SAML V2.0 Metadata Interoperability Profile

Working Draft 01, 9 August 2008

Specification URIs:

TBD

Technical Committee:

OASIS Security Services TC

Chair(s):

Hal Lockhart, BEA Systems, Inc.

Brian Campbell, Ping Identity Corporation

Editors:

Scott Cantor, Internet2

Abstract:

This profile describes a set of rules for SAML metadata producers and consumers to follow such that federated relationships can be interoperably provisioned, and controlled at runtime in a secure, understandable, and self-contained fashion.

Status

This document was last revised or approved by the SSTC on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.

TC members should send comments on this specification to the TC's email list. Others should send comments to the TC by using the "Send A Comment" button on the TC's web page at 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 IPR section of the TC web page (http://www.oasis-open.org/committees/security/ipr.php).

The non-normative errata page for this specification is located at http://www.oasis-open.org/committees/security.

Notices

Table of Contents

1 Introduction....4
  1.1 Notation....4
  1.2 Normative References....5
  1.3 Non-Normative References....5
  1.4 Conformance....5
    1.4.1 SAML V2.0 Metadata Interoperability Profile....5
2 SAML V2.0 Metadata Interoperability Profile....6
  2.1 Required Information....6
  2.2 Profile Overview....6
  2.3 Metadata Acceptance....6
  2.4 Other Assumptions....6
  2.5 Metadata Producer Requirements....6
    2.5.1 Key Representation....7
  2.6 Metadata Consumer Requirements....7
    2.6.1 Key Processing....8
  2.7 Security Considerations....8
Appendix A. Acknowledgements....10
Appendix B. Revision History....11

1 Introduction

The SAML V2.0 metadata specification [SAML2Meta] defines an XML schema and a set of basic processing rules intended to facilitate the use of SAML profiles, and generally any profile or specification involving SAML. Practical experience has shown that the most complex aspects of implementing most SAML profiles, and obtaining interoperability between such implementations, are in the areas of provisioning federated relationships between deployments, and establishing the validity of cryptographic signatures and handshakes. Because the metadata specification was largely intended to solve those exact problems, additional profiling is needed to improve and clarify the use of metadata in addressing those aspects of deployment.

This profile is the product of the implementation experience of several SAML solution providers and has been widely deployed and successfully used in furtherance of the goal of scaling deployment beyond small numbers into the hundreds and thousands of sites, without sacrificing security.

Experience has shown that the most frustrating part of using SAML (and many similar technologies) is that products approach the use of cryptography and trust in wildly inconsistent ways, and often the libraries that such products depend on do the same in their own domains. Key management is hard, and often relies on complicated tools with cryptic output. Standards only help insofar as they can be understood and widely implemented; this has generally not occurred above a basic level of cryptographic correctness. A formal PKI is a tremendously complex, and some would say intractable, goal; it could be argued that SAML itself is a reaction to this. Often, the security of deployments is based on a presumption that required practices like revocation checking are being performed, when in fact they are not.

The purpose of this profile is to guarantee that in a correct implementation, all security considerations not deriving from the particular cryptography used (i.e. algorithm strength, key sizes) can be isolated to metadata exchange and acceptance, and not affect the runtime processing of messages. If a deployment can be shown to rely solely on metadata to derive trust, it can be reasoned about in a much simpler way, and the security exposures can be well understood.

Furthermore, this profile accomplishes a number of practical goals:

- simplifying ordinary implementations and deployments
- reducing the technical foundation required to understand and use implementations
- scaling the provisioning of federated relationships (via processing of metadata batches)
- radically simplifying interactions between existing federated deployments (i.e. interfederation)

Most importantly, these goals can be accomplished without sacrificing security. Too often, the reaction to security complexity is to produce competing approaches that start by rejecting the notion that a substantial degree of security is achievable in the general case.

Another benefit of this profile is to produce a greater awareness of the importance of securing the exchange of metadata. Deployers have sometimes tended to ignore this issue by falling back on the assumption that the underlying PKI would provide the real security of the system, resulting in other exposures due to insecure provisioning of other important information.

1.1 Notation

This specification uses normative text.

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:

Prefix	XML Namespace	Comments
`saml:`	urn:oasis:names:tc:SAML:2.0:assertion	This is the SAML V2.0 assertion namespace defined in the SAML V2.0 core specification [SAML2Core].
`md:`	urn:oasis:names:tc:SAML:2.0:metadata	This is the SAML V2.0 metadata namespace defined in the SAML V2.0 metadata specification [SAML2Meta].
`ds:`	http://www.w3.org/2000/09/xmldsig#	This is the XML Signature namespace [XMLSig].
`xsd:`	http://www.w3.org/2001/XMLSchema	This namespace is defined in the W3C XML Schema specification [Schema1]. In schema listings, this is the default namespace and no prefix is shown.
`xsi:`	http://www.w3.org/2001/XMLSchema-instance	This is the XML Schema namespace for schema-related markup that appears in XML instances [Schema1].

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

1.2 Normative References

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

[RFC3280] R. Housley, et al. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. IETF RFC 3280, April 2002. http://www.ietf.org/rfc/rfc3280.txt.

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

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

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

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

[Schema1] H. S. Thompson et al. *XML Schema Part 1: Structures.* World Wide Web Consortium Recommendation, May 2001. See http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/. Note that this specification normatively references [Schema2], listed below.

[Schema2] Paul V. Biron, Ashok Malhotra. *XML Schema Part 2: Datatypes*. World Wide Web Consortium Recommendation, May 2001. See http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/.

[XMLSig] D. Eastlake et al. *XML-Signature Syntax and Processing*. World Wide Web Consortium Recommendation, February 2002. See http://www.w3.org/TR/xmldsig-core/.

1.3 Non-Normative References

[RFC4346] T. Dierks, E. Rescorla. *The Transport Layer Security (TLS) Protocol Version 1.1*. IETF RFC 4346, April 2006. http://www.ietf.org/rfc/rfc4346.txt.

1.4 Conformance

1.4.1 SAML V2.0 Metadata Interoperability Profile

A metadata producer conforms to this profile if it can produce metadata consistent with the normative text in section 2.5.

A metadata consumer conforms to this profile if it can "accept" metadata in accordance with section 2.3 and process it consistent with the normative text in section 2.6.

2 SAML V2.0 Metadata Interoperability Profile

2.1 Required Information

Identification: `urn:oasis:names:tc:SAML:2.0:profiles:metadata-iop`

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

Description: Given below.

Updates: None.

2.2 Profile Overview

The SAML V2.0 profiles [SAML2Prof] and metadata [SAML2Meta] specifications, and subsequent profiles within OASIS and in other communities, describe the use of SAML metadata as a means of describing deployment capabilities and providing partners with information about endpoints, keys, profile support, processing requirements, etc.

This profile extends these practices by guaranteeing that a given metadata document will be consistently interpreted by any conforming implementation of higher level profiles. To this end, it requires that metadata be usable as a self-contained vehicle for communicating trust such that a user of a conforming implementation can be guaranteed that any and all rules for processing digital signatures, encrypted XML, and transport layer cryptography (e.g. TLS/SSL [RFC4346]) can be derived from the metadata alone, with no additional trust requirements imposed.

This profile requires that all runtime decisions are made solely on the basis of key comparisons, and not on any traditionally certificate-influenced basis. A signed metadata file following this specification is semantically equivalent to a PKI, hence there is little value in the additional layer of complexity provided by certificate validation as in [RFC3280]. Operational experience also shows that managing signed metadata is easier than managing a PKI of the corresponding size and scale.

2.3 Metadata Acceptance

This profile does not seek to constrain the method by which metadata is published or acquired, but only its content and interpretation. It is assumed that, subject to the security and deployment requirements of the participants, some means of exchanging metadata exists that results in the "acceptance" of metadata by a consumer. Acceptance in this profile is defined as an explicit treatment of everything in the metadata as "true", for the purposes of the metadata consumer's operational behavior.

In other words, this profile does not define how metadata is exchanged or how and why it is trusted, but rather assumes that it is exchanged and trusted, and proceeds from that starting point. Dynamic exchange (as described in [SAML2Meta]), manual exchange, the aggregation and signing of metadata by third parties, or any other mechanism, can be used in conjunction with this profile.

The rest of this profile deals with the requirements for producing metadata that will be accepted, and a consumer's obligations having accepted it.

2.4 Other Assumptions

An additional assumption is necessitated by the inability of SAML metadata to express authentication requirements of back-channel communications between SAML entities, such as the SAML SOAP binding [SAML2Bind]. In lieu of extending metadata to capture such requirements, this profile assumes that such communications are secured by means of some combination of TLS/SSL and digital signing. If this assumption does not hold, this profile might need supplementing in some scenarios.

2.5 Metadata Producer Requirements

A producer of metadata that adheres to this profile may be an actual participant in a SAML (or other) profile, or an aggregator of metadata describing many such participants. In either case, the content of the metadata itself is independent of its source and MUST stand alone as a description of the cryptographic requirements for securely communicating with the entity (or entities) described therein, to the extent that the constructs of the SAML V2.0 metadata specification [SAML2Meta] can express these requirements.

Subject to the constraints of the exchange mechanisms in use, a conforming metadata instance MAY be rooted by either an `<md:EntityDescriptor>` or `<md:EntitiesDescriptor>` element. This profile further applies to any `<md:RoleDescriptor>` element (or any derived elements and types) that may be included.

Within the context of a particular role (and the protocols it supports, as expressed in its `protocolSupportEnumeration` attribute), any and all cryptographic keys that are known to be valid at the time of metadata production MUST appear, each in its own `<md:KeyDescriptor>` element, with the appropriate `use` attribute (see section 2.4.1.1 of [SAML2Meta]). This includes not only signing and encryption keys, but also any keys used to establish mutual authentication with technologies such as TLS/SSL.

Signing or transport authentication keys intended for future use MAY be included as a means of preparing for migration from an older to a newer key (i.e. key rollover). Once an allowable period of time has elapsed (with this period dependent on deployment-specific policies), the older key can be removed, completing the change.

Expired keys (those not in use anymore by an entity, for reasons other than compromise) SHOULD be removed once the rollover process to a new key (or keys) has been completed.

Compromised keys MUST be removed from an entity's metadata. The metadata producer MUST NOT rely on the metadata consumer utilizing online or offline mechanisms for verifying the validity of a key (e.g. X.509 revocation lists, OCSP, etc.). The exact time by which a compromise is reflected in metadata is left to the requirements of the parties involved, the metadata's validity period (as defined by a `validUntil` or `cacheDuration` attribute), and the exchange mechanism in use.

2.5.1 Key Representation

Each key included in a metadata role MUST be placed within its own `<md:KeyDescriptor>` element and expressed using the `<ds:KeyInfo>` element within. One or more of the following representations within a `<ds:KeyInfo>` element MUST be present:

- `<ds:KeyValue>`
- `<ds:X509Certificate>` (child element of `<ds:X509Data>`)

In the case of the latter, only a single certificate is permitted. If both forms are used, then they MUST represent the same key.

Any other representation in the form of a `<ds:KeyInfo>` child element (such as `<ds:KeyName>`, `<ds:X509SubjectName>`, `<ds:X509IssuerSerial>`, etc.) MAY appear, but MUST NOT be required in order to identify the key (they are hints only).

In the case of an X.509 certificate, there are no requirements as to the content of the certificate apart from the requirement that it contain the appropriate public key. Specifically, the certificate MAY be expired, not yet valid, carry critical or non-critical extensions or usage flags, and contain any subject or issuer.

2.6 Metadata Consumer Requirements

A metadata consumer MUST have the ability to fully provision and configure itself based on the content of a metadata instance that it has accepted (see section 2.3), within the constraints of the information represented by the SAML V2.0 metadata specification [SAML2Meta] and any profiles that make use of it. A consumer need not provision policy that is outside the scope of metadata, but MUST have the ability to interoperate with the entities described by a metadata instance that it accepts, constrained by whatever default policies it applies.

Subject to the constraints of the exchange mechanism(s) in use, a metadata consumer MUST be able to process instances rooted with either an `<md:EntityDescriptor>` or `<md:EntitiesDescriptor>` element. When processing an `<md:EntitiesDescriptor>` element, each `<md:EntityDescriptor>` element contained within it MUST be processed in accordance with this profile (subject to their validity).

2.6.1 Key Processing

Each key expressed by a `<md:KeyDescriptor>` element within a particular role MUST be accepted when processing messages or assertions in the context of that role. Specifically, any signatures or transport communications (e.g. TLS/SSL sessions) verifiable with a signing key MUST be accepted, and any encryption keys found may be used to encrypt messages or assertions to the containing entity.

Subsequent to accepting a metadata instance, a consumer MUST NOT apply additional criteria of any kind on the acceptance, or validity, of the keys found within it or their use at runtime. Specifically, consumers SHALL NOT apply any online or offline techniques including, but not limited to, X.509 path validation or revocation lists, OCSP responders, etc.

The following key representations within a `<ds:KeyInfo>` element MUST be supported:

- `<ds:KeyValue>`
- `<ds:X509Certificate>` (child element of `<ds:X509Data>`)

In the case of the latter, a metadata consumer MUST extract the public key found in the certificate and MUST NOT honor, interpret, or make use of any of the information found in the certificate other than as an aid in identifying the appropriate key to try (based, for example, on information found at runtime in an XML digital signature's `<ds:KeyInfo>` element or the certificate presented by a transport peer).

A metadata consumer, when implementing authentication of a transport peer via TLS/SSL, MAY retain the checking of server certificate names (e.g. subject cn or subjectAltName) in accordance with [RFC3280], but even then it MUST accept a properly named certificate that contains a public key that corresponds to a valid key found in that peer's metadata, even if the exact certificate presented is not found in that metadata.

2.7 Security Considerations

A number of important exposures arise from the reliance on metadata alone to control runtime trust decisions.

Metadata becomes a critical tool for the revocation of compromised sites and keys, and all of the standard practices in the use of tools like CRLs become relevant to the consumption of metadata. The specification has the mechanisms to address these issues, but they have to be used. Specifically, metadata obtained via an insecure transport should be both signed, and should expire, so that consumers are forced to refresh it often enough to limit the damage from compromised information.

In addition, distributing signed metadata without an expiration over an untrusted channel (e.g. posting it on a public web site) creates an exposure. An attacker can corrupt the channel and substitute an old metadata file containing a compromised key and proceed to use that key together with other attacks to impersonate a site. Repeatedly expiring and reissuing the metadata limits the window of exposure, just as a CRL does.

A broad set of concerns arises in the dynamic exchange of metadata self-published by a site. In such cases, it may seem untenable to trust someone to properly identify their own key, and of course it may be. Rather than constraining the acceptance of that key, this profiles relies on securing the exchange and acceptance of the metadata. Traditional PKI protections can be applied to that document and/or its exchange, subsequently leveraging that protection to establish trust in the key within the metadata.

For example, when using the Well Known Location resolution profile [SAML2Meta], a producer may use an X.509 certificate to sign the metadata. This certificate can be bound to the metadata through its subject or subjectAltName (which might contain a SAML entityID). This ensures the appropriate key/name binding for the signature.

Appendix A. Acknowledgements

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

The editor would also like to acknowledge the following contributors:

- Walter Hoehn, University of Memphis
- Chad LaJoie, SWITCH
- Ian Young, EDINA, University of Edinburgh

Appendix B. Revision History

- Draft 01.