SAML Metadata Guidance Version 1.0

Working Draft 03

30 September 2014

Technical Committee:
OASIS Security Services (SAML) TC

Chairs:
Thomas Hardjono (hardjono@mit.edu), M.I.T.
Nathan Klingenstein (ndk@internet2.edu), Internet2

Editors:
Rainer Hörbe (rainer@hoerbe.at), Individual

Additional artifacts:
This prose specification is one component of a Work Product that also includes:
- XML schemas: (list file names or directory name)
- Other parts (list titles and/or file names)

Related work:
This specification is related to:
- Related specifications (hyperlink, if available)

Declared XML namespaces:
- list namespaces declared within this specification

Abstract:
Summary of the technical purpose of the document.

Status:
This Working Draft (WD) has been produced by one or more TC Members; it has not yet been voted on by the TC or approved as a Committee Draft (Committee Specification Draft or a Committee Note Draft). The OASIS document Approval Process begins officially with a TC vote to approve a WD as a Committee Draft. A TC may approve a Working Draft, revise it, and re-approve it any number of times as a Committee Draft.

URI patterns:
Initial publication URI:
http://docs.oasis-open.org/security/saml-metadata-guide/v1.0/csd01/saml-metadata-guide-v1.0-csd01.doc
Permanent “Latest version” URI:
http://docs.oasis-open.org/security/saml-metadata-guide/v1.0/saml-metadata-guide-v1.0.doc

(Managed by OASIS TC Administration; please don’t modify.)

Table of Contents

1 Introduction

This guide provides an overview of the SAML metadata specification, with a focus on frequently used structures and use cases. The intent is to reach the audience of architects, administrators and developers who want to become familiar with SAML Metadata before going into details with the specification in various normative specifications and XML schema documents.

1.1 Notation

Conventional XML namespace prefixes are used throughout this guide to stand for their respective namespaces as follows:

Prefix	XML Namespace	Comments
`alg:`	urn:oasis:names:tc:SAML:metadata:algsupport	The SAML V2.0 metadata extension namespace defined by [SAML2MetaAlgSup].
`aslo:`	urn:oasis:names:tc:SAML:2.0:protocol:ext:async-slo	The SAML V2.0 metadata extension namespace defined by [SAML-Async-SLO].
`idpdisc:`	urn:oasis:names:tc:SAML:profiles:SSO:idp-discovery-protocol	The SAML V2.0 metadata extension namespace defined by [IdpDisco].
`init:`	urn:oasis:names:tc:SAML:profiles:SSO:request-init	The SAML V2.0 metadata extension namespace defined by [SAML2ReqInit].
`md:`	urn:oasis:names:tc:SAML:2.0:metadata	The SAML V2.0 metadata extension namespace defined by [SAML2Meta].
`mdattr:`	urn:oasis:names:tc:SAML:metadata:attribute	The SAML V2.0 metadata extension namespace defined by [SAML2-MD-EA].
`mdrpi:`	urn:oasis:names:tc:SAML:metadata:rpi	The SAML V2.0 metadata extension namespace defined by [SAML2MetaRPI].
`mdui:`	urn:oasis:names:tc:SAML:metadata:ui	The SAML V2.0 metadata extension namespace defined by [SAML2MetaUI].

1.2 Normative References

Reference	Title	Schema File
[SAML2Meta]	Metadata for the OASIS Security Assertion Markup Language (SAML) V2.0	saml-schema-metadata-2.0.xsd
[SAML2MetaIOP]	SAML V2.0 Metadata Interoperability Profile Version 1.0	-
[SAML2MDext-QR]	Metadata Extension for SAML V2.0 and 3 V1.x Query Requesters	sstc-saml-metadata-ext-query.xsd
[IdpDisco]	Identity Provider Discovery Service Protocol and Profile	sstc-saml-idp-discovery.xsd
[SAML2MDext-EA]	SAML V2.0 Metadata Extension for Entity Attributes	sstc-metadata-attr.xsd
[SAML2Assur]	SAML V2.0 Identity Assurance Profiles	-
[SAML2ReqInit]	Service Provider Request Initiation Protocol and	sstc-request-initiation.xsd

	Profile	
[SAML2MetaAlgSup]	SAML v2.0 Metadata Profile for Algorithm Support	sstc-saml-metadata-algsupport-v1.0.xsd
[SAML2MetaRPI]	SAML V2.0 Metadata Extensions for Registration and Publication Information	saml-metadata-rpi-v1.0.xsd
[SAML2MetaUI]	SAML V2.0 Metadata Extensions for Login and Discovery User Interface	sstc-saml-metadata-ui-v1.0.xsd
[SAML2CB-Ext]	SAML V2.0 Channel Binding Extensions Version 1.0	sstc-saml-channel-binding-ext-v1.0.xsd
[SAML-Async-SLO]	SAML V2.0 Asynchronous Single Logout Profile Extension Version 1.0.	-
[SAML2-MD-EA]	SAML V2.0 Metadata Extension for Entity Attributes Version 1.0	sstc-metadata-attr.xsd
[RFCEntityCat]	The Entity Category SAML Entity Metadata Attribute Type - draft-young-entity-category-02	-
[FeideMaRequ]	Metadata Aggregation Requirements Specification, Andreas Solberg, 2010-01-05. Downloaded from https://rnd.feide.no/2010/01/05/metadata_aggregation_requirements_specification	-

NOTE: The proper format for citation of technical work produced by an OASIS TC (whether Standards Track or Non-Standards Track) is:

[Citation Label]

Work Product title (italicized). Edited by Albert Alston, Bob Ballston, and Calvin Carlson. Approval date (DD Month YYYY). OASIS Stage Identifier and Revision Number (*e.g.*, OASIS Committee Specification Draft 01). Principal URI (version-specific URI, *e.g.*, with stage component: somespec-v1.0-csd01.html). Latest version: (latest version URI, without stage identifiers).

For example:

[OpenDoc-1.2] *Open Document Format for Office Applications (OpenDocument) Version 1.2*. Edited by Patrick Durusau and Michael Brauer. 19 January 2011. OASIS Committee Specification Draft 07. http://docs.oasis-open.org/office/v1.2/csd07/OpenDocument-v1.2-csd07.html. Latest version: http://docs.oasis-open.org/office/v1.2/OpenDocument-v1.2.html.

2 Use Cases

SAML metadata is configuration data used to automatically negotiate agreements between SAML system entities in a trustworthy manner. The registration, aggregation and publication of metadata for participating entities are fundamental services in a federation.

Metadata comprises machine-readable data about entities such as identifiers, protocol and binding support, endpoints, certificates, keys, cryptographic capabilities and security and privacy policies. Several SAML specifications standardize metadata structures and contents, thus supporting interoperable and scaleable deployments.

As SAML specifications evolved over time metadata related specifications are contained in several documents. This non-normative document provides a consolidated overview and with links to the normative specifications.

2.1 Actors

The actors in a federation are federation operators, identity providers, service providers and Discovery Services. To create a specific design for the metadata workflows, [FeideMaRequ] introduced additional terms for the metadata management perspective:

- Metadata publishers (either for a single entity or an aggregated set)
- Metadata consumer
- Metadata aggregator (filters and aggregates)
- Metadata registrar (administers and validates)

A basic federation constellation has the federation operator acting as metadata registrar, aggregator and Publisher. System entities (respectively their operators) play both metadata publisher and consumer as depicted in Fig. 1.

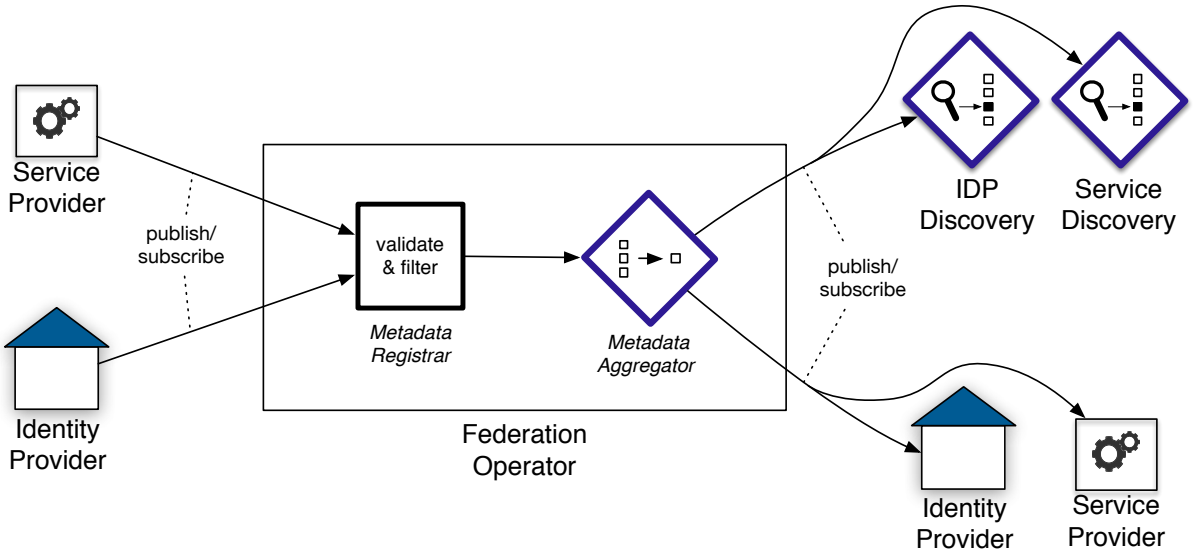


Fig. 1 Baseline constellation and flow of metadata in a federation

In interfederation scenarios entities are aggregated from multiple publishers. Fig. 2 shows a constellation where multiple federations feed their metadata into a single aggregator. Fig. 3 shows an enterprise that exports entities to a federation and consumes metadata from the federation.

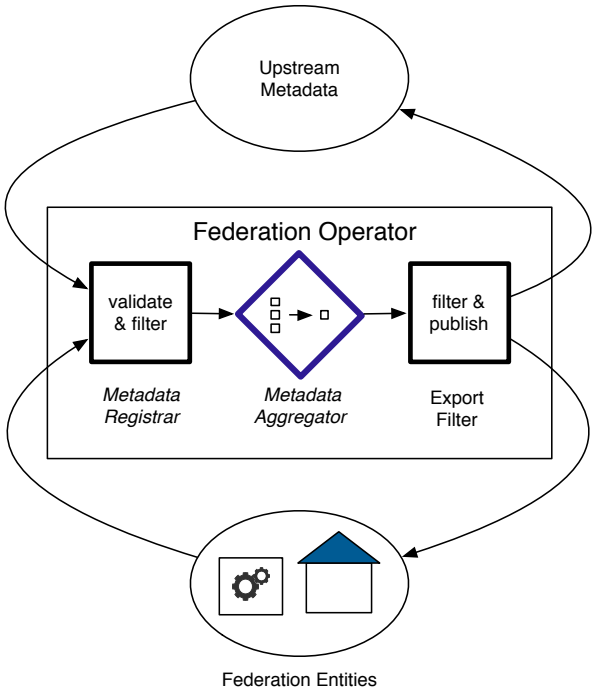


Fig. 2 Federation of federations

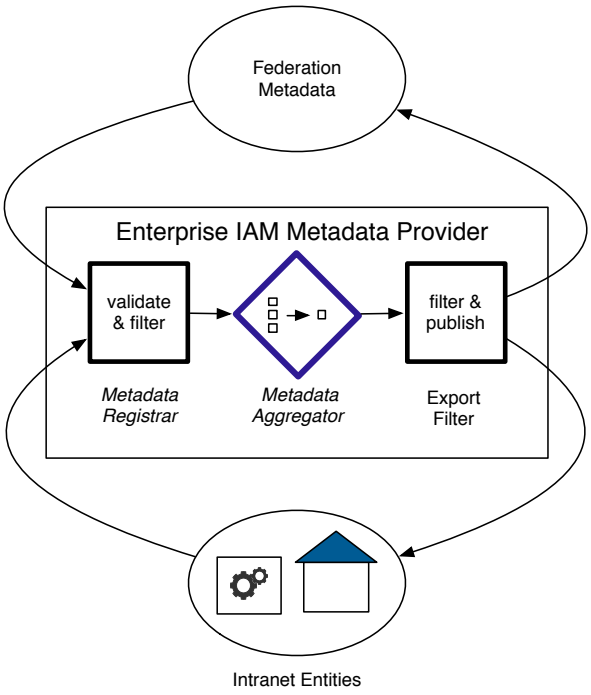


Fig. 3 Enterprise and federation metadata

2.1.1 Use Case Overview

2.1.1.1 Register an Entity

Prerequisite: The operator of the entity and the federation operator have established a trust relationship. This may range from a simple domain-validated certificate to carefully controlled key material that was exchanged out-of-band.

The primary actor is the operator of an entity having the goal to join the federation.

Process: The operator will provide the metadata that will be proofed and validated by the registrar according to the federation's registration policy. There might be agreements outside the metadata, such as commercial or service level contracts.

If the registration is successful the entity is considered a participant in the federation, and its metadata is published on one or more metdata feeds.

Extension: Update Entity

Goal: To make changes in technical capabilites, contact data etc. known to the federation, the "Register Entity" use case is extended to allow updates instead of the submission of a new entity. Otherwise the procedure is identical. This process is also used for key rollover.

Extension: Terminate Entity

Process: The operator sumits a termination request to the registrar, who will then remove the entity from the metadata feeds. This use case may be triffered by the federation operator as well.

2.1.1.2 Consume a Peer's EntityDescriptor

Prerequisite: Two entities that want to exchange SAML messages are registered in metadata.

Goal: Know the peer's configuration without bilateral exchange of configuration data.

Process: The primary actor is the system entity that needs to access the metadata of a peer entity to support a SAML protocol message exchange. Metadata will ususally be obtained in advance. The peer's metadata will typically provide following information:

Configuration/access decision aspect	*Metadata structure*
Is the entity participant in the federation?	`entityID` attribute

Available protocols and bindings; the respective endpoints	`<idpdisc:DiscoveryResponse>`
	`<init:RequestInitiator>`
	`<md:ArtifactResolutionService>`
	`<md:AssertionConsumerService>`
	`<md:AttributeConsumingService>`
	`<md:SingleLogoutService>`
	`<md:SingleSignOnService>`

Organization operating the entity, Point of contact if there are issues	`<md:Organization>` `<md:ContactPerson>`
How is technical trust established?	`<md:KeyDescriptor>`
Available algorithms for cryptographic operations, cipher flexibility	`<alg:DigestMethod>` `<md:EncryptionMethod>` `<alg:SigningMethod>`
Other conditions for technical trust	`WantAuthnRequestsSigned` and `AuthnRequestsSigned` attributes
Semantics of the principal's identifier	`<NameIDFormat>`
Is there an SP's claim on which an IDP can release attributes and be compliant with the privacy rule of purposeful data collection?	`<mdattr:EntityAttributes>/<saml:Attribute Name="http://macedir.org/entity-category">`
How can the entity trust the metadata?	`<ds:Signature>` and `validUntil` attribute

2.1.1.3 Provide IDP Discovery Service

Goal: IDP discovery services – whether as a central service or SP-local – shall be completely based on metadata.

Procedure: SAML metadata provides the extensions in the `mdui:` namespace to describe IDPs and adds related data such as discovery hints.

2.1.1.4 Provide Federation-wide Service Discovery Service

Goal: A common service discovery service shall list all available services. User selecting a service will authenticate using SP-first flows.

SAML metadata provides the extensions in the `mdui:` namespace to describe entities.

Note: There is no target URL element in the metadata. If a request-initiation-protocoal message is sent to the SP, it is up to the SP to redirect the browser to the appropriate start page.

2.1.1.5 Provide User-specific Service Discovery Service

Goal: A service discovery service (typically co-located with an IDP) shall list only accesible services to authenticated user and support IDP-first flows.

SAML metadata provides the extensions in the `mdui:` namespace to describe entities.

Note: There is no target URL element in the metadata. If an unsolicited response, or a request-initiation-protocoal message is sent to the SP, it is up to the SP to redirect the browser to the appropriate start page.

2.1.1.6 Use Case "Aggregate Metadata"

Precondition: A federation has relationships to other federations.

Goal: Make entities from other federations available.

Procedure: The Federation Operator is the primary actor and facilitates the filtering and import of entities from other federation. This is called aggregation.

Metadata needs to be signed by the Federation Operator after pruning existing signatures from the upstream metadata feed.

2.1.1.7 Use Case "Publish Metadata"

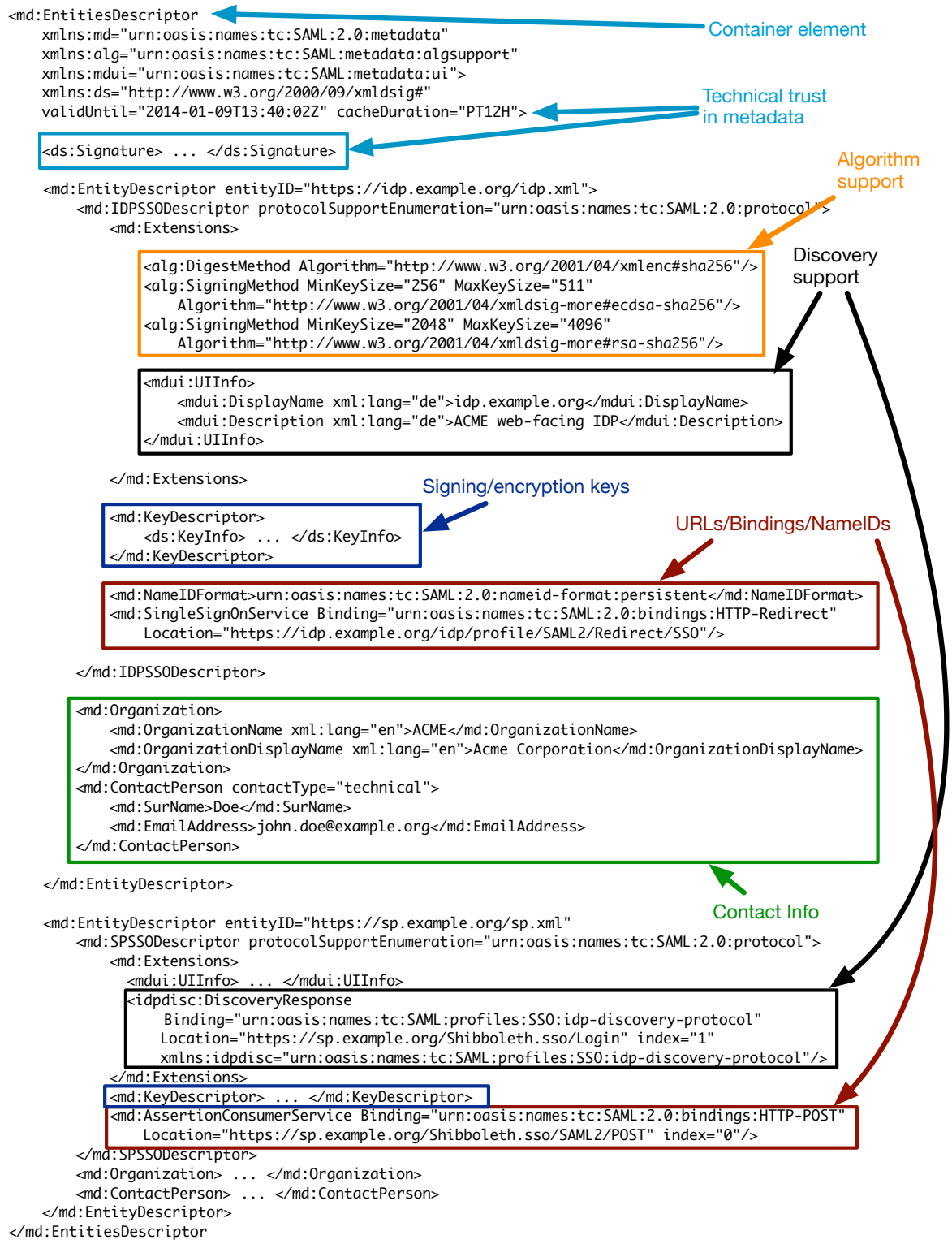
Goal: Make aggregated metadata available to participants.

Procedure: In the standard scenario the metadata signing key is the trust root. Therefore the Federation Operator will remove existing signatures. If metadata is published at a well-known location it will be made available as single XML infoset with a signature at the *EntitiesDescriptor* level. In case of publishing the metadata as an MDX service each *EntityDescriptor* can be signed or TLS can be used as trusted channel.

As the trust path ist validated using metadata, clients shall ignore any X.509-specific attributes. [SAML2MetaIOP]

3 Metadata Structure

3.1 Introduction by Example



3.2 Salient Elements

3.2.1 AttributeAuthorityDescriptor

The use of the <md:AttributeAuthorityDescriptor> role is generally a compatibility requirement for supporting legacy or other SPs that rely on queries for attributes. Another use case is Backend Attribute Exchange, where users authenticate directly to a SP, which will obtain attributes from the Attribute Authority. In most cases, much of the role content will be identical with that from <md:IDPSSODescriptor>.

The order of all this information is significant, which you can refer to the schema for, but the most common elements included would be present in the following order:

- <md:KeyDescriptor> (can be omitted, but rarely)
- <md:AttributeService> (always at least one)
- <md:NameIDFormat> (if any)
- <md:Attribute> (rare today, but may be reasonable to include)

Reference: [SAML2Meta]

3.2.2 AffiliationDescriptor

An affiliation is a list of Service Providers that will receive the same identifier (SAML 2.0 persistent NameID) for a specific user.

Reference: [SAML2Meta]

3.2.3 AttributeConsumingService

SPs support SSO protocols by including one or more <md:AssertionConsumerService> endpoint elements in their metadata. These are the locations to which the IDP will eventually send the user at the SP. By enumerating them in the metadata, the IDP can ensure that the user's information is sent only to authorized locations. However, as attribute release is frequently restricted by privacy considerations, the use of entity categories (see below) may be a more appropriate way than listing requested attributes in the SP's EntityDescriptor.

For technical reasons, these endpoint elements have to carry an additional index XML attribute, which should generally contain a small positive integer. The index values should be unique across all the like-named elements within the role.

Metadata can be used to advertise "service offerings" that can be referenced for service discovery, possibly grouped with an IDP-initiated authentication. An SP can define multiple services or service levels, with accompanying human-readable descriptions, to drive the development of IDP-provided service discovery user interfaces.

Reference: [SAML2Meta]

3.2.4 ContactPerson

Below the <md:EntityDescriptor> (or <md:RoleDescriptor>) level, there are optional <md:ContactPerson> and <md:Organization> elements that provide information about who is standing up a particular entity and how to contact them.

For testing purposes, one will rarely if ever need to supply these elements, but they may be needed for production use. Organization metadata in particular often gets used by other software systems that consume metadata in order to present lists of entities with human-readable names. Examples of such systems include IDPDiscovery services or software to assist users in granting consent for login and release of attributes to SPs.

Reference: [SAML2Meta]

3.2.5 Endpoint

The EndpointType describes a protocol binding endpoint at which a SAML entity can be sent protocol messages. Various protocol or profile-specific metadata elements are bound to this instances of this type, using the Binding, Location and ResponseLocation attributes.

Examples for specific instances are

- `<md:SingleSignOnService>`
- `<md:SingleLogoutService>`
- `<md:AttributeService>`

Reference: [SAML2Meta]

3.2.6 EntityAttribute

The `<md:EntityAttributes>` element provides a generic extension mechanism for carrying an arbitrary set of attribute information for an `<md:EntityDescriptor>` or a group thereof. It is typically used for entity categories.

Reference: [SAML2MDext-EA]

3.2.7 EntityCategory

An EntityCategory assigns category membership semantics to an entity. Categories have been defined for SPs that conform to a data protection Code of Conduct, or are members of a group that is eligible to receive a certain set of user attributes.

Note: As an alternative to organize attribute release via metadata is a list of `<md:RequestedAttribute>` elements as child elements of an `<AttributeConsumingService>` can be included in SP metadata. This is recommended if a small number of attributes are used for a specific service.

Reference: [RFCEntityCat]

3.2.8 EntitiesDescriptor

`<md:EntitiesDescriptor>` is a container, which usually wraps one or more elements of the type `<md:EntityDescriptor>`. It enables a set of IDPs or SPs to be described at once, and then signed as a unit. This is a common way for federations to supply metadata about their members. It is also frequently used in deployments where metadata has to be maintained more or less by hand, because it allows a bunch of information to be maintained in one file, with the entire set of metadata supplied to an IDP or SP with a single configuration element. It is usually easier that way than to individually supply metadata about a single entity at a time using multiple metadata sources.

Reference: [SAML2Meta]

3.2.9 EntityDescriptor

An `<md:EntityDescriptor>` represents a system entity in metadata, which is a SAML service, such as an IDP or an SP. Each entity has an entityID, that is a unique name that distinguishes it from any other entity.

The value of the entityID attribute SHOULD be the canonical URL of the entity's metadata document. Canonical URLs follow the semantic-preserving normalization as specified in RFC 3986 section 6. The entityID should accurately reflect the organization that owns the entity.

Reference: [SAML2Meta]

3.2.10 IDPSSODescriptor

An IDP role typically includes the following descriptive information:

- Public key(s) used by the IDP for authentication and encryption
- Endpoints of various types for communicating with it
- Explicitly supported identifier formats, if any
- Explicitly supported attributes, if any

The order of all this information is significant, which you can refer to the schema for, but the most common elements included would be present in the following order:

- `<md:Extension>` (optional IDP discovery and algorithm support)
- `<md:KeyDescriptor>` (can be omitted, but rarely)
- `<md:ArtifactResolutionService>` (only needed if supporting response by artifact)
- `<md:SingleLogoutService>` (if any)
- `<md:NameIDFormat>` (if any)
- `<md:SingleSignOnService>` (always at least one)

Reference: [SAML2Meta]

3.2.11 KeyDescriptor

A common use of metadata across both IDP and SP roles, among others, is to associate one or more public keys with the system being defined. The `<md:KeyDescriptor>` element is a wrapper around the XML Signature-defined <ds:KeyInfo> element, an extensible container for describing keys.

Because this container is so extensible, there is no prescribed behavior for the use of this element within metadata. Broadly speaking, there are two general approaches: inline and indirect.

Inline Keys

The approach that has been standardized as part of [SAML2MetaIOP] at OASIS is to directly express an explicitly trusted public key using either a `<ds:KeyValue>` or `<ds:X509Certificate>` element. Any number of keys can be included to manage key rollover, and individual keys can be labeled for authentication, encryption, or both.

Note: These keys are used for message-level signing and encryption, and to create secure back channels for transporting SAML messages with SOAP-binding over SSL/TLS. They are **not** used for browser-facing SSL/TLS connections on port 443.

A full description of the security implications of metadata usage is beyond the scope of this material, but be advised that this profile generally requires the enclosing metadata include an expiration time using a validUntil XML attribute. This prevents older signed metadata containing retired or compromised keys from being accepted.

Indirect Key References

The indirect approach involves describing a public key for use as input to a separately-controlled trust evaluation process. This is common to commercial SAML implementations, and may include a wide range of approaches to representing a key, including key "names" using `<ds:KeyName>` or `<ds:X509Subject>` elements, certificate references using the `<ds:X509IssuerSerial>` element, or an actual certificate that is subjected to additional validation using other rules defined outside of metadata.

Reference: [SAML2Meta], [SAML2MetaIOP]

3.2.12 Organization

This is an optional element identifying the organization responsible for the SAML entity described by the element.

Reference: [SAML2Meta]

3.2.13 Roles

Below the <md:EntityDescriptor>, the main information unit is the "role". An <md:EntityDescriptor> contains one or more <md:RoleDescriptors>, which is a super-type for roles such as <md:IDPSSODescriptor> and <md:AttributeAuthorityDescriptor> in the case of metadata about IDPs and <md:SPSSODescriptor> in the case of metadata about SPs.

One important piece of information common to all role elements is the protocolSupportEnumeration attribute, which MUST be present. This attribute contains a space-delimited collection of URIs that represent general classes of protocol support for the role in question. There are URIs defined by the various standards and profiles to represent the fact that an entity acting in a role "supports" a particular protocol family, such as SAML 2.0.

Reference: [SAML2Meta]

3.2.14 SPSSODescriptor

An SP role typically includes the following descriptive information:

- The public key(s) used by the SP for authentication and encryption
- Endpoints of various types for communicating with it
- Explicitly supported identifier formats, if any
- Descriptions of the "services" offered by the SP and the SAML attributes required by them, possibly abstracted into a entity category.

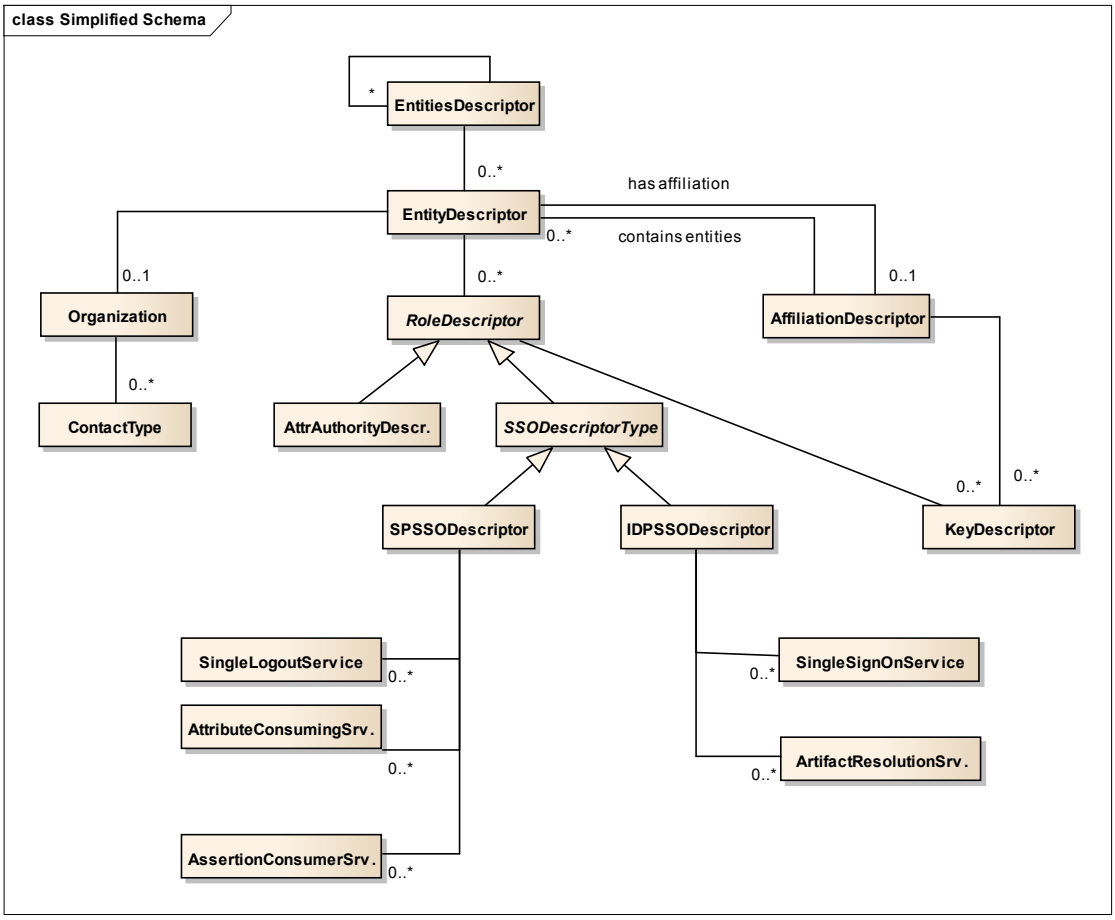
The order of all this information is significant, which you can refer to the schema for, but the most common elements included would be present in the following order:

- <md:Extension> (optional service discovery, algorithm support and entity category)
- <md:KeyDescriptor> (can be omitted, but rarely)
- <md:SingleLogoutService> (if any)
- <md:NameIDFormat> (if any)
- <md:AssertionConsumerService> (always at least one)
- <md:AttributeConsumingService> (rare today, but good practice to include)

Reference: [SAML2Meta]

3.3 Relationships

The diagram below gives an abridged view on the relationship between the more important SAML metadata elements in UML class diagram notation.



Appendix A. Acknowledgments

Material has been used from the Shibboleth Wiki [1]. On behalf of Internet2 and the Shibboleth Consortium, Scott Cantor stated [2] that this material, normally available under a Creative Commons license [3], may be used by OASIS SSTC on the royalty-free terms under which the TC operates.

[1] https://wiki.shibboleth.net/confluence/display/SHIB2/Metadata

[2] https://lists.oasis-open.org/archives/security-services/201409/msg00004.html

[3] http://creativecommons.org/licenses/by-sa/3.0/

Participants:

[Participant Name, Affiliation | Individual Member]
[Participant Name, Affiliation | Individual Member]

Appendix B. Revision History

Revision	Date	Editor	Changes Made
[WD01]	[21 July 2014]	[Rainer Hörbe]	[Initial draft]