[MS-ADFSPP]:
Active Directory Federation Service (AD FS) Proxy Protocol

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **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 make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft Open Specification Promise or the Community Promise. If you would prefer a written license, or if the technologies described in the Open Specifications 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 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, email addresses, logos, people, places, and events 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 specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do 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 are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
10/24/2008	0.01		Initial Availability
12/05/2008	0.2	Minor	Updated the technical content.
01/16/2009	0.2.1	Editorial	Revised and edited the technical content.
02/27/2009	0.2.2	Editorial	Revised and edited the technical content.
04/10/2009	0.2.3	Editorial	Revised and edited the technical content.
05/22/2009	0.2.4	Editorial	Revised and edited the technical content.
07/02/2009	0.2.5	Editorial	Revised and edited the technical content.
08/14/2009	0.2.6	Editorial	Revised and edited the technical content.
09/25/2009	0.3	Minor	Updated the technical content.
11/06/2009	0.3.1	Editorial	Revised and edited the technical content.
12/18/2009	0.3.2	Editorial	Revised and edited the technical content.
01/29/2010	0.4	Minor	Updated the technical content.
03/12/2010	0.4.1	Editorial	Revised and edited the technical content.
04/23/2010	0.4.2	Editorial	Revised and edited the technical content.
06/04/2010	0.4.3	Editorial	Revised and edited the technical content.
07/16/2010	0.4.3	No change	No changes to the meaning, language, or formatting of the technical content.
08/27/2010	1.0	Major	Significantly changed the technical content.
10/08/2010	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
11/19/2010	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
01/07/2011	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
02/11/2011	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
03/25/2011	1.0	No change	No changes to the meaning, language, or formatting of the technical content.
05/06/2011	1.0	No change	No changes to the meaning, language, or formatting of the technical content.

Date	Revision History	Revision Class	Comments
06/17/2011	1.1	Minor	Clarified the meaning of the technical content.
09/23/2011	1.1	No change	No changes to the meaning, language, or formatting of the technical content.
12/16/2011	2.0	Major	Significantly changed the technical content.
03/30/2012	3.0	Major	Significantly changed the technical content.
07/12/2012	3.1	Minor	Clarified the meaning of the technical content.
10/25/2012	3.1	No change	No changes to the meaning, language, or formatting of the technical content.
01/31/2013	3.1	No change	No changes to the meaning, language, or formatting of the technical content.
08/08/2013	4.0	Major	Significantly changed the technical content.

Contents

1 Introduction

The Active Directory Federation Services (AD FS) Proxy Protocol is used by a **security token service (STS)** proxy to obtain configuration data about an STS in order to assist users in selecting an acceptable **security realm** from which to obtain a **security token**. The protocol is also used by an STS to relay Microsoft Web Browser Federated Sign-On Protocol [MS-MWBF] requests back to an STS.

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in RFC 2119. Sections 1.5 and 1.9 are also normative but cannot contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are defined in [MS-GLOS]:

Active Directory

The following terms are specific to this document:

Claim: A declaration made by an entity (for example, name, identity, key, group, privilege, and capability). For more information, see [WSFedPRP] sections 1.4 and 2.

Relying Party: A web application or service that consumes **security token** issued by an **STS**.

Security Realm: Represents a single unit of security administration or trust (for example, a Kerberos realm, as defined in [RFC4120], or a Windows domain, as defined in [MSFT-ADC]).

Security Token: Represents a collection of one or more **claims**.

Security Token Service (STS): A web service that issues **security tokens**. That is, it makes assertions based on evidence that it trusts for consumption by whoever trusts it. For more information, see [WSFedPRP] sections 1.4 and 2. For this protocol, **STS** refers to services that support (either directly or via a front-end) the HTTP protocol defined in this specification.

web browser requester: An HTTP 1.1 web browser client that transmits protocol messages between an **STS** and a relying party.

web service (WS) resource: A destination HTTP 1.1 web application or an HTTP 1.1 resource serviced by the application. In the context of this protocol, it refers to the application or manager of the resource that receives identity information and assertions issued by an **STS** using this protocol. The **WS resource** is a **relying party** in the context of this protocol. For more information, see [WSFedPRP] sections 1.4 and 2.

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

1.2 References

References to Microsoft Open Specifications documentation do not include a publishing year because links are to the latest version of the documents, which are updated frequently. References to other documents include a publishing year when one is available.

A reference marked "(Archived)" means that the reference document was either retired and is no longer being maintained or was replaced with a new document that provides current implementation details. We archive our documents online [Windows Protocol].

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. Please check the archive site, http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-MWBF] Microsoft Corporation, "Microsoft Web Browser Federated Sign-On Protocol".

[RFC1738] Berners-Lee, T., Masinter, L., and McCahill, M., "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

[RFC2396] Berners-Lee, T., Fielding, R., and Masinter, L., "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998, http://www.ietf.org/rfc/rfc2396.txt

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

[RFC2965] Kristol, D., and Montulli, L., "HTTP State Management Mechanism", RFC 2965, October 2000, http://www.ietf.org/rfc/rfc2965.txt

[RFC4122] Leach, P., Mealling, M., and Salz, R., "A Universally Unique Identifier (UUID) URN Namespace", RFC 4122, July 2005, http://www.ietf.org/rfc/rfc4122.txt

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

[SOAP1.1] Box, D., Ehnebuske, D., Kakivaya, G., et al., "Simple Object Access Protocol (SOAP) 1.1", May 2000, http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[SOAP1.2-1/2007] Gudgin, M., Hadley, M., Mendelsohn, N., et al., "SOAP Version 1.2 Part 1: Messaging Framework (Second Edition) ", W3C Recommendation 27, April 2007, http://www.w3.org/TR/2007/REC-soap12-part1-20070427/

[WSDL] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S., "Web Services Description Language (WSDL) 1.1", W3C Note, March 2001, http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[XML] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0 (Fourth Edition)", W3C Recommendation, August 2006, http://www.w3.org/TR/2006/REC-xml-20060816/

[XMLNS] Bray, T., Hollander, D., Layman, A., et al., Eds., "Namespaces in XML 1.0 (Third Edition)", W3C Recommendation, December 2009, http://www.w3.org/TR/2009/REC-xml-names-20091208/

[XMLSCHEMA1] Thompson, H.S., Beech, D., Maloney, M., and Mendelsohn, N., Eds., "XML Schema Part 1: Structures", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[XMLSCHEMA2] Biron, P.V., and Malhotra, A., Eds., "XML Schema Part 2: Datatypes", W3C Recommendation, May 2001, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

[MSFT-ADC] Microsoft Corporation, "Active Directory Collection", March 2003, http://technet2.microsoft.com/WindowsServer/en/library/6f8a7c80-45fc-4916-80d9-16e6d46241f91033.mspx

[RFC4120] Neuman, C., Yu, T., Hartman, S., and Raeburn, K., "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005, http://www.ietf.org/rfc/rfc4120.txt

[WSFedPRP] IBM, BEA Systems, Microsoft, VeriSign, RSA Security, "WS-Federation: Passive Requestor Profile", version 1.0, July 2003, http://msdn.microsoft.com/en-us/library/bb608217.aspx

If you have any trouble finding [WSFedPRP], please check here.

1.3 Overview

The Microsoft Web Browser Federated Sign-On Protocol specified in [MS-MWBF] defines a standard mechanism that may be used by a client to acquire a security token from a security token service (STS). Acquiring a security token is designed to address the following problem related to communicating user information to remote applications and services.

To properly control access to information or resources in remote **web service (WS) resources**, those WS resource must have information about the users that are accessing them. Previous solutions required the WS resource to identify the user and use that identity to access further information about the user. Users were prompted multiple times to supply credentials (for example, user names and passwords) to securely identify themselves and authenticate to multiple WS resources.

Implementations of the Microsoft Web Browser Federated Sign-On Protocol solve this problem by moving the responsibility for authenticating the user away from the remote WS resource to an STS that already has an account for the user. This STS issues security tokens that contain information about the user in the form of **claims**. When accessing a WS resource, the user's web browser presents a security token obtained from an STS to the WS resource. The signature in the security token allows the WS resource to verify its validity, and the claims in the security token convey relevant user information to the WS resource. These claims can then be used for making authorization decisions by the WS resource.

Often an STS must be placed on an internal corporate network, but must also be accessible from external networks such as the Internet. In order to provide service to client requests coming from external networks, an organization may deploy a proxy component for the STS. If the organization authenticates users using SSL client certificate authentication, then a trusted channel must be used to communicate the identity of the user back to the STS. Existing HTTP proxies cannot do this without using a custom protocol.

This specification defines a protocol that enables the proxy to communicate the credentials of a user to an STS for the purpose of generating a security token to participate in a Microsoft Web Browser Federated Sign-On Protocol exchange. In addition, the protocol enables the proxy to assist users in selecting a security realm from which to obtain a security token for the STS. This enables the proxy to reduce the number of requests from external networks that must be serviced by the STS

The protocol is based on SOAP as defined in [SOAP1.1] and [SOAP1.2-1/2007]. The protocol defines the following operations:

- A <GetProxyTrustConfiguration> operation that enables the STS proxy to obtain configuration data from the STS that is necessary to assist users in selecting an acceptable security realm from which to obtain a security token.
- An <LsRequestSecurityToken>, <RequestSecurityTokenWithToken>, and <LsRequestSecurityTokenWithCookie> operations that enable the STS proxy to forward Microsoft Web Browser Federated Sign-On Protocol requests back to the STS, and convert the responses from the STS into Microsoft Web Browser Federated Sign-On Protocol responses.

The protocol specification describes the message processing model in section 3 for the client and the STS to successfully emit or consume protocol messages that are created in accordance with section 2.

1.4 Relationship to Other Protocols

The Active Directory Federation Services (AD FS) Proxy Protocol uses standard web protocols. The reader should be familiar with the IETF specifications.

- Hypertext Transfer Protocol (HTTP) [RFC2616].
- Uniform Resource Identifiers (URIs).
- Uniform Resource Locators (URLs).

URLs and URIs are used to describe the data used in the protocol.

The Active Directory Federation Services (AD FS) Proxy Protocol uses Extensible Markup Language (XML); the following specifications are used to describe the requirements for the XML syntax involved in the protocol. The reader should be familiar with the following W3C specifications.

- Extensible Markup Language (XML) 1.0 (Fourth Edition) ([XML]).
- Namespaces in XML ([XMLNS]).
- SOAP Version 1.1 ([SOAP1.1]).
- SOAP Version 1.2 ([SOAP1.2-1/2007]).
- XML Schema Part 1: Structures Second Edition ([XMLSCHEMA1]).
- XML Schema Part 2: Datatypes Second Edition ([XMLSCHEMA2]).

1.5 Prerequisites/Preconditions

The client MUST be configured with the URL of the server's SOAP service in order to call the service.

1.6 Applicability Statement

The Active Directory Federation Services (AD FS) Proxy Protocol is used by any implementer that requires data about the configuration of an STS in order to validate security tokens from that STS. The software that needs knowledge of an STS’s configuration is often WS resources software that expects to receive security tokens from users that are attempting to access the WS resource.

1.7 Versioning and Capability Negotiation

1.7.1 Versioning

This protocol uses the versioning mechanisms defined in the following specifications.

- SOAP 1.1 ([SOAP1.1]).
- SOAP 1.2 ([SOAP1.2-1/2007]).

The data formatting and message processing of this protocol do not contain any further versioning mechanisms. The data itself is versioned to enable servers to determine whether clients need a full update or have an up-to-date version. This mechanism is described fully in sections 2 and 3 that follow.

1.7.2 Capability Negotiation

None.

1.8 Vendor-Extensible Fields

As specified in section 2, the Active Directory Federation Services (AD FS) Proxy Protocol uses SOAP messages for communication, as specified in [SOAP1.1] and [SOAP1.2-1/2007]. The core functionality of SOAP is to provide extensibility. [SOAP1.2-1/2007] and [SOAP1.1] contains detailed discussion on SOAP messaging framework extensibility model. Vendors may use these SOAP extensibility points as specified.

1.9 Standards Assignments

- There are no standards assignments for the Active Directory Federation Services (AD FS) Proxy Protocol beyond those defined in the following specifications.
- SOAP 1.1 ([SOAP1.1]).
- SOAP 1.2 ([SOAP1.2-1/2007]).

2 Messages

2.1 Transport

The GetProxyTrustConfiguration request, LsRequestSecurityToken request, RequestSecurityTokenWithToken request, and LsRequestSecurityTokenWithCookie request messages MUST be transmitted using the HTTP POST method; they MUST NOT be transmitted using the GET method.

The client role and server role MUST use the HTTPS URL scheme to identify the server endpoints for processing the GetProxyTrustConfiguration request, LsRequestSecurityToken request, RequestSecurityTokenWithToken request, and LsRequestSecurityTokenWithCookie request messages.

2.2 Message Syntax

This section specifies the transport and syntax of request and response messages in normative detail. References to the Protocol Details section are included when knowledge of the protocol details are necessary to understand the context of message transport or syntax.

2.2.1 All Messages

All protocol messages MUST be well-formed XML placed within a SOAP envelope conforming to [SOAP1.2-1/2007] section 5.1 or [SOAP1.1] section 4.

2.2.2 GetProxyTrustConfiguration Request

The SOAP body of the request message MUST conform to the following XML Schema.

```
<s:element name="GetProxyTrustConfiguration">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="proxyVersion"
type="tns:VersionInformation" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="VersionInformation">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="SoftwareVersion" type="s:long" />
    <s:element minOccurs="1" maxOccurs="1" name="Guid" type="s1:guid" />
    <s:element minOccurs="1" maxOccurs="1" name="Version" type="s:long" />
  </s:sequence>
</s:complexType>
<s:simpleType name="guid">
  <s:restriction base="s:string">
    <s:pattern value="[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-
F]{12}" />
  </s:restriction>
</s:simpleType>
```

SoftwareVersion Parameter: The value of this parameter MUST be 1. See sections 3.1.5.1.2.1 (Versioning) and 3.2.5.1.2.1 (Versioning Processing) for details.

2.2.3 GetProxyTrustConfiguration Response

The SOAP body of the response message MUST conform to the following XML Schema.

```
<s:element name="GetProxyTrustConfigurationResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="GetProxyTrustConfigurationResult"
type="s:boolean" />
      <s:element minOccurs="0" maxOccurs="1" name="fsVersion" type="tns:VersionInformation"
/>
      <s:element minOccurs="0" maxOccurs="1" name="proxyInformation"
type="tns:ProxyInformation" />
      <s:element minOccurs="0" maxOccurs="1" name="trustConfig"
type="tns:ArrayOfTrustConfigurationData" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="VersionInformation">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="SoftwareVersion" type="s:long" />
    <s:element minOccurs="1" maxOccurs="1" name="Guid" type="s1:guid" />
    <s:element minOccurs="1" maxOccurs="1" name="Version" type="s:long" />
  </s:sequence>
</s:complexType>
<s:simpleType name="guid">
  <s:restriction base="s:string">
    <s:pattern value="[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-
F]{12}" />
  </s:restriction>
</s:simpleType>
<s:complexType name="ProxyInformation">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="1" name="HostedRealmUriStr" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="LsUrlStr" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="ConfigInfo"
type="tns:ProxyConfigurationInformation" />
  </s:sequence>
</s:complexType>
<s:complexType name="ProxyConfigurationInformation">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="1" name="CookiePath" type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="SuppressRealmCookie" type="s:boolean" />
    <s:element minOccurs="1" maxOccurs="1" name="RealmCookieLifetime" type="s:int" />
  </s:sequence>
</s:complexType>
<s:complexType name="ArrayOfTrustConfigurationData">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="TrustConfigurationData"
nillable="true" type="tns:TrustConfigurationData" />
  </s:sequence>
</s:complexType>
<s:complexType name="TrustConfigurationData">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="trustType" type="tns:TrustTypes" />
    <s:element minOccurs="1" maxOccurs="1" name="trustDisplayName" type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="trustUri" type="s:string" />
    <s:element minOccurs="1" maxOccurs="1" name="trustLsUrl" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="acceptableAuthenticationMethodStrings"
type="tns:ArrayOfString" />
```

```
  </s:sequence>
</s:complexType>
<s:simpleType name="TrustTypes">
  <s:restriction base="s:string">
    <s:enumeration value="TrustedRealm" />
    <s:enumeration value="TrustingRealm" />
    <s:enumeration value="TrustingResource" />
    <s:enumeration value="SelfhostedRealm" />
    <s:enumeration value="UnknownTrustType" />
  </s:restriction>
</s:simpleType>
<s:complexType name="ArrayOfString">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true"
type="s:string" />
  </s:sequence>
</s:complexType>
```

Parameter	Value
SoftwareVersion	The value of this parameter MUST be 1.
HostedRealmUriStr	This parameter MUST be a URI conforming to [RFC2396].
LsUrlStr	This parameter MUST be a URL conforming to [RFC1738].
CookiePath	This parameter MUST conform to a cookie path per [RFC2965].
trustUri	This parameter MUST be a URI conforming to [RFC2396].
trustLsUrl	This parameter MUST be a URL conforming to [RFC1738].
acceptableAuthenticationMethodStrings	This parameter MUST be an empty element or a list of URIs conforming to [RFC2396].

2.2.4 LsRequestSecurityToken Request

The SOAP body of the request message MUST conform to the following XML Schema.

```
<s:element name="LsRequestSecurityToken">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="1" maxOccurs="1" name="credentialTypeUri" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="credentials" type="tns:ArrayOfString" />
      <s:element minOccurs="0" maxOccurs="1" name="accountStoreUri" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="cookie" type="s:base64Binary" />
      <s:element minOccurs="1" maxOccurs="1" name="targetRealmName" type="s:string" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="ArrayOfString">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true"
type="s:string" />
  </s:sequence>
</s:complexType>
```

Parameter	Value
credentialTypeUri	This parameter MUST be a URI conforming to [RFC2396]. The value MUST be "urn:oasis:names:tc:SAML:1.0:am:password" or "urn:ietf:rfc:2246".
credentials	The credential's parameter MUST be either a list of 4 strings or a list of 2 strings. If the parameter is a list of 4 strings, the value of the first string MUST be Username, and the value of the third string MUST be password. If the parameter is a list of 2 strings, then the first string MUST be Certificate. The value of the second string MUST be an X.509 certificate per [WSDL] that is Base64-encoded per [RFC4648].
accountStoreUri	This parameter MUST be a URI conforming to [RFC2396].
targetRealmName	This parameter MUST be a URI conforming to [RFC2396].

2.2.5 LsRequestSecurityToken Response

The SOAP body of the response message MUST conform to the following XML Schema.

```
<s:element name="LsRequestSecurityTokenResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="rstr" type="tns:RSTRResult" />
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="RSTRResult">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="Status" type="tns:RSTRStatus" />
    <s:element minOccurs="0" maxOccurs="1" name="PolicyVersion" type="tns:VersionInformation"
/>
    <s:element minOccurs="0" maxOccurs="1" name="CredentialsVerification"
type="tns:CredentialsVerificationInfo" />
    <s:element minOccurs="0" maxOccurs="1" name="ForeignRealmUri" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="SecurityToken" type="s:base64Binary" />
    <s:element minOccurs="0" maxOccurs="1" name="LogonAcceleratorToken" type="s:base64Binary"
/>
  </s:sequence>
</s:complexType>
<s:simpleType name="RSTRStatus">
  <s:restriction base="s:string">
    <s:enumeration value="Success" />
    <s:enumeration value="WrongPrincipal" />
    <s:enumeration value="NoAcceptableCredential" />
    <s:enumeration value="InvalidTarget" />
    <s:enumeration value="ValidationFailure" />
    <s:enumeration value="GenerationFailure" />
    <s:enumeration value="SidExpansionFailure" />
    <s:enumeration value="NoAccountStores" />
    <s:enumeration value="NoActiveDirectoryForSids" />
    <s:enumeration value="NoAccountStoresForCert" />
    <s:enumeration value="Unset" />
  </s:restriction>
</s:simpleType>
<s:complexType name="VersionInformation">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="SoftwareVersion" type="s:long" />
    <s:element minOccurs="1" maxOccurs="1" name="Guid" type="s1:guid" />
```

```
    <s:element minOccurs="1" maxOccurs="1" name="Version" type="s:long" />
  </s:sequence>
</s:complexType>
<s:complexType name="CredentialsVerificationInfo">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="AccountStoreType"
type="tns:AccountStoreType" />
    <s:element minOccurs="0" maxOccurs="1" name="AccountStoreTypeDisplay" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="AccountStoreUriString" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="AccountStoreDisplayName" type="s:string" />
    <s:element minOccurs="0" maxOccurs="1" name="UserValidationData"
type="tns:UserValidationInfo" />
  </s:sequence>
</s:complexType>
<s:simpleType name="AccountStoreType">
  <s:restriction base="s:string">
    <s:enumeration value="ActiveDirectoryType" />
    <s:enumeration value="LdapDirectoryType" />
    <s:enumeration value="UnknownStoreType" />
  </s:restriction>
</s:simpleType>
<s:complexType name="UserValidationInfo">
  <s:sequence>
    <s:element minOccurs="1" maxOccurs="1" name="ErrorCode" type="s:long" />
    <s:element minOccurs="0" maxOccurs="1" name="AdditionalValidationInfo"
type="tns:ArrayOfString" />
  </s:sequence>
</s:complexType>
<s:complexType name="ArrayOfString">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true"
type="s:string" />
  </s:sequence>
</s:complexType>
```

Parameter	Value
SoftwareVersion	The value of this parameter MUST be 1.
ForeignRealmUri	This parameter MUST be a URI conforming to [RFC2396].
SecurityToken	This parameter MUST be a Base64-encoded [RFC4648] security token conforming to [MS-MWBF] section 2.2.4.2.
AccountStoreUriString	The syntax of this parameter is specified in section 3.2.5.2.2.3.

2.2.6 RequestSecurityTokenWithToken Request

The SOAP body of the request message MUST conform to the following XML Schema.

```
<s:element name="RequestSecurityTokenWithToken">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="inToken" type="s:base64Binary" />
      <s:element minOccurs="0" maxOccurs="1" name="cookie" type="s:base64Binary" />
      <s:element minOccurs="0" maxOccurs="1" name="targetRealmName" type="s:string" />
    </s:sequence>
  </s:complexType>
```

```
</s:element>
```

Parameter	Value
inToken	The syntax of this parameter is specified in section 3.1.1.2.
targetRealmName	The syntax of this parameter is specified in section 2.2.4.

2.2.7 RequestSecurityTokenWithToken Response

The SOAP body of the response message MUST conform to the following XML Schema.

```
<s:element name="RequestSecurityTokenWithTokenResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="rstr" type="tns:RSTRResult" />
    </s:sequence>
  </s:complexType>
</s:element>
```

The RSTRResult schema is specified in section 2.2.5.

2.2.8 LsRequestSecurityTokenWithCookie Request

The SOAP body of the request message MUST conform to the following XML Schema.

```
<s:element name="LsRequestSecurityTokenWithCookie">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="latToken" type="s:base64Binary" />
      <s:element minOccurs="0" maxOccurs="1" name="targetRealmName" type="s:string" />
      <s:element minOccurs="0" maxOccurs="1" name="authMethodUris" type="tns:ArrayOfString"
/>
    </s:sequence>
  </s:complexType>
</s:element>
<s:complexType name="ArrayOfString">
  <s:sequence>
    <s:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true"
type="s:string" />
  </s:sequence>
</s:complexType>
```

Parameter	Value
targetRealmName	This parameter MUST be a URI conforming to [RFC2396].
authMethodUris	The syntax of this parameter is identical to the syntax of <acceptableAuthenticationMethodStrings> defined in section 2.2.3.

2.2.9 LsRequestSecurityTokenWithCookie Response

The SOAP body of the response message MUST conform to the following XML Schema.

```
<s:element name="LsRequestSecurityTokenWithCookieResponse">
  <s:complexType>
    <s:sequence>
      <s:element minOccurs="0" maxOccurs="1" name="rstr" type="tns:RSTRResult" />
    </s:sequence>
  </s:complexType>
</s:element>
```

The RSTRResult schema is specified above in section 2.2.5.

3 Protocol Details

3.1 Client Role Details

This section describes details of protocol processing that must be understood to implement a client that can correctly perform its role in the protocol message exchange.

3.1.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains 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.

The data used by each message exchange is different. The abstract data models for the GetProxyTrustConfiguration, LsRequestSecurityToken, RequestSecurityTokenWithToken, and LsRequestSecurityTokenWithCookie message exchanges can be found in the following sections.

3.1.1.1 GetProxyTrustConfiguration

The client calls this method to get the information required by the client to verify security tokens issued by the server to the client using the protocol specified in [MS-MWBF]. The following data is used in the client's request sent to the server and in the server's response sent to the client.

Name	Description	Corresponding message parameter
Client Policy GUID	This is a globally unique identifier for the policy that is held by the client at the time of a GetProxyTrustConfiguration request.	Request: <GUID> element
Server Policy GUID	This is a globally unique identifier for the policy that is maintained by the server at the time of issuing a GetProxyTrustConfiguration response.	Response: <GUID> element
Client Policy Version	This is a version number for the policy that is held by the client at the time of a GetProxyTrustConfiguration request.	Request: <Version> element
Server Policy Version	This is a version number for the policy that is maintained by the server at the time of issuing a GetProxyTrustConfiguration response.	Response: <Version> element
Hosted Realm URI	This is an identifier for the server. This URI is used in security tokens to identify the server as the issuer of the security token.	Response: <HostedRealmUriStr>
Login Service URL	This is the URL that client SHOULD redirect service requests to using the protocol specified in [MS-MWBF].	Response: <LsUrlStr>
Cookie Path	This is the cookie path per [RFC2965] to use when issuing cookies from the proxy.	Response: <CookiePath>
Suppress Realm Cookie	If true, this parameter indicates that the user's security realm selection SHOULD NOT be cached in a [RFC2965] cookie.	Response: <SuppressRealmCookie>

Name	Description	Corresponding message parameter
Realm Cookie Lifetime	This parameter dictates the lifetime of a [RFC2965] cookie for caching the user's security realm selection.	Response: <RealmCookieLifetime>
List of Security Realm Specific Data	This parameter contains a list of the security realm specific data described in the following table.	Response: <trustConfig>

The following table contains a list of possible values for the List of Security Realm Specific Data parameter in the **GetProxyTrustConfiguration** method.

Name	Description	Corresponding message parameter
Security Realm Type	All security realms with a Trust Type not equal to "TrustedRealm" are ignored.	Response: <trustType>
Security Realm Display Name	The Trust Display Name is the name to display to users who are choosing a security realm.	Response: <trustDisplayName>
Security Realm URI	The Trust URI is the internal identifier of the security realm.	Response: <trustUri>
Security Realm Login Service URL	The Trust Login Service URL is the URL to which users SHOULD be directed when they select the security realm.	Response: <trustLsUrl>
Acceptable Authentication Methods for Security Realm	The Acceptable Authentication Methods for Security Realm is a list of URIs that identify acceptable methods of authentication for the security realm. The list of method URIs is included with the requests to the security realm using the *wauth* parameter described in [MS-MWBF] section 2.2.3.	Response: <acceptableAuthenticationMethodStrings>

3.1.1.2 LsRequestSecurityToken, RequestSecurityTokenWithToken, and LsRequestSecurityTokenWithCookie

At the client, a higher layer may determine whether the server accepts security tokens from a particular user's security realm as described in [MS-MWBF]. The user is represented by an email address. The client calls this method to learn whether the email address belongs to a security realm from which the server will accept tokens using the protocol defined in [MS-MWBF]. The following data is used in the client's request sent to the server and in the server's response sent to the client.

Name	Description	Corresponding message parameter
Incoming Token	This parameter MUST be a Base64-encoded [RFC4648] security token conforming to [MS-MWBF] section 2.2.4.2. This is the security token obtained from the *wresult* parameter.	**RequestSecurityTokenWithToken** Request: <inToken>
Outgoing Security	This parameter MUST be a Base64-encoded [RFC4648] security token conforming to [MS-	All Responses: <SecurityToken>

Name	Description	Corresponding message parameter
Token	[MWBF] section 2.2.4.2. This is the security token to issue in the wresult parameter of [MS-MWBF].	
Incoming Cookie	This parameter MUST be Base64-encoded [RFC4648] data used by the STS to cache data about the user as a [RFC2965] cookie. The protocol does not constrain the format of this data since it is written by the STS for later processing by the STS. STS implementations may use any appropriate data format, and proxy implementations need only retrieve it from the client as an [RFC2965] cookie.	**LsRequestSecurityToken** Request, **RequestSecurityTokenWithToken** Request: <cookie> **LsRequestSecurityTokenWithCookie** Request: <latToken>
Outgoing Cookie	This parameter MUST be Base64-encoded [RFC4648] data used by the STS to cache data about the user as a [RFC2965] cookie. The protocol does not constrain the format of this data since it is written by the STS for later processing by the STS. STS implementations may use any appropriate data format, and proxy implementations need only write it to the client as an [RFC2965] cookie.	All Responses: <LogonAcceleratorToken>
Target Security Realm URI	This parameter identifies the security realm for whom the STS should issue the security token. This parameter is taken from the wtrealm parameter of [MS-MWBF].	All Requests: <TargetRealmName>
Credential Type URI	This parameter identifies whether the Credentials parameter contains a username and password or a certificate.	**LsRequestSecurityToken** Request: <credentialTypeUri>
Credentials	This parameter either contains a username and password, or a certificate. It is used by the STS to look up claims about the user.	**LsRequestSecurityToken** Request: <credentials>
Server Policy Version	This is a version number for the policy that is maintained by the server at the time of issuing a **GetProxyTrustConfiguration** response.	<Version> element
Server Policy GUID	This is a globally unique identifier for the policy that is maintained by the server at the time of issuing a **GetProxyTrustConfiguration** response.	<Guid> element
Foreign Realm Name/URI	This parameter is the security realm identifier for use in caching the **web browser requester**'s security realm selection.	All Responses: <ForeignRealmUri>
Requested Account Store URI	This parameter identifies the store that client requests to be used for generating claims about the user.	**LsRequestSecurityToken** Request: <accountStoreUri>
Response	This parameter either indicates a successful	All Responses: <Status>

Name	Description	Corresponding message parameter
Status	request or provides information on why the request failed.	
Credentials Verification Information	This parameter contains relevant data about the account store used to generate claims about the user. It is only used by the client for error details that may be presented to the web browser requester.	All Responses: <CredentialsVerification>

3.1.2 Timers

There are no protocol-specific timer events that MUST be serviced by an implementation. This protocol does not require timers beyond those that may be used by the underlying transport to transmit and receive messages over HTTP. The protocol does not include provisions for time-based retry for sending protocol messages.

3.1.3 Initialization

The initialization steps required for each of the three protocol message request and response pairs are unrelated to one another. Prior to sending any protocol message, the client MUST be configured with the URL to which the request should be sent. The following sections define the initialization required for the client role prior to sending each request message.

3.1.3.1 GetProxyTrustConfiguration Initialization

The client MAY maintain a cached copy of the data described in the GetProxyTrustConfiguration section.<1>

Prior to emitting a GetProxyTrustConfiguration request, the client MUST obtain the version number and **GUID**, as specified in [RFC4122], section 3, of the currently cached trust information. If no trust information is cached on the client, the client MUST use a version number equal to 0, and a GUID equal to 00000000-0000-0000-0000-000000000000.

3.1.3.2 LsRequestSecurityToken, RequestSecurityTokenWithToken, and LsRequestSecurityTokenWithCookie Initialization

None.

3.1.4 Higher-Layer Triggered Events

The GetProxyTrustConfiguration, LsRequestSecurityToken, RequestSecurityTokenWithToken, and LsRequestSecurityTokenWithCookie message exchanges are triggered by various events. The following sections describe the events that trigger each exchange.

3.1.4.1 GetProxyTrustConfiguration

As described in the GetProxyTrustConfiguration section, the client sends a GetProxyTrustConfiguration request when the client needs the data described in the GetProxyTrustConfiguration section to verify the security tokens issued by the server. Thus, a GetProxyTrustConfiguration request MAY be triggered by the receipt of a security token request at the client, as described in [MS-MWBF]. Implementations MAY choose to improve the performance of handling security token requests by sending a GetProxyTrustConfiguration request and caching the data from the response prior to receiving a request for a security token.<2>

3.1.4.2 LsRequestSecurityToken

When the client is serving as a proxy for an STS in the Requestor STS role described in [MS-MWBF], the client MUST emit an <LsRequestSecurityToken> request message after it authenticates a new web browser requestor requesting a security token using the protocol described in [MS-MWBF]. A new web browser requestor is a web browser requestor that does not present an [RFC2965] session cookie issued by the STS with its security token request.

If a session cookie is presented by the web browser requestor, the client MAY emit an <LsRequestSecurityToken> request message or an <LsRequestSecurityTokenWithCookie> request message, given that no token has been posted in the *wresult* parameter described by [MS-MWBF].<3>

3.1.4.3 RequestSecurityTokenWithToken

When the client is serving as a proxy for an STS in the **relying party** role described in [MS-MWBF], the client MUST emit a RequestSecurityTokenWithToken request message after it receives a security token in the *wresult* parameter of [MS-MWBF] as part of a security token request.

3.1.4.4 LsRequestSecurityTokenWithCookie

When the client is serving as a proxy for an STS in the Requestor STS role described in [MS-MWBF], the client SHOULD emit an LsRequestSecurityTokenWithCookie request message after it receives a session cookie from a web browser requestor requesting a security token using the protocol described in [MS-MWBF], given that no token has been posted in the *wresult* parameter described by [MS-MWBF].<4>

3.1.5 Message Processing Events and Sequencing Rules

The request messages detailed in section 2 are all unrelated to one another. A client MUST emit request messages according to the events that trigger the requests as described above in the Higher-Layer Triggered Events section. The following sections define the message processing rules separately for the GetProxyTrustConfiguration, LsRequestSecurityToken, RequestSecurityTokenWithToken, and LsRequestSecurityTokenWithCookie message exchanges.

3.1.5.1 GetProxyTrustConfiguration

The GetProxyTrustConfiguration exchange MUST consist of a single request message and a single response message. The exchange MUST be initiated by the client with a request message to the server. The following sections describe the client processing for the request and response messages.

3.1.5.1.1 GetProxyTrustConfiguration Request Processing

As described in GetProxyTrustConfiguration Initialization section, the client MUST include the current policy version number and corresponding GUID in the request.

3.1.5.1.2 GetProxyTrustConfiguration Response Processing

Processing the response may be divided into processing the versioning, certificates and other aspects of the response. The following sections address this processing.

3.1.5.1.2.1 Versioning

As detailed in section 3.1.1.1, the response MUST contain a version number and GUID representing the configuration data described in section 2.2.2. This version number and GUID MUST be compared

to the locally cached information. If the GUID from the response is different than the GUID cached locally, then the response contains newer data that MUST be used instead of the locally cached data. If the response GUID and locally cached GUID are identical, but the locally cached version number is less than the response version number, then the response contains newer data that MUST be used instead of the locally cached data. If there is no locally cached data, the version number and GUID MUST be ignored.

3.1.5.1.2.2 STS Data

The STS data contained in the response MUST be cached for use in the protocol described in [MS-MWBF]. The Login Service URL MUST be cached to use for listening for requests according to [MS-MWBF]. The Hosted Realm URI MUST be cached for identifying the server in the *wtrealm* parameter sent to another security realm after the web browser requestor selects the security realm.

3.1.5.1.2.3 Cookie Data

The cookie data returned in the response MUST be cached in order to appropriately issue [RFC2965] cookies. The Cookie Path value MUST be cached, and MUST be used for every cookie sent to the web browser requestor. The SuppressRealmCookie value MUST be cached, and if the value is true then the client MUST NOT use a cookie to save a web browser requestor's selection of security realm. The RealmCookieLifetime value MUST be cached, and if the SuppressRealmCookie value is false, the lifetime of cookies storing the web browser requestor's security realm selection MUST be the value of RealmCookieLifetime in minutes.

3.1.5.1.2.4 Security Realm Data

The security realm data returned in the response MUST be cached in order to offer the web browser requestor the appropriate security realm choices. All security realm entries from the response without a security realm type of "TrustedRealm" MUST be ignored by the client. Security realms with a security realm type of "TrustedRealm" are used to offer the web browser requestor the appropriate security realm choices of security realms where a security token may be obtained.

If the response contains any security realm Accepted Authentication Methods URIs, then the client MUST include those URIs in the *wauth* parameter sent to the Requestor STS as described in [MS-MWBF] section 2.2.3. The security realm Login Service URL MUST be used to direct the [MS-MWBF] request to the appropriate URL after a web browser requestor has selected a security realm. The security realm Display Name MAY be used to provide a human readable identifier for the security realm.<5>

3.1.5.2 LsRequestSecurityToken

The LsRequestSecurityToken exchange MUST consist of a single request message and a single response message. The exchange MUST be initiated by the client with a request message to the server. The following sections describe the client processing for the request and response messages.

3.1.5.2.1 LsRequestSecurityToken Request

As described above in the LsRequestSecurityToken section, when the client is serving as a proxy for an STS in the Requestor STS role described in [MS-MWBF], the client MUST emit an LsRequestSecurityToken request message after it authenticates a new user requesting a security token using the protocol described in [MS-MWBF].

The targetRealmName element MUST be populated by the *wtrealm* parameter of the [MS-MWBF] request for a security token.

The credentialType and credentials elements are determined by the method used at the client for authenticating the [MS-MWBF] web browser requestor. The client MAY use username and password authentication or SSL client certificate authentication.<6>

If SSL client certificate authentication is used, the *credentialTypeUri* parameter MUST be "urn:ietf:rfc:2246". If username and password authentication is used, the credentialTypeUri MUST be "urn:oasis:names:tc:SAML:1.0:am:password".

If SSL client certificate authentication is used, the credentials element MUST contain only two values. The first value MUST equal "Certificate". The value of the second string MUST be an X.509 certificate per [WSDL] that is Base64-encoded per [RFC4648].

If user name and password authentication is used, the credential element MUST contain only four values. The value of the first string MUST be Username. The value of the second string MUST be a username for the web browser requestor. The value of the third string MUST be Password. The value of the fourth string MUST be a password for the web browser requestor.

The client MAY specify an identifier for a particular account store to be used by the server when generating claims for the web browser requestor using the accountStoreUri element.<7>

The client MAY specify an [RFC2965] cookie value that is Base64-encoded per [RFC4648] in the cookie element of the request.<8>

3.1.5.2.2 LsRequestSecurityToken Response

The parameters of the LsRequestSecurityTokenResponse are processed as described in the following sections.

3.1.5.2.2.1 Status

If the Status value is Success, then the request was successful and the client MUST consume the PolicyVersion, ForeignRealmUri, SecurityToken and LogonAcceleratorToken message parameters.

If the Status value is not Success, then the request was not successful and that Status value MAY be used to provide guidance to the web browser requestor. The *CredentialsVerification* parameter MAY also be used to provide guidance on the error to the web browser requestor. Other parameters MUST be ignored and the client MUST fault and return an error to the web browser requestor as described in section 2 of [MS-MWBF].<9>

3.1.5.2.2.2 PolicyVersion

As detailed in the LsRequestSecurityToken Response section, the response MUST contain a version number and GUID representing the configuration data described in the LsRequestSecurityToken, RequestSecurityTokenWithToken, and LsRequestSecurityTokenWithCookie sections. Similarly to a GetProxyTrustConfiguration response message, this version number and GUID MUST be compared to the locally cached information. If the GUID from the response is different than the GUID cached locally, then the server has newer configuration data and the client SHOULD emit a GetProxyTrustConfiguration request to update its local cache. If the response GUID and locally cached GUID are identical, but the locally cached version number is less than the response version number, then the server has newer configuration data and the client SHOULD emit a GetProxyTrustConfiguration request to update its local cache. If there is no locally cached data, the version number and GUID MUST be ignored.<10>

3.1.5.2.2.3 CredentialsVerification

The information found within the CredentialsVerification structure is informational only, and clients MAY ignore it.

3.1.5.2.2.4 ForeignRealmUri

The ForeignRealmUri value MUST be ignored by the client.

3.1.5.2.2.5 SecurityToken

This parameter MUST be a Base64-encoded [RFC4648] security token conforming to [MS-MWBF] section 2.2.4.2. This is the security token to issue in the *wresult* parameter of [MS-MWBF]. Prior to issuing the security token in the *wresult* parameter, the security token MUST be Base64-decoded.

3.1.5.2.2.6 LogonAcceleratorToken

This parameter MUST be Base64-encoded [RFC4648] data used by the STS to cache information about the user as a [RFC2965] cookie. The protocol does not constrain the format of this data since it is written by the STS for later processing by the STS. STS implementations may use any appropriate data format, and proxy implementations need only write the data to the web browser requestor as an [RFC2965] cookie.

3.1.5.3 RequestSecurityTokenWithToken

The RequestSecurityTokenWithToken exchange MUST consist of a single request message and a single response message. The exchange MUST be initiated by the client with a request message to the server. The following sections describe the client processing for the request and response messages.

3.1.5.3.1 RequestSecurityTokenWithToken Request

As described in the RequestSecurityTokenWithToken section, when the client is serving as a proxy for an STS in the relying party role described in [MS-MWBF], the client MUST emit an RequestSecurityTokenWithToken request message after it receives a security token as part of a security token request using the protocol described in [MS-MWBF].

The security token received by the client in the *wresult* parameter described in [MS-MWBF] MUST be Base64-encoded according to [RFC4648], and included in the request in the inToken element of the request.

The *wtrealm* parameter received by the client MUST be included in the targetRealmName element of the request.

If the web browser requestor also presents a cookie as part of the request for a security token, that cookie MUST be included in the cookie element of the request.

3.1.5.3.2 RequestSecurityTokenWithToken Response

Response processing for a RequestSecurityTokenWithToken response MUST be the same as the processing for a LsRequestSecurityToken response as described in the LsRequestSecurityToken Response section, with the exception of the processing of the ForeignRealmUri element.

For RequestSecurityTokenWithToken response messages, if the SuppressRealmCookie configuration value is false, then the client MUST use the value of the ForeignRealmUri element to write an

[RFC2965] cookie to the web browser requestor that records the security realm of the token presented by web browser requestor's using [MS-MWBF]. If the SuppressRealmCookie configuration value is true, then the *ForeignRealmUri* parameter MUST be ignored.

3.1.5.4 LsRequestSecurityTokenWithCookie

The LsRequestSecurityTokenWithCookie exchange MUST consist of a single request message and a single response message. The exchange MUST be initiated by the client with a request message to the server. The following sections describe the client processing for the request and response messages.

3.1.5.4.1 LsRequestSecurityTokenWithCookie Request

As described in the LsRequestSecurityTokenWithCookie section, when the client is serving as a proxy for an STS in the Requestor STS role described in [MS-MWBF], the client MUST emit an LsRequestSecurityTokenWithCookie request message after it receives a session cookie from a user requesting a security token using the protocol described in [MS-MWBF].

The cookie received by the client in MUST be Base64-encoded according to [RFC4648], and included in the request in the latToken element of the request.

The *wtrealm* parameter received by the client MUST be included in the targetRealmName element of the request.

If the security token request includes a *wauth* parameter as described in section 2.2.3 of [MS-MWBF], the URIs of that parameter MUST be included in the authMethodUris list of string elements in the request.

3.1.5.4.2 LsRequestSecurityTokenWithCookie Response

Response processing for an LsRequestSecurityTokenWithCookie response MUST be the same as the processing for a LsRequestSecurityToken response as described in the LsRequestSecurityToken Response section.

3.1.6 Timer Events

There are no protocol-specific timer events that MUST be serviced by an implementation. This protocol does not require timers beyond those that may be used by the underlying transport to transmit and receive messages over HTTPS. The protocol does not include provisions for time-based retry for sending protocol messages.

3.1.7 Other Local Events

This protocol does not have dependencies on any transport protocols other than HTTP 1.1. This protocol relies on this transport mechanism for the correct and timely delivery of protocol messages. The protocol does not take action in response to any changes or failure in machine state or network communications.

3.2 Server Role Details

This section describes details of protocol processing that must be understood to implement a server that can correctly perform its role in the protocol message exchange.

3.2.1 Abstract Data Model

The abstract data model described in section 3.1.1 applies for the server role as well.

3.2.2 Timers

There are no protocol-specific timer events that MUST be serviced by an implementation. This protocol does not require timers beyond those that may be used by the underlying transport to transmit and receive messages over HTTP. The protocol does not include provisions for time-based retry for sending protocol messages.

3.2.3 Initialization

Prior to receiving request messages, the server MUST open an endpoint to listen for request messages. In order to provide the data described in the abstract data model, that data MUST be configured on the server by an administrator.

3.2.4 Higher-Layer Triggered Events

An STS server is triggered on receipt of a protocol message to process that message and respond to the client that sent it.

3.2.5 Message Processing Events and Sequencing Rules

The request messages detailed in section 2 are all unrelated to one another. The following sections define the message processing rules separately for the GetProxyTrustConfiguration, LsRequestSecurityToken, RequestSecurityTokenWithToken, and LsRequestSecurityTokenWithCookie message exchanges.

3.2.5.1 GetProxyTrustConfiguration

The GetProxyTrustConfiguration exchange MUST consist of a single request message and a single response message. The exchange MUST be initiated by the client with a request message to the server. The following sections describe the server processing for the request and response messages.

3.2.5.1.1 GetProxyTrustConfiguration Request Processing

The version number and GUID parameters in GetProxyTrustConfiguration requests MUST be compared to the current version number and GUID of the server's local configuration. If the GUID in the request is different than the GUID of the server's local configuration, then the client has an outdated copy. If the version number in the request is different than the version number of the server's local configuration, then the client has an outdated copy. Otherwise the client has an up-to-date copy. For the corresponding response processing, see section 3.2.5.1.2 below.

3.2.5.1.2 GetProxyTrustConfiguration Response Processing

GetProxyTrustConfiguration response processing can be divided into version processing, certificates processing and other processing. The following sections discuss these processing steps.

3.2.5.1.2.1 Versioning Processing

If the client's version is up-to-date, as described in the preceding section 3.2.5.1.1, then the GetProxyTrustConfigurationResult MUST be set to false, and the fsVersion, proxyInformation, and trustConfig elements described in section 2.2.3 MUST be omitted from the response.

If the client's version is an outdated copy, then the GetProxyTrustConfigurationResult MUST be set to true, and the fsVersion, proxyInformation, and trustConfig elements described in section 2.2.3 MUST be included in the response.

The Version element MUST be set to the version number for the current configuration maintained by the server. The Guid element MUST be set to the GUID for the current configuration maintained by the server.

3.2.5.1.2.2 STS Data

The server MUST maintain a URI to identify itself as described in GetProxyTrustConfiguration section. This URI MUST be included in the response as the HostedRealmUriStr element.

The server MUST maintain a URL that represents the endpoint on which it listens for [MS-MWBF] requests. This URL MUST be included in the response as the LsUrlStr element.

3.2.5.1.2.3 Cookie Data

The server MUST maintain a configuration setting for the cookie path to use for [RFC2965] session cookies. This cookie path MUST be included in the response as the CookiePath element.

The server MUST maintain a configuration setting for whether to issue a cookie caching the web browser requestor's security realm selection. If the selection MUST be cached, then the server MUST include a value of FALSE in the response in the SuppressRealmCookie element. If the selection is not cached, then the server MUST include a value of TRUE in the response in the SuppressRealmCookie element.

The server MUST maintain a configuration setting for how long a web browser requestor's security realm selection should be cached in a cookie. This realm cookie lifetime MUST be included in the response as the RealmCookieLifetime element.

3.2.5.1.2.4 Security Realm Data

The server MUST maintain a list of all security realms from which it accepts security tokens. For each security realm from which the server maintains tokens, the server MUST maintain the following:

A URI to identify the security realm, which MUST be returned in the response in the trustUri element.

The Logon Service URL for the security realm, which MUST be returned in the response in the trustLsUrl element.

The display name of the security realm, which MUST be returned in the response in the trustDisplayName element.

For each security realm from which the server accepts security tokens, the server MUST return a trustType of "TrustedRealm".

The server MAY maintain a list of the accepted authentication methods of the security realm that are identified by URIs. If the server maintains this list, the list of URIs MUST be returned in the acceptableAuthenticationMethodStrings element. If the server does not maintain this list, the acceptableAuthenticationMethodStrings MUST be empty.<11>

3.2.5.2 LsRequestSecurityToken

The LsRequestSecurityToken exchange MUST consist of a single request message and a single response message. The exchange MUST be initiated by the client with a request message to the server. The following sections describe the server processing for the request and response messages.

3.2.5.2.1 LsRequestSecurityToken Request

When the server receives an LsRequestSecurityToken request message, it must respond to it as if it were an [MS-MWBF] request for a security token.

The server MUST consider the targetRealmName element as if it were the *wtrealm* parameter of the [MS-MWBF] request for a security token.

The credentialTypeUri and credentials elements MUST contain information about the method used at the client for authenticating the [MS-MWBF] web browser requestor. The client MAY use username and password authentication or SSL client certificate authentication.<12>

If SSL client certificate authentication was used, the *credentialTypeUri* parameter MUST be "urn:ietf:rfc:2246". If username and password authentication is used, the credentialTypeUri MUST be "urn:oasis:names:tc:SAML:1.0:am:password".

If SSL client certificate authentication was used, the credentials element MUST contain only two values. The first value MUST equal "Certificate". The value of the second string MUST be an X.509 certificate per [WSDL] that is Base64-encoded per [RFC4648].

If username and password authentication was used, the credential element MUST contain only four values. The value of the first string MUST be Username. The value of the second string MUST be a username for the web browser requestor. The value of the third string MUST be Password. The value of the fourth string MUST be a password for the web browser requestor. The credentials provided for the client MUST be used to generate a security token for the user as described in [MS-MWBF].

The client MAY specify an identifier for a particular account store to be used by the server when generating claims for the web browser requestor using the accountStoreUri element.<13>

The client MAY specify an [RFC2965] cookie value that is Base64-encoded per [RFC4648] in the cookie element of the request.<14>

3.2.5.2.2 LsRequestSecurityToken Response

LsRequestSecurityToken response processing can be divided into Status, PolicyVersion, CredentialsVerification, ForeignRealmUri, SecurityToken and LogonAcceleratorToken processing. The response MUST be adequate to be converted into an [MS-MWBF] sign-in response by the client. The following sections discuss these processing steps.

3.2.5.2.2.1 Status

If the security token is successfully generated, the Status value MUST be Success.

If there is an error attempting to generate the security token, the Status value MUST NOT be Success. The following table describes the meaning of various Status values.

Status value	Description
WrongPrincipal	A cookie was included that does not match the credentials or token.
NoAcceptableCredential	The credentials do not represent a directory account.
InvalidTarget	The targetRealmName of the request does not match a supported security realm.
ValidationFailure	The security token or cookie in the request could not be validated.
GenerationFailure	The claims could not be generated.
SidExpansionFailure	An internal error occurred with **Active Directory**.
NoAccountStores	No account store is configured.
NoActiveDirectoryForSids	An internal error occurred with Active Directory.
NoAccountStoresForCert	No account store is configured for the certificate from credentials.
Unset	An internal error occurred and the Status value was not set correctly.

3.2.5.2.2.2 PolicyVersion

As detailed in the LsRequestSecurityToken Response section, the response MUST contain a version number and GUID representing the configuration data described in the LsRequestSecurityToken, RequestSecurityTokenWithToken, and LsRequestSecurityTokenWithCookie sections. Similarly to a GetProxyTrustConfiguration response message, the Version element MUST be set to the version number for the current configuration maintained by the server. The Guid element MUST be set to the GUID for the current configuration maintained by the server.

3.2.5.2.2.3 CredentialsVerification

The information found within the CredentialsVerification structure is informational only, and the server MAY omit it.

The AccountStoreType value SHOULD be ActiveDirectoryType if Active Directory is used for generating claims in the security token returned. The AccountStoreType value SHOULD be LdapDirectoryType if an LDAP directory is used for generating the claims in the security token returned.

The AccountStoreTypeDisplay value SHOULD be a human readable string that identifies the type of account store. The AccountStoreUriString value SHOULD be a URI that uniquely identifies the account store at the server. The AccountStoreDisplayName value SHOULD be a human-readable string that identifies the account store at the server. Windows follows all SHOULD statements for the **CredentialsVerification** element.

The UserValidationData MUST contain an ErrorCode. The ErrorCode value MUST be 0 for a successful validation. When an error occurs, the ErrorCode value depends on the underlying account store used. The UserValidationData MAY contain an AdditionalValidationInfo element with further data.<15>

3.2.5.2.2.4 ForeignRealmUri

The ForeignRealmUri value MUST be the URI "urn:federation:self".

3.2.5.2.2.5 SecurityToken

This parameter MUST be a Base64-encoded [RFC4648] security token conforming to [MS-MWBF] section 2.2.4.2. This is the security token that the server would normally issue in the *wresult* parameter of [MS-MWBF]. The process for generating this value is specified in [MS-MWBF].

3.2.5.2.2.6 LogonAcceleratorToken

This parameter MUST be Base64-encoded [RFC4648] data used by the STS to cache information about the user as a [RFC2965] cookie. The protocol does not constrain the format of this data since it is written by the STS for later processing by the STS. STS implementations may use any data format desired.

3.2.5.3 RequestSecurityTokenWithToken

The RequestSecurityTokenWithToken exchange MUST consist of a single request message and a single response message. The exchange MUST be initiated by the client with a request message to the server. The following sections describe the server processing for the request and response messages.

3.2.5.3.1 RequestSecurityTokenWithToken Request

When the server receives an RequestSecurityTokenWithToken request message, it must respond to it as if it were an [MS-MWBF] request for a security token with a **wresult** populated.

The server MUST consider the targetRealmName element as if it were the *wtrealm* parameter of the [MS-MWBF] request for a security token.

The server MUST consider the inToken element as if it were a Base64-encoded version of the *wresult* parameter of the [MS-MWBF] request for a security token.

The client MAY specify an [RFC2965] cookie value that is Base64-encoded per [RFC4648] in the cookie element of the request.<16>

3.2.5.3.2 RequestSecurityTokenWithToken Response

RequestSecurityTokenWithToken response processing can be divided into Status, PolicyVersion, CredentialsVerification, ForeignRealmUri, SecurityToken and LogonAcceleratorToken processing. The response MUST be adequate to be converted into an [MS-MWBF] sign-in response by the client. The following sections discuss these processing steps.

3.2.5.3.2.1 Status

The server MUST process the Status element in an RequestSecurityTokenWithToken response as specified in the Status section.

3.2.5.3.2.2 PolicyVersion

The server MUST process the PolicyVersion element in a RequestSecurityTokenWithToken response as specified in the PolicyVersion section.

3.2.5.3.2.3 CredentialsVerification

The server MUST not include the CredentialsVerification element in the RequestSecurityTokenWithToken response.

3.2.5.3.2.4 ForeignRealmUri

The foreign realm URI MUST be the URI of the security realm that issued the security token received in the inToken element of the request.

3.2.5.3.2.5 SecurityToken

The server MUST generate the value of the SecurityToken element as specified in [MS-MWBF], treating the inToken value of the request as a *wresult* parameter of [MS-MWBF]. Once the security token is generated, the value MUST be Base64-encoded.

3.2.5.3.2.6 LogonAcceleratorToken

The server MUST process the LogonAcceleratorToken element in an RequestSecurityTokenWithToken response as specified in the LogonAcceleratorToken section.

3.2.5.4 LsRequestSecurityTokenWithCookie

The LsRequestSecurityTokenWithCookie exchange MUST consist of a single request message and a single response message. The exchange MUST be initiated by the client with a request message to the server. The following sections describe the server processing for the request and response messages.

3.2.5.4.1 LsRequestSecurityTokenWithCookie Request

When the server receives an LsRequestSecurityTokenWithCookie request message, it must respond to it as if it were an [MS-MWBF] request for a security token with an [RFC2965] cookie previously set by the server.

The server MUST consider the targetRealmName element as if it were the *wtrealm* parameter of the [MS-MWBF] request for a security token.

The client MUST specify an [RFC2965] cookie value that is Base64-encoded per [RFC4648] in the latToken element of the request.

If the authMethodUris element is present, the server MUST consider the list of URIs in the authMethodUris set as if it were the *wauth* parameter of the [MS-MWBF] request for a security token.

3.2.5.4.2 LsRequestSecurityTokenWithCookie Response

LsRequestSecurityTokenWithCookie response processing can be divided into Status, PolicyVersion, CredentialsVerification, ForeignRealmUri, SecurityToken and LogonAcceleratorToken processing. The response MUST be adequate to be converted into an [MS-MWBF] sign-in response by the client. The following sections discuss these processing steps.

3.2.5.4.2.1 Status

The server MUST process the Status element in an LsRequestSecurityTokenWithCookie response as specified in the Status section.

3.2.5.4.2.2 PolicyVersion

The server MUST process the PolicyVersion element in an LsRequestSecurityTokenWithCookie response as specified in the PolicyVersion section.

3.2.5.4.2.3 CredentialsVerification

The server MUST NOT include the CredentialsVerification element in the LsRequestSecurityTokenWithCookie response.

3.2.5.4.2.4 ForeignRealmUri

The server MUST not include a ForeignRealmUri element in the response.

3.2.5.4.2.5 SecurityToken

The server MUST generate the value of the <SecurityToken> element as specified in [MS-MWBF], using the data cached in the <latToken> element to generate the claims for the user. Once the security token is generated, the value MUST be Base64-encoded.

3.2.5.4.2.6 LogonAcceleratorToken

The server MUST not include a LogonAcceleratorToken element in the response.

3.2.6 Timer Events

There are no protocol-specific timer events that MUST be serviced by an implementation. This protocol does not require timers beyond those that may be used by the underlying transport to transmit and receive messages over HTTPS. The protocol does not include provisions for time-based retry for sending protocol messages.

3.2.7 Other Local Events

This protocol does not have dependencies on any transport protocols other than HTTP 1.1. This protocol relies on this transport mechanism for the correct and timely delivery of protocol messages. The protocol does not take action in response to any changes or failure in machine state or network communications.

4 Protocol Examples

4.1 Service WSDL

The following is a WSDL example describing a service that offers the protocol ([WSDL]). This particular service description also details operations from other protocols.

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tm="http://microsoft.com/wsdl/mime/textMatching/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/"
xmlns:s1="http://microsoft.com/wsdl/types/" xmlns:s="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/"
targetNamespace="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
  <wsdl:types>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/">
      <s:import namespace="http://microsoft.com/wsdl/types/" />
      <s:element name="LsRequestSecurityToken">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="credentialTypeUri" type="s:string"
/>
            <s:element minOccurs="0" maxOccurs="1" name="credentials"
type="tns:ArrayOfString" />
            <s:element minOccurs="0" maxOccurs="1" name="accountStoreUri" type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="cookie" type="s:base64Binary" />
            <s:element minOccurs="0" maxOccurs="1" name="targetRealmName" type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ArrayOfString">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="string" nillable="true"
type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:element name="LsRequestSecurityTokenResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="rstr" type="tns:RSTRResult" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="RSTRResult">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="Status" type="tns:RSTRStatus" />
          <s:element minOccurs="0" maxOccurs="1" name="PolicyVersion"
type="tns:VersionInformation" />
          <s:element minOccurs="0" maxOccurs="1" name="CredentialsVerification"
type="tns:CredentialsVerificationInfo" />
          <s:element minOccurs="0" maxOccurs="1" name="ForeignRealmUri" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="SecurityToken" type="s:base64Binary"
/>
```

```
          <s:element minOccurs="0" maxOccurs="1" name="LogonAcceleratorToken"
type="s:base64Binary" />
        </s:sequence>
      </s:complexType>
      <s:simpleType name="RSTRStatus">
        <s:restriction base="s:string">
          <s:enumeration value="Success" />
          <s:enumeration value="WrongPrincipal" />
          <s:enumeration value="NoAcceptableCredential" />
          <s:enumeration value="InvalidTarget" />
          <s:enumeration value="ValidationFailure" />
          <s:enumeration value="GenerationFailure" />
          <s:enumeration value="SidExpansionFailure" />
          <s:enumeration value="NoAccountStores" />
          <s:enumeration value="NoActiveDirectoryForSids" />
          <s:enumeration value="NoAccountStoresForCert" />
          <s:enumeration value="Unset" />
        </s:restriction>
      </s:simpleType>
      <s:complexType name="VersionInformation">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="SoftwareVersion" type="s:long" />
          <s:element minOccurs="1" maxOccurs="1" name="Guid" type="s1:guid" />
          <s:element minOccurs="1" maxOccurs="1" name="Version" type="s:long" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="CredentialsVerificationInfo">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="AccountStoreType"
type="tns:AccountStoreType" />
          <s:element minOccurs="0" maxOccurs="1" name="AccountStoreTypeDisplay"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="AccountStoreUriString" type="s:string"
/>
          <s:element minOccurs="0" maxOccurs="1" name="AccountStoreDisplayName"
type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="UserValidationData"
type="tns:UserValidationInfo" />
        </s:sequence>
      </s:complexType>
      <s:simpleType name="AccountStoreType">
        <s:restriction base="s:string">
          <s:enumeration value="ActiveDirectoryType" />
          <s:enumeration value="LdapDirectoryType" />
          <s:enumeration value="UnknownStoreType" />
        </s:restriction>
      </s:simpleType>
      <s:complexType name="UserValidationInfo">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="ErrorCode" type="s:long" />
          <s:element minOccurs="0" maxOccurs="1" name="AdditionalValidationInfo"
type="tns:ArrayOfString" />
        </s:sequence>
      </s:complexType>
      <s:element name="RequestSecurityTokenWithToken">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="inToken" type="s:base64Binary" />
            <s:element minOccurs="0" maxOccurs="1" name="cookie" type="s:base64Binary" />
```

```
            <s:element minOccurs="0" maxOccurs="1" name="targetRealmName" type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="RequestSecurityTokenWithTokenResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="rstr" type="tns:RSTRResult" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="LsRequestSecurityTokenWithCookie">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="latToken" type="s:base64Binary" />
            <s:element minOccurs="0" maxOccurs="1" name="targetRealmName" type="s:string" />
            <s:element minOccurs="0" maxOccurs="1" name="authMethodUris"
type="tns:ArrayOfString" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="LsRequestSecurityTokenWithCookieResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="rstr" type="tns:RSTRResult" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetProxyTrustConfiguration">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="proxyVersion"
type="tns:VersionInformation" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetProxyTrustConfigurationResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="GetProxyTrustConfigurationResult"
type="s:boolean" />
            <s:element minOccurs="0" maxOccurs="1" name="fsVersion"
type="tns:VersionInformation" />
            <s:element minOccurs="0" maxOccurs="1" name="proxyInformation"
type="tns:ProxyInformation" />
            <s:element minOccurs="0" maxOccurs="1" name="trustConfig"
type="tns:ArrayOfTrustConfigurationData" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ProxyInformation">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="HostedRealmUriStr" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="LsUrlStr" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="ConfigInfo"
type="tns:ProxyConfigurationInformation" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ProxyConfigurationInformation">
```

```
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="CookiePath" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="SuppressRealmCookie" type="s:boolean"
/>
          <s:element minOccurs="1" maxOccurs="1" name="RealmCookieLifetime" type="s:int" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="ArrayOfTrustConfigurationData">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="TrustConfigurationData"
nillable="true" type="tns:TrustConfigurationData" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="TrustConfigurationData">
        <s:sequence>
          <s:element minOccurs="1" maxOccurs="1" name="trustType" type="tns:TrustTypes" />
          <s:element minOccurs="1" maxOccurs="1" name="trustDisplayName" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="trustUri" type="s:string" />
          <s:element minOccurs="1" maxOccurs="1" name="trustLsUrl" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="acceptableAuthenticationMethodStrings"
type="tns:ArrayOfString" />
        </s:sequence>
      </s:complexType>
      <s:simpleType name="TrustTypes">
        <s:restriction base="s:string">
          <s:enumeration value="TrustedRealm" />
          <s:enumeration value="TrustingRealm" />
          <s:enumeration value="TrustingResource" />
          <s:enumeration value="SelfhostedRealm" />
          <s:enumeration value="UnknownTrustType" />
        </s:restriction>
      </s:simpleType>
      <s:element name="GetFsTrustInformation">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="wsVersion"
type="tns:VersionInformation" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetFsTrustInformationResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="GetFsTrustInformationResult"
type="s:boolean" />
            <s:element minOccurs="0" maxOccurs="1" name="fsVersion"
type="tns:VersionInformation" />
            <s:element minOccurs="0" maxOccurs="1" name="trustInfo"
type="tns:FsInformationData" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="FsInformationData">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="verificationMethod"
type="tns:X509VerificationMethod" />
          <s:element minOccurs="0" maxOccurs="1" name="certificates"
type="tns:FederationCertificates" />
          <s:element minOccurs="0" maxOccurs="1" name="fsDomainAccount" type="s:string" />
          <s:element minOccurs="0" maxOccurs="1" name="hostedRealmUri" type="s:string" />
```

```
          <s:element minOccurs="0" maxOccurs="1" name="lsUrl" type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="X509VerificationMethod">
        <s:complexContent mixed="false">
          <s:extension base="tns:VerificationMethod">
            <s:sequence>
              <s:element minOccurs="0" maxOccurs="1" name="TrustedCertificates"
type="tns:ArrayOfCertInfo" />
              <s:element minOccurs="1" maxOccurs="1" name="RevocationCheckFlags"
type="tns:RevocationFlags" />
            </s:sequence>
          </s:extension>
        </s:complexContent>
      </s:complexType>
      <s:complexType name="VerificationMethod" abstract="true" />
      <s:complexType name="ArrayOfCertInfo">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="CertInfo" nillable="true"
type="tns:CertInfo" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="CertInfo">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="X509Thumbprint" type="s:string" />
        </s:sequence>
      </s:complexType>
      <s:simpleType name="RevocationFlags">
        <s:restriction base="s:string">
          <s:enumeration value="None" />
          <s:enumeration value="CheckEndCert" />
          <s:enumeration value="CheckEndCertCacheOnly" />
          <s:enumeration value="CheckChain" />
          <s:enumeration value="CheckChainCacheOnly" />
          <s:enumeration value="CheckChainExcludeRoot" />
          <s:enumeration value="CheckChainExcludeRootCacheOnly" />
        </s:restriction>
      </s:simpleType>
      <s:complexType name="FederationCertificates">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="1" name="SerializedStore" type="s:base64Binary"
/>
        </s:sequence>
      </s:complexType>
      <s:element name="GetTrustedRealmUri">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="email" type="s:string" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:element name="GetTrustedRealmUriResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="GetTrustedRealmUriResult"
type="s:boolean" />
            <s:element minOccurs="0" maxOccurs="1" name="trustedRealmUri" type="s:string" />
          </s:sequence>
        </s:complexType>
```

```
      </s:element>
      <s:element name="GetClaims">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="1" maxOccurs="1" name="claimType" type="tns:ClaimType" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:simpleType name="ClaimType">
        <s:restriction base="s:string">
          <s:enumeration value="Group" />
          <s:enumeration value="Custom" />
          <s:enumeration value="GroupAndCustom" />
        </s:restriction>
      </s:simpleType>
      <s:element name="GetClaimsResponse">
        <s:complexType>
          <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="groupClaimCollection"
type="tns:ArrayOfGroupClaim" />
            <s:element minOccurs="0" maxOccurs="1" name="customClaimCollection"
type="tns:ArrayOfCustomClaim" />
          </s:sequence>
        </s:complexType>
      </s:element>
      <s:complexType name="ArrayOfGroupClaim">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="GroupClaim" nillable="true"
type="tns:GroupClaim" />
        </s:sequence>
      </s:complexType>
      <s:complexType name="GroupClaim" mixed="true">
        <s:complexContent mixed="false">
          <s:extension base="tns:TrustPolicyEntryBase">
            <s:attribute name="IsSensitive" type="s:boolean" use="required" />
          </s:extension>
        </s:complexContent>
      </s:complexType>
      <s:complexType name="TrustPolicyEntryBase">
        <s:attribute name="uuid" type="s1:guid" use="required" />
        <s:attribute name="Disabled" type="s:boolean" use="required" />
      </s:complexType>
      <s:complexType name="CustomClaim">
        <s:complexContent mixed="false">
          <s:extension base="tns:TrustPolicyEntryBase">
            <s:sequence>
              <s:element minOccurs="0" maxOccurs="1" name="CustomClaimName" type="s:string"
/>
            </s:sequence>
            <s:attribute name="IsSensitive" type="s:boolean" use="required" />
          </s:extension>
        </s:complexContent>
      </s:complexType>
      <s:complexType name="ActiveDirectoryGroupClaim">
        <s:complexContent mixed="true">
          <s:extension base="tns:GroupClaim">
            <s:sequence>
              <s:element minOccurs="0" maxOccurs="1" name="GroupSid" type="s:string" />
            </s:sequence>
```

```
          </s:extension>
        </s:complexContent>
      </s:complexType>
      <s:complexType name="ArrayOfCustomClaim">
        <s:sequence>
          <s:element minOccurs="0" maxOccurs="unbounded" name="CustomClaim" nillable="true"
type="tns:CustomClaim" />
        </s:sequence>
      </s:complexType>
    </s:schema>
    <s:schema elementFormDefault="qualified"
targetNamespace="http://microsoft.com/wsdl/types/">
      <s:simpleType name="guid">
        <s:restriction base="s:string">
          <s:pattern value="[0-9a-fA-F]{8}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-9a-fA-F]{4}-[0-
9a-fA-F]{12}" />
        </s:restriction>
      </s:simpleType>
    </s:schema>
  </wsdl:types>
  <wsdl:message name="LsRequestSecurityTokenSoapIn">
    <wsdl:part name="parameters" element="tns:LsRequestSecurityToken" />
  </wsdl:message>
  <wsdl:message name="LsRequestSecurityTokenSoapOut">
    <wsdl:part name="parameters" element="tns:LsRequestSecurityTokenResponse" />
  </wsdl:message>
  <wsdl:message name="RequestSecurityTokenWithTokenSoapIn">
    <wsdl:part name="parameters" element="tns:RequestSecurityTokenWithToken" />
  </wsdl:message>
  <wsdl:message name="RequestSecurityTokenWithTokenSoapOut">
    <wsdl:part name="parameters" element="tns:RequestSecurityTokenWithTokenResponse" />
  </wsdl:message>
  <wsdl:message name="LsRequestSecurityTokenWithCookieSoapIn">
    <wsdl:part name="parameters" element="tns:LsRequestSecurityTokenWithCookie" />
  </wsdl:message>
  <wsdl:message name="LsRequestSecurityTokenWithCookieSoapOut">
    <wsdl:part name="parameters" element="tns:LsRequestSecurityTokenWithCookieResponse" />
  </wsdl:message>
  <wsdl:message name="GetProxyTrustConfigurationSoapIn">
    <wsdl:part name="parameters" element="tns:GetProxyTrustConfiguration" />
  </wsdl:message>
  <wsdl:message name="GetProxyTrustConfigurationSoapOut">
    <wsdl:part name="parameters" element="tns:GetProxyTrustConfigurationResponse" />
  </wsdl:message>
  <wsdl:message name="GetFsTrustInformationSoapIn">
    <wsdl:part name="parameters" element="tns:GetFsTrustInformation" />
  </wsdl:message>
  <wsdl:message name="GetFsTrustInformationSoapOut">
    <wsdl:part name="parameters" element="tns:GetFsTrustInformationResponse" />
  </wsdl:message>
  <wsdl:message name="GetTrustedRealmUriSoapIn">
    <wsdl:part name="parameters" element="tns:GetTrustedRealmUri" />
  </wsdl:message>
  <wsdl:message name="GetTrustedRealmUriSoapOut">
    <wsdl:part name="parameters" element="tns:GetTrustedRealmUriResponse" />
  </wsdl:message>
  <wsdl:message name="GetClaimsSoapIn">
    <wsdl:part name="parameters" element="tns:GetClaims" />
  </wsdl:message>
```

```
  <wsdl:message name="GetClaimsSoapOut">
    <wsdl:part name="parameters" element="tns:GetClaimsResponse" />
  </wsdl:message>
  <wsdl:portType name="FederationServerServiceSoap">
    <wsdl:operation name="LsRequestSecurityToken">
      <wsdl:input message="tns:LsRequestSecurityTokenSoapIn" />
      <wsdl:output message="tns:LsRequestSecurityTokenSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="RequestSecurityTokenWithToken">
      <wsdl:input message="tns:RequestSecurityTokenWithTokenSoapIn" />
      <wsdl:output message="tns:RequestSecurityTokenWithTokenSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="LsRequestSecurityTokenWithCookie">
      <wsdl:input message="tns:LsRequestSecurityTokenWithCookieSoapIn" />
      <wsdl:output message="tns:LsRequestSecurityTokenWithCookieSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="GetProxyTrustConfiguration">
      <wsdl:input message="tns:GetProxyTrustConfigurationSoapIn" />
      <wsdl:output message="tns:GetProxyTrustConfigurationSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="GetFsTrustInformation">
      <wsdl:input message="tns:GetFsTrustInformationSoapIn" />
      <wsdl:output message="tns:GetFsTrustInformationSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="GetTrustedRealmUri">
      <wsdl:input message="tns:GetTrustedRealmUriSoapIn" />
      <wsdl:output message="tns:GetTrustedRealmUriSoapOut" />
    </wsdl:operation>
    <wsdl:operation name="GetClaims">
      <wsdl:input message="tns:GetClaimsSoapIn" />
      <wsdl:output message="tns:GetClaimsSoapOut" />
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="FederationServerServiceSoap" type="tns:FederationServerServiceSoap">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="LsRequestSecurityToken">
      <soap:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/LsRequestS
ecurityToken" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="RequestSecurityTokenWithToken">
      <soap:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/RequestSec
urityTokenWithToken" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="LsRequestSecurityTokenWithCookie">
```

```
      <soap:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/LsRequestS
ecurityTokenWithCookie" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetProxyTrustConfiguration">
      <soap:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/GetProxyTr
ustConfiguration" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetFsTrustInformation">
      <soap:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/GetFsTrust
Information" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetTrustedRealmUri">
      <soap:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/GetTrusted
RealmUri" style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetClaims">
      <soap:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/GetClaims"
style="document" />
      <wsdl:input>
        <soap:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:binding name="FederationServerServiceSoap12" type="tns:FederationServerServiceSoap">
    <soap12:binding transport="http://schemas.xmlsoap.org/soap/http" />
    <wsdl:operation name="LsRequestSecurityToken">
```

```
      <soap12:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/LsRequestS
ecurityToken" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="RequestSecurityTokenWithToken">
      <soap12:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/RequestSec
urityTokenWithToken" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="LsRequestSecurityTokenWithCookie">
      <soap12:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/LsRequestS
ecurityTokenWithCookie" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetProxyTrustConfiguration">
      <soap12:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/GetProxyTr
ustConfiguration" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetFsTrustInformation">
      <soap12:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/GetFsTrust
Information" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetTrustedRealmUri">
      <soap12:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/GetTrusted
RealmUri" style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
```

```
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
    <wsdl:operation name="GetClaims">
      <soap12:operation
soapAction="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/GetClaims"
style="document" />
      <wsdl:input>
        <soap12:body use="literal" />
      </wsdl:input>
      <wsdl:output>
        <soap12:body use="literal" />
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="FederationServerService">
    <wsdl:port name="FederationServerServiceSoap" binding="tns:FederationServerServiceSoap">
      <soap:address location="https://localhost/adfs/fs/federationserverservice.asmx" />
    </wsdl:port>
    <wsdl:port name="FederationServerServiceSoap12"
binding="tns:FederationServerServiceSoap12">
      <soap12:address location="https://localhost/adfs/fs/federationserverservice.asmx" />
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

4.2 GetProxyTrustConfiguration Request

```
<?xml version="1.0" encoding="utf-8"?>
- <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   - <soap:Body>
         - <GetProxyTrustConfiguration
xmlns="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/">
             - <proxyVersion>
                   - <SoftwareVersion>
                       1
                   </SoftwareVersion>
                   - <Guid>
                       00000000-0000-0000-0000-000000000000
                   </Guid>
                   - <Version>
                       1
                 </Version>
             </proxyVersion>
         </GetProxyTrustConfiguration>
    </soap:Body>
 </soap:Envelope>
```

4.3 GetProxyTrustConfiguration Response

```
<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <soap:Body>
            <GetProxyTrustConfigurationResponse
xmlns="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/">
            <GetProxyTrustConfigurationResult>true</GetProxyTrustConfigurationResult>
            <fsVersion>
                <SoftwareVersion>1</SoftwareVersion>
                <Guid>c8fbb077-6f57-43b9-a8c1-1884fe8813b5</Guid>
                <Version>27</Version>
            </fsVersion>
            <proxyInformation>
                <HostedRealmUriStr>urn:federation:trey research</HostedRealmUriStr>
                <LsUrlStr>https://DSP20A46.adfsrdomlh-
2.nttest.microsoft.com/adfs/ls/</LsUrlStr>
                <ConfigInfo>
                    <CookiePath>/adfs/ls</CookiePath>
                    <SuppressRealmCookie>false</SuppressRealmCookie>
                    <RealmCookieLifetime>30</RealmCookieLifetime>
                </ConfigInfo>
            </proxyInformation>
            <trustConfig>
                <TrustConfigurationData>
                    <trustType>SelfhostedRealm</trustType>
                    <trustDisplayName>Trey Research</trustDisplayName>
                    <trustUri>urn:federation:self</trustUri>
                    <trustLsUrl>https://DSP20A46.adfsrdomlh-
2.nttest.microsoft.com/adfs/ls/</trustLsUrl>
                </TrustConfigurationData>
                <TrustConfigurationData>
                    <trustType>TrustedRealm</trustType>
                    <trustDisplayName>Adatum</trustDisplayName>
                    <trustUri>urn:federation:adatum</trustUri>
                    <trustLsUrl>https://DSP20A52.adfsadomlh-
2.nttest.microsoft.com/adfs/ls/</trustLsUrl>
                </TrustConfigurationData>
                <TrustConfigurationData>
                    <trustType>TrustingRealm</trustType>
                    <trustDisplayName>test resource partner</trustDisplayName>
                    <trustUri>urn:federation:rpsts</trustUri>
                    <trustLsUrl>https://rpsts</trustLsUrl>
                </TrustConfigurationData>
                <TrustConfigurationData>
                    <trustType>TrustingResource</trustType>
                    <trustDisplayName>PKI Claims App</trustDisplayName>
                    <trustUri>https://dsp20a48.adfsrdomlh-
2.nttest.microsoft.com:8081/claims/</trustUri>
                    <acceptableAuthenticationMethodStrings />
                </TrustConfigurationData>
            </trustConfig>
        </GetProxyTrustConfigurationResponse>
    </soap:Body>
</soap:Envelope>
```

4.4 LsRequestSecurityToken Request

```
<?xml version="1.0" encoding="utf-8"?>
```

```
- <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    - <soap:Body>
        - <LsRequestSecurityToken
xmlns="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/">
            - <credentialTypeUri>
                urn:oasis:names:tc:SAML:1.0:am:password
            </credentialTypeUri>
            - <credentials>
                - <string>
                    Username
                </string>
                - <string>
                    testdomain\testuser
                </string>
                - <string>
                    Password
                </string>
                - <string>
                    testpassword
                </string>
            </credentials>
            - <targetRealmName>
                urn:federation:rpsts
            </targetRealmName>
        </LsRequestSecurityToken>
    </soap:Body>
</soap:Envelope>
```

4.5 LsRequestSecurityToken Response

```
<?xml version="1.0" encoding="utf-8"?>
- <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    - <soap:Body>
        - <LsRequestSecurityTokenResponse
xmlns="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/">
            - <rstr>
                - <Status>
                    Success
                </Status>
                - <PolicyVersion>
                    - <SoftwareVersion>
                        1
                    </SoftwareVersion>
                    - <Guid>
                        c8fbb077-6f57-43b9-a8c1-1884fe8813b5
                    </Guid>
                    - <Version>
                        27
                    </Version>
                </PolicyVersion>
                - <CredentialsVerification>
                    - <AccountStoreType>
                        ActiveDirectoryType
                    </AccountStoreType>
```

```
                    - <AccountStoreTypeDisplay>
                        Active Directory
                    </AccountStoreTypeDisplay>
                    - <AccountStoreUriString>
                        urn:federation:activedirectory
                    </AccountStoreUriString>
                    - <AccountStoreDisplayName>
                        Active Directory
                    </AccountStoreDisplayName>
                    - <UserValidationData>
                        - <ErrorCode>
                            0
                        </ErrorCode>
                    </UserValidationData>
                </CredentialsVerification>
                - <ForeignRealmUri>
                    urn:federation:self
                </ForeignRealmUri>
                - <SecurityToken>
                    QBlAHMAVABvAD4APAAvAHcAcwB0ADoAUgBlAHEAdQBlAHMAdABTAG.......
                    Base 64 encoded token ... AHMAZQA+AA==
                </SecurityToken>
<LogonAcceleratorToken>
                    PABzAGEAbQBsADoAQQBzAHMAZQByA... Base 64 encoded token .... BjAG8AbQA=
                </LogonAcceleratorToken>
            </rstr>
        </LsRequestSecurityTokenResponse>
    </soap:Body>
</soap:Envelope>
```

4.6 RequestSecurityTokenWithToken Request

```
<?xml version="1.0" encoding="utf-8"?>
- <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    - <soap:Body>
        - <RequestSecurityTokenWithToken
xmlns="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/">
            - <inToken>
                zAHQAcgBhAHQAbwByAEAAZABkAHMAeQBzADMANwBhADEAMAAuAGMAbwBtAA==
            -- Base 64 encoded token ------

AQBjADEANABuACMAIgAgAC8APgA8AFMAaQBnAG4AYQB0AHUAcgBlAE0AZQB0AGgAbwBkACAAQQBsAGcAbwBy
            </inToken>
            <targetRealmName>https://dsp20a48.adfsrdomlh-
2.nttest.microsoft.com:8081/claims/</targetRealmName>
        </RequestSecurityTokenWithToken>
    </soap:Body>
</soap:Envelope>
```

4.7 RequestSecurityTokenWithToken Response

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

```
    <soap:Body>
        <RequestSecurityTokenWithTokenResponse
xmlns="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/">
            - <rstr>
                - <Status>
                    Success
                </Status>
                - <PolicyVersion>
                    - <SoftwareVersion>
                        1
                    </SoftwareVersion>
                    - <Guid>
                        c8fbb077-6f57-43b9-a8c1-1884fe8813b5
                    </Guid>
                    - <Version>
                        31
                    </Version>
                </PolicyVersion>
                - <ForeignRealmUri>
                    urn:federation:dsp20a52
                </ForeignRealmUri>
                - <SecurityToken>

PAB3AHMAdAA6AFIAZQBxAHUAZQBzAHQAUwBlAGMAdQByAGkAdAB5AFQAbwBrAGUAbgBSAGUAcwBwAG8AbgBzAGUAIAB4A
G0AbABuAHMAOgB3AHMAdAA9ACIAaAB0AHQAcAA6AC8ALwBzAGMAaABlAG0AYQBzAC4AeABtAGwAcwBvAGEAcAAuAG8Acg
BnAC8AdwBzAC8AMgAwADAANQAvADAAMgAvAHQAcgB1AHMAdAAiAD4APAB3AHMAdAA6AF
                    AcwBwAG8AbgBzAGUAPgA=
                </SecurityToken>
<LogonAcceleratorToken>
                    PABzAGEAbQBsADoAQQBzAHMAZQByA... Base 64 encoded token .... BjAG8AbQA=
                </LogonAcceleratorToken>
            </rstr>
        </RequestSecurityTokenWithTokenResponse>
    </soap:Body>
</soap:Envelope>
```

4.8 LsRequestSecurityTokenWithCookie Request

```
<?xml version="1.0" encoding="utf-8"?>
- <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    - <soap:Body>
        - <LsRequestSecurityTokenWithCookie
xmlns="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/">
            - <latToken>
            PABzAGEAbQBsADoAQQBzAHMAZQByAHQAaQBv
            -- base 64 encoded token ------
            dABpAG8AbgBJAEQAPQAiAF8AYwA3ADcAMAAxADIAOQAwAC0ANgBlAGUA
            AaQBzAHQAcgBhAHQAbwByAEAAZABkAHMAeQBzADMANwBhADEAMAAuAGMAbwBtAA==
        </latToken>
            <targetRealmName>https://dsp20a48.adfsrdomlh-2.nttest.microsoft.com:8081/claims/
            </targetRealmName>
        </LsRequestSecurityTokenWithCookie>
    </soap:Body>
</soap:Envelope>
```

4.9 LsRequestSecurityTokenWithCookie Response

```
<?xml version="1.0" encoding="utf-8"?>
- <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    - <soap:Body>
        - <LsRequestSecurityTokenWithCookieResponse
xmlns="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/">
            - <rstr>
                - <Status>
                    Success
                </Status>
                - <PolicyVersion>
                    - <SoftwareVersion>
                        1
                    </SoftwareVersion>
                    - <Guid>
                        c8fbb077-6f57-43b9-a8c1-1884fe8813b5
                    </Guid>
                    - <Version>
                        31
                    </Version>
                </PolicyVersion>
                - <SecurityToken>
                    PAB3AHMAdAA6AFIAZQBxAHUAZQBzAHQAUwBlAGMAdQByAGkAdAB5AFQ
                    --- Base 64 encoded token -------------
                    AbwBrAGUAbgBSAGUAcwBwAG8AbgBzAGUAIAB4AG0AbABuAHMAOgB3AHMAdAA
                    BzAHQAUwBlAGMAdQByAGkAdAB5AFQAbwBrAGUAbgBSAGUAcwBwAG8AbgBzAGUAPgA=
                </SecurityToken>
            </rstr>
            </LsRequestSecurityTokenWithCookieResponse>
    </soap:Body>
</soap:Envelope>
```

5 Security

5.1 Security Considerations for Implementers

Implementers need to ensure that SSL is used to authenticate that the server is the intended server referred to by the server endpoint URL. Implementers also need to ensure that the client role authenticates to the server role such that the server can trust the client to perform SSL client certificate authentication where appropriate. Otherwise there are no specific security considerations beyond those specified in normative references.

5.2 Index of Security Parameters

None.

6 Appendix A: 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:

- Windows Server 2003 R2 operating system
- Windows Server 2008 operating system
- Windows Server 2008 R2 operating system
- Windows Server 2012 operating system
- Windows Server 2012 R2 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 3.1.3.1: After the data described in the GetProxyTrustConfiguration section is obtained for the first time via a GetProxyTrustConfiguration exchange, Windows maintains a cached copy of the data described in the GetProxyTrustConfiguration section.

<2> Section 3.1.4.1: Windows sends a GetProxyTrustConfiguration request when the client service is started, and caches the data from the response. If the cached version of the data described in section GetProxyTrustConfiguration is not available when a security token request is received, Windows will attempt to obtain the data again by sending a GetProxyTrustConfiguration request upon receipt of the security token.

<3> Section 3.1.4.2: Windows always emits an <LsRequestSecurityTokenWithCookie> request message when the web browser requestor presents a session cookie, given that no token has been posted in the *wresult* parameter described by [MS-MWBF].

<4> Section 3.1.4.4: Windows always emits an LsRequestSecurityTokenWithCookie request message when the web browser requestor presents a session cookie, given that no token has been posted in the *wresult* parameter described by [MS-MWBF].

<5> Section 3.1.5.1.2.4: Windows displays the security realm display names to web browser requestors.

<6> Section 3.1.5.2.1: Without modifying the code that ships with this component, Windows can only perform username and password authentication at the client component of this protocol.

<7> Section 3.1.5.2.1: Windows never specifies a particular account store identifier.

<8> Section 3.1.5.2.1: Windows will never emit a cookie value in the LsRequestSecurityToken request, because the presence of a session cookie in the request from the web browser requestor will cause Windows to emit a LsRequestSecurityTokenWithCookie message.

<9> Section 3.1.5.2.2.1: Windows uses the Status value and CredentialsVerification values to populate an error message for displaying to the web browser requestor.

<10> Section 3.1.5.2.2.2: Windows emits a GetProxyTrustConfiguration request if the policy version specified in the response does not match the locally cached version.

<11> Section 3.2.5.1.2.4: Windows allows administrators to enter the accepted authentication methods for a security realm. By default, all methods are acceptable and Windows returns an empty list for the acceptableAuthenticationMethodStrings element.

<12> Section 3.2.5.2.1: In the server role, Windows supports both SSL client certificate authentication as well as username and password authentication.

<13> Section 3.2.5.2.1: If an identifier is specified, and the identifier correctly identifies a configured account store, Windows will honor the request by using only that account store to generate claims about the user.

<14> Section 3.2.5.2.1: If a cookie previously issued by the server is included in the request, Windows will validate that the cookie matches the credentials presented. If the validation fails, Windows will issue a status of "WrongPrincipal". If the validation succeeds, Windows will use the cookie contents to generate the claims as a performance optimization to avoid the account store.

<15> Section 3.2.5.2.2.3: Windows does not include the AdditionalValidationInfo element if the user validation was successful. If there was an error in validating the user, any string that represents the cause of the failure MAY be included here.

<16> Section 3.2.5.3.1: If a cookie previously issued by the server is included in the request, Windows will validate that the cookie matches the credentials presented. If the validation fails, Windows will issue a status of "WrongPrincipal". If the validation succeeds, Windows will use the cookie contents to generate the claims as a performance optimization to avoid the account store.

7 Change Tracking

This section identifies changes that were made to the [MS-ADFSPP] protocol document between the January 2013 and August 2013 releases. 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.
- An extensive rewrite, addition, or deletion of major portions of content.
- The removal of a document from the documentation set.
- Changes made for template compliance.

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 language and 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 or language changes were introduced. The technical content of the document is identical to the last released version, but minor editorial and formatting changes, as well as updates to the header and footer information, and to the revision summary, may have been made.

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.
- New content added for template compliance.
- Content updated for template compliance.
- Content removed for template compliance.
- 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 protocol@microsoft.com.

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
6 Appendix A: Product Behavior	Modified this section to include references to Windows Server 2012 R2 operating system.	Y	Content updated.

8 Index

R

S

T

V