

[MS-ADFSP-IP-Diff]:

Active Directory Federation Services and Proxy Integration Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“[this documentation](#)”) for protocols, file formats, [data portability](#), [computer languages](#), [and standards as well as overviews of the interaction among each of these technologies](#)[support. Additionally, overview documents cover inter-protocol relationships and interactions.](#)
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you [may can](#) make copies of it in order to develop implementations of the technologies [that are](#) described in ~~the Open Specifications~~ [this documentation](#) and [may can](#) distribute portions of it in your implementations [using that use](#) these technologies or [in](#) your documentation as necessary to properly document the implementation. You [may can](#) also distribute in your implementation, with or without modification, any ~~schema, IDL's~~ [schemas, IDLs](#), or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications ~~documentation.~~
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that [may might](#) cover your implementations of the technologies described in the Open Specifications ~~documentation.~~ Neither this notice nor Microsoft's delivery of ~~the this~~ documentation grants any licenses under those [patents](#) or any other Microsoft patents. However, a given Open ~~Specification may~~ [Specifications document might](#) be covered by [the](#) Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in ~~the Open Specification~~ [this documentation](#) are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation [may might](#) be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.
- **Fictitious Names.** The example companies, organizations, products, domain names, ~~e-mail~~ [email](#) addresses, logos, people, places, and events [that are](#) depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

Reservation of Rights. All other rights are reserved, and this notice does not grant any rights other than [as](#) specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications ~~documentation does~~ not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications [documents](#) are intended for use in conjunction with publicly available ~~standards~~ [standards](#) specifications and network programming art, and ~~assumes, as such, assume~~ that the reader either is familiar with the aforementioned material or has immediate access to it.

~~**Preliminary Documentation.** This Open Specification provides documentation for past and current releases and/or for the pre-release version of this technology. This Open Specification is final documentation for past or current releases as specifically noted in the document, as applicable; it is~~

~~preliminary documentation for the pre-release versions. Microsoft will release final documentation in connection with the commercial release of the updated or new version of this technology. As the documentation may change between this preliminary version and the final version of this technology, there are risks in relying on preliminary documentation. To the extent that you incur additional development obligations or any other costs as a result of relying on this preliminary documentation, you do so at your own risk.~~

Revision Summary

Date	Revision History	Revision Class	Comments
8/8/2013	1.0	New	Released new document.
11/14/2013	2.0	Major	Significantly changed the technical content.
2/13/2014	3.0	Major	Significantly changed the technical content.
5/15/2014	3.0	None	No change to the meaning, language, or formatting of the technical content.
6/30/2015	4.0	Major	Significantly changed the technical content.
<u>7/14/2016</u>	<u>5.0</u>	<u>Major</u>	<u>Significantly changed the technical content.</u>

Table of Contents

1	Introduction	11
1.1	Glossary	11
1.2	References	12
1.2.1	Normative References	12
1.2.2	Informative References	13
1.3	Overview	13
1.4	Relationship to Other Protocols	14
1.5	Prerequisites/Preconditions	15
1.6	Applicability Statement	15
1.7	Versioning and Capability Negotiation	15
1.8	Vendor-Extensible Fields	15
1.9	Standards Assignments.....	15
2	Messages.....	16
2.1	Transport	16
2.2	Common Data Types	16
2.2.1	HTTP Headers	16
2.2.1.1	X-MS-Proxy.....	16
2.2.1.2	X-MS-Forwarded-Client-IP.....	16
2.2.1.3	X-MS-Endpoint-Absolute-Path.....	17
2.2.1.4	X-MS-Target-Role	17
2.2.1.5	X-MS-ADFS-Proxy-Client-IP.....	17
2.2.2	Complex Types.....	17
2.2.2.1	Proxy Trust	17
2.2.2.2	Proxy Trust Renewal.....	17
2.2.2.3	Proxy Relying Party Trust	18
2.2.2.4	Configuration.....	18
2.2.2.5	Relying Party Trust List.....	19
2.2.2.6	Relying Party Trust.....	20
2.2.2.7	Relying Party Trust Publishing Settings	20
2.2.2.8	Store Entry List.....	21
2.2.2.9	Store Entry	21
2.2.2.10	Store Entry Key and Value	21
2.2.2.11	Serialized Request with Certificate	21
2.2.2.12	Port Type	23
2.2.2.13	Credential Collection Scheme	23
2.2.2.14	TLS Query Behavior	23
2.2.2.15	Certificate Validation	23
2.2.2.16	Certificate Type	23
2.2.2.17	Proxy Token	24
2.2.2.18	Combined Token	24
2.2.2.19	Proxy Token Wrapper	25
2.2.2.20	Authentication Request.....	25
2.2.2.21	Error Response	26
3	Protocol Details.....	27
3.1	Common Details	27
3.1.1	Abstract Data Model.....	27
3.1.1.1	Server State.....	27
3.1.1.2	Client State	27
3.1.1.3	Relying Party Trust State	28
3.1.2	Timers	28
3.1.3	Initialization	28
3.1.4	Higher-Layer Triggered Events	28
3.1.5	Message Processing Events and Sequencing Rules	28

3.1.6	Timer Events.....	28
3.1.7	Other Local Events.....	28
3.2	Proxy Registration Server Details	28
3.2.1	Abstract Data Model.....	28
3.2.2	Timers	29
3.2.3	Initialization.....	29
3.2.4	Higher-Layer Triggered Events	29
3.2.5	Message Processing Events and Sequencing Rules	29
3.2.5.1	Proxy/EstablishTrust.....	29
3.2.5.1.1	POST	29
3.2.5.1.1.1	Request Body	30
3.2.5.1.1.2	Response Body	30
3.2.5.1.1.3	Processing Details.....	30
3.2.5.2	Proxy/RenewTrust.....	30
3.2.5.2.1	POST	30
3.2.5.2.1.1	Request Body	31
3.2.5.2.1.2	Response Body	31
3.2.5.2.1.3	Processing Details.....	31
3.2.5.3	Proxy/WebApplicationProxy/Trust	31
3.2.5.3.1	GET	31
3.2.5.3.1.1	Request Body	32
3.2.5.3.1.2	Response Body	32
3.2.5.3.1.3	Processing Details.....	32
3.2.5.3.2	POST	32
3.2.5.3.2.1	Request Body	33
3.2.5.3.2.2	Response Body	33
3.2.5.3.2.3	Processing Details.....	33
3.2.5.3.3	DELETE	33
3.2.5.3.3.1	Request Body	34
3.2.5.3.3.2	Response Body	34
3.2.5.3.3.3	Processing Details.....	34
3.2.6	Timer Events.....	34
3.2.7	Other Local Events.....	34
3.3	Proxy Registration Client Details	34
3.3.1	Abstract Data Model.....	34
3.3.2	Timers	34
3.3.3	Initialization.....	34
3.3.4	Higher-Layer Triggered Events	34
3.3.5	Message Processing Events and Sequencing Rules	34
3.3.5.1	Proxy/EstablishTrust.....	35
3.3.5.1.1	POST	35
3.3.5.1.1.1	Request Body	35
3.3.5.1.1.2	Response Body	35
3.3.5.1.1.3	Processing Details.....	35
3.3.5.2	Proxy/RenewTrust.....	35
3.3.5.2.1	POST	35
3.3.5.2.1.1	Request Body	35
3.3.5.2.1.2	Response Body	35
3.3.5.2.1.3	Processing Details.....	35
3.3.5.3	Proxy/WebApplicationProxy/Trust	35
3.3.5.3.1	GET	35
3.3.5.3.1.1	Request Body	36
3.3.5.3.1.2	Response Body	36
3.3.5.3.1.3	Processing Details.....	36
3.3.5.3.2	POST	36
3.3.5.3.2.1	Request Body	36
3.3.5.3.2.2	Response Body	36
3.3.5.3.2.3	Processing Details.....	36

3.3.5.3.3	DELETE	36
3.3.5.3.3.1	Request Body	36
3.3.5.3.3.2	Response Body	36
3.3.5.3.3.3	Processing Details	36
3.3.6	Timer Events.....	36
3.3.7	Other Local Events.....	36
3.4	Service Configuration Server Details.....	37
3.4.1	Abstract Data Model.....	37
3.4.2	Timers	37
3.4.3	Initialization.....	37
3.4.4	High-Layer Triggered Events	37
3.4.5	Message Processing Events and Sequencing Rules	37
3.4.5.1	Proxy/GetConfiguration.....	38
3.4.5.1.1	GET	38
3.4.5.1.1.1	Request Body	38
3.4.5.1.1.2	Response Body	38
3.4.5.1.1.3	Processing Details.....	38
3.4.5.2	Proxy/RelyingPartyTrusts	39
3.4.5.2.1	GET	39
3.4.5.2.1.1	Request Body	39
3.4.5.2.1.2	Response Body	39
3.4.5.2.1.3	Processing Details.....	39
3.4.5.3	Proxy/RelyingPartyTrusts/	39
3.4.5.3.1	GET	39
3.4.5.3.1.1	Request Body	40
3.4.5.3.1.2	Response Body	40
3.4.5.3.1.3	Processing Details.....	40
3.4.6	Timer Events.....	40
3.4.7	Other Local Events.....	40
3.5	Service Configuration Client Details	41
3.5.1	Abstract Data Model.....	41
3.5.2	Timers	41
3.5.3	Initialization.....	41
3.5.4	High-Layer Triggered Events	41
3.5.5	Message Processing Events and Sequencing Rules	41
3.5.5.1	Proxy/GetConfiguration.....	41
3.5.5.1.1	GET	41
3.5.5.1.1.1	Request Body	41
3.5.5.1.1.2	Response Body	41
3.5.5.1.1.3	Processing Details.....	41
3.5.5.2	Proxy/RelyingPartyTrusts	41
3.5.5.2.1	GET	41
3.5.5.2.1.1	Request Body	42
3.5.5.2.1.2	Response Body	42
3.5.5.2.1.3	Processing Details.....	42
3.5.5.3	Proxy/RelyingPartyTrusts/	42
3.5.5.3.1	GET	42
3.5.5.3.1.1	Request Body	42
3.5.5.3.1.2	Response Body	42
3.5.5.3.1.3	Processing Details.....	42
3.5.6	Timer Events.....	42
3.5.7	Other Local Events.....	42
3.6	Proxy Configuration Server Details	42
3.6.1	Abstract Data Model.....	42
3.6.2	Timers	42
3.6.3	Initialization.....	42
3.6.4	High-Layer Triggered Events	43
3.6.5	Message Processing Events and Sequencing Rules	43

3.6.5.1	Proxy/WebApplicationProxy/Store	43
3.6.5.1.1	GET	43
3.6.5.1.1.1	Request Body	44
3.6.5.1.1.2	Response Body	44
3.6.5.1.1.3	Processing Details	44
3.6.5.2	Proxy/WebApplicationProxy/Store/	44
3.6.5.2.1	GET	44
3.6.5.2.1.1	Request Body	45
3.6.5.2.1.2	Response Body	45
3.6.5.2.1.3	Processing Details	45
3.6.5.2.2	POST	45
3.6.5.2.2.1	Request Body	45
3.6.5.2.2.2	Response Body	45
3.6.5.2.2.3	Processing Details	46
3.6.5.2.3	PUT	46
3.6.5.2.3.1	Request Body	46
3.6.5.2.3.2	Response Body	46
3.6.5.2.3.3	Processing Details	46
3.6.5.2.4	DELETE	47
3.6.5.2.4.1	Request Body	47
3.6.5.2.4.2	Response Body	47
3.6.5.2.4.3	Processing Details	48
3.6.6	Timer Events.....	48
3.6.7	Other Local Events.....	48
3.7	Proxy Configuration Client Details	48
3.7.1	Abstract Data Model.....	48
3.7.2	Timers	48
3.7.3	Initialization	48
3.7.4	High-Layer Triggered Events	48
3.7.5	Message Processing Events and Sequencing Rules	48
3.7.5.1	Proxy/WebApplicationProxy/Store	48
3.7.5.1.1	GET	48
3.7.5.1.1.1	Response Body	48
3.7.5.1.1.2	Request Body	49
3.7.5.1.1.3	Processing Details	49
3.7.5.2	Proxy/WebApplicationProxy/Store/	49
3.7.5.2.1	GET	49
3.7.5.2.1.1	Request Body	49
3.7.5.2.1.2	Response Body	49
3.7.5.2.1.3	Processing Details	49
3.7.5.2.2	POST	49
3.7.5.2.2.1	Request Body	49
3.7.5.2.2.2	Response Body	49
3.7.5.2.2.3	Processing Details	49
3.7.5.2.3	PUT	49
3.7.5.2.3.1	Request Body	49
3.7.5.2.3.2	Response Body	49
3.7.5.2.3.3	Processing Details	49
3.7.5.2.4	DELETE	50
3.7.5.2.4.1	Request Body	50
3.7.5.2.4.2	Response Body	50
3.7.5.2.4.3	Processing Details	50
3.7.6	Timer Events.....	50
3.7.7	Other Local Events.....	50
3.8	Application Publishing Server Details	50
3.8.1	Abstract Data Model.....	50
3.8.2	Timers	50
3.8.3	Initialization	50

3.8.4	High-Layer Triggered Events	50
3.8.5	Message Processing Events and Sequencing Rules	50
3.8.5.1	Proxy/RelyingPartyTrusts/{Identifier}/PublishedSettings	51
3.8.5.1.1	POST	51
3.8.5.1.1.1	Request Body	52
3.8.5.1.1.2	Response Body	52
3.8.5.1.1.3	Processing Details	52
3.8.5.1.2	DELETE	52
3.8.5.1.2.1	Request Body	53
3.8.5.1.2.2	Response Body	53
3.8.5.1.2.3	Processing Details	53
3.8.6	Timer Events.....	53
3.8.7	Other Local Events.....	53
3.9	Application Publishing Client Details.....	54
3.9.1	Abstract Data Model.....	54
3.9.2	Timers	54
3.9.3	Initialization.....	54
3.9.4	High-Layer Triggered Events	54
3.9.5	Message Processing Events and Sequencing Rules	54
3.9.5.1	Proxy/RelyingPartyTrusts/{Identifier}/PublishedSettings	54
3.9.5.1.1	POST	54
3.9.5.1.1.1	Request Body	54
3.9.5.1.1.2	Response Body	54
3.9.5.1.1.3	Processing Details	54
3.9.5.1.2	DELETE	54
3.9.5.1.2.1	Request Body	54
3.9.5.1.2.2	Response Body	55
3.9.5.1.2.3	Processing Details	55
3.9.6	Timer Events.....	55
3.9.7	Other Local Events.....	55
3.10	Proxy Runtime Behaviors Server Details.....	55
3.10.1	Abstract Data Model.....	55
3.10.2	Timers	55
3.10.3	Initialization.....	55
3.10.4	High-Layer Triggered Events	55
3.10.5	Message Processing Events and Sequencing Rules	55
3.10.5.1	BackEndProxyTLS	56
3.10.5.1.1	POST	56
3.10.5.1.1.1	Request Body	56
3.10.5.1.1.2	Response Body	56
3.10.5.1.1.3	Processing Details	56
3.10.6	Timer Events.....	57
3.10.7	Other Local Events.....	57
3.11	Proxy Runtime Behaviors Client Details.....	57
3.11.1	Abstract Data Model.....	57
3.11.2	Timers	57
3.11.3	Initialization.....	57
3.11.4	High-Layer Triggered Events	57
3.11.5	Message Processing Events and Sequencing Rules	57
3.11.5.1	End-user X509 Certificate Processing.....	58
3.11.6	Timer Events.....	59
3.11.7	Other Local Events.....	59
3.12	Application Proxy Runtime Behaviors Server Details	59
3.12.1	Abstract Data Model.....	59
3.12.2	Timers	59
3.12.3	Initialization.....	59
3.12.4	High-Layer Triggered Events	59
3.12.5	Message Processing Events and Sequencing Rules	59

3.12.5.1	Issue Preauthentication	59
3.12.5.1.1	Proxy Preauthentication	59
3.12.5.1.1.1	Request Body	60
3.12.5.1.1.2	Response Body	60
3.12.5.1.1.3	Processing Details	60
3.12.5.1.2	SAML-P Extensions for Preauthentication	60
3.12.5.1.3	WS-Fed Extensions for Preauthentication	61
3.12.5.1.4	OAuth Extensions for Preauthentication	61
3.12.5.1.5	Proxy Preauthentication for Active Clients	62
3.12.5.1.5.1	Request Body	62
3.12.5.1.5.2	Response Body	62
3.12.5.1.5.3	Processing Details	63
3.12.6	Timer Events	63
3.12.7	Other Local Events	63
3.13	Application Proxy Runtime Behaviors Client Details	63
3.13.1	Abstract Data Model	63
3.13.2	Timers	63
3.13.3	Initialization	63
3.13.4	High-Layer Triggered Events	63
3.13.5	Message Processing Events and Sequencing Rules	64
3.13.5.1	Preauthentication	64
3.13.5.1.1	Query String Based Preauthentication	64
3.13.5.1.2	HTTP Authorization Header Based Preauthentication	64
3.13.5.2	Initiate Preauthentication	64
3.13.5.2.1	Initiate Redirect-based Preauthentication	64
3.13.5.2.2	Response to [MS-OFBA] Requests	65
3.13.5.2.3	Response to Active Requests	66
3.13.6	Timer Events	67
3.13.7	Other Local Events	67
4	Protocol Examples	68
4.1	Establishing Proxy Trust with the Server	68
4.1.1	Client Request	68
4.1.2	Server Response	68
4.2	Getting Information about All Relying Party Trusts	68
4.2.1	Client Request	68
4.2.2	Server Response	68
4.3	Create a New Set of Published Settings on a Relying Party Trust	69
4.3.1	Client Request	69
4.3.2	Server Response	69
4.4	Remove an Existing Set of Published Settings on a Relying Party Trust	69
4.4.1	Client Request	69
4.4.2	Server Response	69
4.5	Add a Key Value Pair to the Store	69
4.5.1	Client Request	69
4.5.2	Server Response	70
4.6	Retrieve a Value of a Key from the Store	70
4.6.1	Client Request	70
4.6.2	Server Response	70
4.7	Update the Value of a Key Already in the Store	70
4.7.1	Client Request	70
4.7.2	Server Response	70
4.8	Create a new Proxy Relying Party Trust	71
4.8.1	Client Request	71
4.8.2	Server Response	71
4.9	Get the Proxy Relying Party Trust	71
4.9.1	Client Request	71
4.9.2	Server Response	71

5	Security	72
5.1	Security Considerations for Implementers	72
5.2	Index of Security Parameters	72
6	Appendix A: Full JSON Schema	73
7	Appendix B: Product Behavior	79
8	Change Tracking	81
9	Index	87

1 Introduction

This is a specification of the **Active Directory Federation Services and Proxy system** and the protocols that define the interaction behaviors between **Active Directory Federation Services (AD FS)** and the **Web Application Proxy**, or simply **Proxy**. It describes the intended functionality of the system and how the protocols in this system interact.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative ~~and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in [RFC2119]. Sections 1.5 and 1.9 are also normative but do not contain those terms.~~ All other sections and examples in this specification are informative.

1.1 Glossary

~~The~~This document uses the following terms ~~are specific to this document:~~

Active Directory Federation Services (AD FS): A Microsoft implementation of a federation services provider, which provides a security token service (STS) that can issue security tokens to a caller using various protocols such as WS-Trust, WS-Federation, and Security Assertion Markup Language (SAML) version 2.0.

Active Directory Federation Services and Proxy system: A system of features and protocols whereby a client located outside the boundaries of a corporate network can access application services located inside those boundaries.

Coordinated Universal Time (UTC): A high-precision atomic time standard that approximately tracks Universal Time (UT). It is the basis for legal, civil time all over the Earth. Time zones around the world are expressed as positive and negative offsets from UTC. In this role, it is also referred to as Zulu time (Z) and Greenwich Mean Time (GMT). In these specifications, all references to UTC refer to the time at UTC-0 (or GMT).

extended key usage (EKU): An X.509 certificate extension that indicates one or more purposes for which the certificate ~~may~~can be used.

farm configuration: A collection of servers, each of which provide the same services, and to each of which a service request can be routed for load balancing.

internal network: The portion of the corporate network that is protected by a firewall.

non-claims-aware: A characteristic of a network device or application that makes it unable to participate in claims-based authentication.

perimeter network: The portion of the corporate network that is on the outside of the firewall and is exposed to external network traffic.

pre-authentication: In **Active Directory Federation Services (AD FS)**, the act of enforcing authentication of a user on the edge of a protected network boundary.

proxy: A network node that accepts network traffic originating from one network agent and transmits it to another network agent.

token: A set of rights and privileges for a given user.

Web Application Proxy: A set of components that provide proxy services for clients that are requesting access to application services inside the boundaries of a corporate network.

MAY, SHOULD, MUST, SHOULD NOT, MUST NOT: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

[IETF-DRAFT-JWS] Internet Engineering Task Force (IETF), "JSON Web Signature (JWS)", draft-ietf-jose-json-web-signature-10, April 2013, <http://tools.ietf.org/html/draft-ietf-jose-json-web-signature-10>

[IETF-DRAFT-TOKBND] Balfanz, D., Langley, A., Nystroem, M., et al., "Transport Layer Security (TLS) Extension for Token Binding Protocol Negotiation", draft-popov-tokbind-negotiation-00, May 2015, <http://datatracker.ietf.org/doc/draft-popov-tokbind-negotiation>

[MS-OAPX] Microsoft Corporation, "OAuth 2.0 Protocol Extensions".

[MS-OFBA] Microsoft Corporation, "Office Forms Based Authentication Protocol".

[MS-PKAP] Microsoft Corporation, "Public Key Authentication Protocol".

[RFC1422] Kent, S., "Privacy Enhancement for Internet Electronic Mail: Part II: Certificate-Based Key Management", RFC 1422, February 1993, <http://www.rfc-editor.org/rfc/rfc1422.txt>

[RFC1738] Berners-Lee, T., Masinter, L., and McCahill, M., Eds., "Uniform Resource Locators (URL)", RFC 1738, December 1994, <http://www.ietf.org/rfc/rfc1738.txt>

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <http://www.rfc-editor.org/rfc/rfc2119.txt>

[RFC2246] Dierks, T., and Allen, C., "The TLS Protocol Version 1.0", RFC 2246, January 1999, <http://www.rfc-editor.org/rfc/rfc2246.txt>

[RFC2478] Baize, E. and Pinkas, D., "The Simple and Protected GSS-API Negotiation Mechanism", RFC 2478, December 1998, <http://www.ietf.org/rfc/rfc2478.txt>

[RFC2616] Fielding, R., Gettys, J., Mogul, J., et al., "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999, <http://www.rfc-editor.org/rfc/rfc2616.txt>

[RFC2617] Franks, J., Hallam-Baker, P., Hostetler, J., et al., "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999, <http://www.rfc-editor.org/rfc/rfc2617.txt>

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, <http://www.ietf.org/rfc/rfc3280.txt>

[RFC3339] Klyne, G. and Newman, C., "Date and Time on the Internet: Timestamps", RFC 3339, July 2002, <http://www.ietf.org/rfc/rfc3339.txt>

[RFC4158] Cooper, M., Dzambasow, Y., Hesse, P., et al., "Internet X.509 Public Key Infrastructure: Certification Path Building", RFC 4158, September 2005, <http://rfc-editor.org/rfc/rfc4158.txt>

[RFC4559] Jaganathan, K., Zhu, L., and Brezak, J., "SPNEGO-based Kerberos and NTLM HTTP Authentication in Microsoft Windows", RFC 4559, June 2006, <http://www.rfc-editor.org/rfc/rfc4559.txt>

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006, <http://www.rfc-editor.org/rfc/rfc4648.txt>

[RFC793] Postel, J., Ed., "Transmission Control Protocol: DARPA Internet Program Protocol Specification", RFC 793, September 1981, <http://www.rfc-editor.org/rfc/rfc793.txt>

[SAMLCore2] Cantor, S., Kemp, J., Philpott, R., and Maler, E., Eds., "Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V2.0", March 2005, <http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf>

[WSFederation1.2] Kaler, C., McIntosh, M., "Web Services Federation Language (WS-Federation)", Version 1.2, May 2009, <http://docs.oasis-open.org/wsfed/federation/v1.2/os/ws-federation-1.2-spec-os.html>

1.2.2 Informative References

[RFC6101] Freier, A., Karlton, P., and Kocher, P., "The Secure Sockets Layer (SSL) Protocol Version 3.0", RFC 6101, August 2011, <http://www.rfc-editor.org/rfc/rfc6101.txt>

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012, <http://www.rfc-editor.org/rfc/rfc6749.txt>

[WSFederation] Kaler, C., Nadalin, A., Bajaj, S., et al., "Web Services Federation Language (WS-Federation)", Version 1.1, December 2006, <http://specs.xmlsoap.org/ws/2006/12/federation/ws-federation.pdf>

1.3 Overview

The Active Directory Federation Services and Proxy system provides services for authentication, authorization, and access to application services located inside the boundaries of the corporate network for clients that are located outside that boundary. The system is composed of Active Directory Federation Services (AD FS) and the Proxy.

AD FS is located inside the boundaries of the corporate network and can run on one server or multiple servers (also known as a "**farm configuration**"). It is a collection of authentication and authorization services exposed to clients over the HTTP protocol [RFC2616]. AD FS implements a set of application authentication protocols including WS-Federation [WSFederation], SAML-P [SAMLCore2], and OAuth [RFC6749].

The Proxy is a service located at the "edge" of the corporate network. It provides proxy services for clients requesting access to application services inside the corporate network and orchestrates access traffic to these services.

The Proxy directs all authentication traffic to the AD FS in the **internal network** and provisions for certificate-based authentication in particular.

The Proxy publishes application services that are located inside the boundaries of the corporate network and makes them available for access to clients that are outside. It "gates" the access to the network by orchestrating the authentication to the edge through the AD FS before allowing the access to the application service (that is, **pre-authentication**).

AD FS defines and implements a protocol that the Proxy supports and that allows the Proxy to orchestrate access to the network by authenticating requests to the edge.

The following diagram illustrates the various components of the system.

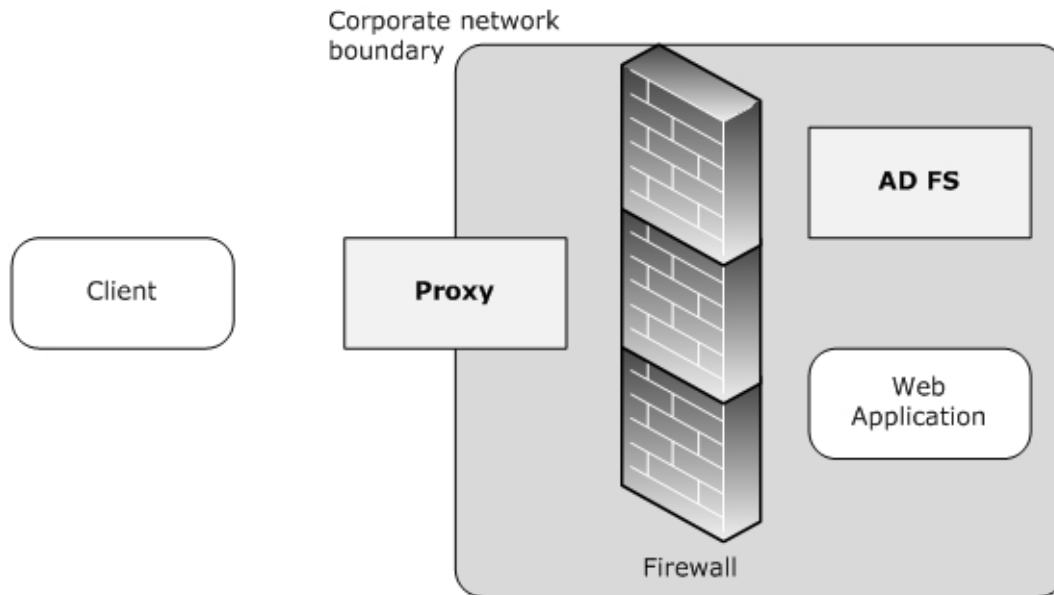


Figure 1: System components

The following components are part of the Active Directory Federation Services and Proxy system:

- **AD FS:** A federation services provider. In this specification this component will be referred to as the server.
- **Proxy:** Both an authentication and an application proxy. In this specification this component will be referred to as the client.

The following components interact with the Active Directory Federation Services and Proxy system:

- **Client:** These components refer to the type of client (for example, browser or rich client) in addition to the identity of the user and the device that is accessing a particular application service.
- **Firewall:** A component that filters traffic flowing between the **perimeter network** and the internal network. In the system described, web traffic is allowed between the Proxy and the AD FS and between the Proxy and the web application.
- **Web Application:** Any web service or application to which a client connects and that typically requires authentication for the user in the client.

This specification describes the distinct areas of interaction between the Proxy and the AD FS.

1.4 Relationship to Other Protocols

The following figure illustrates the relationship of this protocol to other protocols.

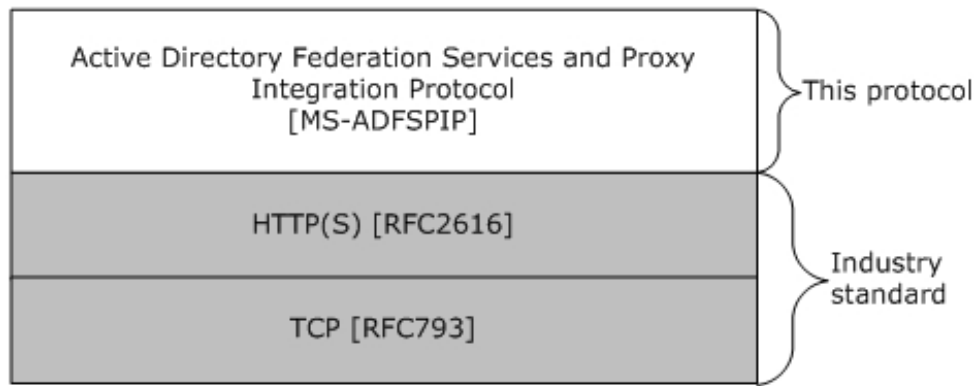


Figure 2: Protocols related to the Active Directory Federation Services and Proxy Integration Protocol

This protocol uses TCP [RFC793] as its transport.

Where specified, this protocol uses base64url encoding ([RFC4648] section 5).

1.5 Prerequisites/Preconditions

No prerequisites or preconditions.

1.6 Applicability Statement

The protocols in the Active Directory Federation Services and Proxy system are applicable to any situation in which the following are important:

1. A proxy for AD FS.
2. Publishing of web applications or services behind-the-firewall to the Internet.
3. Pre-authentication of clients accessing web applications or services behind a firewall.

1.7 Versioning and Capability Negotiation

This protocol does not provide any mechanism for capability negotiation.

1.8 Vendor-Extensible Fields

This protocol does not provide any vendor-extensible fields.

1.9 Standards Assignments

This protocol has not been assigned any standard parameters.

2 Messages

2.1 Transport

The protocol MUST be transported by HTTP/HTTPS [RFC2616]. The protocol requires HTTP/HTTPS ports as specified in section 2.2.2.4, attributes "HttpPort", "HttpsPort" and "HttpsPortForUserTlsAuth", obtained during Proxy (that is, the Web Application Proxy) server registration (section 3.4.5.1).

2.2 Common Data Types

This section defines the set of resource types that are consumed or produced by this protocol. Common element definitions are included in this section.

2.2.1 HTTP Headers

~~Note: Some of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview in the Product Behavior appendix.~~

The following table summarizes the set of HTTP Headers defined by this specification.

Header	Description
X-MS-Endpoint-Absolute-Path	section 2.2.1.3
X-MS-Forwarded-Client-IP	section 2.2.1.2
X-MS-Proxy	section 2.2.1.1
X-MS-Target-Role	section 2.2.1.4
X-MS-ADFS-Proxy-Client-IP	section 2.2.1.5

2.2.1.1 X-MS-Proxy

This header MUST contain the value of the server name of the proxy. This header is included when the proxy is processing client incoming requests as described in the runtime behaviors for the AD FS proxy server details in section 3.6.

```
String = *(%x20-7E)
X-MS-Proxy = String
```

2.2.1.2 X-MS-Forwarded-Client-IP

This header MUST contain the value of the IP address of the client sending the request. This header MUST be included when the proxy is processing incoming requests from clients trying to access the server.

```
String = *(%x20-7E)
X-MS-Forwarded-Client-IP = String
```


2.2.1.3 X-MS-Endpoint-Absolute-Path

This header MUST contain the full URL of the incoming request. This header MUST be included when the proxy is processing incoming requests from clients trying to access the server.

```
String = *(%x20-7E)
X-MS-Endpoint-Absolute-Path = String
```

2.2.1.4 X-MS-Target-Role

This header MUST contain the value "PrimaryComputer" to specify that a given HTTP GET request MUST perform the fetch on a server that has both read and write capabilities on the data.

```
String = *(%x20-7E)
X-MS-Target-Role = String
```

2.2.1.5 X-MS-ADFS-Proxy-Client-IP

~~Note: All of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.~~

The value of this header MUST be set to the IP address of the client's TCP/IP connection to the proxy. This header SHOULD be included when the proxy is processing incoming requests from clients that are trying to access the server. <1>

```
String = *(%x20-7E)
X-MS-ADFS-Proxy-Client-IP = String
```

2.2.2 Complex Types

The following are the defined types used by the protocol details.

2.2.2.1 Proxy Trust

This is a JSON object containing a trust certificate. The format of the object is as follows:

```
{ "SerializedTrustCertificate" : "<certificate>" }
```

certificate: Base64 string encoded ([RFC4648] section 4) X509 certificate [RFC4158].

2.2.2.2 Proxy Trust Renewal

This is a JSON object containing a new trust certificate. The format of the object is as follows:

```
{ "SerializedReplacementCertificate" : "<certificate>" }
```

certificate: Base64 string encoded ([RFC4648] section 4) X509 certificate [RFC4158].

2.2.2.3 Proxy Relying Party Trust

This is a JSON object containing the identifier of the web application for the proxy. The format of the object is as follows:

```
{ "Identifier" : "<web-application-for-client-id>" }
```

web-application-for-client-id: URI of the web application representing the client. The server will issue **tokens** with this value as the audience as described in section 3.13.

2.2.2.4 Configuration

This is a JSON object containing information about the AD FS service. The format of the object is as follows:

```
{
  "ServiceConfiguration" :
  {
    "ServiceHostName" : "<service-host-name>",
    "HttpPort" : "<<http-port-number>>",
    "HttpsPort" : "<<https-port-number>>",
    "HttpsPortForUserTlsAuth" : "<<user-TLS-port-number>>",
    "DeviceCertificateIssuers" : [ "<device-certificate-issuer>", * ],
    "ProxyTrustCertificateLifetime" : "<<trust-renewal-interval>>",
    "DiscoveredUpnSuffixes" : [ "<upn-suffix>", * ],
    "CustomUpnSuffixes" : [ "<upn-suffix>", * ],
    "ServiceHostNameForUserTlsAuth" : "<service-host-name-for-user-tls-auth>"
  },
  "EndpointConfiguration" :
  [
    {
      "Path" : "<endpoint-uri>",
      "PortType" : "<port-type>",
      "AuthenticationSchemes" : "<credential-collection-scheme>",
      "ClientCertificateQueryMode" : "<tls-query-behavior>",
      "CertificateValidation" : "<certificate-validation>",
      "SupportsNtlm" : "<support-ntlm>",
      "ServicePath" : "<service-endpoint-uri>",
      "ServicePortType" : "<service-port-type>"
    }, *
  ],
  "FarmBehavior" : "<farm-behavior-version-number>",
  "IgnoreTokenBinding" : "<ignore-token-binding>"
}
```

service-host-name: Host name of the AD FS service.

service-host-name-for-user-tls-auth: (Optional) Alternate hostname of the AD FS service that implements the endpoint used to authenticate the user using TLS authentication.<2>

http-port-number: Port number for endpoints listening on HTTP.

https-port-number: Port number for endpoints listening on HTTPS.

user-tls-port-number: Port number for user TLS authentication endpoints.

device-certificate-issuer: Base64 string encoded ([RFC4648] section 4) X509 certificate [RFC4158].

trust-renewal-interval: Hint for proxy certificate lifetime.

upn-suffix: Possible User Principal Name (UPN) suffixes for principals that can be preauthorized.

endpoint-uri: URI of endpoint.

port-type: Port Type (section 2.2.2.12) for endpoint.

credential-collection-scheme: Credential Collection Scheme (section 2.2.2.13) for endpoint.

tls-query-behavior: TLS Query Behavior (section 2.2.2.14) for endpoint.

certificate-validation: Certificate Validation (section 2.2.2.15) for endpoint.

support-ntlm: Boolean value that indicates whether the client supports NTLM authentication for SPNEGO-based HTTP authentication [RFC4559].

service-endpoint-uri: URI of endpoint on server. This URI is relative to service-host-name.

service-port-type: Port Type (section 2.2.2.12) for corresponding endpoint on server.

farm-behavior-version-number: (Optional) The following table shows the values of **farm-behavior-version-number** corresponding to the **ad fs behavior level** setting [MS-OAPX] on the server.<3>

<u>ad fs behavior level</u>	<u>farm-behavior-version-number</u>
<u>AD_FS_BEHAVIOR_LEVEL_1</u>	<u>"6.3"</u>
<u>AD_FS_BEHAVIOR_LEVEL_2</u>	<u>"10.0"</u>

If this value is not specified, the value of "6.3" is assumed.

ignore-token-binding: (Optional) A Boolean attribute on the server indicating that token binding information [IETF DRAFT-TOKBND] is not to be retrieved from http.sys for a request and is to be ignored in any existing tokens. The default is true.<4>

2.2.2.5 Relying Party Trust List

This is a JSON array of objects containing web application information. The format of the objects is as follows:

```
[ {  
  "objectIdentifier" : "<object-identifier>",  
  "name" : "<web-application-name>",  
  "publishedThroughProxy" : "<<is-web-application-published>>",  
  "nonClaimsAware" : "<<is-a-non-claims-aware-web-application>>",  
  "enabled" : "<<is-web-application-enabled>>"  
}, + ]
```

object-identifier: The immutable object identifier for the web application on the server.

web-application-name: The name of the web application on the server, unique across web applications.

is-web-application-published: Boolean user configuration declaring this web application as being accessible from outside the internal network through a client.

is-a-non-claims-aware-web-application: Boolean value specifying if the web application is a **non-claims-aware** web application.

enabled: Boolean value specifying if the web application is enabled at the server.

2.2.2.6 Relying Party Trust

This is a JSON object containing detailed web application information. The format of the object is as follows:

```
{
  "objectIdentifier" : "<object-identifier>",
  "name" : "<web-application-name>",
  "publishedThroughProxy" : "<<is-web-application-published>>",
  "nonClaimsAware" : "<<is-a-non-claims-aware-web-application>>",
  "enabled" : "<<is-web-application-enabled>>",
  "identifiers" : [ <web-application-identifier>, * ],
  "proxyTrustedEndpoints" : [ <web-application-at-proxy-endpoint-url>, * ],
  "proxyEndpointMappings" :
    [ { "Key" = "<internal-url>", "Value" = "<external-url>" }, * ]
}
```

object-identifier: The unique object identifier for the web application.

web-application-name: The name of the web application on the server, unique across web applications.

is-web-application-published: Boolean user configuration declaring this web application as accessible from outside the internal network through a client. This value MUST correspond to the value of (proxyTrustedEndpoints.Count > 0).

is-a-non-claims-aware-web-application: Boolean value specifying if the web application is a non-claims-aware web application.

is-web-application-enabled: Boolean value specifying if the web application is enabled at the server.

web-application-identifier: An identifier of the web application on the server.

web-application-at-proxy-endpoint-url: A URL representing an endpoint on the client for the web application where the server will issue tokens to.

internal-url: The internal URL corresponding to the internal-to-external mapping.

external-url: The external URL corresponding to the internal-to-external mapping.

2.2.2.7 Relying Party Trust Publishing Settings

This is a JSON object containing web application publishing information. The format of the object is as follows:

```
{
  "externalUrl" : "<external-url>",
  "internalUrl" : "<internal-url>",
  "proxyTrustedEndpointUrl" : "<web-application-at-proxy-url>"
}
```

external-url: The external URL to be associated with the web application external-to-internal mappings (section 2.2.2.6).

internal-url: The internal URL to be associated with the web application external-to-internal mappings (section 2.2.2.6).

web-application-at-proxy-url: The URL of the endpoint in the client where the server will issue tokens to.

2.2.2.8 Store Entry List

This is a JSON array of store entry objects, which are defined in section 2.2.2.9.

2.2.2.9 Store Entry

This is a JSON object containing store entry information. The format of the object is as follows:

```
{
  "key" : "<entry-key>",
  "version" : "<<entry-version>>",
  "value" : "<entry-value>"
}
```

entry-key: A string that contains the key of the data value for the store entry.

entry-version: A value that specifies the version of the key/value pair for the store entry.

entry-value: The value of the data-blob corresponding to the given key for the store entry.

2.2.2.10 Store Entry Key and Value

This is a JSON object containing the value of a store entry. The format of the object is as follows:

```
{
  "key" : "<entry-key>",
  "value" : "<entry-value>"
}
```

entry-key: A string containing the key of the data value for the store entry.

entry-value: The value of the data-blob corresponding to the given key for the store entry.

2.2.2.11 Serialized Request with Certificate

This is a JSON object containing a serialized [HTTP request that is intended for the target service](#), plus a serialized client certificate and its usage. The format of the object is as follows:

```
{
  "Request" :
  {
    "AcceptTypes" : [ "<accept-type>", * ],
    "Content" : [ <byte>, * ],
    "ContentEncoding" : "<content-encoding>",
    "ContentLength" : "<<content-length>>",
    "ContentType" : "<content-type>",
    "Cookies" :
    [ {
```

```

        "Name" : "<cookie-name>",
        "Value" : "<cookie-value>",
        "Path" : "<cookie-path>",
        "Domain" : "<cookie-domain>",
        "Expires" : "<<cookie-expires>>",
        "Version" : "<<cookie-version>>",
    }, * ],
    "Headers" :
    [ { "Name" : "<header-name>", "Value" : "<header-value>" }, * ],
    "HttpMethod" : "<http-method>",
    "RequestUri" : "<request-uri>",
    "QueryString" : [ { "Name" : "<query-param>", "Value" : "<query-value>" }, * ],
    "UserAgent" : "<user-agent>",
    "UserHostAddress" : "<user-host-address>",
    "UserHostName" : "<user-host-name>",
    "UserLanguages" : [ "<user-language>", * ]
},
"SerializedClientCertificate" : "<serialized-client-certificate>",
"CertificateUsage" : "<certificate-usage>",
}

```

accept-type: A string that represents a MIME accept type supported by the client. This corresponds to a value of the Accept header of the request.

byte: An 8-bit integer in decimal form.

content-encoding: Character set of the entity-body of the request.

content-length: Length in bytes of content sent in the request.

content-type: MIME content type of the request.

cookie-name: Name of the cookie.

cookie-value: Value of the cookie.

cookie-path: Virtual path transmitted with the cookie.

cookie-domain: Domain associated with the cookie.

cookie-expires: Expiration date and time of the cookie.

cookie-version: Version of the cookie.

header-name: Name of header.

header-value: Value of header.

http-method: HTTP data transfer method of the request, for example GET, POST, HEAD.

request-uri: URI of the request.

query-param: Name of the query parameter.

query-value: Value of the query parameter.

user-agent: User agent presented in the request.

user-host-address: IP address and port number to which the request was directed.

user-host-name: DNS name and port number (if provided) specified in the request.

user-language: Natural language preferred for the response.

serialized-client-certificate: Client certificate obtained from TLS handshake base64 string encoded.

certificate-usage: Certificate Type (section 2.2.2.16) for certificate.

2.2.2.12 Port Type

This is an enumeration with the following values:

```
{
  "HttpPort"
  "HttpsPort"
  "HttpsPortForUserTlsAuth"
}
```

2.2.2.13 Credential Collection Scheme

This is an enumeration with the following integer values indicating the type of credential to collect from the client:

```
{
  8
  32768
}
```

8: Indicates basic authentication credentials.

32768: Indicates anonymous authentication.

2.2.2.14 TLS Query Behavior

This is an enumeration with the following values:

```
{
  "None"
  "QueryAndAccept"
  "QueryAndRequire"
}
```

2.2.2.15 Certificate Validation

This is an enumeration with the following values:

```
{
  "None"
  "UserSsl"
  "DeviceIssuedByDrs"
}
```

2.2.2.16 Certificate Type

This is an enumeration with the following values:

```
{
  "User"
  "Device"
}
```

```
}
```

2.2.2.17 Proxy Token

This is a JSON object representing the token issued to the client. The format of the object is defined in [IETF DRAFT-JWS] and is as follows:

```
{
  "ver" : "<version>",
  "aud" : "<audience>",
  "iat" : "<<issued-at>T>L",
  "exp" : "<<expire>T>L",
  "iss" : "<issuer>",
  "relyingpartytrustid" : "<rp-trust-id>",
  "deviceregid" : "<device-registration-id>",
  "authinstant" : "<<auth-instant>T>L",
  "authmethod" : "<auth-method>",
  "upn" : "<upn>"
}
```

version: Token version with a value of 1.0.

audience: Audience for this token. The proxy SHOULD verify that this value matches the value for [Client State].ProxyRelyingPartyTrustIdentifier.

issued-at: Issued at date and time. The proxy SHOULD verify that this value corresponds to a time in the past (before the current time). This is a JSON numeric value representing the number of seconds from 1970-01-01T0:0:0Z **Coordinated Universal Time (UTC)** until the specified UTC date/time. See [RFC3339] for details regarding date/times in general and UTC in particular.

expire: Expiration time of token. The proxy SHOULD verify that this value corresponds to a time in the future (after the current time). This is a JSON numeric value representing the number of seconds from 1970-01-01T0:0:0Z UTC until the specified UTC date/time. See [RFC3339] for details regarding date/times in general and UTC in particular.

issuer: Trusted issuer for this token. The proxy SHOULD verify that this value corresponds to the issuer URI that is published by the server issuing this token through its Federation Metadata [WSFederation1.2].

rp-trust-id: GUID representing application being accessed. The proxy MAY use this value to correlate requests and tokens when listening to multiple requests.

device-registration-id: Identity of the device attempting the access in the form of its certificate thumbprint. The proxy MAY use this value to correlate the client of the request with the client of the token.

auth-instant: Time of authentication. The proxy SHOULD verify that this value corresponds to an earlier time than the issued-at value.

auth-method: Authentication method. The proxy MAY use this value to perform richer authorization of access.

upn: User Principal Name (UPN) of user attempting the access.

2.2.2.18 Combined Token

This is a JSON object containing an access token for the client and an access token for the web application. The format of the object is as follows:


```
{
  "proxy_token" : "<proxy-token>",
  "access_token" : "<access-token>"
}
```

proxy-token: [Proxy Token] (section 2.2.2.17).

access-token: Token issued by the server to the web application.

2.2.2.19 Proxy Token Wrapper

~~Note: All of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.~~

This is a JSON object containing a proxy token as a value on the object. The format of the object is as follows:

```
{
  "authToken" : "<proxy-token>"
}
```

proxy-token: A base64 string encoded ([RFC4648] section 4) [Proxy Token] (section 2.2.2.17).

2.2.2.20 Authentication Request

~~Note: All of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.~~

This is a JSON object containing an authentication request. The format of the object is as follows:

```
{
  "appRealm" : "<web-application-id>",
  "realm" : "<web-application-for-client-id>",
  "username" : "<username>",
  "password" : "<password>",
  "deviceCertificate" : "<device-certificate>",
  "userCertificate" : "<user-certificate>",
  "httpHeaders" :
    [ { "Key" : "<header-name>", "Value" : "<header-value>" }, * ]
}
```

web-application-id: The identifier of the target relying party.

web-application-for-client-id: The identifier of the WAP relying party.

username: The username of the target user.

password: The password of the target user in a base-64-url encoded string.

device-certificate: The certificate used for the device registration in byte[] serialized as the base-64 encoded string.

user-certificate: The certificate to be used to authenticate the user in byte[] serialized as the base-64 encoded string.

header-name: (string) The HTTP header name.

header-value: (string) The value of the corresponding HTTP header.

2.2.2.21 Error Response

~~Note: All of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.~~

This is a JSON object containing exception or error data that the proxy receives from the server to build a response to the client. The format of the object is as follows:

```
{
  "id" : "<<error-id>>",
  "message" : "<message>",
  "type" : "<type>",
}
```

error-id: (DWORD) The identifier of the error encountered. This parameter is not required and can be empty. The following error identifiers can be returned:

- 401 (Unauthorized) – The username, password combination or the user certificate provided is not valid.
- 403 (Forbidden) – The given user is not authorized to access the given relying party. The authorization rules of either the target relying party or the WAP relying party need to be modified.
- 404 (Not Found) – The target relying party or the WAP relying party is not found.
- 412 (Precondition Failed) - If the relying party rules require additional authentication. The additional rules of either the target relying party of the WAP relying party need to be modified.

message: (string) The message corresponding to the error in the user locale of the STS.

type: (string) Additional debug information.

3 Protocol Details

3.1 Common Details

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation of the client and server maintain to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

3.1.1.1 Server State

The following represents the data structure the server MUST hold in order to satisfy these protocol requirements<5>:

```
{
  "ProxyTrustedCertificates" : [ "<certificate-identifier>", * ],
  "ProxyRelyingPartyTrust" : "<web-application-for-proxy>",
  "Configuration" : "<configuration>",
  "RelyingPartyTrusts" : [ "<web-application>", * ],
  "ProxyStore" : [ "StoreEntry" : "<store-entry>", * ]
}
```

certificate-identifier: Data that MUST be used to validate the certificate when presented again.

web-application-for-proxy: Proxy Relying Party Trust (section 2.2.2.3) representing the web application for the client in the server.

configuration: Configuration (section 2.2.2.4) representing service and endpoint configuration.

web-application: Relying Party Trust (section 2.2.2.6) representing an available web application in the server.

store-entry: Store Entry (section 2.2.2.9) containing the triplet of key-version-value of data used by the client for its own consumption.

3.1.1.2 Client State

The following represents the data structure the proxy service MUST hold in order to satisfy these protocol requirements:

```
{
  "TrustCertificate" : "<certificate-with-private-key>",
  "ProxyRelyingPartyTrustIdentifier" : "<web-application-for-client-id>",
  "Configuration" : "<configuration>",
  "RelyingPartyTrusts" : [ "<web-application>", * ]
}
```

certificate-with-private-key: Points to a certificate. The proxy service MUST have a private key for the certificate.

web-application-for-client-id: Identifier of the web application representing the client on the server. This identifier MUST be used by the client when referring to itself on requests to the server.

configuration: Configuration (section 2.2.2.4) obtained from the server.

web-application: Relying Party Trust State (section 3.1.1.3) containing the configuration for a web application on the server.

3.1.1.3 Relying Party Trust State

The following represents the data structure the client MUST hold in order to satisfy these protocol requirements:

```
{  
  "RelyingPartyTrust" : "<web-application>",  
  "RedirectBasedPreauth" : "<<redirect-based-preauth>>"  
}
```

web-application: Relying Party Trust (section 2.2.2.6) representing the web application that the server can issue tokens for.

redirect-based-preauth: Boolean denoting that access from outside the network needs pre-authentication based on HTTP redirects.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Proxy Registration Server Details

3.2.1 Abstract Data Model

None.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

For the system to function properly, the client and the server MUST mutually authenticate each other using client TLS authentication [RFC2246]. For this, the client MUST have the appropriate local configuration to evaluate the trustworthiness of the server TLS certificate and MUST have a client TLS certificate for authenticating itself to the server.

The following resources are required to create and maintain a proper trust configuration between the client and the server.

Resource	Description
Proxy/EstablishTrust	Resource used to establish a trust with the server.
Proxy/RenewTrust	Resource used to renew the trust with the server.

The responses to all the operations can result in the following status codes.

Status code	Description
200	The operation has succeeded.
400	The request is not valid.
401	Unauthorized for specified user credentials or for client TLS certificate.
404	The object does not exist.
405	Invalid verb used in request (GET, DELETE, POST, PUT).
409	The object already exists.
500	Version is not specified where required or any other internal error.
501	Version specified (api-version) is invalid (only valid value is 1).

If the operation authenticates using Integrated Windows authentication [RFC2478], the server MUST validate that the authenticated principal is authorized to do the corresponding operation on the server.

3.2.5.1 Proxy/EstablishTrust

The client MUST first establish a trust with the server in order to act as a Proxy on the system.

3.2.5.1.1 POST

This operation creates a trust based on a Proxy Trust (section 2.2.2.1).

The operation is transported by a HTTP **POST** and can be invoked through the following URIs:

```
adsfs/proxy/EstablishTrust  
adsfs/proxy/PrimaryWriter/EstablishTrust
```

If the operation is invoked through `adsfs/proxy/EstablishTrust`, the request **MUST** authenticate using HTTP Basic authentication [RFC2617].

If the operation is invoked through `adsfs/proxy/PrimaryWriter/EstablishTrust`, the request **MUST** authenticate using Integrated Windows authentication [RFC2478].

The response message for this operation can result in the following status codes.

Status code
200
400
401
500

3.2.5.1.1.1 Request Body

The request body **MUST** be a Proxy Trust (section 2.2.2.1).

3.2.5.1.1.2 Response Body

No response body is returned.

3.2.5.1.1.3 Processing Details

If the operation authenticates using HTTP Basic authentication [RFC2617], the server **MUST** validate that the authenticated principal is authorized to function as a proxy.

The server **MUST** validate that the [Proxy Trust].SerializedTrustCertificate has an **extended key usage (EKU)** for client authentication (1.3.6.1.5.5.7.3.2) ([RFC3280] section 4.2.1.13) and is within the validity period ([RFC1422] section 3.3). If validation fails, the server **MUST** return a HTTP error code of 400.

On successful authentication and authorization, the server **MUST** add [Proxy Trust].SerializedTrustCertificate to [Server State].ProxyTrustedCertificates for future validations.

3.2.5.2 Proxy/RenewTrust

The client **MUST** ensure that the trust with the server remains valid by renewing the trust certificate with the server.

3.2.5.2.1 POST

This operation renews a trust based on a Proxy Trust Renewal (section 2.2.2.2).

The operation is transported by a HTTP **POST** and can be invoked through the following URIs:

```
adsfs/proxy/RenewTrust  
adsfs/proxy/PrimaryWriter/RenewTrust
```

If the operation is invoked through `adfs/proxy/RenewTrust`, the request MUST authenticate using client TLS authentication [RFC2246]. The server MUST validate that the certificate presented by the client during client TLS authentication [RFC2246] can be validated by one of the values of [Server State].ProxyTrustedCertificates. If the certificate cannot be validated, the server MUST return a HTTP error code of 400.

If the operation is invoked through `adfs/proxy/PrimaryWriter/RenewTrust`, the request MUST authenticate using Integrated Windows authentication [RFC2478].

The response message for this operation can result in the following status codes.

Status code
200
400
401
500

3.2.5.2.1.1 Request Body

The request body MUST be Proxy Trust Renewal (section 2.2.2.2).

3.2.5.2.1.2 Response Body

No response body is returned.

3.2.5.2.1.3 Processing Details

The server MUST validate that the [Proxy Trust].SerializedReplacementCertificate has an extended key usage (EKU) for client authentication (1.3.6.1.5.5.7.3.2) ([RFC3280] section 4.2.1.13) and is within the validity period ([RFC1422] section 3.3). If validation fails, the server MUST return a HTTP error code of 400.

The server MUST add [Proxy Trust].SerializedReplacementCertificate to [Server State].ProxyTrustedCertificates for future validations.

3.2.5.3 Proxy/WebApplicationProxy/Trust

The client MUST register with the server as a token recipient with the server before it can function as the Proxy on the system.

3.2.5.3.1 GET

This operation returns a Proxy Relying Party Trust (section 2.2.2.3) corresponding to the web application for the client in the server.

The operation is transported by a HTTP **GET** and can be invoked through the following URI:

```
adfs/proxy/WebApplicationProxy/trust?api-version=1
```

The request MUST authenticate using client TLS authentication [RFC2246]. The server MUST validate that the certificate presented by the client during client TLS authentication [RFC2246] can be validated by one of the values of [Server State].ProxyTrustedCertificates. If the certificate cannot be validated, the server MUST return **an** HTTP error code of 401.

The response message for this operation can result in the following status codes.

Status code
200
400
401
404
500
501

3.2.5.3.1.1 Request Body

The server MUST ignore any request body.

3.2.5.3.1.2 Response Body

The response body MUST be a Proxy Relying Party Trust (section 2.2.2.3).

3.2.5.3.1.3 Processing Details

On successful authentication the server MUST return [Server State].ProxyRelyingPartyTrust (section 3.1.1.1).

3.2.5.3.2 POST

This operation creates the proxy relying party trust based on a Proxy Relying Party Trust (section 2.2.2.3).

The operation is transported by a HTTP **POST** and can be invoked through the following URIs:

```
adfs/proxy/WebApplicationProxy/trust?api-version=1  
adfs/proxy/WebApplicationProxy/PrimaryWriter/trust?api-version=1
```

If the operation is invoked through `adfs/proxy/WebApplicationProxy/trust?api-version=1`, the request MUST authenticate using client TLS authentication [RFC2246].

If the operation is invoked through `adfs/proxy/WebApplicationProxy/PrimaryWriter/trust?api-version=1`, the request MUST authenticate using Integrated Windows authentication [RFC2478].

The response message for this operation can result in the following status codes.

Status code
200
400
401
409

Status code
500
501

3.2.5.3.2.1 Request Body

The request body MUST be a Proxy Relying Party Trust (section 2.2.2.3).

3.2.5.3.2.2 Response Body

No response body is returned.

3.2.5.3.2.3 Processing Details

On successful authentication the server MUST verify that [Server State].ProxyRelyingPartyTrust is not set.

If it is set, the server MUST return a HTTP error code of 409.

If it is not set, the server MUST create the relying party trust for the proxy with an identifier of the received [Proxy Relying Party Trust].Identifier and set the [Server State].ProxyRelyingPartyTrust to the value of the received Proxy Relying Party Trust (section 2.2.2.3).

3.2.5.3.3 DELETE

This operation removes the proxy relying party trust.

The operation is transported by a HTTP **DELETE** and can be invoked through the following URIs:

```

ads/proxy/WebApplicationProxy/trust?api-version=1
ads/proxy/WebApplicationProxy/PrimaryWriter/trust?api-version=1

```

If the operation is invoked through `ads/proxy/WebApplicationProxy/trust?api-version=1`, the request MUST authenticate using client TLS authentication [RFC2246].

If the operation is invoked through `ads/proxy/WebApplicationProxy/PrimaryWriter/trust?api-version=1`, the request MUST authenticate using Integrated Windows authentication [RFC2478].

The response message for this operation can result in the following status codes.

Status code
200
400
401
404
500
501

3.2.5.3.3.1 Request Body

The server MUST ignore any request body.

3.2.5.3.3.2 Response Body

No response body is returned.

3.2.5.3.3.3 Processing Details

On successful authentication the server MUST verify that [Server State].ProxyRelyingPartyTrust is set.

If it is not set the server MUST return a HTTP error code of 404.

If it is set the server MUST remove the relying party trust for the proxy and clear the [Server State].ProxyRelyingPartyTrust value.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

3.3 Proxy Registration Client Details

3.3.1 Abstract Data Model

None.

3.3.2 Timers

None.

3.3.3 Initialization

None.

3.3.4 Higher-Layer Triggered Events

None.

3.3.5 Message Processing Events and Sequencing Rules

See corresponding section on Server Details.

In all operations where the server requires authenticating the proxy using client TLS authentication [RFC2246], the proxy MUST present the certificate on [~~Proxy ServiceClient~~ State-Data].TrustCertificate during client TLS authentication.

3.3.5.1 Proxy/EstablishTrust

See corresponding section on Server Details.

3.3.5.1.1 POST

See corresponding section on Server Details.

3.3.5.1.1.1 Request Body

See corresponding section on Server Details.

3.3.5.1.1.2 Response Body

See corresponding section on Server Details.

3.3.5.1.1.3 Processing Details

[Proxy Trust].SerializedTrustCertificate MUST have an EKU for client authentication (1.3.6.1.5.5.7.3.2) ([RFC3280] section 4.2.1.13) and MUST be within validity period ([RFC1422] section 3.3). The client MUST have the private key of this certificate.

If the server response is a HTTP status code of 200 the proxy MUST set [Client State].TrustCertificate to [Proxy Trust].SerializedTrustCertificate for future authentication to the server.

3.3.5.2 Proxy/RenewTrust

See corresponding section on Server Details.

3.3.5.2.1 POST

See corresponding section on Server Details.

3.3.5.2.1.1 Request Body

See corresponding section on Server Details.

3.3.5.2.1.2 Response Body

See corresponding section on Server Details.

3.3.5.2.1.3 Processing Details

[Proxy Trust].SerializedReplacementCertificate MUST have an EKU for client authentication (1.3.6.1.5.5.7.3.2) ([RFC3280] section 4.2.1.13) and MUST be within validity period ([RFC1422] section 3.3). The proxy MUST have the private key of this certificate.

If the server response is a HTTP status code of 200 the proxy MUST set [Client State].TrustCertificate to [Proxy Trust].SerializedReplacementCertificate for future authentication to the server.<6>

3.3.5.3 Proxy/WebApplicationProxy/Trust

See corresponding section on Server Details.

3.3.5.3.1 GET

See corresponding section on Server Details.

3.3.5.3.1.1 Request Body

See corresponding section on Server Details.

3.3.5.3.1.2 Response Body

See corresponding section on Server Details.

3.3.5.3.1.3 Processing Details

No processing details.

3.3.5.3.2 POST

See corresponding section on Server Details.

3.3.5.3.2.1 Request Body

See corresponding section on Server Details.

3.3.5.3.2.2 Response Body

See corresponding section on Server Details.

3.3.5.3.2.3 Processing Details

If the server response is a HTTP status code of 200 the proxy MUST set [Client State].ProxyRelyingPartyTrustIdentifier to [Proxy Relying Party Trust].Identifier.

3.3.5.3.3 DELETE

See corresponding section on Server Details.

3.3.5.3.3.1 Request Body

See corresponding section on Server Details.

3.3.5.3.3.2 Response Body

See corresponding section on Server Details.

3.3.5.3.3.3 Processing Details

If the server response is a HTTP status code of 200 the proxy MUST clear [Client State].ProxyRelyingPartyTrustIdentifier.

3.3.6 Timer Events

None.

3.3.7 Other Local Events

None.

3.4 Service Configuration Server Details

3.4.1 Abstract Data Model

None.

3.4.2 Timers

None.

3.4.3 Initialization

None.

3.4.4 High-Layer Triggered Events

None.

3.4.5 Message Processing Events and Sequencing Rules

For the proxy to function properly as a proxy component on the system, it MUST retrieve information from the server about the service configuration and the endpoints it listens to, and about the available relying party trusts.

The following resources are required to retrieve server configuration.

Resource	Description
Proxy/GetConfiguration	Resource used to retrieve service and endpoint configuration.
Proxy/RelyingPartyTrusts	Resource used to retrieve all relying party trusts.
Proxy/RelyingPartyTrusts/{Identity}	Resource used to retrieve a particular relying party trust.

The responses to all the operations can result in the following status codes.

Status code	Description
200	The operation has succeeded.
400	The request is not valid.
401	Unauthorized for specified user credentials or for client TLS certificate.
404	The object does not exist.
405	Invalid verb used in request (GET, DELETE, POST, PUT).
409	The object already exists.
500	Version is not specified where required or any other internal error.
501	Version specified (api-version) is invalid (only valid value is values are 1 and 2).<7>

For all operations in this section, the server requires authenticating the proxy using client TLS authentication [RFC2246]. The server MUST validate that the certificate that is presented by the proxy during client TLS authentication can be validated by one of the values of [Server

State].ProxyTrustedCertificates. If the certificate cannot be validated, the server MUST return an HTTP error code of 401.

3.4.5.1 Proxy/GetConfiguration

The server MUST provide configuration for the ~~client's~~client's run-time function.

3.4.5.1.1 GET

This operation returns a Configuration (section 2.2.2.4) containing service and end-point configuration.

The operation is transported by a HTTP **GET** and can be invoked through the following ~~URI~~URIs:

```
adfs/proxy/GetConfiguration?api-version=1
```

~~The request MUST authenticate using client TLS authentication [RFC2246]. The server MUST validate that the certificate presented by the client during client TLS authentication [RFC2246] can be validated by one of the values of [Server State].ProxyTrustedCertificates. If the certificate cannot be validated the server MUST return a HTTP error code of 400.~~

```
adfs/proxy/GetConfiguration?api-version=2
```

If the server is operating on AD FS BEHAVIOR LEVEL 2 or higher ([MS-OAPX] section 3.2.1.1) and has ServiceHostNameForUserTlsAuth configured, it MUST return error 500 to the client for *api-version=1*.

The response message for this operation can result in the following status codes.

Status code
200
400
405
500

3.4.5.1.1.1 Request Body

The server MUST ignore any request body.

3.4.5.1.1.2 Response Body

The response body MUST be a Configuration (section 2.2.2.4).

3.4.5.1.1.3 Processing Details

On successful authentication the server MUST return a [Server State].Configuration (section 3.1.1.1). For requests with *api-version=2*, the Configuration can also contain *ServiceHostNameForUserTlsAuth* (section 2.2.2.4).

3.4.5.2 Proxy/RelyingPartyTrusts

The proxy MUST retrieve information about relying party trusts to obtain relying party trust object identifiers that the proxy MUST use when identifying relying party trusts on requests to the server.

3.4.5.2.1 GET

This operation returns a Relying Party Trust List (section 2.2.2.5) containing all available relying party trusts.

The operation is transported by a HTTP **GET** and can be invoked through the following URI:

```
adsfs/proxy/RelyingPartyTrusts?api-version=1
```

~~The request MUST authenticate using client TLS authentication [RFC2246]. The server MUST validate that the certificate presented by the client during client TLS authentication [RFC2246] can be validated by one of the values of [Server State].ProxyTrustedCertificates. If the certificate cannot be validated the server MUST return a HTTP error code of 401.~~

The response message for this operation can result in the following status codes.

Status code
200
400
401
404
500
501

3.4.5.2.1.1 Request Body

The server MUST ignore any request body.

3.4.5.2.1.2 Response Body

The response body MUST be a Relying Party Trust List (section 2.2.2.5).

3.4.5.2.1.3 Processing Details

On successful authentication the server MUST return a [Server State].RelyingPartyTrusts (section 3.1.1).

3.4.5.3 Proxy/RelyingPartyTrusts/

This resource is available for the client to access data about a specific web application identified by {Identifier}.

3.4.5.3.1 GET

This operation returns a Relying Party Trust (section 2.2.2.6) containing information specific to a relying party trust.

The operation is transported by a HTTP **GET** and can be invoked through the following URI:

```
adfs/proxy/RelyingPartyTrusts/{Identifier}?api-version=1
```

~~The request MUST authenticate using client TLS authentication [RFC2246]. The server MUST validate that the certificate presented by the client during client TLS authentication [RFC2246] can be validated by one of the values of [Server State].ProxyTrustedCertificates. If the certificate cannot be validated the server MUST return a HTTP error code of 401.~~

The response message for this operation can result in the following status codes.

Status code
200
400
401
404
500
501

3.4.5.3.1.1 Request Body

The server MUST ignore any request body.

3.4.5.3.1.2 Response Body

The response body MUST be a Relying Party Trust (section 2.2.2.6).

3.4.5.3.1.3 Processing Details

On successful authentication the server MUST return a [Server State].RelyingPartyTrusts for the relying party trust with [Relying Party Trust].ObjectIdentifier equal to the URI {Identifier} value (section 3.1.1).

3.4.6 Timer Events

None.

3.4.7 Other Local Events

None.

3.5 Service Configuration Client Details

3.5.1 Abstract Data Model

None.

3.5.2 Timers

None.

3.5.3 Initialization

None.

3.5.4 High-Layer Triggered Events

None.

3.5.5 Message Processing Events and Sequencing Rules

See corresponding section on Server Details.

~~In~~For all operations ~~where the server requires authenticating the client using client TLS authentication [RFC2246];~~in this section, the client MUST ~~do~~perform client TLS authentication [RFC2246] using the certificate in [~~Proxy ServiceClient State-Data~~].TrustCertificate.

3.5.5.1 Proxy/GetConfiguration

See corresponding section on Server Details.

3.5.5.1.1 GET

See corresponding section on Server Details.

3.5.5.1.1.1 Request Body

See corresponding section on Server Details.

3.5.5.1.1.2 Response Body

See corresponding section on Server Details.

3.5.5.1.1.3 Processing Details

If the server response is a HTTP status code of 200 the proxy MUST set [Client State].Configuration to Configuration obtained in the response.

3.5.5.2 Proxy/RelyingPartyTrusts

See corresponding section on Server Details.

3.5.5.2.1 GET

See corresponding section on Server Details.

3.5.5.2.1.1 Request Body

See corresponding section on Server Details.

3.5.5.2.1.2 Response Body

See corresponding section on Server Details.

3.5.5.2.1.3 Processing Details

None.

3.5.5.3 Proxy/RelyingPartyTrusts/

See corresponding section on Server Details.

3.5.5.3.1 GET

See corresponding section on Server Details.

3.5.5.3.1.1 Request Body

See corresponding section on Server Details.

3.5.5.3.1.2 Response Body

See corresponding section on Server Details.

3.5.5.3.1.3 Processing Details

None.

3.5.6 Timer Events

None.

3.5.7 Other Local Events

None.

3.6 Proxy Configuration Server Details

3.6.1 Abstract Data Model

None.

3.6.2 Timers

None.

3.6.3 Initialization

None.

3.6.4 High-Layer Triggered Events

None.

3.6.5 Message Processing Events and Sequencing Rules

The proxy MAY use the server store to save and retrieve information about the proxy service or about applications published through the proxy. The server provides resources to set and retrieve information based on a key/value pair entry model.

The following resources are available to store custom proxy configuration on the server.

Resource	Description
Proxy/WebApplicationProxy/Store	Resource used to retrieve all entries in the store.
Proxy/WebApplicationProxy/Store/{Key}	Resource used to add, retrieve, remove, or modify an entry in the store.

The responses to all the operations can result in the following status codes.

Status code	Description
200	The operation has succeeded.
400	The request is not valid.
401	Unauthorized for the specified user credentials or for the client TLS certificate.
404	The object does not exist.
405	Invalid verb used in request (GET, DELETE, POST, PUT).
409	The object already exists.
412	A precondition failed.
500	Version is not specified where required or any other internal error.
501	Version specified (api-version) is invalid (only valid value is 1).

InFor all operations wherein this section, the server requires authenticating the proxy using client TLS authentication [RFC2246], the. The server MUST validate that the certificate that is presented by the proxy during client TLS authentication can be validated by one of the values of [Server State].ProxyTrustedCertificates. If the certificate cannot be validated, the server MUST return an HTTP error code of 401.

3.6.5.1 Proxy/WebApplicationProxy/Store

The proxy MAY retrieve entries from the store by means of this resource.

3.6.5.1.1 GET

This operation returns a Store Entry List (section 2.2.2.8) containing all entries in the store.

The operation is transported by a HTTP **GET** and can be invoked through the following URI:

```
ads/proxy/WebApplicationProxy/Store?api-version=1
```

The request MUST authenticate using client TLS authentication [RFC2246].

The response message for this operation can result in the following status codes.

Status code
200
400
401
404
409
500
501

3.6.5.1.1.1 Request Body

The server MUST ignore any request body.

3.6.5.1.1.2 Response Body

The response body MUST be a Store Entry List (section 2.2.2.8).

3.6.5.1.1.3 Processing Details

Upon successful authentication the server MUST return [Server State].ProxyStore (section 3.1.1).

3.6.5.2 Proxy/WebApplicationProxy/Store/

The client MAY use the store to retrieve, add, remove or modify a particular entry from the store by making requests of this resource.

3.6.5.2.1 GET

This operation returns a Store Entry (section 2.2.2.9) containing its version and value.

The operation is transported by a HTTP **GET** and can be invoked through the following URI:

```
adfs/proxy/WebApplicationProxy/Store/{Key}?api-version=1
```

The request MUST authenticate using client TLS authentication [RFC2246].

The response message for this operation can result in the following status codes.

Status code
200
400
401
404

Status code
500
501

3.6.5.2.1.1 Request Body

The server MUST ignore any request body.

3.6.5.2.1.2 Response Body

The response body MUST be a Store Entry (section 2.2.2.9).

3.6.5.2.1.3 Processing Details

Upon successful authentication the server MUST return the Store Entry (section 2.2.2.9) represented by the object in [Server State].ProxyStore that has a key value with the same string value as {Key}.

If after successful authentication a Store Entry with the same string value as {Key} is not present in [Server State].ProxyStore, the server MUST return an HTTP error code of 404.

3.6.5.2.2 POST

This operation adds a new entry to the store.

The operation is transported by a HTTP **POST** and can be invoked through the following URIs:

```

adsf/proxy/WebApplicationProxy/Store/{Key}?api-version=1
adsf/proxy/PrimaryWriter/WebApplicationProxy/Store/{Key}?api-version=1

```

If the operation is invoked through `adsf/WebApplicationProxy/Store/{Key}`, the request MUST authenticate using client TLS authentication [RFC2246].

If the operation is invoked through `adsf/proxy/PrimaryWriter/WebApplicationProxy/Store/{Key}`, the request MUST authenticate using Integrated Windows authentication [RFC2478].

The response message for this operation can result in the following status codes.

Status code
200
400
401
409
500
501

3.6.5.2.2.1 Request Body

The request body is a Store Entry Key and Value (section 2.2.2.10).

3.6.5.2.2.2 Response Body

No response body is returned.

3.6.5.2.2.3 Processing Details

On successful authentication the server **MUST** validate that the URI value of {Key} is the same as the value of [Store Entry Key and Value].key from the request body.

If it is not the same the server **MUST** return a HTTP error code of 400.

If it is the same the server **MUST** add the entry to the store by adding Store Entry Key and Value with a version of 1 to [Server State].ProxyStore.

If there is an existing value for the key specified then the server **MUST** return a HTTP error code of 409.

3.6.5.2.3 PUT

This operation modifies the value of an existing entry in the store.

The operation is transported by a HTTP **PUT** and can be invoked through the following URIs:

```
adsfs/proxy/WebApplicationProxy/Store/{Key}?api-version=1
adsfs/proxy/PrimaryWriter/WebApplicationProxy/Store/{Key}?api-version=1
```

If the operation is invoked through `adsfs/WebApplicationProxy/Store/{Key}`, the request **MUST** authenticate using client TLS authentication [RFC2246].

If the operation is invoked through `adsfs/proxy/PrimaryWriter/WebApplicationProxy/Store/{Key}`, the request **MUST** authenticate using Integrated Windows authentication [RFC2478].

The response message for this operation can result in the following status codes.

Status code
200
400
401
404
412
500
501

3.6.5.2.3.1 Request Body

The request body is a Store Entry (section 2.2.2.9).

3.6.5.2.3.2 Response Body

No response body is returned.

3.6.5.2.3.3 Processing Details

On successful authentication the server MUST validate that the URI value of {Key} is the same as the value of [Store Entry].key from the request body.

If it is not the same the server MUST return a HTTP error code of 400.

If it is the same the server MUST find a corresponding Store Entry on [Server State].ProxyStore for the corresponding key.

If it is not found the server MUST return a HTTP error code of 404.

If it is found the server MUST validate that the value [Store Entry].version of the entry found is the same as the value of [Store Entry].version from the request body.

If it is not the same the server MUST return a HTTP error code of 412.

If it is the same the server MUST set the value of [Store Entry].value of the corresponding Store Entry on [Server State].ProxyStore to the [Store Entry].value and MUST increment by 1 its value of [Store Entry].version.

3.6.5.2.4 DELETE

This operation ~~modifies~~removes the value of an existing entry in the store.

The operation is transported by ~~an~~ HTTP ~~PUT~~DELETE and can be invoked through the following URIs:

```
adfs/proxy/WebApplicationProxy/Store/{Key}?api-version=1
adfs/proxy/PrimaryWriter/WebApplicationProxy/Store/{Key}?api-version=1
```

If the operation is invoked through adfs/WebApplicationProxy/Store/{Key}, the request MUST authenticate using client TLS authentication [RFC2246].

If the operation is invoked through adfs/proxy/PrimaryWriter/WebApplicationProxy/Store/{Key}, the request MUST authenticate using Integrated Windows authentication [RFC2478].

The response message for this operation can result in the following status codes.

Status code
200
401
404
500
501

3.6.5.2.4.1 Request Body

The server MUST ignore any request body.

3.6.5.2.4.2 Response Body

No response body is returned.

3.6.5.2.4.3 Processing Details

On successful authentication the server **MUST find**looks for a corresponding Store Entry on [Server State].ProxyStore for {Key}.

If it is not found, the server MUST return a HTTP error code of 404.

If it is found, the server MUST remove the Store Entry from [Server State].ProxyStore.

3.6.6 Timer Events

None.

3.6.7 Other Local Events

None.

3.7 Proxy Configuration Client Details

3.7.1 Abstract Data Model

None.

3.7.2 Timers

None.

3.7.3 Initialization

None.

3.7.4 High-Layer Triggered Events

None.

3.7.5 Message Processing Events and Sequencing Rules

For all operations in this section, the client MUST perform client TLS authentication [RFC2246] using the certificate in [Client State].TrustCertificate.

See corresponding section on Server Details.

3.7.5.1 Proxy/WebApplicationProxy/Store

See corresponding section on Server Details.

3.7.5.1.1 GET

See corresponding section on Server Details.

3.7.5.1.1.1 Response Body

See corresponding section on Server Details.

3.7.5.1.1.2 Request Body

See corresponding section on Server Details.

3.7.5.1.1.3 Processing Details

None.

3.7.5.2 Proxy/WebApplicationProxy/Store/

See corresponding section on Server Details.

3.7.5.2.1 GET

See corresponding section on Server Details.

3.7.5.2.1.1 Request Body

See corresponding section on Server Details.

3.7.5.2.1.2 Response Body

See corresponding section on Server Details.

3.7.5.2.1.3 Processing Details

None.

3.7.5.2.2 POST

See corresponding section on Server Details.

3.7.5.2.2.1 Request Body

See corresponding section on Server Details.

3.7.5.2.2.2 Response Body

See corresponding section on Server Details.

3.7.5.2.2.3 Processing Details

None.

3.7.5.2.3 PUT

See corresponding section on Server Details.

3.7.5.2.3.1 Request Body

See corresponding section on Server Details.

3.7.5.2.3.2 Response Body

See corresponding section on Server Details.

3.7.5.2.3.3 Processing Details

None.

3.7.5.2.4 DELETE

See corresponding section on Server Details.

3.7.5.2.4.1 Request Body

See corresponding section on Server Details.

3.7.5.2.4.2 Response Body

See corresponding section on Server Details.

3.7.5.2.4.3 Processing Details

None.

3.7.6 Timer Events

None.

3.7.7 Other Local Events

None.

3.8 Application Publishing Server Details

3.8.1 Abstract Data Model

None.

3.8.2 Timers

None.

3.8.3 Initialization

None.

3.8.4 High-Layer Triggered Events

None.

3.8.5 Message Processing Events and Sequencing Rules

The following resources are available to set the publishing settings to web applications.

Resource	Description
Proxy/RelyingPartyTrusts/{Identity}/ PublishingSettings PublishedSettings	Resource used to publish a particular web application through the client.

The responses to all the operations can result in the following status codes.

Status code	Description
200	The operation has succeeded.
400	The request is not valid.
401	Unauthorized for the specified user credentials or for the client TLS certificate.
404	The object does not exist.
405	Invalid verb used in request (GET, DELETE, POST, PUT).
409	The object already exists.
500	Version is not specified where required or any other internal error.
501	Version specified (api-version) is invalid (only valid value is 1).

In ~~For~~ all operations ~~wherein this section~~, the server requires authenticating the proxy using client TLS authentication [RFC2246], ~~the~~. The server MUST validate that the certificate ~~that is~~ presented by the proxy during client TLS authentication can be validated by one of the values of [Server State].ProxyTrustedCertificates. If the certificate cannot be validated, the server MUST return ~~an~~ HTTP error code of 401.

If the operation authenticates using Integrated Windows authentication [RFC2478], the server MUST validate that the authenticated principal is authorized to do the corresponding operation on the server.

3.8.5.1 Proxy/RelyingPartyTrusts/{Identifier}/~~PublishingSettings~~PublishedSettings

3.8.5.1.1 POST

This operation creates a new set of publishing settings on a relying party trust.

The operation is transported by a HTTP **POST** and can be invoked through the following URIs:

```

adsf/proxy/RelyingPartyTrusts/{Identifier}/PublishingSettingsPublishedSettings?api-version=1
adsf/proxy/PrimaryWriter/RelyingPartyTrusts/{Identifier}/PublishingSettingsPublishedSettings?api-version=1

```

If the operation is invoked through

adsf/proxy/RelyingPartyTrusts/{Identifier}/~~PublishingSettings~~PublishedSettings?api-version=1, the request MUST authenticate using client TLS authentication [RFC2246].

If the operation is invoked through

adsf/proxy/PrimaryWriter/RelyingPartyTrusts/{Identifier}/~~PublishingSettings~~PublishedSettings?api-version=1, the request MUST authenticate using Integrated Windows authentication [RFC2478].

The response message for this operation can result in the following status codes.

Status code
200
400
401
404

Status code
409
500
501

3.8.5.1.1.1 Request Body

The request body MUST be a Relying Party Trust Publishing Settings (section 2.2.2.7).

3.8.5.1.1.2 Response Body

No response body is returned.

3.8.5.1.1.3 Processing Details

If the publishing settings specified in Relying Party Trust Publishing Settings have been set previously on [Server State].RelyingPartyTrusts, the server MUST return a HTTP error code of 409.

If they have not been set the server MUST add the Relying Party Trust Publishing Settings for the relying party trust identifier with {Identifier}. The server MUST add a new URL to [Server State].RelyingPartyTrusts[objectIdentifier:={Identifier}].proxyTrustedEndpoints with the value of [Relying Party Trust Publishing Settings].proxyTrustedEndpointUrl and add a new mapping to [Server State].RelyingPartyTrusts[objectIdentifier:={Identifier}].proxyEndpointMappings with the value of [Relying Party Trust Publishing Settings].internalURL to Key and [Relying Party Trust Publishing Settings].externalURL to Value.

3.8.5.1.2 DELETE

This operation removes the publishing settings for a relying party trust.

The operation is transported by a HTTP **DELETE** and can be invoked through the following URIs:

```

ads/proxy/RelyingPartyTrusts/{Identifier}/PublishingSettingsPublishedSettings?api-version=1
ads/proxy/PrimaryWriter/RelyingPartyTrusts/{Identifier}/PublishingSettingsPublishedSettings?
api-version=1

```

If the operation is invoked through

ads/proxy/RelyingPartyTrusts/{Identifier}/~~PublishingSettingsPublishedSettings~~?api-version=1, the request MUST authenticate using client TLS authentication [RFC2246].

If the operation is invoked through

ads/proxy/PrimaryWriter/RelyingPartyTrusts/{Identifier}/~~PublishingSettingsPublishedSettings~~?api-version=1, the request MUST authenticate using Integrated Windows authentication [RFC2478].

The response message for this operation can result in the following status codes.

Status code
200
400

Status code
401
404
500
501

3.8.5.1.2.1 Request Body

The request body MUST be a Relying Party Trust Publishing Settings (section 2.2.2.7).

3.8.5.1.2.2 Response Body

No response body is returned.

3.8.5.1.2.3 Processing Details

If the publishing settings specified in Relying Party Trust Publishing Settings have not been set previously the server MUST return a HTTP error code of 404.

If they have been set then use the following algorithm for processing this request:

1. If the [Relying Party Trust Publishing Settings].proxyTrustedEndpointUrl is missing or the [Relying Party Trust Publishing Settings].internalUrl is present in the request body, the server MUST return an HTTP error code of 400.
2. If the Relying Party Trust (section 2.2.2.6) with objectIdentifier with the same string value as {Identifier} in [Server State].RelyingPartyTrusts is not found, or if [Server State].RelyingPartyTrusts[objectIdentifier:={Identifier}].proxyTrustedEndpoints with the value of [Relying Party Trust Publishing Settings].proxyTrustedEndpointUrl is not found, or if [Relying Party Trust Publishing Settings].externalUrl is specified and an entry with the matching externalUrl is not found in [Server State].RelyingPartyTrusts[objectIdentifier:={Identifier}].proxyEndpointMappings, the server MUST return an HTTP error code of 404.
3. On the Relying Party Trust (section 2.2.2.6) [Server State].RelyingPartyTrusts[objectIdentifier:={Identifier}], remove the entry from Relying Party Trust Publishing Settings (section 2.2.2.7) that contains in proxyTrustedEndpointUrl the value of [Relying Party Trust Publishing Settings].proxyTrustedEndpointUrl from the request body.
4. If the value of [Relying Party Trust Publishing Settings].externalUrl is present in the request body, remove the entry from [Server State].RelyingPartyTrusts[objectIdentifier:={Identifier}].proxyEndpointMappings that has an externalUrl matching of [Relying Party Trust Publishing Settings].externalUrl from the request body.

3.8.6 Timer Events

None.

3.8.7 Other Local Events

None.

3.9 Application Publishing Client Details

3.9.1 Abstract Data Model

None.

3.9.2 Timers

None.

3.9.3 Initialization

None.

3.9.4 High-Layer Triggered Events

None.

3.9.5 Message Processing Events and Sequencing Rules

See corresponding section on Server Details.

In all operations where the server requires authenticating the client using client TLS authentication [RFC2246], the client MUST ~~use perform client TLS authentication [RFC2246] using~~ the certificate ~~represented by [Proxy Service] in [Client State Data].TrustCertificate during client TLS authentication.~~

3.9.5.1 Proxy/RelyingPartyTrusts/{Identifier}/~~PublishingSettings~~PublishedSettings

See corresponding section on Server Details.

3.9.5.1.1 POST

See corresponding section on Server Details.

3.9.5.1.1.1 Request Body

See corresponding section on Server Details.

3.9.5.1.1.2 Response Body

See corresponding section on Server Details.

3.9.5.1.1.3 Processing Details

If the server response is a HTTP status code of 200 the proxy MUST add a new identifier object to [Client State].RelyingPartyTrusts with the RelyingPartyTrust.Identifier set to {Identifier}.

3.9.5.1.2 DELETE

See corresponding section on Server Details.

3.9.5.1.2.1 Request Body

See corresponding section on Server Details.

3.9.5.1.2.2 Response Body

See corresponding section on Server Details.

3.9.5.1.2.3 Processing Details

If the server response is a HTTP status code of 200 the proxy MUST remove from [Client State].RelyingPartyTrusts the object with RelyingPartyTrust.Identifier with the same string value as {Identifier}.

3.9.6 Timer Events

None.

3.9.7 Other Local Events

None.

3.10 Proxy Runtime Behaviors Server Details

3.10.1 Abstract Data Model

None.

3.10.2 Timers

None.

3.10.3 Initialization

None.

3.10.4 High-Layer Triggered Events

None.

3.10.5 Message Processing Events and Sequencing Rules

The following resource is available to send a request along with the certificate to the server.

Resource	Description
BackEndProxyTLS	Resource used to obtain a request along with the certificate used for client TLS authentication [RFC2246].

The responses to all the operations can result in the following status codes.

Status code	Description
200	The operation has succeeded.
400	The request is not valid.
401	Unauthorized for client TLS certificate.

Status code	Description
500	Internal error.

InFor all operations wherein this section, the server requires authenticating the proxy using client TLS authentication [RFC2246], the. The server MUST validate that the certificate that is presented by the client during client TLS authentication can be validated by one of the values of [Server State].ProxyTrustedCertificates. If the certificate cannot be validated, the server MUST return an HTTP error code of 401.

3.10.5.1 BackEndProxyTLS

The proxy MUST support client TLS authentication [RFC2246] on behalf of the server by obtaining the certificate and forwarding it along with the receiving message to the server.

3.10.5.1.1 POST

This operation obtains a request along with a certificate.

The operation is transported by a HTTP **POST** and can be invoked through the following URI:

```
adfs/backendproxytls
```

The server requires authenticating the client using client TLS authentication [RFC2246].

The response message for this operation can result in the following status codes.

Status code
200
400
401
500

3.10.5.1.1.1 Request Body

The request body MUST be a base64url encoded ([RFC4648] section 5) Serialized Request with Certificate (section 2.2.2.11).

3.10.5.1.1.2 Response Body

No response body is returned.

3.10.5.1.1.3 Processing Details

The server MUST treat [Serialized Request with Certificate].SerializedClientCertificate as the certificate of the end user, and SHOULD assume that the client has already verified the original requester's requester's proof of possession of the private key corresponding to that certificate.

The server MUST process the request as if it was received directly to the endpoint in the server as specified in the request.

3.10.6 Timer Events

None.

3.10.7 Other Local Events

None.

3.11 Proxy Runtime Behaviors Client Details

3.11.1 Abstract Data Model

None.

3.11.2 Timers

None.

3.11.3 Initialization

None.

3.11.4 High-Layer Triggered Events

None.

3.11.5 Message Processing Events and Sequencing Rules

The client SHOULD listen for HTTP requests based on the server characteristics in [Client State].Configuration.

For each object, CurrentEndpointConfiguration in [Client State].Configuration.EndpointConfiguration, the client SHOULD do the following:

1. Listen for HTTP requests whose URLs conform to the following rules:
 1. HostName of the URL is one of the following:
 - [Client State].Configuration.ServiceConfiguration.ServiceHostName
 - [Client State].Configuration.ServiceConfiguration.ServiceHostNameForUserTlsAuth
 - "EnterpriseRegistration.<PossibleUpnSuffix>" where <PossibleUpnSuffix> is one of either [Client State].Configuration.ServiceConfiguration.DiscoveredUpnSuffixes or [Client State].Configuration.ServiceConfiguration.CustomUpnSuffixes.
 2. If CurrentEndpointConfiguration.PortType is "HttpPort", the port component of the URL is [ServiceConfiguration.HttpPort].
 3. If CurrentEndpointConfiguration.PortType is "HttpsPort", the port component of the URL is [ServiceConfiguration.HttpsPort].
 4. If CurrentEndpointConfiguration.PortType is "HttpsPortForUserTlsAuth", the port component of the URL is [ServiceConfiguration.HttpsPortForUserTlsAuth].
 5. The Path component of the URL is a subpath of [CurrentEndpointConfiguration.Path].

2. If `CurrentEndpointConfiguration.ClientCertificateQueryMode` is "QueryAndAccept", and the request does not have a public key authentication header or user agent indicator ([MS-PKAP] section 3.1.5.1.1), then the client SHOULD attempt to retrieve end-user X509 certificate [RFC4158] using client TLS authentication [RFC2246]. If it obtains a certificate the client MUST follow processing in section 3.11.5.1.
3. If `CurrentEndpointConfiguration.ClientCertificateQueryMode` is "QueryAndRequire", then the client SHOULD attempt to retrieve end-user X509 certificate [RFC4158] using client TLS authentication [RFC2246]. If it obtains a certificate, the client MUST follow the processing in section 3.11.5.1. If it does not obtain a certificate, it SHOULD return a HTTP error code of 204.
4. If `CurrentEndpointConfiguration.SupportsNtlm` is true, the client SHOULD ensure that SPNEGO-based authentication requests [RFC4559] with the "Negotiate" auth-scheme are converted to NTLM.
5. If no certificate was obtained in steps 2 or 3, then the client SHOULD replay the request as follows:
 1. The request SHOULD be made to the following URL:
 1. If `CurrentEndpointConfiguration.ServicePortType` is "HttpPort", then form the URL as "http://[ServiceConfiguration.ServiceHostName]:[ServiceConfiguration.HttpPort]/[CurrentEndpointConfiguration.ServicePath]".
 2. If `CurrentEndpointConfiguration.ServicePortType` is "HttpsPort", then form the URL as "https://[ServiceConfiguration.ServiceHostName]:[ServiceConfiguration.HttpsPort]/[CurrentEndpointConfiguration.ServicePath]".
 3. If `CurrentEndpointConfiguration.ServicePortType` is "HttpsPortForUserTlsAuth", then form the URL as "https://[ServiceConfiguration.ServiceHostName]:[ServiceConfiguration.HttpsPortForUserTlsAuth]/[CurrentEndpointConfiguration.ServicePath]".
 2. The client SHOULD add the headers in section 2.2.1 to the request.

3.11.5.1 End-user X509 Certificate Processing

If the client obtains a certificate of the end user then the client SHOULD validate the X509 certificate [RFC4158] based on the `CurrentEndpointConfiguration.CertificateValidation`.

If the `CurrentEndpointConfiguration.CertificateValidation` value is "None" then no validation SHOULD be performed.

If the `CurrentEndpointConfiguration.CertificateValidation` value is "Ssl" then the whole chain validation [RFC4158] of the certificate SHOULD be performed.

If the `CurrentEndpointConfiguration.CertificateValidation` value is "IssuedByDrs" then the client SHOULD validate that the end-user certificate was issued by one of `ServiceConfiguration.DeviceCertificateIssuers`.

Upon successful validation the client MUST construct a request as in section 3.10.5.1. The `[Serialized Request with Certificate].SerializedClientCertificate` MUST be set to the base64 string encoded ([RFC4648] section 4) X509 certificate [RFC4158].

If `CurrentEndpointConfiguration.CertificateValidation` value is "IssuedByDrs" then the `[Serialized Request with Certificate].CertificateUsage` MUST be set to "Device".

If `CurrentEndpointConfiguration.CertificateValidation` value is "Ssl" then the `[Serialized Request with Certificate].CertificateUsage` MUST be set to "User".

The [Serialized Request with Certificate].Request elements values SHOULD be copied from the incoming HTTP request.

The request SHOULD be made to `https://[ServiceConfiguration.ServiceHostName]:[ServiceConfiguration.HttpsPort]/adfs/backendproxys` and the client MUST authenticate with client TLS [RFC2246] using [Client State].TrustCertificate.

3.11.6 Timer Events

None.

3.11.7 Other Local Events

None.

3.12 Application Proxy Runtime Behaviors Server Details

3.12.1 Abstract Data Model

None.

3.12.2 Timers

None.

3.12.3 Initialization

None.

3.12.4 High-Layer Triggered Events

None.

3.12.5 Message Processing Events and Sequencing Rules

3.12.5.1 Issue Preauthentication

The server MUST implement the behaviors in this section if and only if the following is met for a particular incoming request:

1. The request contains the header X-MS-Proxy, as defined in section 2.2.1.1.
2. The [Server State].ProxyRelyingPartyTrust, as defined in section 3.1.1.1, that has the same URI {web-application-for-client-id} (using a case-insensitive comparison) as an object in the [Server State].RelyingPartyTrust array, as defined in section 2.2.2.5, has the enabled `isproperty` set to true.
3. The [Relying Party Trust] being preauthenticated exists and has the value of `publishedThroughProxy` set to true. Note that pre-authentication is different for each protocol; refer to subsequent sections for details.

3.12.5.1.1 Proxy Preauthentication

This operation processes a request for authentication and returns a proxy token as described in section 3.13.5.1 upon success.

The operation is transported by a HTTP **GET** and can be invoked through the following URI:

```
adfs/ls?version=1.0&action=signin&realm={web-application-for-client-id}&apprealm={web-application-id}&returnurl={client-url-to-issue-token}
```

The response message for this operation can result in the following status codes.

Status code	Description
200	The operation has succeeded.
403	The access is forbidden.
500	Internal error.

3.12.5.1.1.1 Request Body

The server **MUST** ignore any request body.

3.12.5.1.1.2 Response Body

No response body is returned.

3.12.5.1.1.3 Processing Details

The server **MUST** validate that {web-application-for-client-id} corresponds to the value of [Server State].ProxyRelyingPartyTrust.objectIdentifier. If validation fails, the server **MUST** return a HTTP error code of 500.

The server **MUST** validate that the request meets the conditions to issue pre-authentication (section 3.12.5.1) for the web application in [Server State].RelyingPartyTrusts with objectIdentifier equal to {web-application-id}.

The server **MUST** validate that the Relying Party Trust (section 2.2.2.6) proxyTrustedEndpoints contains a URL with a scheme, host and port that match those of {client-url-to-issue-token} and that prefix-matches the url-path of {client-url-to-issue-token} (for URL components see [RFC1738] sections 2.1 and 3.1). If validation fails, the server **MUST** return a HTTP error code of 500.

The server performs authentication of the request based on the **server's server's** authentication policy for [Server State].ProxyRelyingPartyTrust. If authentication fails the server **MUST** return a HTTP error code of 403.

If authentication succeeds the server **MUST** return a HTTP status code of 302 with a base64url encoded ([RFC4648] section 5) proxy token (section 3.13.5.1) in the URL query string parameter "authToken".

3.12.5.1.2 SAML-P Extensions for Preauthentication

The server **MUST** validate that the request meets the conditions to issue pre-authentication (section 3.12.5.1) for the web application in [Server State].RelyingPartyTrusts with identifiers containing a string value that matches the <Issuer> element value ([SAMLCore2] section 2.2.5) in the <AuthnRequest> element ([SAMLCore2]).

Upon successful authentication ([SAMLCore2] section 3.4.1.4) the server **MUST** do the following before sending the response to the response URL:

1. Transform the response URL based on the values of [Relying Party Trust].proxyEndpointMappings for the web application by replacing the response URL string portion that matches the Key value (internal URL mapping value) with the value of Value (external URL mapping value). If there is no match the response URL MUST not be changed.
2. If the request is an IdP initiated request the server MUST perform authentication of the request based on the ~~server's~~server's authentication policy for [Server State].ProxyRelyingPartyTrust. If authentication fails the server MUST respond according to [SAMLCore2] defined behavior for failed authentication.
3. If authentication succeeds the server MUST include in the response URL a query string parameter with name "authToken" with a value of a base64url encoded ([RFC4648] section 5) proxy token (section 3.13.5.1).

The server MUST send the response to the response URL.

3.12.5.1.3 WS-Fed Extensions for Preauthentication

If the server implements [WSFederation1.2] then the server MUST implement the following processing.

The server MUST validate that the request meets the conditions to issue pre-authentication (section 3.12.5.1) for the web application in [Server State].RelyingPartyTrusts with identifiers containing a string value that matches the wtrealm query string parameter value.

Upon successful authentication ([WSFederation1.2] section 13.1.1) the server MUST do the following before sending the response to the response URL:

1. Transform the response URL based on the values of [Relying Party Trust].proxyEndpointMappings for the web application by replacing the response URL string portion that matches the Key value (internal URL mapping value) with the value of Value (external URL mapping value). If there is no match the response URL MUST not be changed.
2. If pre-authentication has not happened yet<8> the server MUST perform authentication of the request based on the ~~server's~~server's authentication policy for [Server State].ProxyRelyingPartyTrust. If authentication fails the server MUST respond according to [WSFederation1.2] defined behavior for failed authentication.
3. If authentication succeeds the server MUST include in the response URL a query string parameter with name "authToken" with a value of a base64url encoded ([RFC4648] section 5) proxy token (section 3.13.5.1).

The server MUST send the response to the response URL.

3.12.5.1.4 OAuth Extensions for Preauthentication

If the server implements [the OAuth 2.0 Protocol Extensions](#) [MS-OAPX], then the server MUST implement the following behaviors.

The server MUST validate that the request meets the conditions to issue pre-authentication (section 3.12.5.1) for the web application in [Server State].RelyingPartyTrusts with identifiers containing a URI matching the "resource" query string parameter value.

Upon successful authentication [MS-OAPX], the server MUST do the following before sending the response.

The server performs authentication of the request based on the ~~server's~~server's authentication policy for [Server State].ProxyRelyingPartyTrust. If authentication fails the server MUST respond according to [MS-OAPX] defined behavior for failed authentication.

If authentication succeeds the server MUST generate a proxy token (section 3.13.5.1). The server MUST take the proxy token and combine it with the token targeted for the application in a [Combined Token] (section 2.2.2.18) and base64url encode ([RFC4648] section 5) the results. The server MUST use this [Combined Token] in all references to "token" in [MS-OAPX].

3.12.5.1.5 Proxy Preauthentication for Active Clients

~~Note: All of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.~~

This operation processes a request for authentication, and returns a proxy token as described in section 3.13.5.1 upon success.<9>

The operation is transported by a HTTP POST and can be invoked through the following URI:

```
adfs/proxy/relyingpartytoken?api-version=1
```

The response message for this operation can result in the following status codes.

Status code	Description
200	The operation has succeeded.
400	The request is not valid.
401	Unauthorized for client TLS certificate.
405	Invalid verb used in request (GET, DELETE, PUT).
500	Internal error.
501	The version specified (api-version) is invalid. The only valid value is 1.

~~For this operation, the server requires authenticating the proxy using client TLS authentication [RFC2246]. The server MUST validate that the certificate that is presented by the proxy during client TLS authentication can be validated by one of the values of [Server State].ProxyTrustedCertificates. If the certificate cannot be validated, the server MUST return an HTTP error code of 401.~~

3.12.5.1.5.1 Request Body

~~Note: All of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.~~

The request body MUST be an Authentication Request complex type (section 2.2.2.20).

3.12.5.1.5.2 Response Body

~~Note: All of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.~~

The response body MUST be a Proxy Token Wrapper complex type (section 2.2.2.19) if processing was successful. If processing was not successful and the status code is 400, 401, 500, or 501, the response body can be an Error Response complex type (section 2.2.2.21), but this is not required. The response body MUST be empty in all other cases.

3.12.5.1.5.3 Processing Details

~~Note: All of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.~~

The server MUST validate that {web-application-for-client-id} corresponds to the value of [Server State].ProxyRelyingPartyTrust.objectIdentifier. If validation fails, the server MUST return an HTTP error code of 500.

The server MUST validate that the request meets the conditions to issue pre-authentication (section 3.12.5.1) for the web application in [Server State].RelyingPartyTrusts with objectIdentifier equal to {web-application-id}.

The server performs authentication of the request based on the ~~server's~~server's authentication policy for [Server State].ProxyRelyingPartyTrust. If authentication fails, the server MUST return an HTTP error code of 403.

If authentication succeeds, the server MUST return an HTTP status code of 200 with a Proxy Token Wrapper complex type (section 2.2.2.19) in the response body.

3.12.6 Timer Events

None.

3.12.7 Other Local Events

None.

3.13 Application Proxy Runtime Behaviors Client Details

3.13.1 Abstract Data Model

None.

3.13.2 Timers

None.

3.13.3 Initialization

None.

3.13.4 High-Layer Triggered Events

None.

3.13.5 Message Processing Events and Sequencing Rules

On receiving any request the client needs to identify if the request is preauthenticated to either allow the access or initiate pre-authentication.

3.13.5.1 Preauthentication

A request is preauthenticated if it contains a [Proxy Token] (section 2.2.2.17) signed using JSON Web Signature (JWS) [IETFDRAFT-JWS] with the signing certificate published by the server through the Federation Metadata [WSFederation1.2].

Once a request has been identified as preauthenticated, the proxy MUST allow access by replaying the request to the corresponding internal address without the [Proxy Token].

Other claims might be present as name/value pairs depending on the issuance rules for the proxy configured at the server. It is left to the proxy implementer as to how to use these claims.

3.13.5.1.1 Query String Based Preauthentication

The request is preauthenticated if it contains a valid base64url encoded ([RFC4648] section 5) proxy token (section 3.13.5.1) from the server on the query string parameter "authToken". The token is validated according to section 3.13.5.1.

After successful pre-authentication the proxy MUST remove the authToken parameter with its value before replaying the request to the internal URL.

3.13.5.1.2 HTTP Authorization Header Based Preauthentication

If the request contains a HTTP Authorization header with a valid base64URL encoded ([RFC4648] section 5) [Combined Token] (section 2.2.2.18) then request can be preauthenticated by validating [Combined Token].proxy_token as in section 3.13.5.1.

The client MUST use [Combined Token].proxy_token to authorize the access to the web application.

After successful pre-authentication the client MUST replace the HTTP Authorization header value with a base64URL encoded ([RFC4648] section 5) value of [Combined Token].access_token before replaying the request to the internal URL.

3.13.5.2 Initiate Preauthentication

If the request does not contain a proxy token then the request is unauthenticated and the client MUST initiate pre-authentication.

If the client is servicing a request for the application identified by one of the entries in [Client State].RelyingPartyTrusts then the client MUST initiate pre-authentication as follows:

1. If [Relying Party Trust State].RedirectBasedPreauth is "true" then the client MUST follow processing rules in section 3.13.5.2.1.
2. If [Relying Party Trust State].RedirectBasedPreauth is "false" then the client MUST follow processing rules in section 3.13.5.2.2.

3.13.5.2.1 Initiate Redirect-based Preauthentication

Once a request to a web application has been identified as unauthenticated, the proxy MUST initiate pre-authentication by returning a HTTP 307 Temporary Redirect message to the client, redirecting the client to the following server end-point URL:

"https://" + [Client State].Configuration.ServiceConfiguration.ServiceHostName + ":" + [Client State].Configuration.ServiceConfiguration.HttpsPort + "/adfs/ls"

The redirect URL MUST have the following query string parameters.

Parameter	Value
version	Version of the protocol. It MUST be "1.0".
action	Action on authentication request. It MUST be "signin".
realm	Identifier for the Proxy Relying Party Trust. It MUST be [Client State].ProxyRelyingPartyTrustIdentifier (section 3.1.1.2).
apprealm	objectIdentifier of the application being accessed (section 2.2.2.6).
returnurl	URL of the incoming request.

3.13.5.2.2 Response to [MS-OFBA] Requests

Once a request to a web application has been identified as unauthenticated, the proxy MUST initiate pre-authentication. To do this the proxy MUST identify whether the request is from a Microsoft Office application that relies on the Office Forms Based Authentication (OFBA) Protocol [MS-OFBA].

To identify requests from Microsoft Office clients to application services relying on the OFBA protocol, the proxy MUST check if the request is an HTTP OPTIONS with a particular value on the User-Agent HTTP header or with a particular value on the X-Forms_Based_Auth_Accepted HTTP header (any of them):

Header	Value
User-Agent	Any of the following: "Microsoft Data Access Internet Publishing Provider" "Microsoft-WebDAV-MiniRedir" "non-browser" "MSOffice ##" where ## is an integer number "MSOffice XXXX ##" where XXXX is a value of "Word", "Excel", "PowerPoint" and "OneNote" and ## is an integer number "Mozilla/4.0 (compatible; MS FrontPage)" "Microsoft Office Protocol Discovery"
X-Forms_Based_Auth_Accepted	Any of the following: "t"

If the request is from a Microsoft Office client relying on the OFBA protocol, the server MUST return an HTTP error code of 403 to the client with the following headers:

Header	Value				
X-Forms_Based_Auth_Required	URL for the sign-in request: <table border="1" data-bbox="592 1690 1430 1791"> <thead> <tr> <th>Parameter</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>version</td> <td>Version of the protocol. It MUST be "1.0".</td> </tr> </tbody> </table>	Parameter	Value	version	Version of the protocol. It MUST be "1.0".
Parameter	Value				
version	Version of the protocol. It MUST be "1.0".				

Header	Value								
	<table border="1"> <tr> <td>action</td> <td>Action on authentication request. It MUST be "signin".</td> </tr> <tr> <td>realm</td> <td>Identifier for the Proxy Relying Party Trust. It MUST be [Client State].ProxyRelyingPartyTrustIdentifier (section 3.1.1.2).</td> </tr> <tr> <td>apprealm</td> <td>objectIdentifier of the application being accessed (section 2.2.2.6).</td> </tr> <tr> <td>returnurl</td> <td>URL of the incoming request.</td> </tr> </table>	action	Action on authentication request. It MUST be "signin".	realm	Identifier for the Proxy Relying Party Trust. It MUST be [Client State].ProxyRelyingPartyTrustIdentifier (section 3.1.1.2).	apprealm	objectIdentifier of the application being accessed (section 2.2.2.6).	returnurl	URL of the incoming request.
action	Action on authentication request. It MUST be "signin".								
realm	Identifier for the Proxy Relying Party Trust. It MUST be [Client State].ProxyRelyingPartyTrustIdentifier (section 3.1.1.2).								
apprealm	objectIdentifier of the application being accessed (section 2.2.2.6).								
returnurl	URL of the incoming request.								
X-Forms_Based_Auth_Return_Url	URL of incoming request.								

For requests from non-Microsoft-Office clients accessing services that implement the OFBA protocol [MS-OFBA] that rely on AD FS for authentication, the proxy MUST return an HTTP error code of 401 Unauthorized with the following header.

Header	Value
WWW-Authenticate	"Bearer authorization_uri=https://" + [Client State].Configuration.ServiceConfiguration.ServiceHostName + ":" + [Client State].Configuration.ServiceConfiguration.HttpsPort + "/adfs/oauth2/authorize"

3.13.5.2.3 Response to Active Requests

~~Note: All of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.~~

The proxy MAY choose to preauthenticate requests by making backend requests to the server as specified in section 3.12.5.1.5, provided the proxy deems that the request contains the credentials it needs to be preauthenticated.<10>

The proxy MUST perform client TLS authentication [RFC2246] using the certificate in [Client State].TrustCertificate.

- If the request contains Username and Password in the Authorization header as specified in [RFC2617], they are used as **username** and **password** in the Authentication Request (section 2.2.2.20).
- If the request was made using SSL mutual authentication [RFC6101], the client certificate SHOULD be identified by the proxy as whether it is the proof of the device or the proof of the user.
 - If the client certificate is the proof of the user, it is used as the **userCertificate** in the Authentication Request (section 2.2.2.20).
 - If the client certificate is the proof of the device, it is used as the **deviceCertificate** in the Authentication Request (section 2.2.2.20).
- Any HTTP headers from the incoming request are passed on to the server as the **httpHeaders** in the Authentication Request (section 2.2.2.20).

If the pre-authentication request resulted in an error, the proxy MUST send HTTP 401 to the client.

If the pre-authentication request returned a valid response as specified in section 3.12.5.1.5.2, the value of **authToken** in the Proxy Token Wrapper (section 2.2.2.19) is used for pre-authentication according to the rules specified in section 3.13.5.1.

The proxy MAY allow the "Authorization" header from the incoming HTTP request [RFC2617], to propagate to the backend application.

3.13.6 Timer Events

None.

3.13.7 Other Local Events

None.


```
[{"enabled":true,"name":"Device Registration Service","nonClaimsAware":false,"objectIdentifier":"4646dd08-49eb-e211-9867-00155d6ff01e","publishedThroughProxy":false}, {"enabled":true,"name":"fedpassive","nonClaimsAware":false,"objectIdentifier":"011ab67d-49eb-e211-9867-00155d6ff01e","publishedThroughProxy":false}, {"enabled":true,"name":"integratedWindowsRp","nonClaimsAware":true,"objectIdentifier":"071ab67d-49eb-e211-9867-00155d6ff01e","publishedThroughProxy":true}]
```

4.3 Create a New Set of Published Settings on a Relying Party Trust

4.3.1 Client Request

```
POST https://sts1.contoso.com/adfs/proxy/relyingpartytrusts/7aeee25c-4beb-e211-9867-00155d6ff01e/fedpassive/publishedsettings?api-version=1 HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: sts1.contoso.com
Content-Length: 264
Expect: 100-continue

{"internalUrl":"https://urlInternal","externalUrl":"https://urlExternal","proxyTrustedEndpoint":"https://urlExternal"}
```

4.3.2 Server Response

```
HTTP/1.1 200 OK
Content-Length: 0
```

4.4 Remove an Existing Set of Published Settings on a Relying Party Trust

4.4.1 Client Request

```
DELETE https://sts1.contoso.com/adfs/proxy/relyingpartytrusts/0b153cca-4beb-e211-9867-00155d6ff01e/fedpassive/publishedsettings?api-version=1 HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: sts1.contoso.com
Content-Length: 155
Expect: 100-continue

{"externalUrl":"https://urlExternal","proxyTrustedEndpoint":"https://urlExternal"}
```

4.4.2 Server Response

```
HTTP/1.1 200 OK
Content-Length: 0
```

4.5 Add a Key Value Pair to the Store

4.5.1 Client Request

```
POST
https://sts1.contoso.com/adfs/proxy/webapplicationproxy/store/DLOWTTYDQMB2NAPRXFITNYKZXSVW8D7J0KCQEH0EA?api-version=1 HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: sts1.contoso.com
Content-Length: 33
```

```
Expect: 100-continue  
  
{"value":"SOMEVALUE_THAT_I_HAVE"}
```

4.5.2 Server Response

```
HTTP/1.1 200 OK  
Content-Length: 0
```

4.6 Retrieve a Value of a Key from the Store

4.6.1 Client Request

```
GET https://sts1.contoso.com/adfs/proxy/webapplicationproxy/store/MY_KEY?api-version=1  
HTTP/1.1  
Host: sts1.contoso.com
```

4.6.2 Server Response

```
HTTP/1.1 200 OK  
Cache-Control: no-store  
Pragma: no-cache  
Content-Length: 60  
Content-Type: application/json;charset=UTF-8  
  
{"key":"MY_KEY","version":0,"value":"SOMEVALUE_THAT_I_HAVE"}
```

4.7 Update the Value of a Key Already in the Store

4.7.1 Client Request

```
PUT https://sts1.contoso.com/adfs/proxy/webapplicationproxy/store/MY_KEY?api-version=1  
HTTP/1.1  
Content-Type: application/json;charset=UTF-8  
Host: sts1.contoso.com  
Content-Length: 44  
Expect: 100-continue  
  
{"value":"ANOTHER VALUE___ NEW","version":0}
```

4.7.2 Server Response

```
HTTP/1.1 200 OK  
Cache-Control: no-store  
Pragma: no-cache  
Content-Length: 28  
Content-Type: application/json;charset=UTF-8  
  
{"key":"MY_KEY","version":1}
```

4.8 Create a new Proxy Relying Party Trust

4.8.1 Client Request

```
POST https://sts1.contoso.com/adfs/proxy/webapplicationproxy/trust?api-version=1 HTTP/1.1
Content-Type: application/json;charset=UTF-8
Host: sts1.contoso.com
Content-Length: 35
Expect: 100-continue

{"Identifier":"https://\appProxy"}
```

4.8.2 Server Response

```
HTTP/1.1 200 OK
Content-Length: 0
```

4.9 Get the Proxy Relying Party Trust

4.9.1 Client Request

```
GET https://sts1.contoso.com/adfs/proxy/webapplicationproxy/trust?api-version=1 HTTP/1.1
Host: sts1.contoso.com
```

4.9.2 Server Response

```
HTTP/1.1 200 OK
Cache-Control: no-store
Pragma: no-cache
Content-Length: 35
Content-Type: application/json;charset=UTF-8

{"Identifier":"https://\appProxy"}
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Full JSON Schema

~~Note: Some of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview in the Product Behavior appendix.~~

```
{
  "title" : "Proxy Trust",
  "type" : "object",
  "properties" :
  {
    "SerializedTrustCertificate" : {"type" : "string"}
  }
}

{
  "title" : "Proxy Trust Renewal",
  "type" : "object",
  "properties" :
  {
    "SerializedReplacementCertificate" : {"type" : "string"}
  }
}

{
  "title" : "Proxy Relying Party Trust",
  "type" : "object",
  "properties" :
  {
    "Identifier" : {"type" : "string"}
  }
}

{
  "title" : "Configuration",
  "type" : "object",
  "properties" :
  {
    "ServiceConfiguration" :
    {
      "type" : "object",
      "properties" :
      {
        "ServiceHostName" : {"type" : "string"},
        "HttpPort" : {"type" : "integer"},
        "HttpsPort" : {"type" : "integer"},
        "HttpsPortForUserTlsAuth" : {"type" : "integer"},
        "DeviceCertificateIssuers" :
        {
          "type" : "array",
          "items" : {"type" : "string"}
        },
        "ProxyTrustCertificateLifetime" : {"type" : "integer"},
        "DiscoveredUpnSuffixes" :
        {
          "type" : "array",
          "items" : {"type" : "string"}
        },
        "CustomUpnSuffixes" :
        {
          "type" : "array",
          "items" : {"type" : "string"}
        },
        "ServiceHostNameForUserTlsAuth" : {"type" : "string"}
      }
    }
  }
}
```

```

    },
    "EndpointConfiguration" :
    {
      "type" : "array",
      "items" :
      {
        "type" : "object",
        "properties" :
        {
          "Path" : {"type" : "string"},
          "PortType" :
          {
            "enum" : ["HttpPort", "HttpsPort", "HttpsPortForUserTlsAuth"]
          },
          "AuthenticationSchemes" :
          {
            "enum" : [8, 32768]
          },
          "ClientCertificateQueryMode" :
          {
            "enum" : ["None", "QueryAndAccept", "QueryAndRequire"]
          },
          "CertificateValidation" :
          {
            "enum" : ["None", "User", "DeviceSsl", "IssuedByDrs"]
          },
          "ServicePath" : {"type" : "string"},
          "ServicePortType" :
          {
            "enum" : ["HttpPort", "HttpsPort", "HttpsPortForUserTlsAuth"]
          }
        }
      }
    }
  ],
  "FarmBehavior" : {"type" : "string"},
  "IgnoreTokenBinding" : {"type" : "boolean"}
}

{
  "title" : "Relying Party Trust List",
  "type" : "object",
  "properties" :
  {
    "relyingPartyTrustListArray" :
    {
      "type" : "array",
      "items" :
      {
        "type" : "object",
        "properties" :
        {
          "objectIdentifier" : {"type" : "string"},
          "name" : {"type" : "string"},
          "publishedThroughProxy" : {"type" : "boolean"},
          "nonClaimsAware" : {"type" : "boolean"},
          "enabled" : {"type" : "boolean"}
        }
      }
    }
  }
}

{
  "title" : "Relying Party Trust",
  "type" : "object",
  "properties" :
  {
    "objectIdentifier" : {"type" : "string"},

```

```

"name" : {"type" : "string"},
"publishedThroughProxy" : {"type" : "boolean"},
"nonClaimsAware" : {"type" : "boolean"},
"enabled" : {"type" : "boolean"},
"identifiers" :
{
  "type" : "array",
  "items" : {"type" : "string"}
},
"proxyTrustedEndpoints" :
{
  "type" : "array",
  "items" : {"type" : "string"}
},
"proxyEndpointMappings" :
{
  "type" : "array",
  "items" :
  {
    "type" : "object",
    "properties" :
    {
      "Key" : {"type" : "string"},
      "Value" : {"type" : "string"}
    }
  }
}
}
}

{
  "title" : "Relying Party Trust Publishing Settings",
  "type" : "object",
  "properties" :
  {
    "externalUrl" : {"type" : "string"},
    "internalUrl" : {"type" : "string"},
    "proxyTrustedEndpointUrl" : {"type" : "string"}
  }
}

{
  "title" : "Store Entry List",
  "type" : "object",
  "properties" :
  {
    "storeEntryListArray" :
    {
      "type" : "array",
      "items" : {"type" : "Store Entry"}
    }
  }
}

{
  "title" : "Store Entry",
  "type" : "object",
  "properties" :
  {
    "key" : {"type" : "string"},
    "version" : {"type" : "integer"},
    "value" : {"type" : "string"}
  }
}

{
  "title" : "Store Entry Key and Value",
  "type" : "object",
  "properties" :

```

```

    {
      "key" : {"type" : "string"},
      "value" : {"type" : "string"}
    }
  }

  {
    "title" : "Serialized Request with Certificate",
    "type" : "object",
    "properties" :
    {
      "Request" :
      {
        "type" : "object",
        "properties" :
        {
          "AcceptTypes" : {"type" : "string"},
          "Content" : [ <byte>, * ],
          "ContentEncoding" : {"type" : "string"},
          "ContentLength" : {"type" : "integer"},
          "ContentType" : {"type" : "string"},
          "Cookies" :
          {
            "type" : "object",
            "properties" :
            {
              "Name" : {"type" : "string"},
              "Value" : {"type" : "string"},
              "Path" : {"type" : "string"},
              "Domain" : {"type" : "string"},
              "Expires" : {"type" : "integer"},
              "Version" : {"type" : "integer"}
            }
          },
          "Headers" :
          {
            "type" : "array",
            "items" :
            {
              "type" : "object",
              "properties" :
              {
                "Name" : {"type" : "string"},
                "Value" : {"type" : "string"}
              }
            }
          },
          "HttpMethod" : {"type" : "string"},
          "RequestUri" : {"type" : "string"},
          "QueryString" : {"type" : "string"},
          "UserAgent" : {"type" : "string"},
          "UserHostAddress" : {"type" : "string"},
          "UserHostName" : {"type" : "string"},
          "UserLanguages" : {"type" : "string"}
        }
      },
      "SerializedClientCertificate" : {"type" : "string"},
      "CertificateUsage" :
      {
        "enum" : ["User", "Device"]
      }
    }
  }

  {
    "title" : "Proxy Token",
    "type" : "object",
    "properties" :
    {
      "ver" : {"type" : "number"},

```

```

    "aud" : {"type" : "string"},
    "iat" : {"type" : "integer"},
    "exp" : {"type" : "integer"},
    "iss" : {"type" : "string"},
    "relyingpartytrustid" : {"type" : "string"},
    "deviceregid" : {"type" : "string"},
    "authinstant" : {"type" : "integer"},
    "authmethod" : {"type" : "string"},
    "upn" : {"type" : "string"}
  }
}

{
  "title" : "Combined Token",
  "type" : "object",
  "properties" :
  {
    "proxy_token" : {"type" : "Proxy Token"},
    "access_token" : {"type" : "string"}
  }
}

{
  "title" : "Proxy Token Wrapper",
  "type" : "object",
  "properties" :
  {
    "authToken" : {"type" : "Proxy Token"}
  }
}

{
  "title" : "Authentication Request",
  "type" : "object",
  "properties" :
  {
    "appRealm" : {"type" : " string"},
    "realm" : {"type" : " string"},
    "username" : {"type" : "string"},
    "password" : {"type" : "string"},
    "deviceCertificate" : {"type" : "string"},
    "userCertificate" : {"type" : "string"},

    "httpHeaders" :
    {
      "type" : "object",
      "properties" :
      {
        "Key" : {"type" : "string"},
        "Value" : {"type" : "string"}
      }
    }
  }
}

{
  "title" : "Error Response",
  "type" : "object",
  "properties" :
  {
    "id" : {"type" : "integer"},
    "message" : {"type" : "string"},
    "type" : {"type" : "string"}
  }
}

```


7 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs.

~~Note: Some of the information in this section is subject to change because it applies to an unreleased, preliminary version of the Windows Server operating system, and thus may differ from the final version of the server software when released. All behavior notes that pertain to the unreleased, preliminary version of the Windows Server operating system contain specific references to Windows Server 2016 Technical Preview as an aid to the reader.~~

- Windows Server 2012 R2 operating system
- Windows Server 2016 ~~Technical Preview~~ operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

<1> Section 2.2.1.5: The X-MS-ADFS-Proxy-Client-IP header is not sent by the Web Application Proxy on Windows Server 2012 R2. ~~It is currently implemented by the unreleased preliminary version of Windows Server 2016 Technical Preview.~~

~~<2<2> Section 2.2.2.4: The service-host-name-for-user-tls-auth field is not supported by the Web Application Proxy on Windows Server 2012 R2.~~

~~<3> Section 2.2.2.4: The farm-behavior-version-number field is not supported by the Web Application Proxy on Windows Server 2012 R2.~~

~~<4> Section 2.2.2.4: The ignore-token-binding field is not supported by the Web Application Proxy on Windows Server 2012 R2.~~

<5> Section 3.1.1.1: Any writes to [Server State] require, by default, 5 minutes to propagate to other nodes in the server in an AD FS farm configuration using WID.

<6> Section 3.3.5.2.1.3: Windows does not remove the old certificate from [Server State].

~~<4<7> Section 3.4.5: The following table shows the values of api-version that can be set by the Web Application Proxy in each operating system.~~

<u>Operating System</u>	<u>api-version values supported</u>
<u>Windows Server 2012 R2</u>	<u>1</u>
<u>Windows Server 2016</u>	<u>2</u>

| ~~<8>~~ Section 3.12.5.1.3: Windows validates that the sign-in request comes from a SAML-P IdP initiated request with a query string parameter RelayState containing an identifier of a web application in the server that relies on the WS-Fed protocol for authentication.

| <9> Section 3.12.5.1.5: Preauthentication for active clients is not supported on Windows Server 2012 R2. ~~It is currently implemented by the unreleased preliminary version of Windows Server 2016 Technical Preview.~~

| <10> Section 3.13.5.2.3: Preauthentication of active requests is not supported on Windows Server 2012 R2. ~~It is currently implemented by the unreleased preliminary version of Windows Server 2016 Technical Preview.~~

8 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements or functionality.
- The removal of a document from the documentation set.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

Major and minor changes can be described further using the following change types:

- New content added.
- Content updated.
- Content removed.
- New product behavior note added.
- Product behavior note updated.
- Product behavior note removed.
- New protocol syntax added.
- Protocol syntax updated.
- Protocol syntax removed.
- New content added due to protocol revision.
- Content updated due to protocol revision.
- Content removed due to protocol revision.
- New protocol syntax added due to protocol revision.
- Protocol syntax updated due to protocol revision.
- Protocol syntax removed due to protocol revision.
- Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated**.

Some important terms used in the change type descriptions are defined as follows:

- **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
- **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
1.2.1 Normative References	Added Updated content for this version of Windows Server 2016 Technical Preview .	Y	Content update.
1.2.1 Normative References	70865 : Added reference to [RFC4559].	Y	Content update.
2.2.1 HTTP Headers2 Messages	Updated content for Windows Server 2016 Technical Preview operating system.5374 : Revised quotation marks around values in the JSON definitions for complex types throughout this section to accurately represent data types.	Y N	Content update.
2.2.1.5 X-MS-ADFS-Proxy-Client-IP	Added section with content for Windows Server 2016 Technical Preview operating system.	Y	New content added.
2.2.2.4 Configuration	70865 : Updated object elements (added upn suffix and support-ntlm).5375 : Included information for service-host-name-for-user-tls-auth, farm-behavior-version-number, and ignore-token-binding.	Y	Content update.
2.2.2.11 Serialized Request with Certificate	71135 : Updated73195 : Added details to clarify the definitionpurpose of the Serialized-Request with Certificateserialized request contained by this object to accurately reflect protocol behavior.	Y N	Content update.
2.2.2.17 Proxy Token15 Certificate Validation	708735379 : Updated definitionvalues for audience; revised [Proxy Service State] to [Client State];this enumeration.	Y	Protocol syntax updated. Content update.
2.2.2.19 Proxy Token Wrapper	Added section with content for Windows Server 2016 Technical Preview.	Y	New content added.
2.2.2.20 Authentication Request	Added section with content for Windows Server 2016 Technical Preview.	Y	New content added.
2.2.2.21 Error Response	Added section with content for Windows Server 2016 Technical Preview.	Y	New content added.
3.1.1.1 Server State	70873 : Revised snippet to read ProxyTrustedCertificates from ProxyTrustedCertificates.	N	Protocol syntax updated.
3.1.1.3 Relying Party Trust State	70873 : Removed unrelated content — pre-auth-required.	N	Content removed.
3.1.1.3 Relying Party Trust State	71136 : Updated section reference — Relying Party Trust.5374 : Removed quotation marks around the redirect-based-preauth value in the JSON definition for	Y N	Content update.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
	<u>this structure to accurately represent its Boolean data type.</u>		
<u>3.23.5.1.1.3 Message Processing DetailsEvents and Sequencing Rules</u>	70873 : Corrected erroneous ADM element: revised [Server State] from [5381 : Changed [Proxy Service State Data].TrustCertificate to [Client State].TrustCertificate.	Y	Content updated due to protocol revisionupdate.
<u>3.2.5.2.1 POST3.4.5 Message Processing Events and Sequencing Rules</u>	70873 : Corrected erroneous ADM element: revised [Server State].ProxyTrustedCertificates from [Service State Data].ProxyTrustCertificates-5381 : Added content that applies to all operations in this section.	Y	Content updated due to protocol revisionupdate.
<u>3.24.5.2.1.3 Message Processing DetailsEvents and Sequencing Rules</u>	70873 : Corrected erroneous ADM element: revised [Server State].ProxyTrustedCertificates from [Service State Data].ProxyTrustCertificates-Updated content for this version of Windows Server.	Y	Content updated due to protocol revisionupdate.
<u>3.24.5.31.1 GET</u>	70873 : Corrected erroneous ADM element: revised [Server State].ProxyTrustedCertificates from [Service State Data].ProxyTrustCertificates-5381 : Removed content because it is specified in the parent section.	Y	Content updated due to protocol revisionupdate.
<u>3.24.5.3.1.3 Processing Details1_GET</u>	70873 : Corrected erroneous ADM element: revised [Server State] from [Service State Data].Updated content for this version of Windows Server.	Y	Content updated due to protocol revisionupdate.
<u>3.24.5.3.21.1.3 Processing Details</u>	70873 : Corrected erroneous ADM element: revised [Server State] from [Service State Data].Updated content for this version of Windows Server.	Y	Content updated due to protocol revisionupdate.
<u>3.4.5.2.5.3.3.3 Processing Details1_GET</u>	70873 : Corrected erroneous ADM element: revised [Server State] from [Service State Data]-5381 : Removed content because it is specified in the parent section.	Y	Content updated due to protocol revisionupdate.
<u>3.4.5.3.5.1.1.3 Processing Details GET</u>	70873 : Corrected erroneous ADM element: revised [Client State] from [Proxy Service State]-5381 : Removed content because it is specified in the parent section.	Y	Content updated due to protocol revisionupdate.
<u>3.3.5.2.1.35 Message Processing DetailsEvents and Sequencing Rules</u>	70873 : Corrected erroneous ADM element: revised [Client State] from 5381 : Changed [Proxy Service State Data].TrustCertificate to [Client State].TrustCertificate.	Y	Content updated due to protocol revisionupdate.
<u>3.36.5.3.2.3 Message Processing DetailsEvents and Sequencing Rules</u>	70873 : Corrected erroneous ADM element: revised [Client State] from [Proxy Service State]-5381 : Updated content about client TLS authentication to indicate that it applies to all operations in this section.	Y	Content updated due to protocol revisionupdate.
<u>3.3.5.3.3.3 Processing Details</u>	70873 : Corrected erroneous ADM element: revised [Client State] from [Proxy Service State].	Y	Content updated due to protocol revision.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
3.4.5.1.1 GET	70873 : Corrected erroneous ADM element: revised [Server State].ProxyTrustedCertificates from [Service State Data].ProxyTrustCertificates.	Y	Content updated due to protocol revision.
3.4.5.1.1.3 Processing Details	70873 : Corrected erroneous ADM element: revised [Server State] from [Service State Data].	Y	Content updated due to protocol revision.
3.46.5.2.1-GET4 DELETE	70873 : Corrected erroneous ADM element: revised [Server State].ProxyTrustedCertificates from [Service State Data].ProxyTrustCertificates.73173 : Revised the description to call out the correct operation and behavior.	Y	Content updated due to protocol revisionupdate.
3.46.5.2.1-3 Processing Details4.2 Response Body	70873 : Corrected erroneous ADM element: revised [Server State] from [Service State Data].73173 : Clarified description.	Y N	Content updated due to protocol revisionupdate.
3.4.5.3.1 GET	70873 : Corrected erroneous ADM element: revised [Server State].ProxyTrustedCertificates from [Service State Data].ProxyTrustCertificates.	Y	Content updated due to protocol revision.
3.4.5.3.1.3 Processing Details	70873 : Corrected erroneous ADM element: revised [Server State] from [Service State Data].	Y	Content updated due to protocol revision.
3.5.5.1.1.3 Processing Details	70873 : Corrected erroneous ADM element: revised [Client State] from [Proxy Service State].	Y	Content updated due to protocol revision.
3.67.5 Message Processing Events and Sequencing Rules	70873 : Corrected erroneous ADM element: revised [Server State].ProxyTrustedCertificates from [Service State Data].ProxyTrustCertificates.5381 : Added content that applies to all operations in this section.	Y	Content updated due to protocol revisionupdate.
3.68.5.1.1-3 Message Processing Details Events and Sequencing Rules	70873 : Added content.5378 : Changed "PublishingSettings" to "PublishedSettings" in the resource URI listed.	Y	Content update.
3.8.5 Message Processing Events and Sequencing Rules	70873 : Corrected erroneous ADM element: revised [Server State].ProxyTrustedCertificates from [Service State Data].ProxyTrustCertificates.5381 : Updated content about client TLS authentication to indicate that it applies to all operations in this section.	Y	Content updated due to protocol revisionupdate.
3.8.5.1.1-3 Processing Details POST	70873 : Corrected erroneous ADM element: revised [Server State] from [Service State Data].5378 : Changed "PublishingSettings" to "PublishedSettings" where it appears in URIs.	Y	Content updated due to protocol revisionupdate.
3.8.5.1.2-3 Processing Details DELETE	70873 : Corrected erroneous ADM element: revised [Server State] from [Service State Data].5378 : Changed "PublishingSettings" to "PublishedSettings" where it appears in URIs.	Y	Content updated due to protocol revisionupdate.
3.109.5 Message Processing Events	70873 : Corrected erroneous ADM element: revised [Server State].ProxyTrustedCertificates from [Service	Y	Content updated due to protocol

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
and Sequencing Rules	State Data].ProxyTrustCertificates.5381 : Updated to indicate that the client uses the certificate in [Client State].TrustCertificate.		revision update.
3.1110.5 Message Processing Events and Sequencing Rules	70865 : Updated message processing rules.5381 : Updated content about client TLS authentication to indicate that it applies to all operations in this section.	Y	Content update.
3.12.5.1.5 Proxy Preauthentication for Active Clients	Added section with content for Windows Server 2016 Technical Preview.	Y	New content added.
3.12.5.1.5.1 Request Body	Added section with content for Windows Server 2016 Technical Preview.	Y	New content added.
3.12.5.1.5.2 Response Body	Added section with content for Windows Server 2016 Technical Preview.	Y	New content added.
3.1211.5.1.5.3 Message Processing DetailsEvents and Sequencing Rules	Added section with Updated content for this version of Windows Server 2016 Technical Preview.	Y	New content added. Content update.
3.1312.5.2-1 Initiate Redirect-basedIssue Preauthentication	70873 : Corrected snippet: revised [Client State] from [Proxy Service Client].5380 : Updated processing rule regarding [Server State].ProxyRelyingPartyTrust.	Y	Content updated due to protocol revision update.
3.1312.5.2-1 Initiate Redirect-based.5 Proxy Preauthentication for Active Clients	71136 : Updated the definition of apprealm5381 : Added content that applies to the operation in this section.	Y	Content update.
3.13.5.2.2 Response to [MS-OFBA] Requests	71136 : Updated the definition of apprealm.	Y	Content update.
3.13.5.2.3 Response to Active Requests	Added section with content for Windows Server 2016 Technical Preview.5381 : Added content that indicates that the proxy performs client TLS authentication using the certificate in [Client State].TrustCertificate.	Y	New content added.Content update.
6 Appendix A: Full JSON Schema	5379 : Updated values for the <u>CurrentEndpointConfiguration.CertificateValidation</u> enumeration.	Y	Content update.
6 Appendix A: Full JSON Schema	5374 : Added definitions for the "Content" property to <u>DiscoveredUpnSuffixes</u> and <u>CustomUpnSuffixes</u> properties of the "Serialized-Request with Certificate" <u>ServiceConfiguration</u> object.	Y	Content update.
6 Appendix A: Full JSON Schema	Updated content for <u>this version of Windows Server 2016 Technical Preview operating system.</u>	Y	Content update.
7 Appendix B: Product Behavior	Removed the Client product from the applicability list.	Y	Content update.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
7 Appendix B: Product Behavior	Updated the product behavior notes to include Windows Server 2016 Technical Preview.	Y	Content update.

9 Index

A

- Applicability 15
- Application proxy runtime behaviors client
 - Abstract data model 63
 - Initialization 63
 - Message processing events and sequencing rules 64
 - Other local events 67
 - Timer events 67
 - Timers 63
- Application proxy runtime behaviors server
 - Abstract data model 59
 - Initialization 59
 - Other local events 63
 - Timer events 63
 - Timers 59
- Application publishing client
 - Abstract data model 54
 - Initialization 54
 - Message processing events and sequencing rules 54
 - Other local events 55
 - Timer events 55
 - Timers 54
- Application publishing server
 - Abstract data model 50
 - Initialization 50
 - Message processing events and sequencing rules 50
 - Other local events 53
 - Timer events 53
 - Timers 50

C

- Capability negotiation 15
- Change tracking 81
- Common
 - Abstract data model 27
 - Higher-layer triggered events 28
 - Initialization 28
 - Message processing events and sequencing rules 28
 - Other local events 28
 - Timer events 28
 - Timers 28
- Common data types 16

F

- Fields - vendor-extensible 15
- Full JSON schema 73

G

- Glossary 11

I

- Implementer - security considerations 72
- Index of security parameters 72
- Informative references 13
- Introduction 11

J

JSON schema 73

M

Messages
 transport 16

N

Normative references 12

O

Overview (synopsis) 13

P

Parameters - security index 72

Preconditions 15

Prerequisites 15

Product behavior 79

Proxy configuration client

 Abstract data model 48

 Initialization 48

 Message processing events and sequencing rules 48

 Other local events 50

 Timer events 50

 Timers 48

Proxy configuration server

 Abstract data model 42

 Initialization 42

 Message processing events and sequencing rules 43

 Other local events 48

 Timer events 48

 Timers 42

Proxy registration client

 Abstract data model 34

 Higher-layer triggered events 34

 Initialization 34

 Message processing events and sequencing rules 34

 Other local events 36

 Timer events 36

 Timers 34

Proxy registration server

 Abstract data model 28

 Higher-layer triggered events 29

 Initialization 29

 Message processing events and sequencing rules 29

 Other local events 34

 Timer events 34

 Timers 29

Proxy runtime behaviors client

 Abstract data model 57

 Initialization 57

 Message processing events and sequencing rules 57

 Other local events 59

 Timer events 59

 Timers 57

Proxy runtime behaviors server

 Abstract data model 55

 Initialization 55

 Message processing events and sequencing rules 55

- Other local events 57
- Timer events 57
- Timers 55

R

- References
 - informative 13
 - normative 12
- Relationship to other protocols 14

S

- Security
 - implementer considerations 72
 - parameter index 72
- Service configuration client
 - Abstract data model 41
 - Initialization 41
 - Message processing events and sequencing rules 41
 - Other local events 42
 - Timer events 42
 - Timers 41
- Service configuration server
 - Abstract data model 37
 - Initialization 37
 - Message processing events and sequencing rules 37
 - Other local events 40
 - Timer events 40
 - Timers 37
- Standards assignments 15

T

- Tracking changes 81
- Transport 16
 - common data types 16

V

- Vendor-extensible fields 15
- Versioning 15