



2 SAML V2.0 Metadata Interoperability Profile 3 Version 2.0

4 Working Draft 01 5 2 March 2010

6 Specification URIs:

7 This Version:

8 TBD

9 Previous Version:

10 TBD

11 Latest Version:

12 <http://docs.oasis-open.org/security/saml/Post2.0/sstc-metadata-iop-2.0.html>

13 <http://docs.oasis-open.org/security/saml/Post2.0/sstc-metadata-iop-2.0.odt> (Authoritative)

14 <http://docs.oasis-open.org/security/saml/Post2.0/sstc-metadata-iop-2.0.pdf>

15 Latest Approved Version:

16 TBD

17 Technical Committee:

18 OASIS Security Services TC

19 Chair(s):

20 Thomas Hardjono, M.I.T.

21 Editors:

22 Scott Cantor, Internet2

23 Related Work:

24 This specification supercedes the SAML V2.0 Metadata Interoperability Profile [MetalOP].

25 Abstract:

26 This profile describes a set of rules for SAML metadata producers and consumers to follow such
27 that federated relationships can be interoperably provisioned, and controlled at runtime in a
28 secure, understandable, and self-contained fashion. Runtime use of assymmetric keys and/or
29 Kerberos credentials is included.

30 Status

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

34 TC members should send comments on this specification to the TC's email list. Others
35 should send comments to the TC by using the "Send A Comment" button on the TC's
36 web page at <http://www.oasis-open.org/committees/security>.
37 For information on whether any patents have been disclosed that may be essential to
38 implementing this specification, and any offers of patent licensing terms, please refer to the IPR
39 section of the TC web page (<http://www.oasis-open.org/committees/security/ipr.php>).
40 The non-normative errata page for this specification is located at [http://www.oasis-
open.org/committees/security](http://www.oasis-
41 open.org/committees/security).

42 Notices

43 Copyright © OASIS Open 2010. All Rights Reserved.

44 All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual
45 Property Rights Policy (the "OASIS IPR Policy"). The full Policy may be found at the OASIS website.

46 This document and translations of it may be copied and furnished to others, and derivative works that
47 comment on or otherwise explain it or assist in its implementation may be prepared, copied, published,
48 and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice
49 and this section are included on all such copies and derivative works. However, this document itself may
50 not be modified in any way, including by removing the copyright notice or references to OASIS, except as
51 needed for the purpose of developing any document or deliverable produced by an OASIS Technical
52 Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be
53 followed) or as required to translate it into languages other than English.

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

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

61 OASIS requests that any OASIS Party or any other party that believes it has patent claims that would
62 necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to
63 notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such
64 patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced
65 this specification.

66 OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any
67 patent claims that would necessarily be infringed by implementations of this specification by a patent
68 holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR
69 Mode of the OASIS Technical Committee that produced this specification. OASIS may include such
70 claims on its website, but disclaims any obligation to do so.

71 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that
72 might be claimed to pertain to the implementation or use of the technology described in this document or
73 the extent to which any license under such rights might or might not be available; neither does it represent
74 that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to
75 rights in any document or deliverable produced by an OASIS Technical Committee can be found on the
76 OASIS website. Copies of claims of rights made available for publication and any assurances of licenses
77 to be made available, or the result of an attempt made to obtain a general license or permission for the
78 use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS
79 Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any
80 information or list of intellectual property rights will at any time be complete, or that any claims in such list
81 are, in fact, Essential Claims.

82 The name "OASIS" is a trademark of OASIS, the owner and developer of this specification, and should be
83 used only to refer to the organization and its official outputs. OASIS welcomes reference to, and
84 implementation and use of, specifications, while reserving the right to enforce its marks against
85 misleading uses. Please see <http://www.oasis-open.org/who/trademark.php> for above guidance.

86 **Table of Contents**

87	1 Introduction.....	5
88	1.1 Notation.....	6
89	1.2 Normative References.....	6
90	1.3 Non-Normative References.....	7
91	2 SAML V2.0 Metadata Interoperability Profile.....	8
92	2.1 Required Information.....	8
93	2.2 Profile Overview.....	8
94	2.3 Metadata Exchange and Acceptance.....	8
95	2.4 Implementation Constraints.....	9
96	2.4.1 Peer Authentication.....	9
97	2.5 Metadata Producer Requirements.....	9
98	2.5.1 Key and Kerberos Principal Name Representation.....	9
99	2.6 Metadata Consumer Requirements.....	10
100	2.6.1 Key and Kerberos Principal Name Processing.....	10
101	2.7 Security Considerations.....	11
102	3 Conformance.....	13
103	3.1 SAML V2.0 Metadata Interoperability Profile Version 2.0.....	13
104	3.1.1 Public Key Conformance Mode.....	13
105	3.1.1.1 Strict Public Key Conformance Mode.....	13
106	3.1.2 Kerberos Conformance Mode.....	13
107	Appendix A. Acknowledgements.....	14
108	Appendix B. Revision History.....	15
109		

1 Introduction

110

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

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

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

131 Of course, it is the case that some deployments, at least to date, have overcome some or all of these
132 problems. They may have a mature PKI, possibly one that existed long before their use of SAML, or they
133 may require such a PKI for other purposes. In such cases, it is obviously less beneficial to deploy a
134 second trust infrastructure based on SAML metadata. Another factor in this profile's usefulness is the
135 relationship between SAML and the other security technologies involved in a deployment; if the use of
136 SAML is subordinated to a secondary role, this profile may be less applicable.

137 The purpose of this profile is to guarantee that in a correct implementation, all security considerations not
138 deriving from the particular cryptography used (i.e., algorithm strength, key sizes) can be isolated to
139 metadata exchange and acceptance, and not affect the runtime processing of messages. In other words,
140 given a metadata instance and presuming that it is successfully processed and has not been updated or
141 superseded, it should be possible with no other information supplied to determine whether a given
142 credential (e.g., a key or certificate) will be accepted by an implementation when used to secure a SAML
143 protocol or assertion.

144 If an implementation can be shown to rely solely on the acceptance of metadata to derive trust, it can be
145 reasoned about in a much simpler way, and the security exposures can be well understood. Furthermore,
146 this profile accomplishes a number of additional practical goals:

- 147 • simplifying ordinary implementations and deployments
- 148 • reducing the technical foundation required to understand and use implementations
- 149 • scaling the provisioning of federated relationships (via processing of metadata batches)
- 150 • facilitating the use of XML encryption without dependency on weaker methods for establishing
151 knowledge of public keys (e.g., guessing based on TLS server certificates)
- 152 • radically simplifying interactions between existing federated deployments (i.e. interfederation)

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

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

160 Finally, note that, in addition to SAML V2.0 itself, this profile is applicable to any set of use cases
161 supported by SAML metadata, including SAML V1.x profiles (as in [SAML1Meta]) and any other
162 specifications that may profile SAML metadata.

163 1.1 Notation

164 This specification uses normative text.

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

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

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

173 Listings of XML schemas appear like this.

174 Example code listings appear like this.

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

Prefix	XML Namespace	Comments
saml:	urn:oasis:names:tc:SAML:2.0:assertion	This is the SAML V2.0 assertion namespace [SAML2Core].
md:	urn:oasis:names:tc:SAML:2.0:metadata	This is the SAML V2.0 metadata namespace [SAML2Meta].
krb:	urn:oasis:names:tc:SAML:2.0:attribute:kerberos	This is the SAML V2.0 Kerberos Attribute Profile Version 1.0 namespace [KrbAttr].
ds:	http://www.w3.org/2000/09/xmldsig#	This is the XML Signature namespace [XMLSig].

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

181 1.2 Normative References

- 182 **[KrbAttr]** OASIS Committee Draft, *SAML V2.0 Kerberos Attribute Profile Version 1.0*,
183 December 2009. <http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-attribute-kerberos.pdf>
184
- 185 **[RFC2119]** S. Bradner. *Key words for use in RFCs to Indicate Requirement Levels*. IETF
186 RFC 2119, March 1997. <http://www.ietf.org/rfc/rfc2119.txt>
- 187 **[RFC2818]** E. Rescorla. *HTTP Over TLS*. IETF RFC 2818, May 2000.
188 <http://www.ietf.org/rfc/rfc2818.txt>

189 **[RFC4120]** C. Neuman et al. *The Kerberos Network Authentication Service (V5)*. IETF RFC
190 4120, July 2005. <http://www.ietf.org/rfc/rfc4120.txt>

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

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

197 **[SAML2Errata]** OASIS Approved Errata, *SAML V2.0 Errata*, October 2009. [http://docs.oasis-
198 open.org/security/saml/v2.0/sstc-saml-approved-errata-2.0.pdf](http://docs.oasis-open.org/security/saml/v2.0/sstc-saml-approved-errata-2.0.pdf)

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

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

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

208 **1.3 Non-Normative References**

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

211 **[RFC5280]** D. Cooper, et al. *Internet X.509 Public Key Infrastructure Certificate and*
212 *Certificate Revocation List (CRL) Profile*. IETF RFC 5280, May 2008.
213 <http://www.ietf.org/rfc/rfc5280.txt>

214 **[SAML1Meta]** OASIS Standard, *Metadata Profile for the OASIS Security Assertion Markup*
215 *Language (SAML) V1.x*, November 2007. [http://docs.oasis-
216 open.org/security/saml/Post2.0/sstc-saml1x-metadata-os.pdf](http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml1x-metadata-os.pdf)

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

220 **2 SAML V2.0 Metadata Interoperability Profile**

221 **2.1 Required Information**

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

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

224 **Description:** Given below.

225 **Updates:** SAML V2.0 Metadata Interoperability Profile [MetaIOP].

226 **2.2 Profile Overview**

227 The SAML V2.0 profiles [SAML2Prof] and metadata [SAML2Meta] specifications, and subsequent profiles
228 within OASIS and in other communities (e.g., [SAML1Meta]), describe the use of SAML metadata as a
229 means of describing deployment capabilities and providing partners with information about endpoints,
230 keys, profile support, processing requirements, etc.

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

237 This profile requires that all runtime decisions are made on the basis of key or Kerberos principal name
238 comparisons, and not on any traditionally certificate-influenced basis. This permits a signed metadata file
239 that conforms to this specification to be semantically equivalent to an X.509-based public key
240 infrastructure (PKI); hence there is little value in the additional layer of complexity provided by certificate
241 validation as specified in [RFC5280]. Operational experience also shows that managing signed metadata
242 is easier than managing a PKI of the corresponding size and scale.

243 This revision of the specification introduces the capability to use a conventional Kerberos [RFC4120]
244 infrastructure.

245 **2.3 Metadata Exchange and Acceptance**

246 This profile does not constrain the method(s) by which metadata is published or acquired, but only its
247 content and interpretation. It is assumed that, subject to the security and deployment requirements of the
248 participants, some means of exchanging metadata exists that results in the "acceptance" of metadata by a
249 consumer. Acceptance in this profile is defined as an explicit treatment of everything in the metadata as
250 "true", for the purposes of the metadata consumer's operational behavior. The truth of a given set of
251 metadata is of course contingent upon the metadata not being superseded by newer metadata, which may
252 conflict with, and therefore render obsolete, the earlier information.

253 In other words, this profile does not define how (or how often) metadata is exchanged or how and why it is
254 trusted, but rather assumes that it is exchanged and trusted, and proceeds from that starting point.
255 Dynamic exchange (as described in [SAML2Meta]), manual exchange, the aggregation and signing of
256 metadata by third parties, or any other mechanism, can be used in conjunction with this profile. Note that
257 verification of metadata signatures, if applicable, is considered to be part of this prerequisite step.

258 The rest of this profile deals with the requirements for producing metadata, and a conformant consumer's
259 obligations having accepted it.

260 Finally, note that accepting metadata does not imply that a relying party will interoperate with a specific
261 asserting party; it implies only that if it does so, it does so in a predictable fashion based on the metadata it
262 accepts about that party.

263 **2.4 Implementation Constraints**

264 **2.4.1 Peer Authentication**

265 An additional constraint is necessitated by the inability of SAML metadata to express the authentication
266 requirements of back-channel communications between SAML-using entities, such as via the SAML
267 SOAP binding [SAML2Bind]. In lieu of extending metadata to capture such requirements, this profile
268 assumes that such communications are secured by means of some combination of TLS/SSL and digital
269 signing. If this assumption cannot be made, this profile might need to be supplemented in such scenarios.

270 **2.5 Metadata Producer Requirements**

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

276 Subject to any constraints of the exchange mechanisms in use, a conforming metadata instance may be
277 rooted by either an `<md:EntityDescriptor>` or `<md:EntitiesDescriptor>` element. Any
278 `<md:RoleDescriptor>` element (or any derived element or type) appearing in the metadata instance
279 **MUST** conform to this profile's requirements.

280 Within the context of a particular role (and the protocols it supports, as expressed in its
281 `protocolSupportEnumeration` attribute), any and all cryptographic keys and Kerberos principal
282 names that are known by the producer to be valid at the time of metadata production **MUST** appear within
283 that role's element, in the manner described below in section 2.5.1. This includes not only signing and
284 encryption keys, but also any keys used to establish mutual authentication with technologies such as
285 TLS/SSL.

286 Signing or transport authentication keys or Kerberos principal names intended for future use **MAY** be
287 included as a means of preparing for migration from an older to a newer key or Kerberos principal name
288 (i.e., key rollover or change of Kerberos principal name). Once an allowable period of time has elapsed
289 (with this period dependent on deployment-specific policies), the older key or Kerberos principal name can
290 be removed, completing the change. Expired keys or Kerberos principal names (those not in use anymore
291 by an entity, for reasons other than compromise) **SHOULD** be removed once the rollover process to a
292 new key (or keys) or new Kerberos principal name (or names) has been completed.

293 Compromised keys **MUST** be removed from an entity's metadata. A Kerberos principal whose secret key
294 has been compromised **MUST** also have its name removed from an entity's metadata until a new secret
295 key has been securely established. The metadata producer **MUST NOT** rely on the metadata consumer
296 utilizing online or offline mechanisms for verifying the validity of a key (e.g., X.509 revocation lists, OCSP,
297 etc.). The exact time by which a compromise is reflected in metadata is left to the requirements of the
298 parties involved, the metadata's validity period (as defined by a `validUntil` or `cacheDuration`
299 attribute), and the exchange mechanism in use.

300 **2.5.1 Key and Kerberos Principal Name Representation**

301 Each key or Kerberos principal name included in a metadata role **MUST** be placed within its own
302 `<md:KeyDescriptor>` element, with the appropriate `use` attribute (see section 2.4.1.1 of [SAML2Meta],
303 as revised by E62 in [SAML2Errata]), and expressed using the `<ds:KeyInfo>` element.

304 One or more of the following representations within a `<ds:KeyInfo>` element MUST be present:

- 305 • `<ds:KeyValue>`
- 306 • `<ds:X509Certificate>` (child element of `<ds:X509Data>`)
- 307 • `<krb:KerberosSname>` (child element of `<krb:KerberosData>`)
- 308 • `<krb:KerberosCname>` (child element of `<krb:KerberosData>`)

309 In the case of `<ds:X509Certificate>`, only a single certificate is permitted. If both `<ds:KeyValue>`
310 and `<ds:X509Certificate>` are used, then they MUST represent the same key.

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

314 In the case of an X.509 certificate, there are no requirements as to the content of the certificate apart from
315 the requirement that it contain the appropriate public key. Specifically, the certificate may be expired, not
316 yet valid, carry critical or non-critical extensions or usage flags, and contain any subject or issuer. The use
317 of the certificate structure is merely a matter of notational convenience to communicate a key and has no
318 semantics in this profile apart from that. However, it is RECOMMENDED that certificates be unexpired.

319 In the case of a Kerberos principal name, there are no special requirements except that the Kerberos
320 principal MUST exist within the claimed realm. The `<krb:KerberosCname>` representation MUST only
321 be used for metadata roles that sign or encrypt plaintext. The `<krb:KerberosSname>` representation
322 MUST only be used for metadata roles that validate signatures or decrypt ciphertext.

323 2.6 Metadata Consumer Requirements

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

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

334 2.6.1 Key and Kerberos Principal Name Processing

335 Each key or Kerberos principal name expressed by a `<md:KeyDescriptor>` element within a particular
336 role MUST be treated as valid when processing messages or assertions in the context of that role.
337 Specifically, any signatures or transport communications (e.g., TLS/SSL sessions) verifiable with a signing
338 key or Kerberos principal name MUST be treated as valid, and any encryption keys or Kerberos principal
339 names found MAY be used to encrypt messages or assertions (or encryption keys) intended for the
340 containing entity.

341 Subsequent to accepting a metadata instance, a consumer SHOULD NOT apply additional criteria of any
342 kind on the acceptance, or validity, of the keys or Kerberos principal names found within it or their use at
343 runtime, with the exception of Kerberos protocol exchanges between a metadata consumer and Kerberos
344 Key Distribution Center, as these may be necessary for the protocol itself to operate (such as requesting a
345 service ticket for an entity described in the metadata that is subsequently used to sign a SAML message).

346 Specifically, consumers SHOULD NOT apply any online or offline techniques including, but not limited to,
347 X.509 path validation or revocation lists, OCSP responders, etc. Use of such mechanisms, if unavoidable,
348 will cause interoperability issues. In any case, consumers MUST support the acceptance of certificates
349 without CRL distribution points or Authority Information Access extensions.

350 The semantics of the following key and Kerberos principal name representations within a `<ds:KeyInfo>`
351 element are defined:

- 352 • `<ds:KeyValue>`
- 353 • `<ds:X509Certificate>` (child element of `<ds:X509Data>`)
- 354 • `<krb:KerberosSname>` (child element of `<krb:KerberosData>`)
- 355 • `<krb:KerberosCname>` (child element of `<krb:KerberosData>`)

356 In the case of `<ds:KeyValue>`, the key itself is explicitly identified. In the case of
357 `<ds:X509Certificate>`, a metadata consumer MUST extract the public key found in the certificate
358 and SHOULD NOT honor, interpret, or make use of any of the information found in the certificate other
359 than as an aid in identifying the key used (based, for example, on information found at runtime in an XML
360 digital signature's `<ds:KeyInfo>` element or the certificate presented by a transport peer).

361 In the case that a candidate key is identified, a signature can be directly evaluated based on whether it is
362 verifiable with the key. Authentication of a transport peer can be evaluated by extracting the key presented
363 by the peer (often in the form of an X.509 certificate) and comparing it by value to the candidate key. This
364 process has the effect of decoupling the certificates that may be present in metadata from those
365 presented at runtime, provided that the public keys are in fact the same.

366 A metadata consumer, when implementing authentication of a transport peer via TLS/SSL, MAY retain the
367 checking of server certificate names (e.g., `subjectAltName` or `Common Name`) in accordance with
368 [RFC2818]. Note that this constrains the certificates that may be used at runtime for TLS/SSL server
369 authentication, but does not affect certificates that might appear in metadata, because the eventual
370 comparison is based solely on the key.

371 In the case that a candidate Kerberos principal name is identified, a signature can be evaluated by
372 authenticating the Kerberos AP-REQ message included in the `<ds:KeyInfo>` element of the signature
373 and comparing the service ticket's `cname` field by value with the Kerberos principal name given in
374 metadata. Authentication of a transport peer can be evaluated by comparing the Kerberos principal name
375 claimed by the peer's authenticated service ticket with the Kerberos principal name given in metadata.

376 **2.7 Security Considerations**

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

379 Metadata becomes a critical tool for the revocation of compromised sites and keys, and all of the standard
380 practices in the use of tools like CRLs become relevant to the consumption of metadata. The specification
381 has the mechanisms to address these issues, but they have to be used. Specifically, metadata obtained
382 via an insecure transport should be both signed, and should expire, so that consumers are forced to
383 refresh it often enough to limit the damage from compromised information. Either the `validUntil` or
384 `cacheDuration` attribute may be appropriate to mitigate this threat, depending on the exchange
385 mechanism.

386 In addition, distributing signed metadata without an expiration over an untrusted channel (e.g., posting it
387 on a public web site) creates an exposure. An attacker can corrupt the channel and substitute an old
388 metadata file containing a compromised key and proceed to use that key together with other attacks to
389 impersonate a site. Repeatedly expiring (using a `validUntil` attribute) and reissuing the metadata limits
390 the window of exposure, just as a CRL does. Note that the `cacheDuration` attribute does not prevent
391 this attack.

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

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

401 **3 Conformance**

402 **3.1 SAML V2.0 Metadata Interoperability Profile Version 2.0**

403 **3.1.1 Public Key Conformance Mode**

404 A metadata producer supports the "Public Key Conformance Mode" of this profile if it can produce
405 metadata consistent with the normative text in section 2.5 and if it supports the production of
406 `<ds:KeyValue>` or `<ds:X509Certificate>` elements in accordance with section 2.5.1.

407 A metadata consumer supports the "Public Key Conformance Mode" of this profile if it can process
408 metadata consistent with the normative text in section 2.6 and if it supports the consumption of
409 `<ds:KeyValue>` or `<ds:X509Certificate>` elements in accordance with section 2.6.1.

410 **3.1.1.1 Strict Public Key Conformance Mode**

411 A metadata consumer supports the "Strict Public Key Conformance Mode" of this profile if it supports the
412 "Public Key Conformance Mode", per section 3.1.1, and does so without the imposition of certificate path
413 validation, revocation lists, OCSP, or other related technologies when evaluating certificates.

414 **3.1.2 Kerberos Conformance Mode**

415 A metadata producer supports the "Kerberos Conformance Mode" of this profile if it can produce
416 metadata consistent with the normative text in section 2.5 and if it supports the production of
417 `<krb:KerberosCname>` or `<krb:KerberosSname>` elements in accordance with section 2.5.1.

418 A metadata consumer supports the "Kerberos Conformance Mode" of this profile if it can process
419 metadata consistent with the normative text in section 2.6 and if it supports the consumption of
420 `<krb:KerberosCname>` or `<krb:KerberosSname>` elements in accordance with section 2.6.1.

421 **Appendix A. Acknowledgements**

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

- 424 • TBD

425 The editor would also like to acknowledge the following contributors:

- 426 • Walter Hoehn, University of Memphis
- 427 • Chad LaJoie, SWITCH
- 428 • Ian Young, EDINA, University of Edinburgh
- 429 • Josh Howlett, JANET

430 **Appendix B. Revision History**

- 431 ● Draft 01, revised from CS-01 of original profile with new Kerberos material and softened language
432 on certificate processing.