



WS-Trust 1.3 Errata 01

OASIS Approved Errata

25 April 2012

Specification URLs

This version:

<http://docs.oasis-open.org/ws-sx/ws-trust/v1.3/errata01/os/ws-trust-1.3-errata01-os.doc> (Authoritative)
<http://docs.oasis-open.org/ws-sx/ws-trust/v1.3/errata01/os/ws-trust-1.3-errata01-os.html>
<http://docs.oasis-open.org/ws-sx/ws-trust/v1.3/errata01/os/ws-trust-1.3-errata01-os.pdf>

Previous version:

<http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-errata-cd-03.doc> (Authoritative)
<http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-errata-cd-03.html>
<http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-errata-cd-03.pdf>

Latest version:

<http://docs.oasis-open.org/ws-sx/ws-trust/v1.3/errata01/ws-trust-1.3-errata01.doc> (Authoritative)
<http://docs.oasis-open.org/ws-sx/ws-trust/v1.3/errata01/ws-trust-1.3-errata01.html>
<http://docs.oasis-open.org/ws-sx/ws-trust/v1.3/errata01/ws-trust-1.3-errata01.pdf>

Technical Committee:

OASIS Web Services Secure Exchange (WS-SX) TC

Chairs:

Kelvin Lawrence (klawrenc@us.ibm.com), IBM
Chris Kaler (ckaler@microsoft.com), Microsoft

Editors:

Anthony Nadalin (tonynad@microsoft.com), Microsoft
Marc Goodner (mgoodner@microsoft.com), Microsoft
Abbie Barbir (abbie.bibir@bankofamerica.com), Bank of America

Related work:

This specification is related to:

- *WS-Trust 1.3.* 19 March 2007. OASIS Standard.
<http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.html>

Abstract:

This document lists errata for *WS-Trust 1.3* produced by the WS-SX Technical Committee.

Status:

This document was last revised or approved by the OASIS Web Services Secure Exchange (WS-SX) TC on the above date. The level of approval is also listed above. Check the “Latest version” location noted above for possible later revisions of this document.

Technical Committee members should send comments on this specification to the Technical Committee’s email list. Others should send comments to the Technical Committee by using the “Send A Comment” button on the Technical Committee’s web page at <http://www.oasis-open.org/committees/ws-sx/>.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the

Intellectual Property Rights section of the Technical Committee web page (<http://www.oasis-open.org/committees/ws-sx/ipr.php>).

Citation format:

When referencing this specification the following citation format should be used:

[WS-Trust-1.3-errata]

WS-Trust 1.3 Errata 01. 25 April 2012. OASIS Approved Errata.

<http://docs.oasis-open.org/ws-sx/ws-trust/v1.3/errata01/os/ws-trust-1.3-errata01-os.html>.

Notices

Copyright © OASIS Open 2012. All Rights Reserved.

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

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

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

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

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

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

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

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

Table of Contents

| | | |
|-----|---|----|
| 1 | Issues Addressed | 5 |
| 2 | Typographical/Editorial Errors | 6 |
| 2.1 | Normative references | 6 |
| 2.2 | Normative language capitalization changes | 6 |
| 2.3 | Section 9.2 Key and Encryption Requirements | 12 |
| 2.4 | WSDL changes | 12 |
| 2.5 | Schema Changes | 12 |
| 3 | Normative Errors | 13 |
| 4 | References | 14 |
| | Appendix A. Acknowledgements | 15 |

1 1 Issues Addressed

- 2 The following issues related to WS-Trust 1.3 as recorded in the [WS-SX Issues] have been addressed in
3 this document.

| Issue | Description |
|--------------|--|
| ER012 | Review normative RFC 2119 language in WS-Trust |
| ER016 | SecondaryParameters element missing from WS-Trust schema xsd |
| ER019 | Wrong target namespace in WSDL for WS-Trust 1.3 |
| ER021 | Bearer key type URI inconsistent |
| i169 | Sample wsdl in conflict w WS-I BSP in WS-Trust1.3, 1.4 |
| i170 | Update XML Signature references to refer to XML Signature, Second Edition, update c14n reference in ws-trust |
| i171 | Incorrect URI provided for Canonical XML 1.0 when defining C14n abbreviation |

4

5 2 Typographical/Editorial Errors

6 2.1 Normative references

7 Insert after line 185

W3C Recommendation, "Canonical XML Version 1.1", 2 May 2008.
<http://www.w3.org/TR/2008/REC-xml-c14n11-20080502/>

10

11 Insert after line 201

[W3C Recommendation, D. Eastlake et al. XML Signature Syntax and Processing (Second Edition). 10 June 2008.
<http://www.w3.org/TR/2008/REC-xmldsig-core-20080610/>

15 2.2 Normative language capitalization changes

16 The following changes do not affect the normative meaning of the text, they are only to properly capitalize
17 2119 terms. The changes listed below document the changes as they appear in the text. There were
18 many instances of the terms OPTIONAL and REQUIRED in the schema exemplar descriptions that
19 appeared un-capitalized that are not captured below but that have also been addressed. All other 2119
20 terms that remain un-capitalized are used in their English sense.

21 Line 212

22 Authentication of requests is based on a combination of OPTIONAL network and transport-provided
23 security and information (claims) proven in the message

24

25 Line 231

26 This model is illustrated in the figure below, showing that any requestor MAY also be a service, and that
27 the Security Token Service is a Web service (that is, it MAY express policy and require security tokens).

28

29 Line 242

30 In the figure above the arrows represent possible communication paths; the requestor MAY obtain a
31 token from the security token service, or it MAY have been obtained indirectly. The requestor then
32 demonstrates authorized use of the token to the Web service. The Web service either trusts the issuing
33 security token service or MAY request a token service to validate the token (or the Web service MAY
34 validate the token itself).

35

36 In summary, the Web service has a policy applied to it, receives a message from a requestor that possibly
37 includes security tokens, and MAY have some protection applied to it using [WS-Security] mechanisms.

38

39 Line 254

40 In brokered trust models, the signature MAY NOT verify the identity of the claimant – it MAY verify the
41 identity of the intermediary, who MAY simply assert the identity of the claimant.

42

43 Line 259

44 The trust engine MAY need to externally verify or broker tokens

45

46 Line 265

47 In this specification we define how security tokens are requested and obtained from security token
48 services and how these services MAY broker trust and trust policies so that services can perform step 3.
49

50 Line 280

51 As part of a message flow, a request MAY be made of a security token service to exchange a security
52 token (or some proof) of one form for another

53

54 Line 289

55 the security token service generating the new token MAY NOT need to trust the authority that issued the
56 original token provided by the original requestor since it does trust the security token service that is
57 engaging in the exchange for a new security token

58

59 Line 300

60 An administrator or other trusted authority MAY designate that all tokens of a certain type are

61

62 Line 303

63 or the security token service MAY provide this function as a service to trusting services.

64

65 Line 306

66 These mechanisms are non-normative and are NOT REQUIRED in any way.

67

68 Line 313

69 Trust hierarchies – Building on the trust roots mechanism, a service MAY choose to allow hierarchies of
70 trust so long as the trust chain eventually leads to one of the known trust roots. In some cases the
71 recipient MAY require the sender to provide the full hierarchy. In other cases, the recipient MAY be able
72 to dynamically fetch the tokens for the hierarchy from a token store.

73

74 Line 335

75 or they MAY return a token with their chosen parameters that the requestor MAY then choose to discard
76 because it doesn't meet their needs

77

78 Line 339

79 Other specifications MAY define specific bindings and profiles of this mechanism for additional purposes.

80

81 Line 341

82 in some cases an anonymous request MAY be appropriate

83

84 Line 343

85 If not a fault SHOULD be generated (but is NOT REQUIRED to be returned for denial-of-service reasons).

86

87 Line 415 (this one changes a “shouldn’t”)

88 In general, the returned token SHOULD be considered opaque to the requestor. That is, the requestor
89 SHOULD NOT be required to parse the returned token.

90

91 Line 429

92 and the value of the OPTIONAL @Context attribute
93
94 Line 432
95 In such cases, the RSTR MAY be passed in the body or in a header block.
96
97 Line 475
98 the ellipses below represent the different containers in which this element MAY appear
99
100 Line 518
101 This binding supports the OPTIONAL use of exchanges during the token acquisition process as well as
102 the OPTIONAL use of the key extensions described in a later section.
103
104 Line 522
105 the following OPTIONAL elements
106
107 Line 561
108 This REQUIRED attribute contains a URI that indicates the syntax used to specify the set of requested
109 claims along with how that syntax SHOULD be interpreted.
110
111 Line 574
112 The format is assumed to be understood by the requestor because the value space MAY be
113
114 Line 580
115 The issuer is not obligated to honor this range – they MAY
116
117 Line 587
118 The difference in time SHOULD be minimized.
119
120 Line 697
121 Each request MAY generate more than one RSTR sharing the same Context attribute value
122
123 Line 711
124 Note: that these operations require that the service can either succeed on all the RST requests or MUST
125 NOT perform any partial operation.
126
127 Line 722
128 If any error occurs in the processing of the RSTC or one of its contained RSTs, a SOAP fault MUST be
129 generated for the entire batch request so no RSTC element will be returned.
130
131 Line 741
132 the following OPTIONAL elements
133
134 Line 833

135 The token issuer can OPTIONALY provide
136
137 Line 990
138 As a result, the proof-of-possession tokens, and possibly lifetime and other key parameters elements,
139 MAY be different
140
141 Line 1071
142 If confidentiality protection of the <wst:IssuedTokens> header is REQUIRED then the entire header
143 MUST be encrypted using the <wsse11:EncryptedHeader> construct.
144
145 Line 1131
146 and the OPTIONAL <wst:Lifetime> element
147
148 Line 1167
149 This OPTIONAL element indicates that returned tokens SHOULD allow requests for postdated tokens.
150
151 Line 1225
152 If a client needs to ensure the validity of a token, it MUST validate the token at the issuer.
153
154 Line 1292
155 this section defines an OPTIONAL binding
156
157 Line 1354
158 The result MAY be a status, a new token, or both.
159
160 Line 1370
161 The request provides a token upon which the request is based and OPTIONAL tokens. As well, the
162 OPTIONAL <wst:TokenType> element
163
164 Line 1371
165 This MAY be any supported token type or it MAY be the following URI indicating that only status is
166 desired:
167
168 Line 1378
169 which is OPTIONAL
170
171 Line 1467
172 However, there are many scenarios where a set of exchanges between the parties is REQUIRED prior to
173 returning (e.g., issuing) a security token.
174
175 Line 1487
176 with the issued security token and OPTIONAL proof-of-possession token
177
178 Line 1502

179 (and MAY contain initial negotiation/challenge information)
180
181 Line 1504
182 Optionally, this MAY return token information
183
184 Line 1572
185 Exchange requests MAY also utilize existing binary formats
186
187 Line 1579
188 ellipses below indicate that this element MAY be placed in different containers
189
190 Line 1602
191 In some cases it MAY be necessary to provide a key exchange token so that the other party (either
192 requestor or issuer) can provide entropy or key material as part of the exchange. Challenges MAY NOT
193 always provide a usable key as the signature may use a signing-only certificate.
194
195 Line 1606
196 The section describes two OPTIONAL elements
197
198 Line 1608
199 ellipses below indicate that this element MAY be placed in different containers
200
201 Line 1617
202 This OPTIONAL element is used to indicate that the receiving party (either the original requestor or
203 issuer) SHOULD provide a KET to the other party on the next leg of the exchange.
204
205 Line 1822
206 This MAY be built into the exchange messages
207
208 Line 1832
209 To this end, the following computed key algorithm is defined to be OPTIONALLY used in these scenarios
210
211 Line 1837
212 However, until the exchange is actually completed it MAY be (and is often) inappropriate to use the
213 computed keys. As well, using a token that hasn't been returned to secure a message may (no change,
214 English) complicate processing since it crosses the boundary of the exchange and the underlying
215 message security. This means that it MAY NOT be appropriate to sign the final leg of the exchange using
216 the key derived from the exchange.
217
218 Line 1874
219 This <wst:CombinedHash> element is OPTIONAL
220
221 Line 1878
222 since all types of requests MAY issue security tokens they could apply to other bindings

223
224 Line 1924
225 The syntax for these OPTIONAL elements is as follows
226
227 Line 1950
228 That is, requestors SHOULD be familiar with the recipient policies
229
230 Line 1996
231 This element either contains a security token or a <wsse:SecurityTokenReference> element that
232 references the security token containing the key that SHOULD be used in the returned token.
233
234 Line 2037
235 EncryptionAlgorithm – used to indicate the symmetric algorithm that the STS SHOULD use to encrypt the
236 T (e.g. AES256)
237
238 Line 2043
239 EncryptionAlgorithm – used to indicate the symmetric algorithm that the STS SHOULD use to encrypt T
240 for RP (e.g. AES256)
241 KeyWrapAlgorithm – used to indicate the KeyWrap algorithm that the STS SHOULD use to wrap the
242 generated key that is used to encrypt the T for RP
243
244 Line 2052
245 EncryptionAlgorithm – used to indicate the symmetric algorithm that the STS SHOULD use to encrypt T
246 for RP (e.g. AES256)
247
248 Line 2059
249 EncryptionAlgorithm - used to indicate the symmetric algorithm that the STS SHOULD use to encrypt T
250 for RP (e.g. AES256)
251 KeyWrapAlgorithm – used to indicate the KeyWrap algorithm that the STS SHOULD use to wrap the
252 generated key that is used to encrypt the T for RP
253
254 Line 2140
255 This OPTIONAL element, of type xs:boolean, specifies whether the requested security token SHOULD be
256 marked as "Forwardable"
257
258 Line 2145
259 This OPTIONAL element, of type xs:boolean, specifies whether the requested security token SHOULD be
260 marked as "Delegatable".
261
262 Line 2224
263 Arbitrary types MAY be used to specify participants
264
265 Line 2248
266 OPTIONAL the <wst:TokenType> element can be specified in the request and can indicate
267

268 Line 2363
269 Other specifications and profiles MAY provide additional details on key exchange
270
271 Line 2376
272 In these cases both parties SHOULD contribute entropy to the key exchange by means of the
273 <wst:entropy> element
274
275 Line 2403
276 If the requestor provides key material that the recipient doesn't accept, then the issuer SHOULD reject the
277 request.
278
279 Line 2492
280 A third party MAY also act as a broker to transfer keys
281
282 Line 2631
283 The perfect forward secrecy property MAY be achieved by

284 **2.3 Section 9.2 Key and Encryption Requirements**

285 Changed Bearer URI in table at 1956 from
286 <http://docs.oasis-open.org/ws-sx/wstrust/200512/Bearer>
287 to
288 <http://docs.oasis-open.org/ws-sx/ws-trust/200512/Bearer>

289 **2.4 WSDL changes**

290 The WSDL was replaced with a more representative example that better illustrates usage of the protocol.

291 **2.5 Schema Changes**

292 Missing SecondaryParameters element added to section 3.1 in external schema file.
293 Corrected Bearer URI in KeyTypeEnum simple type from
294 <http://docs.oasis-open.org/wssx/wstrust/200512/Bearer>
295 to
296 <http://docs.oasis-open.org/ws-sx/ws-trust/200512/Bearer>
297

298

3 Normative Errors

299

None.

300

4 References

- 301 [WS-SX Issues] WS-SX TC Issues List
302 <http://docs.oasis-open.org/ws-sx/issues/Issues.xml>
303 [WS-Trust] OASIS Standard, "WS-Trust 1.3", March 2007
304 <http://docs.oasis-open.org/ws-sx/ws-trust/200512>

305 Appendix A. Acknowledgements

306 The following individuals have participated in the creation of this specification and are gratefully
307 acknowledged.

308

309 TC Members during the development of this specification:

310 Don Adams, Tibco Software Inc.

311 Jan Alexander, Microsoft Corporation

312 Steve Anderson, BMC Software

313 Donal Arundel, IONA Technologies

314 Howard Bae, Oracle Corporation

315 Abbie Barbir, Nortel Networks Limited

316 Charlton Barreto, Adobe Systems

317 Mighael Botha, Software AG, Inc.

318 Toufic Boubez, Layer 7 Technologies Inc.

319 Norman Brickman, Mitre Corporation

320 Melissa Brumfield, Booz Allen Hamilton

321 Lloyd Burch, Novell

322 Scott Cantor, Internet2

323 Greg Carpenter, Microsoft Corporation

324 Steve Carter, Novell

325 Symon Chang, BEA Systems, Inc.

326 Ching-Yun (C.Y.) Chao, IBM

327 Martin Chapman, Oracle Corporation

328 Kate Cherry, Lockheed Martin

329 Henry (Hyenvui) Chung, IBM

330 Luc Clement, Systinet Corp.

331 Paul Cotton, Microsoft Corporation

332 Glen Daniels, Sonic Software Corp.

333 Peter Davis, Neustar, Inc.

334 Martijn de Boer, SAP AG

335 Werner Dittmann, Siemens AG

336 Abdeslem DJAOUI, CCLRC-Rutherford Appleton Laboratory

337 Fred Dushin, IONA Technologies

338 Petr Dvorak, Systinet Corp.

339 Colleen Evans, Microsoft Corporation

340 Ruchith Fernando, WSO2

341 Mark Fussell, Microsoft Corporation

342 Vijay Gajjala, Microsoft Corporation

343 Marc Goodner, Microsoft Corporation

344 Hans Granqvist, VeriSign

345 Martin Gudgin, Microsoft Corporation
346 Tony Gullotta, SOA Software Inc.
347 Jiandong Guo, Sun Microsystems
348 Phillip Hallam-Baker, VeriSign
349 Patrick Harding, Ping Identity Corporation
350 Heather Hinton, IBM
351 Frederick Hirsch, Nokia Corporation
352 Jeff Hodges, Neustar, Inc.
353 Will Hopkins, BEA Systems, Inc.
354 Alex Hristov, Otecia Incorporated
355 John Hughes, PA Consulting
356 Diane Jordan, IBM
357 Venugopal K, Sun Microsystems
358 Chris Kaler, Microsoft Corporation
359 Dana Kaufman, Forum Systems, Inc.
360 Paul Knight, Nortel Networks Limited
361 Ramanathan Krishnamurthy, IONA Technologies
362 Christopher Kurt, Microsoft Corporation
363 Kelvin Lawrence, IBM
364 Hubert Le Van Gong, Sun Microsystems
365 Jong Lee, BEA Systems, Inc.
366 Rich Levinson, Oracle Corporation
367 Tommy Lindberg, Dajeil Ltd.
368 Mark Little, JBoss Inc.
369 Hal Lockhart, BEA Systems, Inc.
370 Mike Lyons, Layer 7 Technologies Inc.
371 Eve Maler, Sun Microsystems
372 Ashok Malhotra, Oracle Corporation
373 Anand Mani, CrimsonLogic Pte Ltd
374 Jonathan Marsh, Microsoft Corporation
375 Robin Martherus, Oracle Corporation
376 Miko Matsumura, Infravio, Inc.
377 Gary McAfee, IBM
378 Michael McIntosh, IBM
379 John Merrells, Sxip Networks SRL
380 Jeff Mischkinsky, Oracle Corporation
381 Prateek Mishra, Oracle Corporation
382 Bob Morgan, Internet2
383 Vamsi Motukuru, Oracle Corporation
384 Raajmohan Na, EDS
385 Anthony Nadalin, IBM
386 Andrew Nash, Reactivity, Inc.

387 Eric Newcomer, IONA Technologies
388 Duane Nickull, Adobe Systems
389 Toshihiro Nishimura, Fujitsu Limited
390 Rob Philpott, RSA Security
391 Denis Pilipchuk, BEA Systems, Inc.
392 Darren Platt, Ping Identity Corporation
393 Martin Raeppler, SAP AG
394 Nick Ragouzis, Enosis Group LLC
395 Prakash Reddy, CA
396 Alain Regnier, Ricoh Company, Ltd.
397 Irving Reid, Hewlett-Packard
398 Bruce Rich, IBM
399 Tom Rutt, Fujitsu Limited
400 Maneesh Sahu, Actional Corporation
401 Frank Siebenlist, Argonne National Laboratory
402 Joe Smith, Apani Networks
403 Davanum Srinivas, WSO2
404 Yakov Sverdlov, CA
405 Gene Thurston, AmberPoint
406 Victor Valle, IBM
407 Asir Vedamuthu, Microsoft Corporation
408 Greg Whitehead, Hewlett-Packard
409 Ron Williams, IBM
410 Corinna Witt, BEA Systems, Inc.
411 Kyle Young, Microsoft Corporation