td>Human readable name of service provider</td>
<td><code>&lt;element name=&quot;ProviderName&quot; type=&quot;string&quot; minOccurs='0' maxOccurs='1'/&gt;</code></td>
</tr>
<tr>
<td>IDPList</td>
<td>Optional</td>
<td>List of identity providers that the service provider recognizes and from which the ECP may choose to service the request</td>
<td><code>&lt;element ref=&quot;samlp:IDPList&quot; minOccurs='0'/&gt;</code></td>
</tr>
<tr>
<td>IsPassive</td>
<td>Optional</td>
<td>If &quot;true&quot; then the ECP should not interact with the principal</td>
<td><code>&lt;element name=&quot;IsPassive&quot; type='boolean' minOccurs='0' maxOccurs='1'/&gt;</code></td>
</tr>
</tbody>
</table>

### 4.2.4.3 SP>ECP Request Example

The following is an example of the SOAP authentication request from the service provider to the ECP:

```xml
<soap:Envelope
    xmlns:lecp="urn:saml2:lecp:2004-05"
    xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Header>
        <paos:Request xmlns:paos="urn:liberty:paos:2003-08"
                       responseConsumerURL="http://identity-service.example.com/abc"
                       service="urn:oasis:names:tc:SAML:2.0:profiles:ecp">
        </paos:Request>
    </soap:Header>
    <paos:Request mustUnderstand="1" role="next">
        <providerID>http://Serviceprovider.example.com</providerID>
        <providerName>Service Provider X</providerName>
        <IDPList>
            <IDPEntry>
                <providerID>http://IdentityProvider.example.com</providerID>
                <providerName>Identity Provider X</providerName>
                <loc>http://IdentityProvider.example.com/saml2/sso</loc>
            </IDPEntry>
            ... more entries ...
        </IDPList>
    </paos:Request>
</soap:Envelope>
```
4.2.4.4 ECP>IDP Request Example

As noted above, the PAOS header is removed from the SOAP message by the ECP before the
authentication request is forwarded to the identity provider. An example authentication request from the
ECP to the identity provider is as follows:

```xml
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
               xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol">
  <soap:Header/>
  <soap:Body>
    <samlp:AuthnRequest> ...
  </soap:Body>
</soap:Envelope>
```

4.2.5 Steps 4, 5: Authentication Response SOAP Message: IDP>ECP>SP

The identity provider may return an authentication response (or fault) when presented with an
authentication request. An authentication response is conveyed in a SOAP message with a <Response>
element in the SOAP body, targeted at the service provider as the ultimate SOAP receiver. The identity
provider’s response message also contains a SOAP LECP Response header targeted at the ECP, which
removes it and MAY add a SOAP <paos:Response> header block before forwarding the SOAP
response to the service provider using an HTTP POST.

The <paos:Response> SOAP header block in the response is generally used to correlate this response
to an earlier request from the service provider. In this profile the correlation refToMessageID attribute is
not required since the <Response> element’s InResponseTo attribute may be used for this purpose, but
if the <paos:Request> SOAP Header block had a messageID then the <paos:Response> SOAP
header block MUST be used.

4.2.5.1 ECP Response Header Block : IDP > ECP

The ECP response SOAP header block MUST be used on the response from the identity provider to the
ECP. The schema definition for this header block includes the mustUnderstand, role and
assertionConsumerURL attributes and defines no element values:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>mustUnderstand</td>
<td>1</td>
</tr>
<tr>
<td>actor/role</td>
<td>next</td>
</tr>
</tbody>
</table>
assertionConsumerURL This is set by the identity provider based on the <AuthnRequest> message or the service provider's metadata obtained and verified by the identity provider. The ECP MUST confirm that this value corresponds to the value the ECP obtained from the responseConsumerURL in the PAOS Request. Since the responseConsumerURL SHOULD be relative and the assertionConsumerURL is a full URL some processing will be required.

This mechanism is used for security purposes to confirm the correct response destination. If the values do not match, then the ECP MUST generate a SOAP Fault response to the service provider and not return the authentication response.

The ECP SOAP header has no element content.

After removing the ECP Response SOAP header and adding a PAOS Response SOAP header, the ECP does an HTTP POST of the response SOAP message to the service provider.

4.2.5.2 IDP>ECP Response Example

```xml
<soap:Envelope xmlns:lecp="urn:liberty:lecp:2003-08"
 xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Header>
  <lecp:Response mustUnderstand="1" role="next"
 assertionConsumerURL="http://ServiceProvider.example.com/lecp_assertion_consumer"/>
 </soap:Header>
 <soap:Body>
  <samlp:Response> ... </samlp:Response>
 </soap:Body>
</soap:Envelope>
```

4.2.5.3 PAOS Response Header Block : ECP>SP

The PAOS SOAP Response header has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>refToMessageID</td>
<td>Allows correlation with the PAOS request. This optional attribute MUST be used by if the corresponding PAOS request specified the MessageID attribute. Note that the equivalent functionality is provided with the &lt;AuthnRequest&gt; and &lt;Response&gt; correlation.</td>
</tr>
<tr>
<td>mustUnderstand</td>
<td>1</td>
</tr>
<tr>
<td>actor/role</td>
<td>next</td>
</tr>
</tbody>
</table>

The PAOS response SOAP header has no element content.

4.2.5.4 ECP>SP Response Example

```xml
<soap:Envelope xmlns:paos="urn:liberty:paos:2003-08"
 xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
 <soap:Header>
  <paos:Response refToMessageID="6c3a4f8b9c2d" role="next" mustUnderstand="1"/>
 </soap:Header>
 <soap:Body>
  <samlp:Response> ... </samlp:Response>
 </soap:Body>
</soap:Envelope>
```
4.2.6 Step 6: HTTP service response: SP>ECP

Once the service provider has received an authentication response in an HTTP request, it may respond with the service data in the HTTP response.

4.2.7 Security Considerations

1. The `<AuthnRequest>` and `<Response>` elements SHOULD be signed.
2. The PAOS header should be integrity protected, such as with SOAP Message Security or through the use of SSL/TLS over every link.
3. The service provider should be authenticated to the ECP, for example with server-side TLS authentication.
4. The ECP should be authenticated to the identity provider, such as by maintaining an authenticated session.

4.3 Identity Provider Discovery Profile

This section defines a profile by which a service provider can discover which identity providers a principal is using with the Web Browser SSO profile. In deployments having more than one identity provider, service providers need a means to discover which identity provider(s) a principal uses. The discovery profile relies on a cookie that is written in a domain that is common between identity providers and service providers in a deployment. The domain that the deployment predetermines is known as the common domain in this profile, and the cookie containing the list of identity providers is known as the common domain cookie.

Which entities host web servers in the common domain is a deployment issue and is outside the scope of this profile.

4.3.1 Common Domain Cookie

The name of the cookie MUST be _saml_idp. The format of the cookie value MUST be a set of one or more base-64 encoded URI values separated by a single space character. Each URI is the unique identifier of an identity provider, the providerId as defined in [Metadata]. The final set of values is then URL encoded.

The common domain cookie writing service (see below) SHOULD append the identity provider's providerId to the list. If the providerId is already present in the list, it MAY remove and append it when authentication of the principal occurs. The intent is that the most recently established identity provider session is the last one in the list.

The cookie MUST be set with no Path prefix or a Path prefix of "/". The Domain MUST be set to "/[common-domain]" where [common-domain] is the common domain established within the deployment for use with this profile. The cookie MUST be marked as secure.

Cookie syntax should be in accordance with [RFC2965] or [NetscapeCookie]. The cookie MAY be either session-only or persistent. This choice may be made within a deployment, but should apply uniformly to all identity providers in the deployment.

4.3.2 Setting the Common Domain Cookie

After the identity provider authenticates a principal, it MAY set the common domain cookie. The means by which the identity provider sets the cookie are implementation-specific so long as the cookie is successfully set with the parameters given above. One possible implementation strategy follows and
should be considered non-normative. The identity provider may:

- Have previously established a DNS and IP alias for itself in the common domain.
- Redirect the user agent to itself using the DNS alias using a URL specifying "https" as the URL scheme. The structure of the URL is private to the implementation and may include session information needed to identify the user-agent.
- Set the cookie on the redirected user agent using the parameters specified above.
- Redirect the user agent back to itself, or, if appropriate, to the service provider.

### 4.3.3 Obtaining the Common Domain Cookie

When a service provider needs to discover which identity providers a principal uses, it invokes an exchange designed to present the common domain cookie to the service provider after it is read by an HTTP server in the common domain.

If the HTTP server in the common domain is operated by the service provider or if other arrangements are in place, the service provider MAY utilize the HTTP server in the common domain to relay its `<AuthnRequest>` to the identity provider for an optimized single sign-on process.

The specific means by which the service provider reads the cookie are implementation-specific so long as it is able to cause the user agent to present cookies that have been set with the parameters given in section Section 3.6.1. One possible implementation strategy is described as follows and should be considered non-normative. Additionally, it may be sub-optimal for some applications.

- Have previously established a DNS and IP alias for itself in the common domain.
- Redirect the user agent to itself using the DNS alias using a URL specifying "https" as the URL scheme. The structure of the URL is private to the implementation and may include session information needed to identify the user-agent.
- Set the cookie on the redirected user agent using the parameters specified above.
- Redirect the user agent back to itself, or, if appropriate, to the identity provider.

### 4.4 Single Logout Profiles

**TODO:** profile use of Single Logout with Redirect/POST and SOAP bindings. No clear use case for artifact?

#### 4.4.1 Synchronous Single Logout Profile

#### 4.4.2 Asynchronous Single Logout Profile
5 Name Identifier Management Profiles

TODO: profile use of NameID mgmt for Redirect/POST and SOAP bindings. No clear use case for artifact?

5.1 Synchronous Name Identifier Management Profile

5.2 Asynchronous Name Identifier Management Profile
6 Name Identifier Mapping Profile

[SAMLCore] defines a Name Identifier Mapping Protocol for mapping a principal's name identifier into a
different name identifier for the same principal. This profile describes the use of this protocol with a
synchronous binding, such as the SOAP binding defined in [SAMLBind], and additional guidelines for
protecting the privacy of the principal with encryption and limiting the use of the mapped identifier.

6.1 Required Information

Contact information: security-services-comment@lists.oasis-open.org
Description: Given below.
Updates: None.

6.2 Profile Overview

The message exchange and basic processing rules that govern this profile are largely defined by section
3.9 of [SAMLCore] that defines the messages to be exchanged, and the binding used to exchange the
messages. Section 3.2 of [SAMLBind] defines the binding of the message exchange to SOAP 1.1. Unless
specifically noted here, all requirements defined in those specifications apply.

6.3 Profile Description

In this profile, a requester uses a synchronous binding to send a <NameIDMappingRequest> message
directly to an identity provider containing a name identifier for a principal that is shared between them.
Note that this identifier need not itself be persistent, and MAY be encrypted (perhaps obtained from the
previous use of this profile).

The requester MUST authenticate to the identity provider, either using a mechanism permitted by the
binding, or by signing the <NameIDMappingRequest> message.

If the identity provider receiving the request recognizes the principal, can support the requester's
{NameIDPolicy} for that principal, and is willing to fulfill the request based on authentication of the
requester and any applicable policies, then it responds with a successful <NameIDMappingResponse>
in the binding-specific response containing the requested name identifier. The resulting identifier MAY be
encrypted and time limited, as described below. Otherwise the response will contain an error status code.

The responding identity provider MUST authenticate to the requester, either using a mechanism permitted
by the binding, or by signing the <NameIDMappingResponse> message.

6.4 Use of Encryption

Section 2.3.3 of [SAMLCore] defines the use of encryption to apply confidentiality to a name identifier. In
most cases, the identity provider SHOULD encrypt the mapped name identifier it returns to the requester
to protect the privacy of the principal. The requester may extract the <EncryptedID> element and place
it in subsequent protocol messages or assertions.
6.5 Limiting Use of Mapped Identifier

Additional limits on the use of the resulting identifier MAY be applied by the identity provider by returning
the mapped name identifier in the form of an `<Assertion>` containing the identifier in its `<Subject>` but
without any statements. The assertion is then encrypted and the result used as the `<EncryptedData>`
element in the `<EncryptedID>` returned to the requester. The assertion MAY include a `<Conditions>`
element to limit use, as defined by [SAMLCore], such as time-based constraints or use by specific relying
parties, and MUST be signed for integrity protection.

6.6 Use of Metadata

[SAMLMeta][SAMLMetadata] defines an endpoint element `<md:IDPDescriptor>, <md:NameIDMappingService>`, to describe the location(s) to which a
requester may send requests using this profile. The `<Binding>` attribute of the element MUST contain the
identifier of the protocol binding supported by that endpoint. The `<ResponseLocation>` attribute MUST be
omitted from endpoints that support this profile.

The identity provider, if encrypting the resulting identifier for a particular provider, can use that provider's
`<md:KeyDescriptor>` element with a `use` attribute of encryption to determine an appropriate
encryption key and algorithm to use.
7 Attribute Profiles

7.1 Guidelines
This section provides a checklist of items that MUST be addressed by each attribute profile.

1. A human readable string name for the profile.
2. Unique URI to be used for the NameFormat attribute of the <saml:AttributeDesignator> element.
3. Syntax and restrictions on class of strings acceptable as the value of the name attribute of the <saml:AttributeDesignator> element when the selected NameFormat attribute value is present.
4. Additional attributes (together with required namespaces) defined by the profile that may be used with the <saml:AttributeDesignator> element.
5. Rules for determining the equality of <saml:AttributeDesignator> elements as defined by the profile.
6. Syntax and restrictions on values acceptable in a <saml:AttributeValue> element, when the selected NameFormat attribute value is present and whether the xsi:type attribute must be present.
7. Additional attributes (together with required namespaces) defined by the profile that may be used with the <saml:Attribute> element.

7.2 Basic Attribute Profile

7.2.1 NameFormat Value
URI: urn:oasis:names:tc:SAML:2.0:attribute-name:basic

7.2.2 Attribute Names
The class of strings acceptable as the value of the Name attribute of the <saml:AttributeDesignator> element MUST be drawn from the set of values belonging to the primitive type Name as defined in Section 3.3.6 of [XML-Schema-Part2].

No additional attributes are defined for the <saml:Attribute> or <saml:AttributeDesignator> elements.

7.2.3 Attribute Values
The type of the <saml:AttributeValue> element MUST be drawn from one of the values of the types defined in Section 3.3 of [XML-Schema-Part2]. The xsi:type attribute MUST be present and be given the appropriate type.

7.2.4 AttributeDesignator Comparison
Two <saml:AttributeDesignator> elements are equal iff the name attribute values are equal in the sense of Section 3.3.6 of [XML-Schema-Part2].
7.2.5 Example

TBD

7.3 X.500/LDAP Profile

There is a substantial body of work describing standard syntaxes for X.500/LDAP attributes. This includes RFC2256 [RFC2256], which describes an overview of the attribute types and object classes defined by the ISO and ITU-T committees in the X.500 documents, in particular those intended for use by directory clients. Several authors have built upon these approaches to develop additional attribute types and some of these have been widely implemented. For example, the inetOrgPerson object class defined in RFC2798 [RFC2798] has received wide implementation amongst LDAP vendors. Other efforts include the definition of eduPerson object class by the EDUCAUSE/Internet2 task force [eduPersonSchema].

7.3.1 NameFormat Value

URI: urn:oasis:names:tc:SAML:2.0:attribute-name:X500-LDAP

7.3.2 Attribute Names

Following [Morgan], we adopt the URN oid namespace described in [RFC3061]. In this approach the Attribute name is based on the OID assigned to the X.500/LDAP attribute type.

Example:

urn:oid:1.3.6.1.4.1.299

X.500 conventions require that every object-class be identified with a unique OID. This ensures attribute names are unambiguous.

For purposes of human readability there is also a requirement to carry an optional string name together with the OID. This modeled by the optional attribute FriendlyName with namespace urn:oasis:names:tc:SAML:2.0:attribute-name:X500-LDAP. The FriendlyName attribute is drawn from [RFCXXX].

7.3.3 Attribute Values

We need to define a convention for carrying different attribute syntaxes within XML. [RFC2252] explains that octet strings are the canonical representation for X.500/LDAP attribute syntaxes. The primitive type hexbinary (Section 3.2.15 of [XML-Schema-Part2]) allows octet strings to be represented by their hexadecimal representation. The hexbinary type MUST be used when <saml:AttributeValue> contains an X.500/LDAP attribute value.

7.3.4 AttributeDesignator Comparison

<saml:AttributeDesignator> elements are equal if and only if the Name attribute values are equal in the sense of [RFC3061]. The FriendlyName attribute plays no role in the comparison.

7.3.5 Example

TBD
7.4 UUID Profile

7.4.1 UUIDs and GUIDs

UUIDs (Universally Unique Identifiers), also known as GUIDs (Globally Unique Identifiers), are used to define objects and subjects such that they are guaranteed uniqueness across space and time. UUIDs were originally used in the Network Computing System (NCS), and then used in the Open Software Foundation's (OSF) Distributed Computing Environment (DCE). Recently GUIDs have been used in Microsoft's Active Directory/Windows 2000/2003 platform.

A UUID is a 128 bit number, generated such that it should never be duplicated within the domain of interest. UUIDs are used to represent a wide range of objects including, but not limited to, subjects/users, groups of users and node names. A UUID, represented as a hexadecimal string, is as follows:

```
f81d4fae-7dec-11d0-a765-00a0c91e6bf6
```

In DCE and Microsoft Windows the UUID is usually presented to the administrator in the form of a “friendly name”. For instance the above UUID could represent the user john.hughes@entegrity.com.

7.4.2 DCE Attribute Names

DCE is able to transport a wide range of authorization data within its Privilege Attribute Certificate (PAC). The DCE PAC entry types that are supported in SAML 2.0 are provided in the below table.

<table>
<thead>
<tr>
<th>AttributeName</th>
<th>DCE Entry Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principal</td>
<td>Principal</td>
<td>The DCE Principal of the identity, in UUID form</td>
</tr>
<tr>
<td>PGroup</td>
<td>Group</td>
<td>The Primary DCE Group of the identity, in UUID form</td>
</tr>
<tr>
<td>NGroups</td>
<td>numGroups</td>
<td>The number of groups on the following group set</td>
</tr>
<tr>
<td>GroupSet</td>
<td>Groups</td>
<td>A list of UUIDs from the groups list, in UUID form</td>
</tr>
</tbody>
</table>

The attribute name space used for DCE attributes is:

```
URI:urn:oasis:names:tc:SAML:2.0:attribute-namespace#dce
```

7.4.3 Attribute Values

Whilst the UUID is guaranteed to be unique in across space and time, the friendly name is not. Hence each attribute defined in the previous sections will carry the UUID as well as a friendly name. TBD – on how to define.
8 References


[Anders] A suggestion on how to implement SAML browser bindings without using "Artifacts",  


[HTML401] HTML 4.01 Specification, W3C Recommendation 24 December 1999,  
http://www.w3.org/TR/html4.  


[MSURL] Microsoft technical support article,  
http://support.microsoft.com/support/kb/articles/Q208/4/27.ASP.  

https://www.projectliberty.org/specs/liberty-paos-v1.0.pdf  

[Rescorla-Sec] E. Rescorla et al., Guidelines for Writing RFC Text on Security Considerations,  


[RFC2045] Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies,  

[RFC2119] S. Bradner, Key words for use in RFCs to Indicate Requirement Levels, IETF RFC 2119,  


[RFC2617] HTTP Authentication: Basic and Digest Access Authentication, IETF RFC 2617,  

[SAMLBind] Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0, DRAFT  

http://www.oasis-open.org/committees/security/.  

[SAMLMeta] Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0, DRAFT  


http://www.oasis-open.org/committees/security/.  

[SAMLReqs] Darren Platt et al., SAML Requirements and Use Cases, OASIS, April 2002,  
http://www.oasis-open.org/committees/security/.  


[SESSION] RL “Bob” Morgan, Support of target web server sessions in Shibboleth,  
http://middleware.internet2.edu/shibboleth/docs/draft-morgan-shibboleth-session-00.txt


[RFC2798] W. Smith, Definition of the inetOrgPerson LDAP Object class, April 2000

[eduPersonSchema] eduPerson.ldif, Available from http://www.educase.edu/eduperson


A. Acknowledgments

The editors would like to acknowledge the contributions of the OASIS Security Services Technical Committee, whose voting members at the time of publication were:

- TBD
## B. Revision History

<table>
<thead>
<tr>
<th>Rev</th>
<th>Date</th>
<th>By Whom</th>
<th>What</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>02/16/04</td>
<td>Frederick Hirsch</td>
<td>Split new profiles document from bindings and profiles, removed bindings section. Added ECP profile, added and formatted references.</td>
</tr>
<tr>
<td>2</td>
<td>03/02/04</td>
<td>Frederick Hirsch</td>
<td>Removed URL Size restriction section – this is located in the bindings document. Minor cleanup in section 2.1</td>
</tr>
<tr>
<td>3</td>
<td>03/27/04</td>
<td>Frederick Hirsch</td>
<td>Changes to reflect core 8, review comments, corrections.</td>
</tr>
<tr>
<td>4</td>
<td>03/30/04</td>
<td>Frederick Hirsch</td>
<td>Additional review comments, corrections.</td>
</tr>
<tr>
<td>6</td>
<td>04/16/04</td>
<td>Scott Cantor</td>
<td>Replaced 1.1 SSO profiles with new proposal, added discovery profile, revised confirmation method descriptions, removed binding-related duplications, added placeholders for additional profiles.</td>
</tr>
<tr>
<td>7</td>
<td>05/09/04</td>
<td>Scott Cantor</td>
<td>Added NameIdentifierMapping profile</td>
</tr>
<tr>
<td>8</td>
<td>05/14/04</td>
<td>Frederick Hirsch</td>
<td>Changes based on 5/11/04 SSTC conference call – replace Identifier with ID in elements, in elements and attributes replace Authentication with Authn. Specifically, changed &lt;AuthenticationStatement&gt;, &lt;NameIdentifierMappingRequest&gt;, &lt;NameIdentifierMappingResponse&gt;, &lt;EncryptedIdentifier&gt;, &lt;NameIdentifierMappingService&gt;</td>
</tr>
<tr>
<td>9</td>
<td>05/30/04</td>
<td>Scott Cantor</td>
<td>Sync'd confirmation data sections to new schema in core-14, relaxed NameIDMapping profile requirement for SOAP binding, started clean-up of ECP, adjusted SSO profile to reflect bindings-12, added back sendervouches.</td>
</tr>
<tr>
<td>10</td>
<td>06/07/04</td>
<td>Prateek Mishra</td>
<td>Added attribute profiles materials from hughes-mishra-baseline-attributes-04 with John Hughes updates</td>
</tr>
<tr>
<td>11</td>
<td>06/13/04</td>
<td>Scott Cantor</td>
<td>Added metadata considerations to profiles, minor editorial cleanups, new section headers for profiles</td>
</tr>
</tbody>
</table>
C. Notices

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification, can be obtained from the OASIS Executive Director.

OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to implement this specification. Please address the information to the OASIS Executive Director.

Copyright © OASIS Open 2003-2004. All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to OASIS, except as needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an “AS IS” basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

JavaScript is a trademark or registered trademark of Sun Microsystems, Inc. in the United States and other countries.