Kantara Initiative eGovernment Implementation Profile of SAML V2.0 ~~May 14, 20106~~

~~Working Draft 0~~

~~Version 2.0~~

~~Document identifier~~Version:

~~draft-kantara-egov-saml2-profile-2.0~~Working Draft 07

Date:

May 14, 2010

~~Location:~~

~~TBD~~

Editor~~s~~:

Scott Cantor, Internet2

Contributors:

Kantara eGovernment W~~C~~orking Group

Andreas Åkre Solberg, UNINETT

Abstract:

This document contains an implementation profile for eGovernment use of SAML V2.0, suitable for the purposes of testing conformance of implementations of SAML V2.0. It is not a deployment profile, and does not provide for or reflect specific behavior expected of implementations when used within a particular deployment context.

Filename:

draft-kantara-egov-saml2-profile-2.0-07

Notice:

Table of Contents

1 Introduction

SAML V2.0 is a rich and extensible standard that must be profiled to be used interoperably, and the profiles that typically emerge from the broader standardization process usually remain fairly broad and include a number of options and features that increase the burden for implementers and make deployment-time decisions more difficult.

The Kantara Initiative eGovernment Implementation Profile provides a SAML V2.0 conformance specification for Identity Provider and Service Provider implementations operating in eGovernment federations and deployments. The profile is based on the SAML V2.0 specifications created by the Security Services Technical Committee (SSTC) of OASIS, and related specifications approved by that body. It constrains and supplements the base SAML V2.0 features, elements, and attributes required for eGovernment federations and deployments.

Implementation profiles define the features that software implementations must support such that deployers can be assured of the ability to meet their own (possibly varied) deployment requirements. Deployment profiles define specific options and constraints to which deployments are required to conform; they guide product configuration and federation operations, and provide criteria against which actual deployments may be tested. This document does not include a deployment profile, but reflects the features deemed necessary or desirable from software implementations in support of a variety of deployment profiles planned and in use. This includes requirements deemed useful to further the eventual goal of interfederation between deployments.

1.1 Notation

This specification uses normative text to describe the use of SAML capabilities.

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this specification are to be interpreted as described in [RFC2119]:

> …they MUST only be used where it is actually required for interoperation or to limit behavior which has potential for causing harm (e.g., limiting retransmissions)…

These keywords are thus capitalized when used to unambiguously specify requirements over protocol and application features and behavior that affect the interoperability and security of implementations. When these words are not capitalized, they are meant in their natural-language sense.

```
Listings of XML schemas appear like this.
```

```
Example code listings appear like this.
```

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

- The prefix `saml2:` stands for the SAML 2.0 assertion namespace, `urn:oasis:names:tc:SAML:2.0:assertion`
- The prefix `saml2p:` stands for the SAML 2.0 protocol namespace, `urn:oasis:names:tc:SAML:2.0:protocol`
- The prefix `md:` stands for the SAML 2.0 metadata namespace, `urn:oasis:names:tc:SAML:2.0:metadata`
- The prefix `idpdisc:` stands for the Identity Provider Discovery Service Protocol and Profile [IdPDisco] namespace, `urn:oasis:names:tc:SAML:profiles:SSO:idp-discovery-protocol`

- The prefix `mdattr:` stands for the Metadata Extension for Entity Attributes Version 1.0 [MetaAttr] namespace, `urn:oasis:names:tc:SAML:metadata:attribute`

This specification uses the following typographical conventions in text: `<ns:Element>`, `Attribute`, **Datatype**, `OtherCode`.

2 SAML V2.0 Implementation Profile

This profile specifies behavior and options that implementations of a selected set of SAML V2.0 profiles [SAML2Prof] are required to support. The requirements specified are *in addition to* all normative requirements of the original profiles, as modified by the Approved Errata [SAML2Err], and readers should be familiar with all relevant reference documents. Any such requirements are not repeated here except where deemed necessary to highlight a point of discussion or draw attention to an issue addressed in errata, but remain implied.

SAML leaves substantial latitude to implementations with regard to how software is architected and combined with authentication and application infrastructure. Where the terms "Identity Provider" and "Service Provider" are used, they should be understood to include the total software footprint intended to provide the desired functionality; no specific assumptions are made as to how the required features are exposed to deployers, only that there is some method for doing so.

2.1 Required Information

Identification: http://kantarainitiative.org/eGov/profiles/SAML2.0/v2.0

Contact information: http://kantarainitiative.org/confluence/display/eGov/Home

Description: Given below

Updates: Liberty Alliance eGov Profile for SAML 2.0 [eGov15]

2.2 Metadata and Trust Management

Identity Provider, Service Provider, and Discovery Service implementations MUST support the use of SAML V2.0 Metadata [SAML2Meta] in conjunction with their support of the SAML V2.0 profiles referenced by subsequent sections. Additional expectations around the use of particular metadata elements related to profile behavior may be encountered in those sections.

2.2.1 Metadata Profiles

Implementations MUST support the SAML V2.0 Metadata Interoperability Profile Version 1.0 [MetaIOP].

In addition, ~~I~~implementations MUST ~~also ~~support ~~an alternative to that profile's language on ~~the use of the `<md:KeyDescriptor>` element as follows:

- Implementations MUST support the `<ds:X509Certificate>` element as input to subsequent requirements. Support for other key representations, and for other mechanisms for credential distribution, is OPTIONAL.
- Implementations MUST support some form of path validation of signing, TLS, and encryption credentials used to secure SAML exchanges against one or more trusted certificate authorities. Support for PKIX [RFC5280] is RECOMMENDED; implementations SHOULD document the behavior of the validation mechanisms they employ, particular with respect to limitations or divergence from PKIX [RFC5280].
- Implementations MUST support the use of OCSP [RFC2560] and ~~c~~Certificate ~~r~~Revocation ~~l~~Lists (CRLs) obtained via the "CRL Distribution Point" X.509 extension [RFC5280] for revocation checking of those credentials.
- Implementations MAY support additional constraints on the contents of certificates used by particular entities, such as "subjectAltName" or "DN", key usage constraints, or policy extensions, but SHOULD document such features and make them optional to enable where possible.

Note that these metadata profiles are intended to be mutually exclusive within a given deployment context; they are alternatives, rather than complimentary or compatible uses of the same metadata information.

Implementations SHOULD support the SAML V2.0 Metadata Extension for Entity Attributes Version 1.0 [MetaAttr] and provide policy controls on the basis of SAML attributes supplied via this extension mechanism.

2.2.2 Metadata Exchange

It is OPTIONAL for implementations to support the generation or exportation of metadata, but implementations MUST support the publication of metadata using the Well-Known-Location method defined in section 4.1 of [SAML2Meta] (under the assumption that entityID values used are suitable for such support).

Implementations MUST support the following mechanisms for the importation of metadata:

- local file
- remote resource at fixed location accessible via HTTP 1.1 [RFC2616] or HTTP 1.1 over TLS/SSL [RFC2818]

In the case of HTTP resolution, implementations MUST support use of the "ETag" and "Last-Modified" headers for cache management. Implementations SHOULD support the use of more than one fixed location for the importation of metadata, but MAY leave their behavior unspecified if a single entity's metadata is present in more than one source.

Importation of multiple entities' metadata contained within an `<md:EntitiesDescriptor>` element MUST be supported.

Finally, implementations SHOULD allow for the automated updating/reimportation of metadata without service degradation or interruption.

2.2.2.1 Metadata Verification

Verification of metadata, if supported, MUST include XML signature verification at least at the root element level, and SHOULD support the following mechanisms for signature key trust establishment:

- Direct comparison against known keys.
- Some form of path-based certificate validation against one or more trusted certificate authorities, along with certificate revocation lists and/or OCSP [RFC2560]. Support for PKIX [RFC5280] is RECOMMENDED; implementations SHOULD document the behavior of the validation mechanisms they employ, particular with respect to limitations or divergence from PKIX [RFC5280].

2.3 Name Identifiers

In conjunction with their support of the SAML V2.0 profiles referenced by subsequent sections, Identity Provider and Service Provider implementations MUST support the following SAML V2.0 name identifier formats, in accordance with the normative obligations associated with them by [SAML2Core]:

- `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`
- `urn:oasis:names:tc:SAML:2.0:nameid-format:transient`

Support for other formats is OPTIONAL.

2.4 Attributes

In conjunction with their support of the SAML V2.0 profiles referenced by subsequent sections, Identity Provider and Service Provider implementations MUST support the generation and consumption of `<saml2:Attribute>` elements that conform to the SAML V2.0 X.500/LDAP Attribute Profile [SAML-X500].

The ability to support `<saml2:AttributeValue>` elements whose values are not simple strings (e.g., `<saml2:NameID>`, or other XML values) is OPTIONAL. Such content could be base64-encoded as an alternative.

2.5 Browser Single Sign-On

This section defines an implementation profile of the SAML V2.0 Web Browser SSO Profile [SAML2Prof].

2.5.1 Identity Provider Discovery

Service Provider and Discovery Service implementations MUST support the Identity Provider Discovery Service Protocol Profile in conformance with section 2.4.1 of [IdPDisco].

2.5.2 Authentication Requests

2.5.2.1 Binding and Security Requirements

Identity Provider and Service Provider implementations MUST support the use of the HTTP-Redirect binding [SAML2Bind] for the transmission of `<saml2p:AuthnRequest>` messages, including the generation or verification of signatures in conjunction with this binding.

Support for other bindings is OPTIONAL.

2.5.2.2 Message Content

In addition to standard core- and profile-driven requirements, Service Provider implementations MUST support the inclusion of at least the following `<saml2p:AuthnRequest>` child elements and attributes (when appropriate)~~:~~:

- `AssertionConsumerServiceURL`
- `ProtocolBinding`
- `ForceAuthn`
- `IsPassive`
- `AttributeConsumingServiceIndex`
- `<saml2p:RequestedAuthnContext>`
- `<saml2p:NameIDPolicy>`
-

Identity Provider implementations MUST support all `<saml2p:AuthnRequest>` child elements and attributes defined by [SAML2Core], but MAY provide that support in the form of returning appropriate errors when confronted by particular request options. However, implementations MUST fully support the options enumerated above~~.~~, and be configurable to utilize those options in a useful manner as defined by [SAML2Core].

Implementations MAY limit their support of the `<saml2p:RequestedAuthnContext>` element to the value "exact" for the `Comparison` attribute~~.~~, but MUST otherwise support any allowable content of the element.

Identity Provider implementations MUST support verification of requested `AssertionConsumerServiceURL` locations via comparison to `<md:AssertionConsumerService>` elements supplied via metadata using case-sensitive string comparison. It is OPTIONAL to support other means of comparison (e.g., canonicalization or other manipulation of URL values) or alternatve verification mechanisms.

2.5.3 Responses

2.5.3.1 Binding and Security Requirements

Identity Provider and Service Provider implementations MUST support the use of the HTTP-POST and HTTP-Artifact bindings [SAML2Bind] for the transmission of `<saml2p:Response>` messages.

Support for other bindings, and for artifact types other than `urn:oasis:names:tc:SAML:2.0:artifact-04`, is OPTIONAL.

Identity Provider and Service Provider implementations MUST support the generation and consumption of unsolicited `<saml2p:Response>` messages (i.e., responses that are not the result of a `<saml2p:AuthnRequest>` message).

Identity Provider implementations MUST support the issuance of `<saml2p:Response>` messages (with appropriate status codes) in the event of an error condition, provided that the user agent remains available and an acceptable location to which to deliver the response is available. The criteria for "acceptability" of a response location are not formally specified, but are subject to Identity Provider policy and reflect its responsibility to protect users from being sent to untrusted or possibly malicious parties. Note that this is a stronger requirement than the comparable language in [SAML2Prof].

Identity Provider and Service Provider implementations MUST support the signing of `<saml2:Assertion>` elements in responses; support for signing of the `<saml2p:Response>` element is OPTIONAL.

Identity Provider and Service Provider implementations MUST support the use of XML Encryption via the `<saml2:EncryptedAssertion>` element when using the HTTP-POST binding; support for the `<saml2:EncryptedID>` and `<saml2:EncryptedAttribute>` elements is OPTIONAL.

2.5.3.2 Message Content

The Web Browser SSO Profile allows responses to contain any number of assertions and statements. Identity Provider implementations MUST allow the number of `<saml2:Assertion>`, `<saml2:AuthnStatement>`, and `<saml2:AttributeStatement>` elements in the `<saml2p:Response>` message to be limited to one.

In turn, Service Provider implementations MAY limit support to a single instance of those elements when processing `<saml2p:Response>` messages.

Identity Provider implementations MUST support the inclusion of a `Consent` attribute in `<saml2p:Response>` messages, and a `SessionIndex` attribute in `<saml2:AuthnStatement>` elements.

Service Provider implementations that provide some form of session semantics MUST support the `<saml2:AuthnStatement>` element's `SessionNotOnOrAfter` attribute.

Service Provider implementations MUST support the acceptance/rejection of assertions based on the content of the `<saml2:AuthnStatement>` element's `<saml2:AuthnContext>` element.

Implementations also MUST support the acceptance/rejection of particular <saml2:AuthnContext> content based on the identity of the Identity Provider. [IAP] provides one such mechanism via SAML V2.0 metadata and is RECOMMENDED; though this specification is in draft form, the technical details are not expected to change prior to eventual approval.

2.5.4 Artifact Resolution

Pursuant to the requirement in section 2.5.3.1 for support of the HTTP-Artifact binding [SAML2Bind] for the transmission of `<saml2p:Response>` messages, implementations MUST support the SAML V2.0 Artifact Resolution profile [SAML2Prof] as constrained by the following subsections.

2.5.4.1 Artifact Resolution Requests

Identity Provider and Service Provider implementations MUST support the use of the SAML SOAP (using HTTP as a transport) binding [SAML2Bind] for the transmission of `<saml2p:ArtifactResolve>` messages.

Implementations MUST support the use of SAML message signatures and TLS server authentication to authenticate requests; support for TLS client authentication, or other forms of authentication in conjunction with the SAML SOAP binding, is OPTIONAL.

2.5.4.2 Artifact Resolution Responses

Identity Provider and Service Provider implementations MUST support the use of the SAML SOAP (using HTTP as a transport) binding [SAML2Bind] for the transmission of `<saml2p:ArtifactResponse>` messages.

Implementations MUST support the use of SAML message signatures and TLS server authentication to authenticate responses; support for TLS client authentication, or other forms of authentication in conjunction with the SAML SOAP binding, is OPTIONAL.

2.6 Browser Holder of Key Single Sign-On

This section defines an implementation profile of the SAML V2.0 Holder-of-Key Web Browser SSO Profile Version 1.0 [HoKSSO].

The implementation requirements defined in section 2.5 for the non-holder-of-key profile apply to implementations of this profile.

2.7 SAML 2.0 Proxying

Section 3.4.1.5 of [SAML2Core] defines a formalized approach to proxying the SAML 2.0 Authentication Request protocol between multiple Identity Providers. This section defines an implementation profile for this behavior suitable for composition with the Single Sign-On profiles defined in sections 2.5 and 2.6.

The requirements of the profile are imposed on Identity Provider implementations acting as a proxy. These requirements are in addition to the technical requirements outlined in section 3.4.1.5.1 of [SAML2Core], which also MUST be supported.

2.7.1 Authentication Requests

Proxying Identity Provider implementations MUST support the mapping of incoming to outgoing `<saml2p:RequestedAuthnContext>` and `<saml2p:NameIDPolicy>` elements, such that deployers may choose to pass through values or map between different vocabularies as required.

Proxying Identity Provider implementations MUST support the suppression/eliding of <saml2p:RequesterID> elements from outgoing <saml2p:AuthnRequest> messages to allow for hiding the identity of the Service Provider from proxied Identity Providers.

2.7.2 Responses

Proxying Identity Provider implementations MUST support the mapping of incoming to outgoing <saml2:AuthnContext> elements, such that deployers may choose to pass through values or map between different vocabularies as required.

Proxying Identity Provider implementations MUST support the suppression of <saml2:AuthenticatingAuthority> elements from outgoing <saml2:AuthnContext> elements to allow for hiding the identity of the proxied Identity Provider from Service Providers.

2.8 Single Logout

This section defines an implementation profile of the SAML V2.0 Single Logout Profile [SAML2Prof].

For clarification, the technical requirements for each message type below reflect the intent to normatively require initiation of logout by a Service Provider using either the front- or back-channel, and initiation/propagation of logout by an Identity Provider using the back-channel.

2.8.1 Logout Requests

2.8.1.1 Binding and Security Requirements

Identity Provider implementations MUST support the SAML SOAP (using HTTP as a transport) binding [SAML2Bind] for the issuance of <saml2p:LogoutRequest> messages, and MUST support the SAML SOAP (using HTTP as a transport) and HTTP-Redirect bindings [SAML2Bind] for the reception of <saml2p:LogoutRequest> messages.

Service Provider implementations MUST support the SAML SOAP (using HTTP as a transport) binding [SAML2Bind] for both issuance and reception of <saml2p:LogoutRequest> messages.

Support for other bindings is OPTIONAL.

Implementations MUST support the use of SAML message signatures and TLS server authentication to authenticate <saml2p:LogoutRequest> messages; support for TLS client authentication, or other forms of authentication in conjunction with the SAML SOAP binding, is OPTIONAL.

Identity Provider and Service Provider implementations MUST support the use of XML Encryption via the <saml2:EncryptedID> element when using the HTTP-Redirect binding.

2.8.1.2 User Interface Behavior

Identity Provider implementations MUST support both user-initiated termination of the local session only and user-initiated Single Logout. Upon receipt of a <saml2p:LogoutRequest> message via a front-channel binding, Identity Provider implementations MUST support user intervention governing the choice of propagating logout to other Service Providers, or limiting the operation to the Identity Provider. Of course, implementations MUST return status information to the requesting entity (e.g. partial logout indication) as appropriate.

Service Provider implementations MUST support both user-initiated termination of the local session only and user-initiated Single Logout.

Identity Provider implementations MUST also support the administrative initiation of Single Logout for any active session, subject to appropriate policy.

2.8.2 Logout Responses

2.8.2.1 Binding and Security Requirements

Identity Provider implementations MUST support the SAML SOAP (using HTTP as a transport) and HTTP-Redirect bindings [SAML2Bind] for the issuance of `<saml2p:LogoutResponse>` messages, and MUST support the SAML SOAP (using HTTP as a transport) binding [SAML2Bind] for the reception of `<saml2p:LogoutResponse>` messages.

Service Provider implementations MUST support the SAML SOAP (using HTTP as a transport) binding [SAML2Bind] for both issuance and reception of `<saml2p:LogoutResponse>` messages.

Support for other bindings is OPTIONAL.

Implementations MUST support the use of SAML message signatures and TLS server authentication to authenticate `<saml2p:LogoutResponse>` messages; support for TLS client authentication, or other forms of authentication in conjunction with the SAML SOAP binding, is OPTIONAL.

3 Conformance Classes

3.1 Standard

Conforming Identity Provider and/or Service Provider implementations MUST support the normative requirements in sections 2.2, 2.3, 2.4, and 2.5.

3.1.1 Signature and Encryption Algorithms

Implementations MUST support the signature and digest algorithms identified by the following URIs in conjunction with the creation and verification of XML Signatures [XMLSig]:

- `http://www.w3.org/2001/04/xmldsig-more#rsa-sha256` (defined in [RFC4051])
- `http://www.w3.org/2001/04/xmlenc#sha256` (defined in [XMLEnc])

Implementations SHOULD support the signature and digest algorithms identified by the following URIs in conjunction with the creation and verification of XML Signatures [XMLSig]:

- `http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256` (defined in [RFC4051])

Implementations MUST support the block encryption algorithms identified by the following URIs in conjunction with the use of XML Encryption [XMLEnc]:

- `http://www.w3.org/2001/04/xmlenc#tripledes-cbc`
- `http://www.w3.org/2001/04/xmlenc#aes128-cbc`
- `http://www.w3.org/2001/04/xmlenc#aes256-cbc`

Implementations MUST support the key transport algorithms identified by the following URIs in conjunction with the use of XML Encryption [XMLEnc]:

- `http://www.w3.org/2001/04/xmlenc#rsa-1_5`
- `http://www.w3.org/2001/04/xmlenc#rsa-oaep-mgf1p`

Implementations SHOULD support the key agreement algorithms identified by the following URIs in conjunction with the use of XML Encryption [XMLEnc]:

- `http://www.w3.org/2009/xmlenc11#ECDH-ES` (defined in [XMLEnc11])

  (This is a Last Call Working Draft of XML Encryption 1.1, and this normative requirement is contingent on W3C ratification of this specification without normative changes to this algorithm's definition.)

Support for other algorithms is OPTIONAL.

3.2 Standard with Logout

Conforming Identity Provider and/or Service Provider implementations MUST meet the conformance requirements in section 3.1, and MUST in addition support the normative requirements in section 2.8.

3.3 Full

Conforming Identity Provider and/or Service Provider implementations MUST meet the conformance requirements in section 3.1, and MUST in addition support the normative requirements in sections 2.6, 2.7, and 2.8.

4 References

4.1 Normative References

[RFC2119] IETF RFC 2119, *Key words for use in RFCs to Indicate Requirement Levels*, March 1997. http://www.ietf.org/rfc/rfc2119.txt

[RFC2560] IETF RFC 2560, *X.509 Internet Public Key Infrastructure Online Certificate Status Protocol,* June 1999. http://www.ietf.org/rfc/rfc2560.txt

[RFC2616] IETF RFC 2616, *Hypertext Transfer Protocol – HTTP/1.1*, June 1999. http://www.ietf.org/rfc/rfc2616.txt

[RFC2818] IETF RFC 2818, *HTTP Over TLS*, May 2000. http://www.ietf.org/rfc/rfc2818.txt

[RFC4051] IETF RFC 4051, *Additional XML Security Uniform Resource Identifiers,* April 2005. http://www.ietf.org/rfc/rfc4051.txt

[RFC5280] IETF RFC 5280, *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile,* May 2008. http://www.ietf.org/rfc/rfc5280.txt

[HoKSSO] OASIS Committee Specification, *SAML V2.0 Holder-of-Key Web Browser SSO Profile Version 1.0*, July 2009. http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-holder-of-key-browser-sso-cs-01.pdf

[IAP] OASIS Committee Draft, *Identity Assurance Profiles, Version 1.0*, September 2009. http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-assurance-profile-cd-01.pdf

[IdPDisco] **OASIS Committee Specification, *Identity Provider Discovery Service Protocol and Profile*, March 2008. http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-id**p-discovery.pdf

[MetaAttr] OASIS Committee Specification, *SAML V2.0 Metadata Extension for Entity Attributes Version 1.0*, August 2009. http://docs.oasis-open.org/security/saml/Post2.0/sstc-metadata-attr.pdf

[MetaIOP] OASIS Committee Specification, *SAML V2.0 Metadata Interoperability Profile Version 1.0*, August 2009. http://docs.oasis-open.org/security/saml/Post2.0/sstc-metadata-iop.pdf

[SAML2Core] OASIS Standard, *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0*, March 2005. http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf

[SAML2Meta] OASIS Standard, *Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0*, March 2005. http://docs.oasis-open.org/security/saml/v2.0/saml-metadata-2.0-os.pdf

[SAML2Bind] OASIS Standard, *Bindings for the OASIS Security Assertion Markup Language (SAML) V2.0*, March 2005. http://docs.oasis-open.org/security/saml/v2.0/saml-bindings-2.0-os.pdf

[SAML2Prof] OASIS Standard, *Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0*, March 2005. http://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf

[SAML2Err] OASIS Approved Errata, *SAML V2.0 Errata*, Dec 2009. http://www.oasis-open.org/committees/download.php/37166/sstc-saml-approved-errata-2.0-02.pdf

[SAML-X500] OASIS Committee Specification, *SAML V2.0 X.500/LDAP Attribute Profile*, March 2008. http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-attribute-x500.pdf

[XMLEnc] D. Eastlake et al. *XML Encryption Syntax and Processing*. World Wide Web Consortium Recommendation. http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/

[XMLEnc11] D. Eastlake et al. *XML Encryption Syntax and Processing Version 1.1*. World Wide Web Consortium Last Call Working Draft. http://www.w3.org/TR/2010/WD-xmlenc-core1-20100513/

[XMLSig] D. Eastlake et al. *XML-Signature Syntax and Processing, Second Edition*. World Wide Web Consortium Recommendation, June 2008. http://www.w3.org/TR/xmldsig-core/

Non-Normative References

[eGov15] Kyle Meadors, *Liberty Alliance eGov Profile for SAML 2.0 Version 1.5*. http://www.projectliberty.org/liberty/content/download/4711/32210/file/Liberty_Alliance_eGov_Profile_1.5_Final.pdf

Appendix A. ~~Change Log~~Revision History

- Draft 01: first working draft based on similar document created by InCommon Federation
- Draft 02: first round of feedback incorporated, deployment section dropped, new section on Artifact Resolution added, artifact added for SSO responses, SOAP added for logout, discovery moved under SSO, language on non-string attributes added, changed SHOULD to MUST for IdP support of selected AuthnRequest features
- Draft 03: moved Artifact Resolution into a SSO profile subsection, new language on SOAP security and SLO bindings, added metadata publication via WKL, added language on IdP error handling, added Holder of Key SSO profile, added Conformance Classes
- Draft 04: added UI language around SLO, layered conformance language and added MTI algorithms, added section for Proxying
- Draft 05: revised language for IdP error handling, added text on ACS checking, added proxying privacy language, heavily revised metadata section and added a "pseudo-profile" for combining certificates in metadata with PKI as an IOP alternative
- Draft 06: added normative reference to RFC5280 in path validation text, expanded algorithm requirements, added sentence on administrative logout
- Draft 07, clarifications on AuthnContext support and reference to IAP, additional algorithm reference, change to boilerplate sections to match Kantara template