[MS-OCAUTHWS]:

OC Authentication Web Service Protocol

Revision Summary

Date	Revision History	Revision Class	Comments
3/31/2010	0.1	Major	Initial Availability
4/30/2010	0.2	Editorial	Revised and edited the technical content
6/7/2010	0.3	Editorial	Revised and edited the technical content
6/29/2010	0.4	Editorial	Changed language and formatting in the technical content.
7/23/2010	0.4	No Change	No changes to the meaning, language, or formatting of the technical content.
9/27/2010	1.0	Major	Significantly changed the technical content.
11/15/2010	1.0	No Change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	1.0	No Change	No changes to the meaning, language, or formatting of the technical content.
3/18/2011	1.0	No Change	No changes to the meaning, language, or formatting of the technical content.
6/10/2011	1.0	No Change	No changes to the meaning, language, or formatting of the technical content.
1/20/2012	2.0	Major	Significantly changed the technical content.
4/11/2012	2.0	No Change	No changes to the meaning, language, or formatting of the technical content.
7/16/2012	2.0	No Change	No changes to the meaning, language, or formatting of the technical content.
10/8/2012	2.0	No Change	No changes to the meaning, language, or formatting of the technical content.
2/11/2013	2.0	No Change	No changes to the meaning, language, or formatting of the technical content.
7/30/2013	2.0	No Change	No changes to the meaning, language, or formatting of the technical content.
11/18/2013	2.1	Minor	Clarified the meaning of the technical content.
2/10/2014	2.1	No Change	No changes to the meaning, language, or formatting of the technical content.
4/30/2014	2.2	Minor	Clarified the meaning of the technical content.
7/31/2014	2.3	Minor	Clarified the meaning of the technical content.
10/30/2014	2.4	Minor	Clarified the meaning of the technical content.
3/30/2015	3.0	Major	Significantly changed the technical content.

Table of Contents

1 Introduction

The OC Authentication Web Service Protocol defines the message formats, server behavior, and client behavior for the purposes of authentication and certificate enrollment.

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 [RFC2119]. Sections 1.5 and 1.9 are also normative but do not contain those terms. All other sections and examples in this specification are informative.

1.1 Glossary

The following terms are specific to this document:

authentication: The act of proving an identity to a server while providing key material that binds the identity to subsequent communications.

base64 encoding: A binary-to-text encoding scheme whereby an arbitrary sequence of bytes is converted to a sequence of printable ASCII characters, as described in [RFC4648].

certificate: (1) A certificate is a collection of attributes (1) and extensions that can be stored persistently. The set of attributes in a certificate can vary depending on the intended usage of the certificate. A certificate securely binds a public key to the entity that holds the corresponding private key. A certificate is commonly used for **authentication** and secure exchange of information on open networks, such as the Internet, extranets, and intranets. Certificates are digitally signed by the issuing **certification authority (CA)** and can be issued for a user, a computer, or a service. The most widely accepted format for certificates is defined by the ITU-T X.509 version 3 international standards. For more information about attributes and extensions, see [RFC3280] and [X509] sections 7 and 8.

(2) When referring to X.509v3 certificates, that information consists of a public key, a distinguished name (DN) (3) of some entity assumed to have control over the private key corresponding to the **public key** in the certificate, and some number of other attributes and extensions assumed to relate to the entity thus referenced. Other forms of certificates can bind other pieces of information.

certificate chain: A sequence of **certificates (1)**, where each certificate in the sequence is signed by the subsequent certificate. The last certificate in the chain is normally a self-signed certificate.

certification: The **certificate (1)** request and issuance process whereby an end entity (EE) first makes itself known to a **certification authority (CA)** (directly, or through a registration authority) through the submission of a certificate enrollment request, prior to that **CA** issuing a **certificate (1)** or **certificates (1)** for that EE.

certification authority (CA): A third party that issues **public key certificates (1)**. Certificates serve to bind public keys to a user identity. Each user and certification authority (CA) can decide whether to trust another user or CA for a specific purpose, and whether this trust should be transitive. For more information, see [RFC3280].

endpoint: A device that is connected to a computer network.

fully qualified domain name (FQDN): An unambiguous domain name (2) that gives an absolute location in the Domain Name System's (DNS) hierarchy tree, as defined in [RFC1035] section 3.1 and [RFC2181] section 11.

globally unique identifier (GUID): A term used interchangeably with universally unique identifier (UUID) in Microsoft protocol technical documents (TDs). Interchanging the usage of these terms does not imply or require a specific algorithm or mechanism to generate the value.

Specifically, the use of this term does not imply or require that the algorithms specified in [RFC4122] or [C706] must be used for generating the **GUID**. See also universally unique identifier (UUID).

Hypertext Transfer Protocol (HTTP): An application-level protocol for distributed, collaborative, hypermedia information systems (text, graphic images, sound, video, and other multimedia files) on the World Wide Web.

Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS): An extension of **HTTP** that securely encrypts and decrypts webpage requests.

Integrated Windows authentication: A configuration setting that enables negotiation of **authentication** protocols in Internet Information Services (IIS). Integrated Windows authentication is more secure than Basic authentication, because the user name and password are hashed instead of plaintext.

Kerberos: An **authentication** system that enables two parties to exchange private information across an otherwise open network by assigning a unique key (called a ticket) to each user that logs on to the network and then embedding these tickets into messages sent by the users. For more information, see [MS-KILE].

NT LAN Manager (NTLM) Authentication Protocol: A protocol using a challenge-response mechanism for **authentication** in which clients are able to verify their identities without sending a password to the server. It consists of three messages, commonly referred to as Type 1 (negotiation), Type 2 (challenge) and Type 3 (authentication). For more information, see [MS-NLMP].

private key: One of a pair of keys used in public-key cryptography. The private key is kept secret and is used to decrypt data that has been encrypted with the corresponding public key. For an introduction to this concept, see [CRYPTO] section 1.8 and [IEEE1363] section 3.1.

proxy: A computer, or the software that runs on it, that acts as a barrier between a network and the Internet by presenting only a single network address to external sites. By acting as a go-between that represents all internal computers, the proxy helps protects network identities while also providing access to the Internet.

public key: One of a pair of keys used in public-key cryptography. The public key is distributed freely and published as part of a digital certificate. For an introduction to this concept, see [CRYPTO] section 1.8 and [IEEE1363] section 3.1.

Security Assertion Markup Language (SAML): The set of specifications that describe security assertions encoded in XML, profiles for attaching assertions to protocols and frameworks, request/response protocols used to obtain assertions, and the protocol bindings to transfer protocols, such as **SOAP** and HTTP.

security association (SA): A simplex "connection" that provides security services to the traffic carried by it. See [RFC4301] for more information.

security token: An opaque message or data packet produced by a Generic Security Services (GSS)-style **authentication** package and carried by the application protocol. The application has no visibility into the contents of the token.

security token service (STS): A web service that issues claims (2) and packages them in encrypted security tokens.

server: A replicating machine that sends replicated files to a partner (client). The term "server" refers to the machine acting in response to requests from partners that want to receive replicated files.

Session Initiation Protocol (SIP): An application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. **SIP** is defined in [RFC3261].

SOAP: A lightweight protocol for exchanging structured information in a decentralized, distributed environment. **SOAP** uses XML technologies to define an extensible messaging framework, which provides a message construct that can be exchanged over a variety of underlying protocols. The framework has been designed to be independent of any particular programming model and other implementation-specific semantics. SOAP 1.2 supersedes SOAP 1.1. See [SOAP1.2-1/2003].

SOAP fault: A container for error and status information within a **SOAP message**. See [SOAP1.2-1/2007] section 5.4 for more information.

SOAP message: An XML document consisting of a mandatory SOAP envelope, an optional SOAP header, and a mandatory SOAP body. See [SOAP1.2-1/2007] section 5 for more information.

Transport Layer Security (TLS): A security protocol that supports confidentiality and integrity of messages in client and server applications communicating over open networks. **TLS** supports server and, optionally, client authentication by using X.509 certificates (as specified in [X509]). **TLS** is standardized in the IETF TLS working group. See [RFC4346].

Uniform Resource Identifier (URI): A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [RFC3986].

Uniform Resource Locator (URL): A string of characters in a standardized format that identifies a document or resource on the World Wide Web. The format is as specified in [RFC1738].

user agent server (UAS): A logical entity that generates a response to a **Session Initiation Protocol (SIP)** request. The response either accepts, rejects, or redirects the request. The role of the UAS lasts only for the duration of that transaction. If a process responds to a request, it acts as a UAS for that transaction. If it initiates a request later, it assumes the role of a user agent client (UAC) for that transaction.

web application: A software application that uses **HTTP** as its core communication protocol and delivers information to the user by using web-based languages such as HTML and XML.

web service: A unit of application logic that provides data and services to other applications and can be called by using standard Internet transport protocols such as **HTTP**, Simple Mail Transfer Protocol (SMTP), or File Transfer Protocol (FTP). Web services can perform functions that range from simple requests to complicated business processes.

Web Services Description Language (WSDL): An XML format for describing network services as a set of endpoints that operate on messages that contain either document-oriented or procedure-oriented information. The operations and messages are described abstractly and are bound to a concrete network protocol and message format in order to define an endpoint. Related concrete endpoints are combined into abstract endpoints, which describe a network service. WSDL is extensible, which allows the description of endpoints and their messages regardless of the message formats or network protocols that are used.

web ticket: A security token that is sent by a protocol client to a web application during **authentication**. The security token can be included in either the body or the header of an HTTP message.

WSDL message: An abstract, typed definition of the data that is communicated during a **WSDL operation**, as described in [WSDL].

WSDL operation: A single action or function of a web service. The execution of a WSDL operation typically requires the exchange of messages between the service requestor and the service provider.

X.509: An ITU-T standard for public key infrastructure subsequently adapted by the IETF, as specified in [RFC3280].

XML namespace: A collection of names that is used to identify elements, types, and attributes in XML documents identified in a URI reference [RFC3986]. A combination of XML namespace and local name allows XML documents to use elements, types, and attributes that have the same names but come from different sources. For more information, see [XMLNS-2ED].

XML schema: A description of a type of XML document that is typically expressed in terms of constraints on the structure and content of documents of that type, in addition to the basic syntax constraints that are imposed by XML itself. An XML schema provides a view of a document type at a relatively high level of abstraction.

XML schema definition (XSD): The World Wide Web Consortium (W3C) standard language that is used in defining XML schemas. Schemas are useful for enforcing structure and constraining the types of data that can be used validly within other XML documents. XML schema definition refers to the fully specified and currently recommended standard for use in authoring **XML schemas**.

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

1.2 References

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.

[IETFDRAFT-OAuth2.0] Hammer-Lahav, E., Ed., Recordon, D., and Hardt, D., "The OAuth 2.0 Authorization Protocol", draft-ietf-oauth-v2-22, http://tools.ietf.org/html/draft-ietf-oauth-v2-23

[MS-OAUTH2EX] Microsoft Corporation, "OAuth 2.0 Authentication Protocol Extensions".

[MS-OCER] Microsoft Corporation, "Client Error Reporting Protocol".

[MS-SIPAE] Microsoft Corporation, "Session Initiation Protocol (SIP) Authentication Extensions".

[MS-SIPRE] Microsoft Corporation, "Session Initiation Protocol (SIP) Routing Extensions".

[MS-WCCE] Microsoft Corporation, "Windows Client Certificate Enrollment Protocol".

[MS-WSPOL] Microsoft Corporation, "Web Services: Policy Assertions and WSDL Extensions".

[MS-WSTEP] Microsoft Corporation, "WS-Trust X.509v3 Token Enrollment Extensions".

[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

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000, http://www.rfc-editor.org/rfc/rfc2818.txt

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and Schooler, E., "SIP: Session Initiation Protocol", RFC 3261, June 2002, http://www.ietf.org/rfc/rfc3261.txt

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

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

[SAMLCore] Maler, E., Mishra, P., Philpott, R., et al., "Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML) V1.1", September 2003, http://www.oasis-open.org/committees/download.php/3406/oasis-sstc-saml-core-1.1.pdf

[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] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation, June 2003, http://www.w3.org/TR/2003/REC-soap12-part1-20030624

[SOAP1.2/2] Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., and Nielsen, H.F., "SOAP Version 1.2 Part 2: Adjuncts", W3C Recommendation, June 2003, http://www.w3.org/TR/2003/REC-soap12-part2-20030624

[WS-MetaDataExchange] Ballinger, K. et al., "Web Services Metadata Exchange (WS-MetadataExchange) Version 1.1", August 2006, http://specs.xmlsoap.org/ws/2004/09/mex/WS-MetadataExchange.pdf

[WS-Trust1.3] Nadalin, A., Goodner, M., Gudgin, M., Barbir, A., Granqvist, H., "WS-Trust 1.3", OASIS Standard 19 March 2007, http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.html

[WSA1.0 Core] Gudgin, M., Ed., Hadley, M., Ed., and Rogers, Tony, Ed., "Web Services Addressing 1.0 - Core", W3C Recommendation 9 May 2006, http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/ws-addr-core.pdf

[WSA1.0 Metadata] Gudgin, M., Ed., Hadley, M., Ed., Rogers, T., Ed., Yalcinalp, U., Ed., "Web Services Addressing 1.0 - Metadata", W3C Recommendation, September 2007, http://www.w3.org/TR/2007/REC-ws-addr-metadata-20070904

[WSA1.0] World Wide Web Consortium, "Web Services Addressing 1.0 - WSDL Binding", W3C Candidate Recommendation, May 2006, http://www.w3.org/TR/2006/CR-ws-addr-wsdl-20060529/

[WSDLSOAP] Angelov, D., Ballinger, K., Butek, R., et al., "WSDL 1.1 Binding Extension for SOAP 1.2", W3C Member Submission, April 2006, http://www.w3.org/Submission/2006/SUBM-wsdl11soap12-20060405/

[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

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

[WSSE 1.0] Nadalin, A., Kaler, C., Hallam-Baker, P., and Monzillo, R., Eds., "Web Services Security: SOAP Message Security 1.0 (WS-Security 2004)", OASIS Standard 200401, March 2004, http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf

[WSSP1.2-2012] OASIS, "WS-SecurityPolicy 1.2", April 2012, http://docs.oasis-open.org/ws-sx/ws-securitypolicy/v1.2/ws-securitypolicy.pdf

[WSSX509TP] OASIS Standard, "Web Services Security X.509 Certificate Token Profile", March 2004, http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf

[WSS] OASIS, "Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)", February 2006, http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf

[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., 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., Ed. and Malhotra, A., Ed., "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-OCDISCWS] Microsoft Corporation, "Lync Autodiscover Web Service Protocol".

[RFC2315] Kaliski, B., "PKCS #7: Cryptographic Message Syntax Version 1.5", RFC 2315, March 1998, http://www.ietf.org/rfc/rfc2315.txt

[RFC2986] Nystrom, M. and Kaliski, B., "PKCS#10: Certificate Request Syntax Specification", RFC 2986, November 2000, http://www.ietf.org/rfc/rfc2986.txt

[RFC5280] Cooper, D., Santesson, S., Farrell, S., et al., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008, http://www.ietf.org/rfc/rfc5280.txt

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 5652, September 2009, http://www.rfc-editor.org/rfc/rfc5652.txt

1.3 Protocol Overview (Synopsis)

This protocol can be used to generate a **security token**, which can subsequently be used for **authentication** with other services. This protocol also allows a protocol client to request **X.509** v3 **certificates (2)**, which can subsequently be used for certificate-based authentication.

This protocol is used by the Web Ticket Service, described in section 1.3.1, by the Certificate Provisioning Service, described in section 1.3.2, and by the Authentication Broker Service, described in section 1.3.3.

1.3.1 Web Ticket Service

The Web Ticket Service is a **security token service (STS)**. The type of credentials that a client presents to the Web Ticket Service is described in section 3.2. The security token returned in the response is called a **Web ticket**.

The client presents the Web ticket as its credentials when authenticating to certain **Web applications**. See the individual Web application specifications for details, for example, Lync Autodiscover Web Service described in [MS-OCDISCWS] or Certificate Provisioning Service described in this document. The Web ticket can be presented in the body of the **Hypertext Transfer Protocol (HTTP)** message or in the HTTP header, depending on the type of Web application.

1.3.1.1 Web Service Web Applications

The following figure illustrates this protocol for Web applications that are **Web services**.

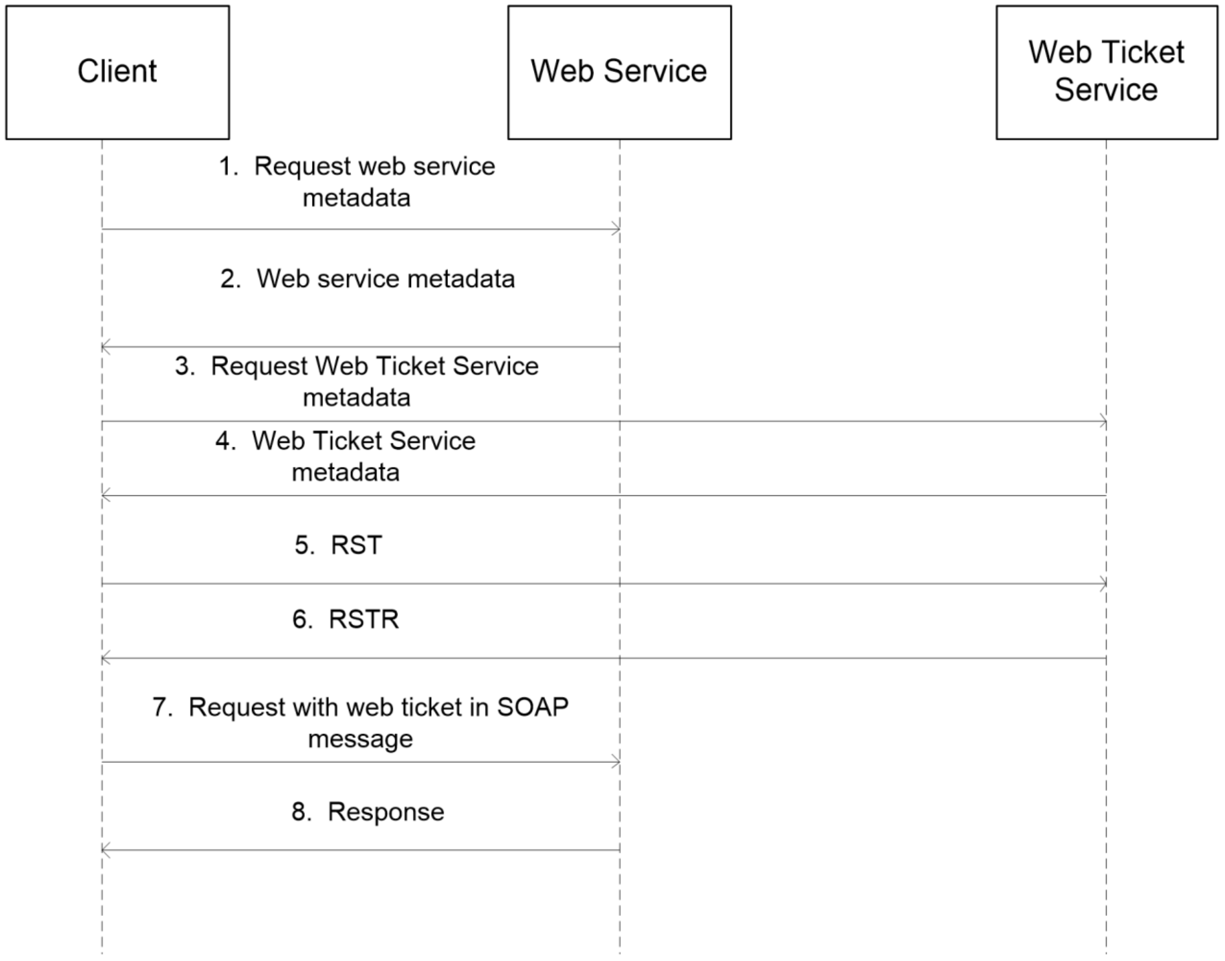
Client
Web Service
Web Ticket Service
1. Request web service metadata
2. Web service metadata
3. Request Web Ticket Service metadata
4. Web Ticket Service metadata
5. RST
6. RSTR
7. Request with web ticket in SOAP message
8. Response

Figure 1: This protocol for Web service Web applications

1. The client requests the metadata for the Web service using WS Metadata Exchange protocol as described in [WS-MetaDataExchange].
2. The Web service metadata is returned. The client discovers the **Uniform Resource Locator (URL)** of the Web Ticket Service. See details in section 3.2.
3. The client requests the metadata for the Web Ticket Service.
4. The Web Ticket Service metadata is returned. The following authentication types can be associated with the bindings in the metadata: **Integrated Windows authentication**, OCS-signed certificate authentication, and Live ID authentication. For details, see section 3.2.
5. The client sends an **RST** (Request Security Token). For details, see section 3.2.4.1.1.1.
6. The Web Ticket Service responds with an **RSTR** (Request Security Token Response). For details, see section 3.2.4.1.1.2.
7. The client sends a request to the Web service, with the Web ticket attached. For details, see section 3.2.

8. The Web service sends a response.

1.3.1.2 Non-Web Service Web Applications

The following figure illustrates this protocol for Web applications that are non-Web services.

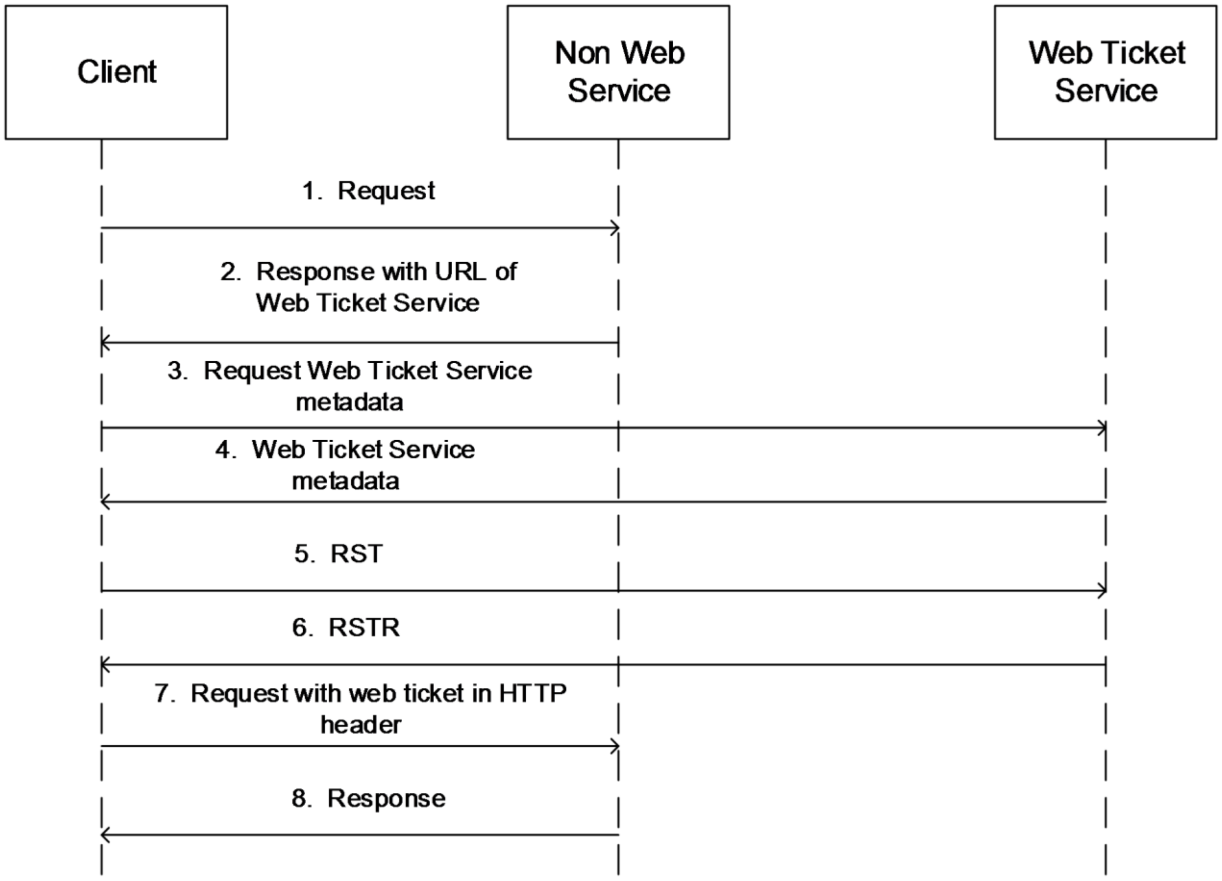
Client
Non Web Service
Web Ticket Service
1. Request
2. Response with URL of Web Ticket Service
3. Request Web Ticket Service metadata
4. Web Ticket Service metadata
5. RST
6. RSTR
7. Request with web ticket in HTTP header
8. Response

Figure 2: This protocol for non-Web service Web applications

1. The client sends a GET or POST HTTP request to the non-Web service Web application with content defined by the requirements of that application.

2. A response with status code 401 and a HTTP header containing the URL of the Web Ticket Service. For details, see section 3.2.

3. The client requests the Web Ticket Service's metadata using WS Metadata Exchange protocol as described in [WS-MetaDataExchange].

4. The Web Ticket Service metadata is returned. The following authentication types can be associated with the bindings in the metadata: Integrated Windows authentication, OCS-signed certificate authentication, and Live ID authentication. For details, see section 3.2.

5. The client sends an **RST** (Request Security Token). For details, see section 3.2.4.1.1.1.

6. The Web Ticket Service responds with a **RSTR** (Request Security Token Response). For details, see section 3.2.4.1.1.2.

7. The client sends a request to the non-Web service Web application, with the Web ticket in an HTTP header. For details, see section 3.2.

8. The Web service sends a response.

1.3.2 Certificate Provisioning Service

The Certificate Provisioning Service provides an X.509 v3 certificate (2) for the authenticated user to the client. The client can use the obtained certificate (2) for authentication against other services. One example of an authentication mechanism that uses this certificate (2) can be found in [MS-SIPAE] section 4.4.

1.3.3 Authentication Broker Service

The Authentication Broker Service provides a web service-based **TLS** implementation. This is to be used by a client that does not have local support for TLS and wishes to use TLS-DSK authentication mechanism with the **SIP** server which is detailed in [MS-SIPAE].

The following diagram illustrates the sequence of events. Details of the call flows are explained in section 3.3.

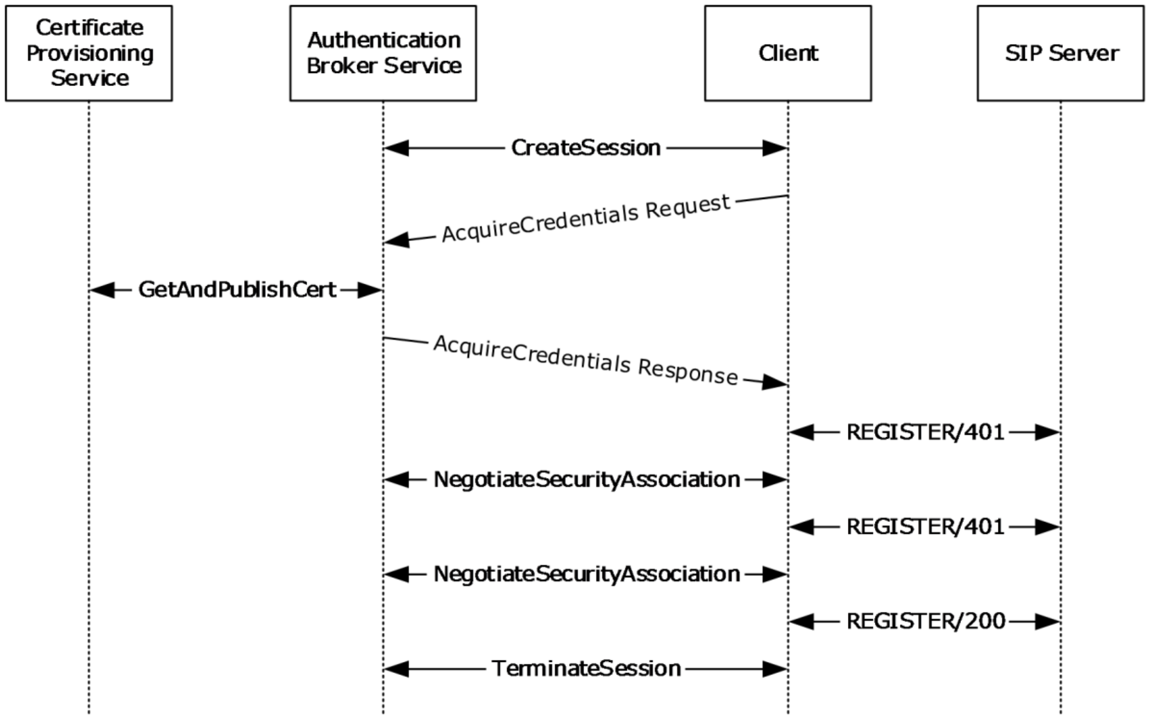
Certificate Provisioning Service
Authentication Broker Service
Client
SIP Server
CreateSession
AcquireCredentials Request
GetAndPublishCert
AcquireCredentials Response
REGISTER/401
NegotiateSecurityAssociation
REGISTER/401
NegotiateSecurityAssociation
REGISTER/200
TerminateSession

Figure 3: Sequence of events for Authentication Broker Service

1.4 Relationship to Other Protocols

The Web Ticket Service and Web applications that accept Web tickets as client credentials use **Simple Object Access Protocol (SOAP)** over **Hypertext Transfer Protocol over Secure Sockets Layer (HTTPS)**, as described in [RFC2818], SOAP 1.1, as described in [SOAP1.1], and WS-Trust 1.3, as described in [WS-Trust1.3], as shown in the following figure.

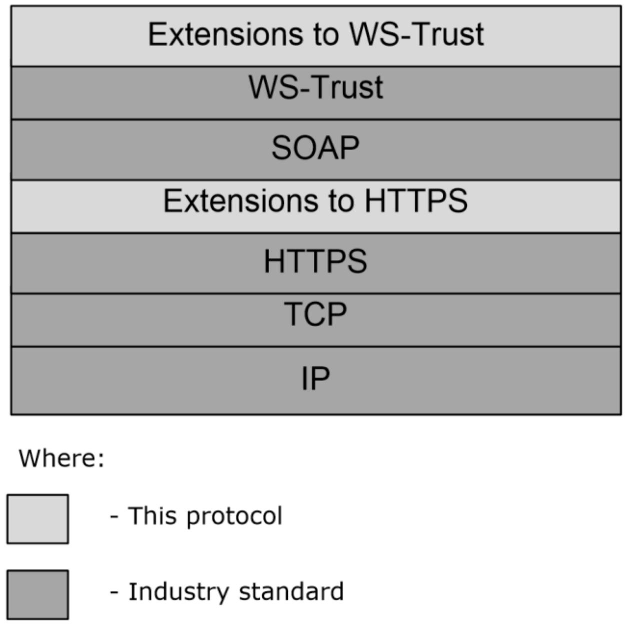

Figure 4: This protocol in relation to other protocols

1.5 Prerequisites/Preconditions

This protocol facilitates the issuance of X.509 v3 certificates (2). A server implementation of the protocol requires the functionality of a **certification authority (CA)**, capable of interpreting requests in PKCS#10, as described in [RFC2986], and generating the appropriate certificate (2).

Protocol clients are required to be able to understand PKCS#7 format, as described in [RFC2315] and [RFC5652], and X.509 v3 certificate (2) format, as described in [RFC5280], which are used by the server to send the **certificate chain** and the certificate (2).

A protocol client needs to retrieve the Web Ticket Service URL before using this protocol. The two ways for the client to do so are shown in the figures in section 1.3.1.1. If the client retrieves it from a Web service, the URL ought to be read from the metadata document of a participating Web service, from the **wsp:Policy/sp:IssuedToken/sp:Issuer/wsa10:Address** element associated with the service's binding that accepts a Web ticket, as described in [WSSP1.2-2012] and [WS-MetaDataExchange]. If the client retrieves it from a non-Web service, the Web application is required to return it in a 401 response in an HTTP header extension named **X-MS-WebTicketURL**, as described in [MS-OCDISCWS].

In order to use the Authentication Broker Service, a protocol client needs to retrieve the Internal/External AuthBroker Service URL, which is included as part of the User type in the response of the Lync Autodiscover Web Service [MS-OCDISCWS]. The section below shows a sample response.

```
<AutodiscoverResponse AccessLocation="Internal"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

  <User>

    <Link token="Internal/Autodiscover"
href="https://pool1.contoso.com/Autodiscover/AutodiscoverService.svc/root"/>

    <Link token="Internal/AuthBroker" href="https://pool1.contoso.com/Reach/sip.svc"/>
    <Link token="Internal/Ucwa" href="https://pool1.contoso.com/Ucwa/discovery"/>

    <Link token="External/Autodiscover" 
href="https://pool1external.contoso.com/Autodiscover/AutodiscoverService.svc/root"/>

    <Link token="External/AuthBroker" 
href="https://pool1external.contoso.com/Reach/sip.svc"/>

    <Link token="External/Ucwa" 
href="https://pool1external.contoso.com/Ucwa/discovery"/>

    <Link token="Internal/Mcx" href="https://pool1.contoso.com/Mcx/McxService.svc"/>

    <Link token="External/Mcx" 
href="https://pool1external.contoso.com/Mcx/McxService.svc"/>

  </User>

</AutodiscoverResponse>
```

1.6 Applicability Statement

This protocol is applicable when clients require authentication with servers using X.509 v3 certificates (2).

1.7 Versioning and Capability Negotiation

None.

1.8 Vendor-Extensible Fields

This protocol provides extensibility by the use of **any** and **anyAttribute** in the schema, as specified in [XMLSCHEMA1]. Vendors can choose to include their own elements by taking advantage of this extensibility.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

This protocol uses the **SOAP message** protocol for formatting request and response messages, as specified in [SOAP1.2/1] and [SOAP1.2/2]. It transmits those messages using HTTPS, as specified in [RFC2818].

2.2 Common Message Syntax

This section contains common definitions that are used by this protocol. The syntax of the definitions uses **XML schema**, as specified in [XMLSCHEMA1] and [XMLSCHEMA2], and **WSDL**, as specified in [WSDL].

The table in section 2.2.1 lists common namespaces.

2.2.1 Namespaces

This specification defines and references various **XML namespaces** using the mechanisms specified in [XMLNS]. Although this specification associates a specific XML namespace prefix for each XML namespace that is used, the choice of any particular XML namespace prefix is implementation-specific and not significant for interoperability.

Prefix	Namespace URI	Reference
xs	http://www.w3.org/2001/XMLSchema	[XMLSCHEMA1]
xsi	http://www.w3.org/2001/XMLSchema-instance	[XMLSCHEMA1]
xml	http://www.w3.org/XML/1998/namespace	[XMLSCHEMA1]
wst	http://docs.oasis-open.org/ws-sx/ws-trust/200512/	[WS-Trust1.3]
tns	http://schemas.microsoft.com/OCS/AuthWebServices/	
soap	http://schemas.xmlsoap.org/wsdl/soap/	[WSDL]
wsdl	http://schemas.xmlsoap.org/wsdl/	[WSDL]
wstep	http://schemas.microsoft.com/windows/pki/2009/01/enrollment	[MS-WSTEP]
auth	http://schemas.xmlsoap.org/ws/2006/12/authorization	[WSFederation]
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd	[WSSE 1.0]
wsp	http://schemas.xmlsoap.org/ws/2004/09/policy	[MS-WSPOL]
saml	urn:oasis:names:tc:SAML:1.0:assertion	[SAMLCore]
af	urn:component:Microsoft.Rtc.WebAuthentication.2010	
http	http://schemas.microsoft.com/ws/06/2004/policy/http	[MS-WSPOL]
wsaw	http://www.w3.org/2006/05/addressing/wsdl	[WSA1.0]
sp	http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702	[WSSP1.2-2012]
wsa10	http://www.w3.org/2005/08/addressing	[WSA1.0 Core]

Prefix	Namespace URI	Reference
wsx	http://schemas.xmlsoap.org/ws/2004/09/mex	
soap12	http://schemas.xmlsoap.org/wsdl/soap12	[WSDLSOAP]
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd	
wsap	http://schemas.xmlsoap.org/ws/2004/08/addressing/policy	
msc	http://schemas.microsoft.com/ws/2005/12/wsdl/contract	[MS-WSPOL]
wsam	http://www.w3.org/2007/05/addressing/metadata	[WSA1.0 Metadata]
soapenc	http://schemas.xmlsoap.org/soap/encoding	

2.2.2 Messages

This specification does not define any common WSDL message definitions.

2.2.3 Elements

This specification does not define any common XML schema element definitions.

2.2.4 Complex Types

The following table summarizes the set of common XML schema complex type definitions defined by this specification. XML schema complex type definitions that are specific to a particular operation are described with the operation.

Complex type	Description
af:OCSDiagnosticsFaultType	Authentication-specific error information in the **SOAP fault** detail. It is returned for some failures during Live ID authentication or Web ticket verification at a Web service.
af:MSWebAuthenticationType	WS-Policy assertion that describes the Live ID environment.
af:BindingType	WS-Policy assertion that the protocol client can communicate with the associated port. The absence of this assertion means that the client MUST NOT communicate with the associated WSDL port.
tns:ErrorInfoType	The base type of all the types that describe errors in any operation.

2.2.4.1 af:OCSDiagnosticsFaultType

The **af:OCSDiagnosticsFaultType** element is a child element of **s:Fault/s:detail**, as defined in [SOAP1.1].

```
<xs:complexType name="OCSDiagnosticsFaultType">
```

```
    <xs:sequence>
      <xs:element name="Ms-Diagnostics-Fault" type="af:MsDiagnosticsFaultType" minOccurs="1" />
      <xs:any processContents="lax" namespace="##any" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>
```

The **af:Ms-Diagnostics-Fault** element is a child element of **af:OCSDiagnosticsFaultType** element. It describes the authentication-specific error information.

```
  <xs:complexType name="MsDiagnosticsFaultType">
    <xs:sequence>
      <xs:element name="ErrorId" type="xs:positiveInteger" minOccurs="1" maxOccurs="1" />
      <xs:element name="Reason" type="xs:string" minOccurs="1" maxOccurs="1" />
      <xs:any processContents="lax" namespace="##any" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>
```

The **af:ErrorId** element carries a unique positive integer value for each specific error condition.

The **af:Reason** element carries a string that provides a reason for an explanation of specific error.

Error IDs and reason string used by OC Authentication Web Service are documented in Section 7.22 of [MS-OCER].

2.2.4.2 af:MSWebAuthenticationType

The **af:MSWebAuthenticationType** element is a WS-Policy assertion and a child element of the **wsp:Policy** element. It contains policy elements that provide information about a security token service that can issue tokens accepted by OC Authentication Web Service.

```
  <xs:complexType name="MSWebAuthenticationType">
    <xs:sequence>
      <xs:element name="Policy" type="wsp:Policy" minOccurs="1" />
      <xs:any processContents="lax" namespace="##any" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>
```

The **af:LiveIdEnvironmentType** element is a child element of the **wsp:Policy** element inside **af:MSWebAuthenticationType**. It describes the environment in which the security token service operates.

```
  <xs:simpleType name="LiveIdEnvironmentType">
    <xs:restriction base="xs:string" >
      <xs:enumeration value="PRODUCTION" />
      <xs:enumeration value="PPE" />
      <xs:enumeration value="INT" />
    </xs:restriction>
  </xs:simpleType>
```

The **"PRODUCTION"** enumeration value indicates production environment.

The **"PPE"** enumeration value indicates pre-production environment.

The **"INT"** enumeration value indicates integration environment.

2.2.4.3 af:BindingType

The **af:BindingType** element is a WS-Policy assertion and a child element of the **wsp:Policy** element.

```
<xs:complexType name="BindingType">
  <xs:sequence>
    <xs:any processContents="lax" namespace="##any" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax" />
</xs:complexType>
```

2.2.4.4 tns:ErrorInfoType

The **tns:ErrorInfoType** type is defined as follows.

```
<xs:complexType name="ErrorInfoType">
  <xs:sequence>
    <xs:element name="Description" type="xs:string" minOccurs="0" maxOccurs="1" />
    <xs:element name="AdditionalContext" minOccurs="0" maxOccurs="1">
      <xs:complexType>
        <xs:sequence>
          <xs:any processContents="lax" namespace="##any" minOccurs="0" maxOccurs="unbounded"
/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax" />
</xs:complexType>
```

tns:Description: Contains a textual description of the error.
tns:AdditionalContext: Can contain any implementation-defined context.

2.2.5 Simple Types

The following table summarizes the set of common XML schema simple type definitions defined by this specification. XML schema simple type definitions that are specific to a particular operation are described with the operation.

Simple type	Description
tns:ResponseClassType	Specifies whether the response for an operation is success, warning, or failure.

2.2.5.1 tns:ResponseClassType

The **tns:ResponseClassType** type is defined as follows.

```
<xs:simpleType name="ResponseClassType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Success" />
    <xs:enumeration value="Warning" />
    <xs:enumeration value="Error" />
  </xs:restriction>
</xs:simpleType>
```

The enumeration values have the usual meaning, and are used by the server to represent the class of the response.

2.2.6 Attributes

The following table summarizes the set of common XML schema attribute definitions defined by this specification. XML schema attribute definitions that are specific to a particular operation are described with the operation.

Attribute	Description
tns:ResponseClass	An instance of **ResponseClassType** that specifies the class of **Response**.

2.2.6.1 ResponseClass

The **ResponseClass** attribute is defined as follows.

```
<xs:attribute name="ResponseClass" type="tns:ResponseClassType" use="required" />
```

This attribute is an instance of type **ResponseClassType**, which is defined in section 2.2.5.1. It appears as a required attribute in all the responses of the **GetAndPublishCert** operation, which is defined in section 3.1.4.1.

2.2.7 Groups

This specification does not define any common XML schema group definitions.

2.2.8 Attribute Groups

This specification does not define any common XML schema attribute group definitions.

3 Protocol Details

The client side of this protocol is simply a pass-through. That is, no additional timers or other state is required on the client side of this protocol. Calls made by the higher-layer protocol or application are passed directly to the transport, and the results returned by the transport are passed directly back to the higher-layer protocol or application.

3.1 Certificate Provisioning Service Server Details

The Certificate Provisioning Service hosts a message **endpoint** that receives **GetAndPublishCert** messages. When received, the server uses the **certification** request, which is part of the message, to generate and sign a certificate (2). It then stores the certificate (2) in an implementation-defined manner, so that it can be used to verify a client certificate (2) presented for authentication. After that, it sends the certificate (2) to the client as part of **GetAndPublishCertResponse**, as specified in section 3.1.4.1.2.2.

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 server SHOULD keep the following states:

Certificate Issuer: A **proxy**, with which the server can communicate with a CA (2), used for generating X.509 v3 certificates (2). The nature of the proxy is implementation-dependent.

Trusted Certificate Authorities: A list of CAs (2) whose certificate chains are required to be trusted by the protocol clients in order for them to create Transport Layer Security (TLS) connections with the server. This list MUST have sufficient data that the certificates (2) in the chain can be located.

3.1.2 Timers

None.

3.1.3 Initialization

The CA (2) that would be used for generating X.509 v3 certificates (2) SHOULD be initialized with at least one **public key**/**private key** pair, used for signing the certificates (2).

The certificate (2) issuer proxy SHOULD be constructed and initialized, so that it can communicate with the CA (2).

The Trusted Certificate Authorities list SHOULD be initialized.

3.1.4 Message Processing Events and Sequencing Rules

The following table summarizes the list of operations as defined by this specification:

Operation	Description
GetAndPublishCert	A mechanism for clients to get a certificate (2), which can then be used for authentication purposes.

3.1.4.1 GetAndPublishCert

This operation is defined as part of the **CertProvisioningService portType**.

```
<wsdl:operation name="GetAndPublishCert">
    <wsdl:input message="tns:GetAndPublishCertMsg" />
    <wsdl:output message="tns:GetAndPublishCertResponseMsg" />
</wsdl:operation>
```

GetAndPublishCert generates a X.509 v3 certificate (2) using the PKCS#10 certification request in the request, and then stores the certificate (2) in an implementation-specific manner, so that it can be used to verify client certificates (2) supplied during authentication.
If an error occurs during processing, an error response MUST be sent using the **ErrorInfo** element in **GetAndPublishCertResponse**, as specified in section 3.1.4.1.2.2.
SOAP faults SHOULD NOT be used for error reporting.

3.1.4.1.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
tns:GetAndPublishCertMsg	The request for certificate provisioning.
tns:GetAndPublishCertResponseMsg	The response for certificate provisioning.

3.1.4.1.1.1 tns:GetAndPublishCertMsg

The **tns:GetAndPublishCertMsg** represents the incoming message and is defined as follows.

```
<wsdl:message name="GetAndPublishCertMsg">
    <wsdl:part name="request" element="tns:GetAndPublishCert" />
</wsdl:message>
```

tns:GetAndPublishCert: Refers to the **GetAndPublishCert** definition in section 3.1.4.1.2.1.

3.1.4.1.1.2 tns:GetAndPublishCertResponseMsg

The **tns:GetAndPublishCertResponseMsg** represents the outgoing message and is defined as follows.

```
<wsdl:message name="GetAndPublishCertResponseMsg">
    <wsdl:part name="response" element="tns:GetAndPublishCertResponse" />
</wsdl:message>
```

tns:GetAndPublishCertResponse: Refers to the **GetAndPublishCertResponse** definition in section 3.1.4.1.2.2.

3.1.4.1.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
tns:GetAndPublishCert	Container for the client request for certificate provisioning.
tns:GetAndPublishCertResponse	Container for the response to a request for certificate provisioning.
wst:RequestSecurityToken	Used to request a security token (for any purpose).
wst:RequestSecurityTokenResponse	Used to return a security token or response to a security token request.

3.1.4.1.2.1 tns:GetAndPublishCert

The **tns:GetAndPublishCert** element contains the client request, and is defined as follows.

```
<xs:element name="GetAndPublishCert" type="tns:GetAndPublishCertType" />
```

tns:GetAndPublishCertType: Refers to the **GetAndPublishCertType** definition in section 3.1.4.1.3.1.

3.1.4.1.2.2 tns:GetAndPublishCertResponse

The **tns:GetAndPublishCertResponse** element contains the response from server, and is defined as follows.

```
<xs:element name="GetAndPublishCertResponse" type="tns:GetAndPublishCertResponseType" />
```

tns:GetAndPublishCertResponseType: Refers to the **GetAndPublishCertResponseType** definition in section 3.1.4.1.3.2.

3.1.4.1.2.3 wst:RequestSecurityToken

The **wst:RequestSecurityToken** element is defined in [WS-Trust1.3] section 3.1, and further extended in [MS-WSTEP] section 3.1.4.1.2.5. For this protocol, this element MUST be a child of the **GetAndPublishCert** element and has the following extra restrictions:

1. **/wst:RequestedSecurityToken/wst:RequestType** MUST be "http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue".

2. **/wst:RequestedSecurityToken/wst:TokenType** MUST be "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3".

3. **/wst:RequestedSecurityToken/wsse:BinarySecurityToken** MUST contain a PKCS#10 Certification Signing Request (CSR) ([MS-WCCE] section 2.2.2.6.1) encoded with **base64 encoding**

4. **/wst:RequestedSecurityToken/wsBinarySecurityToken/@EncodingType** MUST be "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd#base64binary".

5. The **/wst:RequestedSecurityToken/wsse:BinarySecurityToken/@ValueType** attribute MUST be "http://schemas.microsoft.com/OCS/AuthWebServices.xsd#PKCS10".

Any optional element or attribute not mentioned in this section SHOULD be ignored.

The server SHOULD be able to process **ValidityPeriod** and **ValidityPeriodUnits**, as specified in [MS-WCCE] section 3.1.1.4.3.1.1.

3.1.4.1.2.4 wst:RequestSecurityTokenResponse

The **wst:RequestSecurityTokenResponse** element is defined in [WS-Trust1.3] section 3.2, and is further extended in [MS-WSTEP] section 3.1.4.1.3.4. For this protocol, this element is a child of the **GetAndPublishCertResponse** element.

In case of an error, this element MUST NOT be present in the **GetAndPublishCertResponse**.

In case of success, the following restrictions MUST be adhered to:

1. **/wst:RequestSecurityTokenResponse/wstep:DispositionMessage** MUST be "Issued".
2. **/wst:RequestSecurityTokenResponse /wstep:DispositionMessage/@lang** attribute MUST be "en-US".
3. **/wst:RequestSecurityTokenResponse/wst:TokenType** MUST be "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3".
4. **/wst:RequestSecurityTokenResponse/wst:RequestedSecurityToken** MUST contain **BinarySecurityToken**, which MUST contain the X.509 v3 certificate (2) using base64 encoding.
5. The **Common Name** of the **Subject (Section 4.1.2.6 of** [RFC3280]**)** in the returned certificate (2) MUST have the same value as the **Entity** attribute in the client request.
6. **SubjectKeyIdentifier (Section 4.2.1.2 of** [RFC3280]**)** in the returned certificate (2) SHOULD contain the value of the **DeviceId** attribute in the client request.
7. **/wst:RequestSecurityTokenResponse/wst:RequestedSecurityToken/wsse:BinarySecurityToken/@ValueType** MUST be "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3".
8. **/wst:RequestSecurityTokenResponse/wst:RequestedSecurityToken/wsse:BinarySecurityToken/@EncodingType** MUST be "http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd#base64binary".
9. **/wst:RequestSecurityTokenResponse/wsse:BinarySecurityToken** MUST contain the **BinarySecurityToken** that came as part of the incoming request.

Any element or attribute not mentioned in this section SHOULD be ignored.

3.1.4.1.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

Complex type	Description
tns:GetAndPublishCertType	Describes the client request for certificate provisioning.
tns:GetAndPublishCertResponseType	Describes the server response to a request for certificate provisioning.
tns:GetAndPublishCertErrorInfoType	Describes any failure in a **GetAndPublishCert** operation.

3.1.4.1.3.1 tns:GetAndPublishCertType

The **tns:GetAndPublishCertType** type describes the client request and is defined as follows.

```
<xs:complexType name="GetAndPublishCertType">
  <xs:sequence>
    <xs:element ref="wst:RequestSecurityToken" minOccurs="1" maxOccurs="1" />
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="DeviceId" type="xs:string" use="required" />
  <xs:attribute name="Entity" type="xs:anyURI" use="required" />
  <xs:anyAttribute namespace="##other" processContents="lax" />
</xs:complexType>
```

wst:RequestSecurityToken: Refers to the **RequestSecurityToken**, as defined in section 3.1.4.1.2.3.
DeviceId: Refers to the **DeviceId**, as defined in section 3.1.4.1.5.1.
Entity: Refers to the **Entity**, as defined in section 3.1.4.1.5.2.

3.1.4.1.3.2 tns:GetAndPublishCertResponseType

The **tns:GetAndPublishCertResponseType** type describes the server response and is defined as follows.

```
<xs:complexType name="GetAndPublishCertResponseType">
  <xs:sequence>
    <xs:element ref="wst:RequestSecurityTokenResponse" minOccurs="0" maxOccurs="1" />
    <xs:element name="ErrorInfo" type="tns:GetAndPublishCertErrorInfoType" minOccurs="0"
maxOccurs="1" />
  </xs:sequence>
  <xs:attribute name="DeviceId" type="xs:string" use="required" />
  <xs:attribute name="Entity" type="xs:anyURI" use="required" />
  <xs:attribute name="ResponseClass" type="tns:ResponseClassType" use="required" />
  <xs:anyAttribute namespace="##other" processContents="lax" />
</xs:complexType>
```

wst:RequestSecurityTokenResponse: Refers to **RequestSecurityTokenResponse** element in section 3.1.4.1.2.4.
ErrorInfo: This element contains information about the error that occurred, if the operation is not successful. It MUST be an instance of the **GetAndPublishCertErrorInfoType**, as defined in section 3.1.4.1.3.3.
DeviceId: Refers to the **DeviceId** definition in section 3.1.4.1.5.1. This attribute contains the same value as the one contained in the **DeviceId** attribute of the client request.
Entity: Refers to the **Entity** definition in section 3.1.4.1.5.2. This attribute contains the same value as the one contained in **Entity** attribute of the client request.
ResponseClass: Refers to the **ResponseClass** definition in section 2.2.6.1.

3.1.4.1.3.3 tns:GetAndPublishCertErrorInfoType

The **tns:GetAndPublishCertErrorInfoType** type is defined as follows.

```
<xs:complexType name="GetAndPublishCertErrorInfoType">
  <xs:complexContent>
    <xs:extension base="ErrorInfoType">
      <xs:sequence />
      <xs:attribute name="ResponseCode" type="GetAndPublishCertResponseCodeType"
use="required" />
    </xs:extension>
  </xs:complexContent>
```

```
    </xs:complexType>
```

It is used to describe any failure in a **GetAndPublishCert** operation.
tns:ResponseCode: It MUST be an instance of a **GetAndPublishCertResponseCodeType**, as defined in section 3.1.4.1.4.1, and contains a code that describes the failure.

3.1.4.1.4 Simple Types

The following table summarizes the XML schema simple type definitions that are specific to this operation.

Simple type	Description
tns:GetAndPublishResponseCodeType	The status of the certificate provisioning request.

3.1.4.1.4.1 tns:GetAndPublishResponseCodeType

The **tns:GetAndPublishResponseCodeType** type is defined as follows.

```
    <xs:simpleType name="GetAndPublishCertResponseCodeType">
      <xs:restriction base="xs:string">
        <xs:enumeration value="NoError" />
        <xs:enumeration value="InternalError" />
        <xs:enumeration value="InvalidPublicKey" />
        <xs:enumeration value="InvalidValidityPeriod" />
        <xs:enumeration value="InvalidEKU" />
        <xs:enumeration value="InvalidSipUri" />
        <xs:enumeration value="InvalidCSR" />
        <xs:enumeration value="DataStoreUnavailable" />
        <xs:enumeration value="InvalidDeviceId" />
        <xs:enumeration value="RequestMalformed" />
        <xs:enumeration value="AccountDisabled" />
        <xs:enumeration value="UserImproperlyProvisioned" />
      </xs:restriction>
    </xs:simpleType>
```

NoError: Indicates success.

InternalError: Indicates an unexpected server error.

InvalidPublicKey: Indicates that the certification request did not contain a valid public key.

InvalidValidityPeriod: Indicates that the CSR contained an invalid or unacceptable validity period.

InvalidEKU: Indicates that the CSR contained invalid Enhanced Key Usage.

InvalidSipUri: Indicates that the **Entity**, as defined in section 3.1.4.1.5.2, is invalid.

InvalidCSR: Indicates that the CSR is invalid.

DataStoreUnavailable: Indicates that the store where the certificate (2) was supposed to be stored was not available.

InvalidDeviceId: Indicates that the **DeviceId**, as defined in section 3.1.4.1.5.1, is invalid.

RequestMalformed: Indicates that the **wst:RequestSecurityToken**, as defined in section 3.1.4.1.2.3, is invalid.

AccountDisabled: Indicates that the account of the user operating the client is disabled.

UserImproperlyProvisioned: Indicates that the user is not provisioned on a server that supports this protocol.

3.1.4.1.5 Attributes

The following table summarizes the XML schema attribute definitions that are specific to this operation.

Attribute	Description
DeviceId	Part of **GetAndPublishCertType**, as specified in section 3.1.4.1.3.1, and **GetAndPublishCertResponseType**, as specified in section 3.1.4.1.3.2.
Entity	Part of **GetAndPublishCertType** and **GetAndPublishCertResponseType.**

3.1.4.1.5.1 DeviceId

The **DeviceId** attribute is part of **GetAndPublishCertType** and **GetAndPublishCertResponseType**, and is defined as follows.

```
<xs:attribute name="DeviceId" type="xs:string" use="required" />
```

This is an identifier for the device on which the client is operating, and serves to identify a device unique among the various devices that the same user might be using simultaneously. It MUST be unique for each device being used by the same user. **DeviceId** MUST be convertible to a **GUID**. If the client uses an identifier for the device with any other service, which uses the certificate (2) retrieved using the **GetAndPublishCert** operation for authentication, **DeviceId** and the aforementioned identifier MUST be equal or it MUST be possible for the **DeviceId** to be generated using the identifier using a deterministic mathematical transformation. This transformation MUST be known to the certificate (2) verification engine.

3.1.4.1.5.2 Entity

The **Entity** attribute is part of **GetAndPublishCertType** and **GetAndPublishCertResponseType**, and is defined as follows.

```
<xs:attribute name="Entity" type="xs:anyURI" use="required" />
```

This is an identifier for the user who is using the client. It MUST be same as the Session Initiation Protocol (SIP) **Uniform Resource Identifier (URI)** for the authenticated user, as specified in [RFC3261] section 19.1, without the "sip:" prefix.

3.1.4.1.6 Groups

This specification does not define any common XML schema group definitions.

3.1.4.1.7 Attribute Groups

This specification does not define any common XML schema attribute group definitions.

3.1.5 Timer Events

None.

3.1.6 Other Local Events

None.

3.2 Web Ticket Service Server Details

The Web Ticket Service issues Web tickets using its **IssueToken** operation, which follows the protocol described in [WS-Trust1.3], except where indicated in section 3.2.4.1.1.1 and section 3.2.4.1.1.2.

Clients MUST authenticate to the Web Ticket Service using one of the following authentication protocols:

- Integrated Windows authentication
- OCS-signed certificate authentication
- Live ID authentication
- OAuth2 authentication

Integrated Windows authentication follows the **Kerberos** and the **NT LAN Manager (NTLM) Authentication Protocol**, as specified in [RFC4559]. If Integrated Windows authentication fails, the errors defined in section 3.2.4.1 are returned.

Certificate (2) authentication signed by a **user agent server (UAS)** follows SOAP Message Security 1.1, as specified in [WSS], to validate an X.509 security token, as specified in [WSSX509TP]. If OCS-signed certificate (2) authentication fails, the errors defined in section 3.2.4.1 are returned. The certificate signed by the UAS can be obtained from the Certificate Provisioning Service described in section 3.1 of this document.

The Live ID token is presented as a **Security Assertion Markup Language (SAML)** token, as specified in [SAMLCore], and verified using SOAP Message Security 1.1, as specified in [WSS]. The way in which the client retrieves the SAML token is out of the scope of this document. The type of Live ID environment for which the server is configured is specified in the Web service metadata as MSWebAuthentication policy assertion. See section 2.2.4.2 for MSWebAuthentication policy assertion schema. If Live ID authentication fails, the errors defined in section 3.2.4.1 are returned.

The OAuth2 authentication follows the OAuth 2.0 Authorization Protocol described in [IETFDRAFT-OAuth2.0] with Extensions described in [MS-OAUTH2EX]. The protocol server extracts the OAuth2 token from the Authorization header of the HTTP request and validates that:

- the token carries an actor token that was issued by the Authorization Server that protocol server trusts;
- the actor token is signed by a certificate associated with the Authorization Server that issued the token;
- the actor token nameid (name identifier) claim value matches the issuer claim in the token;
- both the token itself and actor token carry audience claim with a value in the following format: 00000004-0000-0ff1-ce00-000000000000/<host_fqdn>@<realm>, where:
    - 00000004-0000-0ff1-ce00-000000000000 is identifier associated with the protocol server described in the document,

    - <host_fqdn> is a placeholder which represents the **fully qualified domain name (FQDN)** of the protocol server,
    - <realm> is a place holder which represents a realm value configured for the protocol server;
- the token carries at least one of the following claims: nameid (name identifier), smtp (e-mail address), sip (SIP address) and values in these claims match corresponding values of exactly one user in the UAS database.

If validation of OAuth2 token fails, the errors defined in section 3.2.4.1 are returned.

Sending the Web Ticket as Credentials to a Web Service Web Application

After the client receives a Web ticket from the Web Ticket Service, the client MUST attach the Web ticket, as it would a SAML token, to its requests to a participating Web service.

If the Web ticket fails validation, **OCSDiagnosticsFaults**, as described in section 2.2.4.1, SHOULD be returned. The following table describes the relevant **OCSDiagnosticsFaults**.

faultcode	ErrorId	Reason
wsse:InvalidSecurityToken	28032	The Web ticket is invalid.
wsse:InvalidSecurityToken	28033	The Web ticket has expired.
wsse:InvalidSecurityToken	28034	Proof Web tickets are only valid at the same Web server where they were requested.

The Web service MAY also return faults specified in [WSSE 1.0].

The Web ticket can be sent as a signed security token or a proof-of-possession token, as specified in [WS-Trust1.3].

Sending the Web Ticket as Credentials to a Non-Web Service Web Application

After the client receives a Web ticket from the Web Ticket Service, the client MUST send the Web ticket in an HTTP header extension in its request to participating non-Web services.

```
X-MS-WebTicket = ticket-data *(";" ticket-extns)
ticket-data = "opaque" "=" base64-ticket
base64-ticket = 1*(ALPHA / DIGIT / "+" / "/")     ; base-64 encoded SAML token
ticket-extns: 1*(ALPHA / DIGIT / "-") "=" 1*(ALPHA / DIGIT / "-")
```

The Web ticket, or SAML token, used to construct the **base64-ticket** MUST be a signed security token, as specified in [WS-Trust1.3].

If the Web ticket fails validation, an error response MUST be returned with an HTTP extension header called **X-Ms-diagnostics**, as described in section 3.2.4.1. The following table describes the relevant fault codes.

Faultcode	ErrorId	Reason
wsse:InvalidSecurityToken	28032	The Web ticket is invalid.
wsse:InvalidSecurityToken	28033	The Web ticket has expired.

3.2.1 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 Web Ticket Service SHOULD keep the following states:

Fully Qualified Domain Name of the Web Server Farm: This fully qualified domain name (FQDN) is used to verify the address in the **wst:RequestSecurityToken/wsp:AppliesTo/wsa10:EndpointReference/wsa10:Address** element of the RST. The logic for determining this FQDN is implementation-dependent.

3.2.2 Timers

None.

3.2.3 Initialization

None.

3.2.4 Message Processing Events and Sequencing Rules

The following table summarizes the list of operations as defined by this specification:

Operation	Description
IssueToken	Provides a Web ticket given one of the following credentials: ▪ Integrated Windows authentication ▪ Live ID ▪ A certificate (2) signed by a UAS. The operation is at the Web Ticket Service.

3.2.4.1 IssueToken

The **IssueToken** interface provides an operation that returns a Web ticket for a client.

```
<wsdl:portType name="IWebTicketService">
    <wsdl:operation name="IssueToken">
        <wsdl:input wsaw:Action="http://docs.oasis-open.org/ws-sx/ws-trust/200512/RST/Issue"
message="tns:IWebTicketService_IssueToken_InputMessage"/>
        <wsdl:output wsaw:Action="http://docs.oasis-open.org/ws-sx/ws-
trust/200512/RSTRC/IssueFinal" message="tns:IWebTicketService_IssueToken_OutputMessage"/>
    </wsdl:operation>
</wsdl:portType>
```

If there is an error while processing the credentials of the user, then depending on the authentication type used, the response message contains the error details in a custom HTTP header or in a SOAP fault.

HTTP X-Ms-diagnostics Header

The **X-Ms-diagnostics** header is an HTTP header that is returned if Integrated Windows authentication or certificate (2) authentication signed by the UAS fails at the Web Ticket Service for the reasons in this section.

The header has the following format.

```
X-Ms-diagnostics = errorId ";" source ";" reason ";" fault
errorId = 1*DIGIT
source = DQUOTE 1*(ALPHA / DIGIT / "-" / "." / "_" / "~") DQUOTE
                                        ; Fully qualified domain name of server
token = DQUOTE 1*( ALPHA / DIGIT / "-" / "." / "_" / "~") DQUOTE
fault = DQUOTE 1*(ALPHA) ":" 1*(ALPHA) DQUOTE
```

The HTTP response code and the details of the **X-Ms-diagnostics** header are described later for each authentication type.

The following table lists Integrated Windows authentication errors.

Type of error	Response code	ErrorId	token	faultcode
The user was authenticated but could not be found in the UAS database.	403	28000	User is not SIP enabled.	wsse:FailedAuthentication
Some unexpected error occurred in the system.	500	28001	Internal error while processing Integrated Windows authentication or authorization.	wsse:FailedAuthentication

SOAP Faults

The following **OCSDiagnosticsFaults**, as defined in section 2.2.4.1, are returned for Live ID authentication failures, OCS-signed certificate (2) failures, or if there are internal errors processing the RST after Integrated Windows authentication or certificate (2) credentials signed by the UAS are successfully verified. The following table lists SOAP errors.

faultcode	ErrorId	Reason
wsse:SecurityTokenUnavailable	28028	The Live ID token encryption key cannot be resolved. Check that the token is obtained for this site in the appropriate Live ID environment.
wsse:SecurityTokenUnavailable	28017	The Live ID token signing key cannot be resolved. Check that the token is obtained from the appropriate Live ID environment.
wsse:UnsupportedSecurityToken	28018	The Live ID token was produced with the incorrect site policy.
wsse:FailedAuthentication	28019	The Live ID token identity is not associated with a user account.
wsse:InvalidSecurity	28020	There is no valid security token.
wsse:UnsupportedSecurityTokenType	28021	The security token type is unsupported.
wsse:InvalidSecurityToken	28022	There is no valid subject statement.
wsse:InvalidSecurity	28023	There is no valid message security.
wsse:FailedAuthentication	28024	Authentication failed.

The "key cannot be resolved" errors above indicate that protocol server could not locate the key referenced in the token in local or remote stores that it knows about. The "incorrect site policy" error above indicates that Live ID token presented to the protocol server was constructed using policy that the server does not understand.
The following table lists certificate (2) authentication errors while processing the contents of a certificate (2) signed by the UAS.

faultcode	ErrorId	Reason
wsse:FailedAuthentication	28011	The certificate (2) is expired.
wsse:FailedAuthentication	28012	The certificate (2) is invalid.
wsse:FailedAuthentication	28013	The certificate (2) is not found.
wsse:FailedAuthentication	28014	The user was not found when queried in the database.
wsse:FailedAuthentication	28015	There was an internal error while processing a certificate (2) authentication or authorization provided by the UAS.

The following table lists internal failures that occur after Integrated Windows authentication and UAS certificate (2) credentials are successfully verified.

SubCode	ErrorId	Reason
wsse:InvalidSecurity	28025	There is no valid security principal.
wsse:InvalidSecurity	28026	There is no valid security identity.
wsse:InvalidSecurity	28027	There is no valid message security.

The following table lists failures that occur while processing the RST.

SubCode	ErrorId	Reason
wst:RequestFailed	28035	The SIP URI in the claim type requirements of the Web ticket request does not match the SIP URI associated with the presented credentials.

3.2.4.1.1 Messages

The following table summarizes the set of WSDL message definitions that are specific to this operation.

Message	Description
tns:IWebTicketService_IssueToken_InputMessage	A request for a token to be issued.
tns:IWebTicketService_IssueToken_OutputMessage	The response to a request for a token to be issues.

3.2.4.1.1.1 tns:IWebTicketService_IssueToken_InputMessage

The **tns:IWebTicketService_IssueToken_InputMessage** represents the incoming message and is defined as follows.

```
<wsdl:message name="IWebTicketService_IssueToken_InputMessage">
    <wsdl:part name="rst" type="q1:MessageBody"
xmlns:q1="http://schemas.microsoft.com/Message"/>
```

Refer to the **q1:MessageBody** definition in section 3.2.4.1.3.1.

3.2.4.1.1.2 tns:IWebTicketService_IssueToken_OutputMessage

The **tns:IWebTicketService_IssueToken_OutputMessage** represents the incoming message and is defined as follows.

```
<wsdl:message name="IWebTicketService_IssueToken_OutputMessage">
    <wsdl:part name="IssueTokenResult" type="q2:MessageBody" 
xmlns:q2="http://chemas.microsoft.com/Message"/>
  </wsdl:message>
```

Refer to the **q2:MessageBody** definition in section 3.2.4.1.3.2.

3.2.4.1.2 Elements

Elements are defined in the **XML schema definition (XSD)** associated with [WS-Trust1.3].

3.2.4.1.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

Complex type	Description
q1:MessageBody	Describes the type of **tns:IWebTicketService_IssueToken_InputMessage**.
q2:MessageBody	Describes the type of **tns:IWebTicketService_IssueToken_OutputMessage**.
wst:RequestSecurityTokenMsg	The alternative type of **q1:MessageBody**, as defined in section 3.2.4.1.3.1.
wst:RequestSecurityTokenResponseMsg	The alternative type of **q2:MessageBody**, as defined in section 3.2.4.1.3.2.

Other complex types are defined in the XSD associated with [WS-Trust1.3].

3.2.4.1.3.1 q1:MessageBody

The **q1:MessageBody** type is defined as follows:

```
<xs:complexType name="MessageBody">
    <xs:sequence>
      <xs:any namespace="##any" maxOccurs="unbounded" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

3.2.4.1.3.2 q2:MessageBody

Refer to the **q1:MessageBody** definition in section 3.2.4.1.3.1.

3.2.4.1.3.3 wst:RequestSecurityTokenMsg

The **wst:RequestSecurityTokenMsg** is the alternative type of **q1:MessageBody**, and is defined in [WS-Trust1.3], with the exception that only the following elements need to be in the message:

The **wst:RequestSecurityTokenMsg** is an incoming message, and is defined in **[WS-Trust1.3]**, with the exception that only the following elements need to be in the message:

/wst:RequestSecurityToken/@Context: A required attribute that MUST be set to a **universally unique identifier (UUID)**.

/wst:RequestSecuritytoken/wst:TokenType: A required element that MUST be set to "http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV1.1".

/wst:RequestSecurityToken/wst:RequestType: A required element that MUST be set to "http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue".

/wst:RequestSecurityToken/wsp:AppliesTo/wsa:EndpointReference/wsa:Address: A required element that MUST be set to the **HTTP URL** of the service for which the token is being requested. For example, the element could be set to the HTTP URL of the Certificate Provisioning Web Service. The server MUST validate that this address is part of the server farm.

/wst:RequestSecurityToken/wst:Entropy/wst:BinarySecret: This required element specifies a base64 encoded sequence of cryptographically random octets representing the requestor's entropy. The key size MUST be obtained from the WS-Policy, as specified in **[MS-WSPOL]**, for the Web Ticket Service and SHOULD NOT be less than 128 bits. The entropy size SHOULD be the same size as the key size.

/wst:RequestSecuritytoken/wst:KeyType: A required element that MUST be set to "http://docs.oasis-open.org/ws-sx/ws-trust/200512/SymmetricKey".

/wst:RequestSecuritytoken/wst:Claims: An optional element representing a specific claim. Its **Dialect** attribute MUST be set to "urn:component:Microsoft.Rtc.WebAuthentication.2010:authclaims".

/wst:RequestSecuritytoken/wst:Claims/auth:ClaimType: An optional element, as specified in **[WSFederation]**, representing a specific claim type. If this element is present, its **Uri** attribute MUST be set to "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/uri".

/wst:RequestSecuritytoken/wst:Claims/auth:ClaimType/auth:Value: An optional element, as specified in **[WSFederation]**, representing the **SIP URI** of the user for which the **Web ticket** will be created. If this element is included, the SIP URI MUST match the credentials submitted with the RST. If the element is not included, the server SHOULD use the credentials submitted with the RST to determine the SIP URI. If the SIP URI does not match the credentials, the server SHOULD respond with a fault message carrying fault code **wst:RequestFailed** as described in section 3.2.4.1.

If any one of the above required elements is not supplied or the element syntax does not conform to the syntax requirement specified in this section, the server SHOULD respond with a fault message carrying fault code **wst:InvalidRequest** as described in Section 3 of **[WS-Trust1.3]**.

3.2.4.1.3.4 wst:RequestSecurityTokenResponseMsg

The **wst:RequestSecurityTokenResponseMsg** is the alternative type of **q2:MessageBody**, and is defined in [WS-Trust1.3], with the exception that only the following elements need be in the message:

/wst:RequestSecurityTokenResponse/@Context: A required attribute that MUST be set to the value from the corresponding request.

/wst:RequestSecuritytokenResponse/wst:TokenType: A required element that MUST be set to "http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-1.1#SAMLV1.1".

/wst:RequestSecurityTokenResponse/wst:RequestedSecurityToken/saml:Assertion: A required element that MUST be returned. This element and its contents SHOULD be treated as an opaque XML token by the User Agent.

/wst:RequestSecurityTokenResponse/wst:Lifetime/wsu:Created: An optional element that indicates the **Coordinated Universal Time (UTC)** when the token was created.

/wst:RequestSecurityTokenResponse/wst:Lifetime/wsu:Expires: A required element that indicates the **UTC** time when the token expires.

/wst:RequestSecurityTokenResponse/wst:RequestedUnattachedReference: An optional element that indicates how to reference the returned token when that token does not support references using **URI** fragments (XML ID). This information is included because the token is considered opaque to the requestor.

/wst:RequestSecurityTokenResponse/wst:RequestedAttachedReference: An optional element that indicates how to reference the token when it is not placed inside the message. This information is included because the token is considered opaque to the requestor.

/wst:RequestSecurityTokenResponse/wsp:AppliesTo/wsa:EndpointReference/wsa:Address: A required element that MUST be set to the **URL** of the **HTTP** URL of the server farm or service to which the ticket applies. Clients SHOULD perform a prefix match on this URL to determine which services the ticket applies to.

/wst:RequestSecurityTokenResponse/wst:RequestedProofToken/wst:ComputedKey: This required element MUST be set to the element specified in the **ComputedKeyAlgorithm** element of the metadata from the Web Ticket Service's binding. For example, it could be set to http://docs.oasis-open.org/ws-sx/ws-trust/200512/CK/PSHA1.

/wst:RequestSecurityTokenResponse/wst:Entropy/wst:BinarySecret: This required element specifies a base64 encoded sequence of cryptographically random octets representing the Web Ticket Service's entropy. The size of the element SHOULD be the same as the KeySize specified in the WS-Policy associated with the binding at a **Web service** that accepts a **Web ticket**.

3.2.4.1.4 Simple Types

Simple types are defined in the XSD associated with [WS-Trust1.3].

3.2.4.1.5 Attributes

Attributes are defined in the XSD associated with [WS-Trust1.3].

3.2.4.1.6 Groups

This specification does not define any common XML schema group definitions.

3.2.4.1.7 Attribute Groups

This specification does not define any common XML schema attribute group definitions.

3.2.5 Timer Events

None.

3.2.6 Other Local Events

None.

3.3 Authentication Broker Service Server Details

The Authentication Broker Service requires a session to be created using **CreateAuthBrokerSession** (as specified in section 3.3.4.1) in order provide the TLS implementation data for authentication with the SIP **server**. The service requires a valid **Web Ticket** which can be obtained using the Web Ticket Service (section 3.2). The client is also required to provide a list of client-supported hash algorithms**.** The response from **CreateAuthBrokerSession** contains the **SessionId** that will be used for remaining requests, as well as the server-supported hash algorithms.

AuthBrokerAcquireCredential (as specified in section 3.3.4.3) is called by the client in order to acquire a valid certificate for the user. This is passed the **SessionId** and the **SIPInstance**. The **server (2)** will need to acquire a new certificate from the Certificate Provisioning Service (section 3.1) or locate a previously obtained certificate.

Once the above two calls are completed, the client will then initiate authentication with the SIP server. When the TLS protocol implementation is required to generate responses, the client will make a call to **AuthBrokerNegotiateSecurityAssociation** (as specified in section 3.3.4.4)**,** passing the target and gssapi-data provided by the **server (2)** to generate the gssapi-data required for the response.

There are three conditions under which this call will function:

- client_hello – **AuthBrokerNegotiateSecurityAssociation** will generate an **SA** and a client_hello handshake message when no **input** element is provided. The result is encoded using the base64 algorithm, and returned in the response.
- gssapi-data challenge – The server locates the SA that it created for the client_hello response, decodes the value of the "input" parameter using the base64 algorithm, and passes it along with the security context information stored in the SA to the TLS implementation. The server obtains or locates a previously obtained certificate (1) and calls the TLS implementation to generate an output token that carries the TLS certificate, client_key_exchange, certificate_very, change_cipher, _spec, and finished handshake messages. The server then encodes the TLS token and returns it to the protocol client as part of the response.
- Finished_handshake – The server locates the SA that it created for the client_hello response, decodes the value of the "input" parameter using the base64 algorithm and passes it, along with security context information stored in the SA, to the TLS implementation for validation. Once information is validated, the server computes, or derives, client and server authentication keys as described in [MS-SIPAE] section 3.2.5.1.

After the client is done with the requests, the session is terminated by calling **TerminateAuthBrokerSession** (section 3.3.4.2)**.**

3.3.1 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 Authentication Broker Service SHOULD keep the following states:

- The session identifier of the user.
- A certificate store that can be used to save and retrieve certificates
- The store of the challenge that is associated with the first **AuthBrokerNegotiateSecurityAssociation** (section 3.3.4.4) that is used to generate the response for subsequent calls.

3.3.2 Timers

The server keeps track of the session using a session expiration timer that automatically terminates the session after a period of inactivity.

3.3.3 Initialization

None.

3.3.4 Message Processing Events and Sequencing Rules

The following table summarizes the list of operations as defined by this specification:

Operation	Description
CreateAuthBrokerSession	Creates a session to be used as part of the client authentication with the SIP server using TLS-DSK.
TerminateAuthBrokerSession	Ends the client session with Authentication Broker Service.
AuthBrokerAcquireCredential	Associates a specific **SIPInstance** with the session. This can be called once per session.
AuthBrokerNegotiateSecurityAssociation	Provides gssapi response data for challenges issues by the SIP server**.** Also provides client and server authentication keys once the finish handshake is received. This can be called multiple times per session.

3.3.4.1 CreateAuthBrokerSession

Creates a session to be used as part of the client authentication with the SIP server using TLS-DSK.

```
<wsdl:operation name="CreateAuthBrokerSession">
<wsdl:input wsaw:Action="http://tempuri.org/IAuthBroker/CreateAuthBrokerSession"
message="tns:IAuthBroker_CreateAuthBrokerSession_InputMessage" />
<wsdl:output wsaw:Action="http://tempuri.org/IAuthBroker/CreateAuthBrokerSessionResponse"
message="tns:IAuthBroker_CreateAuthBrokerSession_OutputMessage" />
</wsdl:operation>
```

3.3.4.1.1 Messages

The following table summarizes the XML schema message definitions that are specific to this operation.

Operation	Description
tns:IAuthBroker_CreateAuthBrokerSession_InputMessage	The request for **CreateAuthBrokerSession**.

Operation	Description
tns:IAuthBroker_CreateAuthBrokerSession_OutputMessage	The response for **CreateAuthBrokerSession**.

3.3.4.1.1.1 tns:IAuthBroker_CreateAuthBrokerSession_InputMessage

The request **WSDL message** for the **CreateAuthBrokerSession** **WSDL operation**.

```
<wsdl:message name="IAuthBroker_CreateAuthBrokerSession_InputMessage">
  <wsdl:part name="parameters" element="tns:CreateAuthBrokerSession" />
  </wsdl:message>
```

CreateAuthBrokerSession: Refers to the **CreateAuthBrokerSession** definition in section 3.3.4.1.2.1.

3.3.4.1.1.2 tns:IAuthBroker_CreateAuthBrokerSession_OutputMessage

The response WSDL message for the **CreateAuthBrokerSession** WSDL operation.

```
<wsdl:message name="IAuthBroker_CreateAuthBrokerSession_OutputMessage">
  <wsdl:part name="parameters" element="tns:CreateAuthBrokerSessionResponse" />
  </wsdl:message>
```

CreateAuthBrokerSessionResponse: Refers to the **CreateAuthBrokerSessionResponse** definition in section 3.3.4.1.2.2.

3.3.4.1.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
tns:CreateAuthBrokerSession	Container for the client request to create a session.
tns:CreateAuthBrokerSessionResponse	Container for the response to a request to create a session.

3.3.4.1.2.1 tns:CreateAuthBrokerSession

The container for the client request to **CreateAuthBrokerSession**.

```
<xs:element name="CreateAuthBrokerSession">
- <xs:complexType>
- <xs:sequence>
  <xs:element minOccurs="0" name="supportedHashAlgorithms" nillable="true"
type="q5:ArrayOfstring" xmlns:q5="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
/>
  </xs:sequence>
  </xs:complexType>
  </xs:element>
```

supportedHashAlgorithms: An array of the supported hash algorithms made by the requestor.

3.3.4.1.2.2 tns:CreateAuthBrokerSessionResponse

The container for the response to a request to **CreateAuthBrokerSession**.

```
<xs:element name="CreateAuthBrokerSessionResponse">
- <xs:complexType>
- <xs:sequence>
  <xs:element minOccurs="0" name="CreateAuthBrokerSessionResult" nillable="true"
type="q6:CreateAuthBrokerSessionResponse"
xmlns:q6="http://schemas.datacontract.org/2004/07/Microsoft.Rtc.Internal.WebRelay.Sip" />
  </xs:sequence>
  </xs:complexType>
  </xs:element>
```

CreateAuthBrokerSessionResult: The result of the request. The type is defined in section 3.3.4.1.3.1.

3.3.4.1.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

Element	Description
tns:CreateAuthBrokerSessionResponse	Describes the server response for creating a new session.

3.3.4.1.3.1 tns:CreateAuthBrokerSessionResponse

Describes the server response for creating a new session.

```
<xs:complexType name="CreateAuthBrokerSessionResponse">
<xs:sequence>
<xs:element minOccurs="0" name="HashAlgorithm" nillable="true" type="xs:string" />
<xs:element minOccurs="0" name="SessionId" nillable="true" type="xs:string" />
</xs:sequence>
</xs:complexType>
```

HashAlgorithm: The hash algorithm that will be used for the session. It is determined based on the **supportedHashAlgorithms** provided by the caller.

SessionId: A value that is unique to the session.

3.3.4.1.4 Simple Types

None.

3.3.4.1.5 Attributes

None.

3.3.4.1.6 Groups

None.

3.3.4.1.7 Attribute Groups

None.

3.3.4.2 TerminateAuthBrokerSession

Ends the client session with Authentication Broker Service.

```
<wsdl:operation name="TerminateAuthBrokerSession">
  <wsdl:input wsaw:Action="http://tempuri.org/IAuthBroker/TerminateAuthBrokerSession"
message="tns:IAuthBroker_TerminateAuthBrokerSession_InputMessage" />
  <wsdl:output
wsaw:Action="http://tempuri.org/IAuthBroker/TerminateAuthBrokerSessionResponse"
message="tns:IAuthBroker_TerminateAuthBrokerSession_OutputMessage" />
```

</wsdl:operation>

3.3.4.2.1 Messages

The following table summarizes the XML schema message definitions that are specific to this operation.

Message	Description
tns:IAuthBroker_TerminateAuthBrokerSession_InputMessage	The request for **TerminateAuthBrokerSession**.
tns:IAuthBroker_TerminateAuthBrokerSession_OutputMessage	The response for **TerminateAuthBrokerSession**.

3.3.4.2.1.1 tns:IAuthBroker_TerminateAuthBrokerSession_InputMessage

The request WSDL message for the **TerminateAuthBrokerSession** WSDL operation.

```
<wsdl:message name="IAuthBroker_TerminateAuthBrokerSession_InputMessage">
<wsdl:part name="parameters" element="tns:TerminateAuthBrokerSession" />
</wsdl:message>
```

3.3.4.2.1.2 tns:IAuthBroker_TerminateAuthBrokerSession_OutputMessage

The response WSDL message for the **TerminateAuthBrokerSession** WSDL operation.

```
<wsdl:message name="IAuthBroker_TerminateAuthBrokerSession_OutputMessage">
<wsdl:part name="parameters" element="tns:TerminateAuthBrokerSessionResponse" />
</wsdl:message>
```

3.3.4.2.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
tns:TerminateAuthBrokerSession	Container for the client request to **TerminateAuthBrokerSession**.
tns:TerminateAuthBrokerSessionResponse	Container for the response to the client request to **TerminateAuthBrokerSession**.

3.3.4.2.2.1 tns:TerminateAuthBrokerSession

The container for the client request to **TerminateAuthBrokerSession**.

```
<xs:element name="TerminateAuthBrokerSession">
  <xs:complexType>
   <xs:sequence>
<xs:element minOccurs="0" name="sessionID" nillable="true" type="xs:string" />
</xs:sequence>
</xs:complexType>
</xs:element>
```

sessionID: The **SessionId** that was returned from **CreateAuthBrokerSessionResponse**.

3.3.4.2.2.2 tns:TerminateAuthBrokerSessionResponse

The container for the response to the client request to **TerminateAuthBrokerSession**.

```
<xs:element name="TerminateAuthBrokerSessionResponse">
  <xs:complexType>
<xs:sequence />
</xs:complexType>
</xs:element>
```

3.3.4.2.3 Complex Types

None.

3.3.4.2.4 Simple Types

None.

3.3.4.2.5 Attributes

None.

3.3.4.2.6 Groups

None.

3.3.4.2.7 Attribute Groups

None.

3.3.4.3 AuthBrokerAcquireCredential

Associates a specific **SIPInstance** with the session.

```
<wsdl:operation name="AuthBrokerAcquireCredential">
  <wsdl:input wsaw:Action="http://tempuri.org/IAuthBroker/AuthBrokerAcquireCredential"
message="tns:IAuthBroker_AuthBrokerAcquireCredential_InputMessage" />
  <wsdl:output
wsaw:Action="http://tempuri.org/IAuthBroker/AuthBrokerAcquireCredentialResponse"
message="tns:IAuthBroker_AuthBrokerAcquireCredential_OutputMessage" />
  </wsdl:operation>
```

3.3.4.3.1 Messages

The following table summarizes the XML schema message definitions that are specific to this operation.

Message	Description
tns:IAuthBroker_AuthBrokerAcquireCredential_InputMessage	The request for **AuthBrokerAcquireCredential**.
tns:IAuthBroker_AuthBrokerAcquireCredential_OutputMessage	The response for **AuthBrokerAcquireCredential**.

3.3.4.3.1.1 tns:IAuthBroker_AuthBrokerAcquireCredential_InputMessage

The request WSDL message for the **AuthBrokerAcquireCredential** WSDL operation.

```
<wsdl:message name="IAuthBroker_AuthBrokerAcquireCredential_InputMessage">
<wsdl:part name="parameters" element="tns:AuthBrokerAcquireCredential" />
</wsdl:message>
```

3.3.4.3.1.2 tns:IAuthBroker_AuthBrokerAcquireCredential_OutputMessage

The response WSDL message for the **AuthBrokerAcquireCredential** WSDL operation.

```
<wsdl:message name="IAuthBroker_AuthBrokerAcquireCredential_OutputMessage">
<wsdl:part name="parameters" element="tns:AuthBrokerAcquireCredentialResponse" />
</wsdl:message>
```

3.3.4.3.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Elements	Description
tns:AuthBrokerAcquireCredential	Container for the client request to **AuthBrokerAcquireCredential**.
tns:AuthBrokerAcquireCredentialResponse	Container for the response to the client request to **AuthBrokerAcquireCredential**.

3.3.4.3.2.1 tns:AuthBrokerAcquireCredential

The container for the client request to **AuthBrokerAcquireCredential**.

```
<xs:element name="AuthBrokerAcquireCredential">
  <xs:complexType>
     <xs:sequence>
       <xs:element minOccurs="0" name="sessionid" nillable="true" type="xs:string" />
       <xs:element minOccurs="0" name="sipInstance" nillable="true" type="xs:string" />
     </xs:sequence>
  </xs:complexType>
</xs:element>
```

sessionid: The **SessionId** that was returned from **CreateAuthBrokerSessionResponse**.

sipInstance: The **SIPInstance** that uniquely identifies the endpoint, as defined in [MS-SIPRE] section 4.2.

3.3.4.3.2.2 tns:AuthBrokerAcquireCredentialResponse

The container for the response to the client request to **AuthBrokerAcquireCredential**.

```
<xs:element name="AuthBrokerAcquireCredentialResponse">
   <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" name="AuthBrokerAcquireCredentialResult" type="xs:long" />
      </xs:sequence>
   </xs:complexType>
</xs:element>
```

AuthBrokerAcquireCredentialResult: The remaining lifetime, in seconds, of the certificate on the server on the server. The value will be zero if the certificate has expired or if obtaining the certificate failed.

3.3.4.3.3 Complex Types

None.

3.3.4.3.4 Simple Types

None.

3.3.4.3.5 Attributes

None.

3.3.4.3.6 Groups

None.

3.3.4.3.7 Attribute Groups

None.

3.3.4.4 AuthBrokerNegotiateSecurityAssociation

Provides gssapi response data for challenges issues by the SIP server and client and server authentication keys once the handshake is complete.

```
<wsdl:operation name="AuthBrokerNegotiateSecurityAssociation">
  <wsdl:input
wsaw:Action="http://tempuri.org/IAuthBroker/AuthBrokerNegotiateSecurityAssociation"
message="tns:IAuthBroker_AuthBrokerNegotiateSecurityAssociation_InputMessage" />
  <wsdl:output
wsaw:Action="http://tempuri.org/IAuthBroker/AuthBrokerNegotiateSecurityAssociationResponse"
message="tns:IAuthBroker_AuthBrokerNegotiateSecurityAssociation_OutputMessage" />
  </wsdl:operation>
```

3.3.4.4.1 Messages

The following table summarizes the XML schema message definitions that are specific to this operation.

Message	Description
tns:IAuthBroker_AuthBrokerNegotiateSecurityAssociation_InputMessage	The request for **AuthBrokerNegotiateSecurityAssociation**.
tns:IAuthBroker_AuthBrokerNegotiateSecurityAssociation_OutputMessage	The response for **AuthBrokerNegotiateSecurityAssociation**.

3.3.4.4.1.1 tns:IAuthBroker_AuthBrokerNegotiateSecurityAssociation_InputMessage

The request WSDL message for the **AuthBrokerNegotiateSecurityAssociation** WSDL operation.

```
<wsdl:message name="IAuthBroker_AuthBrokerNegotiateSecurityAssociation_InputMessage">
<wsdl:part name="parameters" element="tns:AuthBrokerNegotiateSecurityAssociation" />
</wsdl:message>
```

3.3.4.4.1.2 tns:IAuthBroker_AuthBrokerNegotiateSecurityAssociation_OutputMessage

The response WSDL message for the **AuthBrokerNegotiateSecurityAssociation** WSDL operation.

```
<wsdl:message name="IAuthBroker_AuthBrokerNegotiateSecurityAssociation_OutputMessage">
<wsdl:part name="parameters" element="tns:AuthBrokerNegotiateSecurityAssociationResponse" />
</wsdl:message>
```

3.3.4.4.2 Elements

The following table summarizes the XML schema element definitions that are specific to this operation.

Element	Description
tns:AuthBrokerNegotiateSecurityAssociation	Container for the client request to **AuthBrokerNegotiateSecurityAssociation**.
tns:AuthBrokerNegotiateSecurityAssociationResponse	Container for the response to the client request to **AuthBrokerNegotiateSecurityAssociation**.

3.3.4.4.2.1 AuthBrokerNegotiateSecurityAssociation

The container for the client request to **AuthBrokerNegotiateSecurityAssociation**.

```
<xs:element name="AuthBrokerNegotiateSecurityAssociation">
  <xs:complexType>
  <xs:sequence>
<xs:element minOccurs="0" name="sessionid" nillable="true" type="xs:string" />
<xs:element minOccurs="0" name="target" nillable="true" type="xs:string" />
<xs:element minOccurs="0" name="input" nillable="true" type="xs:string" />
</xs:sequence>
</xs:complexType>
</xs:element>
```

sessionid: The **SessionId** that was returned from **CreateAuthBrokerSessionResponse**.

target: The targetname, as specified in [MS-SIPAE] section 2.2.1, contained in the response from the SIP server.

input: The value of the gssapi-data, as specified in [MS-SIPAE] section 2.2.1, contained in the response from the SIP server**Error! Hyperlink reference not valid.**. Do not set if this is the first message of the handshake.

3.3.4.4.2.2 AuthBrokerNegotiateSecurityAssociationResponse

The container for the response to the client request to **AuthBrokerNegotiateSecurityAssociation**.

```
<xs:element name="AuthBrokerNegotiateSecurityAssociationResponse">
  <xs:complexType>
  <xs:sequence>
<xs:element minOccurs="0" name="AuthBrokerNegotiateSecurityAssociationResult" nillable="true"
type="q7:NegotiateSaResponse"
xmlns:q7="http://schemas.datacontract.org/2004/07/Microsoft.Rtc.Internal.WebRelay.Sip" />
</xs:sequence>
</xs:complexType>
</xs:element>
```

AuthBrokerNegotiateSecurityAssociationResult: A **NegotiateSaResponse**, as defined in section 3.3.4.4.3.1, that describes the server response for the **AuthBrokerNegotiateSecurityAssociation** request.

3.3.4.4.3 Complex Types

The following table summarizes the XML schema complex type definitions that are specific to this operation.

Complex Type	Description
tns:NegotiateSaResponse	Describes the server response for the **AuthBrokerNegotiateSecurityAssociation** request.
tns:SAReturnData	Describes the SA return data type.
tns:AuthReturnValuePair	Describes the base for the SA return data type.

3.3.4.4.3.1 tns:NegotiateSaResponse

Describes the server response for the **AuthBrokerNegotiateSecurityAssociation** request.

```
<xs:complexType name="NegotiateSaResponse">
  <xs:complexContent mixed="false">
  <xs:extension base="tns:SAReturnData">
  <xs:sequence>
<xs:element minOccurs="0" name="ClientSigningKey" nillable="true" type="xs:string" />
<xs:element minOccurs="0" name="ServerSigningKey" nillable="true" type="xs:string" />
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
```

For more information on **tns:SAReturnData** see section 3.3.4.4.3.2.

ClientSigningKey: The key generated to be used by the client as part of [MS-SIPAE].

SeverSigningKey: The key generated to be used by the server as part of [MS-SIPAE].

3.3.4.4.3.2 tns:SAReturnData

Describes the SA return data type.

```
<xs:complexType name="SAReturnData">
  <xs:complexContent mixed="false">
  <xs:extension base="tns:AuthReturnValuePair">
  <xs:sequence>
<xs:element minOccurs="0" name="MaxSignature" type="xs:unsignedInt" />
</xs:sequence>
</xs:extension>
</xs:complexContent>
</xs:complexType>
```

For more information on **tns:AuthReturnValuePair** see section 3.3.4.4.3.3.

MaxSignature: The maximum size of the security token in bytes. This value is 0 unless **ClientSigningKey** and **ServerSigningKey** are populated.

3.3.4.4.3.3 tns:AuthReturnValuePair

Describes the base for the SA return data type.

```
<xs:complexType name="AuthReturnValuePair">
 <xs:sequence>
<xs:element minOccurs="0" name="OutString" nillable="true" type="xs:string" />
<xs:element minOccurs="0" name="SecurityStatus" type="xs:int" />
</xs:sequence>
</xs:complexType>
```

OutString: The value of the gssapi-data challenge. For more details on how this is computed see section 3.3.

SecurityStatus: The status of the request. Once OK is returned, negotiation is complete and **ClientSigningKey**, **ServerSigningKey**, and **MaxSignature** of **NegotiateSaResponse** will be populated. The table below describes possible values of this field.

Success/Informational Values

SecurityStatus	Description
0x00090312	OK
0x00090313	ContinueNeeded
0x00090314	CompAndContinue
0x00090317	ContentExpired
0x00090320	CredentialsNeeded
0x00090321	Renegotiate

Error Values

SecurityStatus	Description
0x80090300	OutOfMemory
0x80090301	InvalidHandle
0x80090302	Unsupported
0x80090303	TargetUnknown
0x80090304	InternalError
0x80090305	PackageNotFound
0x80090306	NotOwner
0x80090307	CannotInstall
0x80090308	InvalidToken
0x80090309	CannotPack
0x8009030A	QopNotSupported
0x8009030B	NoImpersonation
0x8009030C	LogonDenied

SecurityStatus	Description
0x8009030D	UnknownCredentials
0x8009030E	NoCredentials
0x8009030F	MessageAltered
0x80090310	OutOfSequence
0x80090311	NoAuthenticatingAuthority
0x80090318	IncompleteMessage
0x80090320	IncompleteCredentials
0x80090321	BufferNotEnough
0x80090322	WrongPrincipal
0x80090324	TimeSkew
0x80090325	UntrustedRoot
0x80090326	IllegalMessage
0x80090327	CertUnknown
0x80090328	CertExpired
0x80090331	AlgorithmMismatch
0x80090332	SecurityQosFailed
0x8009033E	SmartcardLogonRequired
0x80090343	UnsupportedPreauth

3.3.4.4.4 Simple Types

None.

3.3.4.4.5 Attributes

None.

3.3.4.4.6 Groups

None.

3.3.4.4.7 Attribute Groups

None.

3.3.5 Timer Events

None.

3.3.6 Other Local Events

None.

4 Protocol Examples

4.1 GetAndPublishCert

This section contains an example of a request and response for a **GetAndPublishCert** operation.

4.1.1 Request

The following example is a request in a **GetAndPublishCert** operation.

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <To s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">https://server.contoso.com/Ce
rtProv/CertProvisioningService.svc</To>
    <Action s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://schemas.microsoft.com/
OCS/AuthWebServices/GetAndPublishCert</Action>
  </s:Header>
  <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <GetAndPublishCert DeviceId="{161CCE75-E0C7-5F60-BDD1-054099725B0B}"
Entity="alice@contoso.com" xmlns="http://schemas.microsoft.com/OCS/AuthWebServices/">
      <RequestSecurityToken xmlns="http://docs.oasis-open.org/ws-sx/ws-trust/200512/">
        <TokenType>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-
profile-1.0#X509v3</TokenType>
        <RequestType>http://docs.oasis-open.org/ws-sx/ws-trust/200512/Issue</RequestType>
        <BinarySecurityToken EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd#base64binary"
ValueType="http://schemas.microsoft.com/OCS/AuthWebServices.xsd#PKCS10"
xmlns="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
-----BEGIN NEW CERTIFICATE REQUEST-----
          MIIDmjCCAwMCAQAwGDEWMBQGA1UEAwwNdGVzdEB0ZXN0LmNvbTCBnzANBgkqhkiG
          9w0BAQEFAAOBjQAwgYkCgYEAtvIRlSA9B8KyYvaxpkJIiJ/gpZbsQ0PbKKpmJST0
          wbEu1+5uYGu1jrlBapHHQuP8BHhsL8GBeYBytkeUifUGJLYckx4EAX4yC84NRyLw
          4gq757DmEm0tka2d0Yi45dyZXjRPX4vKaMTvCIutnZisw/8G1TSWWWxUL9aQqhkH
          ancCAwEAAaCCAkAwGgYKKwYBBAGCNw0CAzEMFgo2LjAuNjAwMi4yMFYGCSsGAQQB
          gjcVFDFJMEcCAQkMKG5hbWVuZHJhLXIyazgucmVkbW9uZC5jb3JwLm1pY3Jvc29m
          dC5jb20MD1JFRE1PTkRcbmFrdW1hcgwHY2VydHJlcTB0BgorBgEEAYI3DQICMWYw
          ZAIBAR5cAE0AaQBjAHIAbwBzAG8AZgB0ACAARQBuAGgAYQBuAGMAZQBkACAAQwBy
          AHkAcAB0AG8AZwByAGEAcABoAGkAYwAgAFAAcgBvAHYAaQBkAGUAcgAgAHYAMQAu
          ADADAQAwgZ8GCisGAQQBgjcNAgExgZAwLB4cAHYAYQBsAGkAZABpAHQAeQBQAGUA
          cgBpAG8AZB4MAE0AbwBuAHQAaABzMCweJgBWAGEAbABpAGQAaQB0AHkAUABlAHIA
          aQBvAGQAVQBuAGkAdABzHgIANjAyHiYAQwBlAHIAdABpAGYAaQBjAGEAdABlAFQA
          ZQBtAHAAbABhAHQAZR4IAFUAcwBlAHIwgbEGCSqGSIb3DQEJDjGBozCBoDAXBgkr
          BgEEAYI3FAIECh4IAFUAcwBlAHIwCwYDVR0PBAQDAgWgMBMGA1UdJQQMMAoGCCsG
          AQUFBwMCMEQGCSqGSIb3DQEJDwQ3MDUwDgYIKoZIhvcNAwICAgCAMA4GCCqGSIb3
          DQMEAgIAgDAHBgUrDgMCBzAKBggqhkiG9w0DBzAdBgNVHQ4EFgQUF6WGh2KP4bGp
          6EKbyH+Ta43+sNUwDQYJKoZIhvcNAQEFBQADgYEAHxyeh68rKO4qRH7q30PXRqh/
          CD0egJZG43mzvoqBsvk1O1PiWl/tI9RJcxommgojHHth5KE9Up3dInvcSL9JrCHv
          AbTbpq4mLkQeU/ZduBNKMw7h1kEDqqn8L4ELmH5H7wkk5VE382Nc28ZeHyBZvvRH
          dq9NY8SqvRrO9r8o5f4=
          -----END NEW CERTIFICATE REQUEST-----</BinarySecurityToken>
        <RequestID
xmlns="http://schemas.microsoft.com/windows/pki/2009/01/enrollment">4792483c-70b5-4591-b138-
1a503a26d65b</RequestID>
      </RequestSecurityToken>
    </GetAndPublishCert>
  </s:Body>
</s:Envelope>
```

4.1.2 Response

The following example is a response in a **GetAndPublishCert** operation.

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <Action s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://schemas.microsoft.com/
OCS/AuthWebServices/GetAndPublishCertResponse</Action>
  </s:Header>
  <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <GetAndPublishCertResponse ResponseClass="Success" DeviceId="{161CCE75-E0C7-5F60-BDD1-
054099725B0B}" Entity="alice@contoso.com"
xmlns="http://schemas.microsoft.com/OCS/AuthWebServices/">
      <RequestSecurityTokenResponse xmlns="http://docs.oasis-open.org/ws-sx/ws-
trust/200512/">
        <TokenType>http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-
profile-1.0#X509v3</TokenType>
        <DispositionMessage xml:lang="en-US"
xmlns="http://schemas.microsoft.com/windows/pki/2009/01/enrollment"
>Issued</DispositionMessage>
        <BinarySecurityToken EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd#base64binary"
ValueType="http://schemas.microsoft.com/OCS/AuthWebServices.xsd#PKCS10"
xmlns="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">---
--BEGIN NEW CERTIFICATE REQUEST-----
          MIIDmjCCAwMCAQAwGDEWMBQGA1UEAwwNdGVzdEB0ZXN0LmNvbTCBnzANBgkqhkiG
          9w0BAQEFAAOBjQAwgYkCgYEAtvIRlSA9B8KyYvaxpkJIiJ/gpZbsQ0PbKKpmJST0
          wbEu1+5uYGu1jrlBapHHQuP8BHhsL8GBeYBytkeUifUGJLYckx4EAX4yC84NRyLw
          4gq757DmEm0tka2d0Yi45dyZXjRPX4vKaMTvCIutnZisw/8G1TSWWWxUL9aQqhkH
          ancCAwEAAaCCAkAwGgYKKwYBBAGCNw0CAzEMFgo2LjAuNjAwMi4yMFYGCSsGAQQB
          gjcVFDFJMEcCAQkMKG5hbWVuZHJhLXIyazgucmVkbW9uZC5jb3JwLm1pY3Jvc29m
          dC5jb20MD1JFRE1PTkRcbmFrdW1hcgwHY2VydHJlcTB0BgorBgEEAYI3DQICMWYw
          ZAIBAR5cAE0AaQBjAHIAbwBzAG8AZgB0ACAARQBuAGgAYQBuAGMAZQBkACAAQwBy
          AHkAcAB0AG8AZwByAGEAcABoAGkAYwAgAFAAcgBvAHYAaQBkAGUAcgAgAHYAMQAu
          ADADAQAwgZ8GCisGAQQBgjcNAgExgZAwLB4cAHYAYQBsAGkAZABpAHQAeQBQAGUA
          cgBpAG8AZB4MAE0AbwBuAHQAaABzMCweJgBWAGEAbABpAGQAaQB0AHkAUABlAHIA
          aQBvAGQAVQBuAGkAdABzHgIANjAyHiYAQwBlAHIAdABpAGYAaQBjAGEAdABlAFQA
          ZQBtAHAAbABhAHQAZR4IAFUAcwBlAHIwgbEGCSqGSIb3DQEJDjGBozCBoDAXBgkr
          BgEEAYI3FAIECh4IAFUAcwBlAHIwCwYDVR0PBAQDAgWgMBMGA1UdJQQMMAoGCCsG
          AQUFBwMCMEQGCSqGSIb3DQEJDwQ3MDUwDgYIKoZIhvcNAwICAgCAMA4GCCqGSIb3
          DQMEAgIAgDAHBgUrDgMCBzAKBggqhkiG9w0DBzAdBgNVHQ4EFgQUF6WGh2KP4bGp
          6EKbyH+Ta43+sNUwDQYJKoZIhvcNAQEFBQADgYEAHxyeh68rKO4qRH7q30PXRqh/
          CD0egJZG43mzvoqBsvk1O1PiWl/tI9RJcxommgojHHth5KE9Up3dInvcSL9JrCHv
          AbTbpq4mLkQeU/ZduBNKMw7h1kEDqqn8L4ELmH5H7wkk5VE382Nc28ZeHyBZvvRH
          dq9NY8SqvRrO9r8o5f4=
          -----END NEW CERTIFICATE REQUEST-----</BinarySecurityToken>
        <RequestedSecurityToken>
          <BinarySecurityToken EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd#base64binary" ValueType="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3"
xmlns="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
            MIIDUzCCAj+gAwIBAgIK9VHsicQY22Nt2DAJBgUrDgMCHQUAMCAxHjAcBgNVBAMT
            FUNvbW11bmljYXRpb25zIFNlcnZlcjAeFw0xMDAyMTMwODM5MTFaFw0xMDA4MTIw
            ODM5MTFaMCoxKDAmBgNVBAMTH25rMUBvY3NkZXYubnR0ZXN0Lm1pY3Jvc29mdC5j
            b20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBALbyEZUgPQfCsmL2saZCSIif
            4KWW7END2yiqZiUk9MGxLtfubmBrtY65QWqRx0Lj/AR4bC/BgXmAcrZHlIn1BiS2
            HJMeBAF+MgvODUci8OIKu+ew5hJtLZGtndGIuOXcmV40T1+LymjE7wiLrZ2YrMP/
            BtU0lllsVC/WkKoZB2p3AgMBAAGjggEPMIIBCzATBgNVHSUEDDAKBggrBgEFBQcD
            AjAvBgNVHQ4EKAQmezE2MUNDRTc1LUUwQzctNUY2MC1CREQxLTA1NDA5OTcyNUIw
            Qn0wYQYDVR0jBFowWIApbmFtZW5kcmEtdjJrOC5vY3NkZXYubnR0ZXN0Lm1pY3Jv
            c29mdC5jb22hK4IpbmFtZW5kcmEtdjJrOC5vY3NkZXYubnR0ZXN0Lm1pY3Jvc29m
            dC5jb20wNAYDVR0SBC0wK4IpbmFtZW5kcmEtdjJrOC5vY3NkZXYubnR0ZXN0Lm1p
            Y3Jvc29mdC5jb20wKgYDVR0RBCMwIYEfbmsxQG9jc2Rldi5udHRlc3QubWljcm9z
            b2Z0LmNvbTAJBgUrDgMCHQUAA4IBAQDJqQNY46t0+CLmyjdt83k/gXPTzIrzyotQ
            L+wdgkUn+kYpXCeuu5kPQ5CQothvJPgmF5f6r97/L3n19mWoBQgWzeZkVToSrjT5
            YaJ7Djs1UPhAL8LSH9nzAqkTh7eMtWdtcwTactjIWWVF+63L1JaCbCR7q87WY/zO
            36/YHnJ8OXXDeMs6Nvt3dfvkReIRgAF7ecIYo89FtyGP5sCHocQCRKbHIDJLGHbD
            6P1K+10W8cf4UuZmceCfh6J3rp0XpXhHydc/4vZvxuUWJfw7pOrFBldXZYgi0uKV
            jPwlPkDaGxUM+7yBirmMHQOjv4s79eeUPHvDhPnsjZMja2AP6eim
          </BinarySecurityToken>
        </RequestedSecurityToken>
        <RequestID
xmlns="http://schemas.microsoft.com/windows/pki/2009/01/enrollment">4792483c-70b5-4591-b138-
1a503a26d65b</RequestID>
      </RequestSecurityTokenResponse>
    </GetAndPublishCertResponse>
  </s:Body>
</s:Envelope>
```

4.2 IssueToken

This section contains an example of a request and response for an **IssueToken** operation.

4.2.1 Request

The following example is a request in an **IssueToken** operation.

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <RequestSecurityToken Context="2fdf3b92-4341-4eeb-b898-44ef4994cd55"
xmlns="http://docs.oasis-open.org/ws-sx/ws-trust/200512">
      <TokenType>http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
1.1#SAMLV1.1</TokenType>
      <RequestType>http://schemas.xmlsoap.org/ws/2005/02/trust/Issue</RequestType>
      <AppliesTo xmlns="http://schemas.xmlsoap.org/ws/2004/09/policy">
        <EndpointReference xmlns="http://www.w3.org/2005/08/addressing">
          <Address>https://pool0.vdomain.com/GroupExpansion/Service.svc</Address>
        </EndpointReference>
      </AppliesTo>
      <Entropy>
        <BinarySecret>pElGrLu4aRHp9KKXicKdS3hnHi+6sXCgHEZiqPomYgk=</BinarySecret>
      </Entropy>
      <KeyType>http://docs.oasis-open.org/ws-sx/ws-trust/200512/SymmetricKey</KeyType>
    </RequestSecurityToken>
  </s:Body>
</s:Envelope>
```

4.2.2 Response

The following example is a response in an **IssueToken** operation.

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body>
    <RequestSecurityTokenResponseCollection xmlns="http://docs.oasis-open.org/ws-sx/ws-
trust/200512">
      <RequestSecurityTokenResponse Context="2fdf3b92-4341-4eeb-b898-44ef4994cd55">
        <TokenType>http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
1.1#SAMLV1.1</TokenType>
        <RequestedSecurityToken>
          <saml:Assertion MajorVersion="1" MinorVersion="1" AssertionID="SamlSecurityToken-
7e62744e-bb8b-4f79-a4c8-623c866adf8c"
Issuer="https://Server.Vdomain.com/webticket/webticketservice.svc" IssueInstant="2010-02-
11T21:40:47.004Z" xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
            <saml:Conditions NotBefore="2010-02-11T21:40:47.004Z" NotOnOrAfter="2010-02-
11T22:40:47.004Z">
              <saml:AudienceRestrictionCondition>
                <saml:Audience>https://pool0.vdomain.com/</saml:Audience>
              </saml:AudienceRestrictionCondition>
            </saml:Conditions>
            <saml:AuthenticationStatement 
AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:unspecified" AuthenticationInstant="2010-02-
11T21:40:47.225Z">
              <saml:Subject>
                <saml:NameIdentifier 
Format="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/uri">sip:v_luser1@vdomain.com</saml
:NameIdentifier>
                <saml:SubjectConfirmation>
                  <saml:ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:holder-of-
key</saml:ConfirmationMethod>
                  <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
                    <e:EncryptedKey xmlns:e="http://www.w3.org/2001/04/xmlenc#">
                      <e:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#kw-
aes256"></e:EncryptionMethod>
                      <KeyInfo>
                        <KeyName>8cc79744ef14800</KeyName>
                      </KeyInfo>
                      <e:CipherData>

<e:CipherValue>wyI/Nw4+7Z58OyNf3saoPfiqp04n5X7EBqrmec2T9TphxDMwb6+fkw==</e:CipherValue>
                      </e:CipherData>
                    </e:EncryptedKey>
                  </KeyInfo>
                </saml:SubjectConfirmation>
              </saml:Subject>
            </saml:AuthenticationStatement>
            <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
              <SignedInfo>
                <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></CanonicalizationMethod>
                <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"></SignatureMethod>
                <Reference URI="#SamlSecurityToken-7e62744e-bb8b-4f79-a4c8-623c866adf8c">
                  <Transforms>
                    <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"></Transform>
                    <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"></Transform>
                  </Transforms>
                  <DigestMethod 
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"></DigestMethod>
                  <DigestValue>enTQ3mTVzgi6mbLytyjK1vIXfxCbJQz8/niGwWqc74k=</DigestValue>
                </Reference>
              </SignedInfo>

<SignatureValue>KaFH+iScjrxSfVfkINKvWj4hmlcGty0sgirY4Ws5OIa39nGIAkBH29ieZNRy8tGWYbUTvqb8LvP/x/rmB
ViB/G1zYJLMSxFyigZYnIfU2zRM61PORQVNMXhJXe1lhkvJAqGmQjDt0C+3vj01gbvifzJdSXvG109PLaHN2s2lbKZPOAAHxa
VlsczkXtKEV/4GfmzDgga2zdK+1R7cNx+A4QdwolbWcCpzx1Jj2+UekSpVZ7huVazxbF9foemiMUhruQR+Z7GE3nP12UU5WPw
9Cl+26B7a9DR2/MZM+Ax0g3FojhhzGpZbF//T/XIRIoBPD4mloYzh5XYdaK4bZskqzQ==</SignatureValue>
              <KeyInfo>
                <o:SecurityTokenReference xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-secext-1.0.xsd">
                  <o:KeyIdentifier ValueType="http://docs.oasis-open.org/wss/oasis-wss-soap-
message-security-1.1#ThumbprintSHA1">cooXvCIM4bC0T0+4uxdrK7jU64I=</o:KeyIdentifier>
                </o:SecurityTokenReference>
              </KeyInfo>
            </Signature>
          </saml:Assertion>
        </RequestedSecurityToken>
        <Lifetime>
          <Created xmlns="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">2010-02-11T21:40:47.0048342Z</Created>
          <Expires xmlns="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
utility-1.0.xsd">2010-02-11T22:40:47.0048342Z</Expires>
        </Lifetime>
        <RequestedAttachedReference>
          <o:SecurityTokenReference xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-secext-1.0.xsd">
```

```
            <o:KeyIdentifier ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
profile-1.0#SAMLAssertionID">SamlSecurityToken-7e62744e-bb8b-4f79-a4c8-
623c866adf8c</o:KeyIdentifier>
          </o:SecurityTokenReference>
        </RequestedAttachedReference>
        <RequestedUnattachedReference>
          <o:SecurityTokenReference xmlns:o="http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-secext-1.0.xsd">
            <o:KeyIdentifier ValueType="http://docs.oasis-open.org/wss/oasis-wss-saml-token-
profile-1.0#SAMLAssertionID">SamlSecurityToken-7e62744e-bb8b-4f79-a4c8-
623c866adf8c</o:KeyIdentifier>
          </o:SecurityTokenReference>
        </RequestedUnattachedReference>
        <AppliesTo xmlns="http://schemas.xmlsoap.org/ws/2004/09/policy">
          <EndpointReference xmlns="http://www.w3.org/2005/08/addressing">
            <Address>https://pool0.vdomain.com/</Address>
          </EndpointReference>
        </AppliesTo>
        <RequestedProofToken>
          <ComputedKey>http://docs.oasis-open.org/ws-sx/ws-trust/200512/CK/PSHA1</ComputedKey>
        </RequestedProofToken>
        <Entropy>
          <BinarySecret>rrVofgKABHqpcvaUYgcSkFFt2+ef+dQltq5QDCWa7C8=</BinarySecret>
        </Entropy>
      </RequestSecurityTokenResponse>
    </RequestSecurityTokenResponseCollection>
  </s:Body>
</s:Envelope>
```

4.3 CreateAuthBrokerSession

This section contains an example of a request and response for a **CreateAuthBrokerSession** operation.

4.3.1 Request

The following example is a request in a **CreateAuthBrokerSession** operation.

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IAuthBroker/CreateAuthBrokerSession</a:Action>
    <a:MessageID>urn:uuid:70de6ed0-5279-44db-956a-84109a5a1a95</a:MessageID>
    <a:To
s:mustUnderstand="1">https://webpoolbl20d10.infra.contoso.com/Reach/Sip.svc/AuthBroker</a
:To>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <Timestamp xmlns="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
        <Created>2014-08-26T19:35:03.868Z</Created>
        <Expires>2014-08-26T19:40:03.868Z</Expires>
      </Timestamp>
      <saml:Assertion MajorVersion="1" MinorVersion="1" AssertionID="SamlSecurityToken-
5b1c2eaf-5806-4f8c-840d-e1254d4ff134"
Issuer="https://bl20d10fes05.infra.contoso.com:4443/54aae93a-2d06-573b-b07b-768c3ba1fc56"
IssueInstant="2014-08-26T19:39:38.343Z"
xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
        <saml:Conditions NotBefore="2014-08-26T19:39:38.343Z" NotOnOrAfter="2014-08-
27T03:11:37.343Z">
          <saml:AudienceRestrictionCondition>
```

```
            <saml:Audience>https://webpoolbl20d10.infra.contoso.com/</saml:Audience>
          </saml:AudienceRestrictionCondition>
        </saml:Conditions>
        <saml:AuthenticationStatement
AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:unspecified"
AuthenticationInstant="2014-08-26T19:39:38.343Z">
          <saml:Subject>
            <saml:NameIdentifier
Format="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/uri">sip:user1@contoso.com<
/saml:NameIdentifier>
            <saml:SubjectConfirmation>
              <saml:ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:holder-of-
key</saml:ConfirmationMethod>
              <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
                <e:EncryptedKey xmlns:e="http://www.w3.org/2001/04/xmlenc#">
                  <e:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#kw-
aes256"></e:EncryptionMethod>
                  <KeyInfo>
                    <KeyName>54aae93a-2d06-573b-b07b-
768c3ba1fc56:8d18f6a2ac8b8c8</KeyName>
                  </KeyInfo>
                  <e:CipherData>

<e:CipherValue>8UIhCyHsgQ4jBTxA5Hj++xCls28GcPgtFE+8sTukXUpJ3OaVwS3Y+Q==</e:CipherValue>
                  </e:CipherData>
                </e:EncryptedKey>
              </KeyInfo>
            </saml:SubjectConfirmation>
          </saml:Subject>
        </saml:AuthenticationStatement>
        <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
            <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></CanonicalizationMethod>
            <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"></SignatureMethod>
            <Reference URI="#SamlSecurityToken-5b1c2eaf-5806-4f8c-840d-e1254d4ff134">
              <Transforms>
                <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"></Transform>
                <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></Transform>
              </Transforms>
              <DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"></DigestMethod>
              <DigestValue>wO4at1m6CXhFb6epvW1m12RtgeDt5GO9OTQ82k4riu4=</DigestValue>
            </Reference>
          </SignedInfo>

<SignatureValue>LW+jhAgKDFKX3F1Qkz/vq0PJIwP+1jO+E1iNDhqoz49mP6ZtRnZXXfGbKN1nbwJE1tNgx2EwG
KccDko2BIZAAzwIhRo26IwQYk01Ho8U04wByVJku9/aLEbrNyTHSIXidlaLpabxUicEcjkLTt/GZUHL6ElqViMyk8
qoKXZjHD9AN/9bnhXIkJz4Nj/bBv5hj67GboE/y4PZSlEJdS20y6Ksy7OHJtZaCp0+7iiqIdKVKkJsWKkiq/2sJ5g
6a3HW/MPiQsIKgGdTR4GAdnlxffFnP5YljjyAlKqv9gdcrfwbVGBJzFqw6nRPNLrMvFQ5IMXy0xW/8TCZAM+efnOq
0Q==</SignatureValue>
          <KeyInfo>
            <o:SecurityTokenReference>
              <o:KeyIdentifier ValueType="http://docs.oasis-open.org/wss/oasis-wss-soap-
message-security-1.1#ThumbprintSHA1">diYUC56Gpspzzp8aEwelSAqzdYY=</o:KeyIdentifier>
            </o:SecurityTokenReference>
          </KeyInfo>
        </Signature>
      </saml:Assertion>
    </o:Security>
  </s:Header>
  <s:Body>
    <a:CreateAuthBrokerSession
xmlns="http://schemas.microsoft.com/2003/10/Serialization/Arrays"
xmlns:a="http://tempuri.org/">
      <a:supportedHashAlgorithms>
```

```
        <string>SHA1</string>
        <string>SHA256</string>
        <string>SHA384</string>
        <string>SHA512</string>
      </a:supportedHashAlgorithms>
    </a:CreateAuthBrokerSession>
  </s:Body>
</s:Envelope>
```

4.3.2 Response

The following example is a response in a **CreateAuthBrokerSession** operation.

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IAuthBroker/CreateAuthBrokerSessionResponse</a:Ac
tion>
    <a:RelatesTo>urn:uuid:70de6ed0-5279-44db-956a-84109a5a1a95</a:RelatesTo>
  </s:Header>
  <s:Body>
    <CreateAuthBrokerSessionResponse xmlns="http://tempuri.org/">
      <CreateAuthBrokerSessionResult
xmlns:b="http://schemas.datacontract.org/2004/07/Microsoft.Rtc.Internal.WebRelay.Sip"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <b:HashAlgorithm>SHA1</b:HashAlgorithm>
        <b:SessionId>79b37b28-26e4-49e0-9df1-2aed7f5c4f2a</b:SessionId>
      </CreateAuthBrokerSessionResult>
    </CreateAuthBrokerSessionResponse>
  </s:Body>
</s:Envelope>
```

4.4 TerminateAuthBrokerSession

This section contains an example of a request and response for a **TerminateAuthBrokerSession** operation.

4.4.1 Request

The following example is a request in a **TerminateAuthBrokerSession** operation.

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IAuthBroker/TerminateAuthBrokerSession</a:Action>
    <a:MessageID>urn:uuid:13cd6f39-7fb9-4421-9951-5bf76f0f3547</a:MessageID>
    <a:To
s:mustUnderstand="1">https://webpoolbl20d10.infra.contoso.com/Reach/Sip.svc/AuthBroker</a
:To>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <Timestamp xmlns="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
        <Created>2014-08-26T19:35:08.088Z</Created>
        <Expires>2014-08-26T19:40:08.088Z</Expires>
      </Timestamp>
      <saml:Assertion MajorVersion="1" MinorVersion="1" AssertionID="SamlSecurityToken-
5b1c2eaf-5806-4f8c-840d-e1254d4ff134"
Issuer="https://bl20d10fes05.infra.contoso.com:4443/54aae93a-2d06-573b-b07b-768c3ba1fc56"
```

```
IssueInstant="2014-08-26T19:39:38.343Z"
xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
        <saml:Conditions NotBefore="2014-08-26T19:39:38.343Z" NotOnOrAfter="2014-08-
27T03:11:37.343Z">
          <saml:AudienceRestrictionCondition>
            <saml:Audience>https://webpoolbl20d10.infra.contoso.com/</saml:Audience>
          </saml:AudienceRestrictionCondition>
        </saml:Conditions>
        <saml:AuthenticationStatement
AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:unspecified"
AuthenticationInstant="2014-08-26T19:39:38.343Z">
          <saml:Subject>
            <saml:NameIdentifier
Format="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/uri">sip:user1@lyncnadbr.co
m</saml:NameIdentifier>
            <saml:SubjectConfirmation>
              <saml:ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:holder-of-
key</saml:ConfirmationMethod>
              <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
                <e:EncryptedKey xmlns:e="http://www.w3.org/2001/04/xmlenc#">
                  <e:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#kw-
aes256"></e:EncryptionMethod>
                  <KeyInfo>
                    <KeyName>54aae93a-2d06-573b-b07b-
768c3ba1fc56:8d18f6a2ac8b8c8</KeyName>
                  </KeyInfo>
                  <e:CipherData>

<e:CipherValue>8UIhCyHsgQ4jBTxA5Hj++xCls28GcPgtFE+8sTukXUpJ3OaVwS3Y+Q==</e:CipherValue>
                  </e:CipherData>
                </e:EncryptedKey>
              </KeyInfo>
            </saml:SubjectConfirmation>
          </saml:Subject>
        </saml:AuthenticationStatement>
        <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
            <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></CanonicalizationMethod>
            <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"></SignatureMethod>
            <Reference URI="#SamlSecurityToken-5b1c2eaf-5806-4f8c-840d-e1254d4ff134">
              <Transforms>
                <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"></Transform>
                <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></Transform>
              </Transforms>
              <DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"></DigestMethod>
              <DigestValue>wO4at1m6CXhFb6epvW1m12RtgeDt5GO9OTQ82k4riu4=</DigestValue>
            </Reference>
          </SignedInfo>

<SignatureValue>LW+jhAgKDFKX3F1Qkz/vq0PJIwP+1jO+E1iNDhqoz49mP6ZtRnZXXfGbKN1nbwJE1tNgx2EwG
KccDko2BIZAAzwIhRo26IwQYk01Ho8U04wByVJku9/aLEbrNyTHSIXidlaLpabxUicEcjkLTt/GZUHL6ElqViMyk8
qoKXZjHD9AN/9bnhXIkJz4Nj/bBv5hj67GboE/y4PZSlEJdS20y6Ksy7OHJtZaCp0+7iiqIdKVKkJsWKkiq/2sJ5g
6a3HW/MPiQsIKgGdTR4GAdnlxffFnP5YljjyAlKqv9gdcrfwbVGBJzFqw6nRPNLrMvFQ5IMXy0xW/8TCZAM+efnOq
0Q==</SignatureValue>
          <KeyInfo>
            <o:SecurityTokenReference>
              <o:KeyIdentifier ValueType="http://docs.oasis-open.org/wss/oasis-wss-soap-
message-security-1.1#ThumbprintSHA1">diYUC56Gpspzzp8aEwelSAqzdYY=</o:KeyIdentifier>
            </o:SecurityTokenReference>
          </KeyInfo>
        </Signature>
      </saml:Assertion>
    </o:Security>
  </s:Header>
  <s:Body>
```

```
    <TerminateAuthBrokerSession xmlns="http://tempuri.org/">
      <sessionID>79b37b28-26e4-49e0-9df1-2aed7f5c4f2a</sessionID>
    </TerminateAuthBrokerSession>
  </s:Body>
</s:Envelope>
```

4.4.2 Response

The following example is a response in a **TerminateAuthBrokerSession** operation.

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IAuthBroker/TerminateAuthBrokerSessionResponse</a
:Action>
    <a:RelatesTo>urn:uuid:13cd6f39-7fb9-4421-9951-5bf76f0f3547</a:RelatesTo>
  </s:Header>
  <s:Body>
    <TerminateAuthBrokerSessionResponse xmlns="http://tempuri.org/"/>
  </s:Body>
</s:Envelope>
```

4.5 AuthBrokerAcquireCredential

This section contains an example of a request and response for an **AuthBrokerAcquireCredential** operation.

4.5.1 Request

The following example is a request in an **AuthBrokerAcquireCredential** operation.

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IAuthBroker/AuthBrokerAcquireCredential</a:Action
>
    <a:MessageID>urn:uuid:8cabe6b3-10e1-437f-adf1-208d0231064e</a:MessageID>
    <a:To
s:mustUnderstand="1">https://webpoolbl20d10.infra.contoso.com/Reach/Sip.svc/AuthBroker</a
:To>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <Timestamp xmlns="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
        <Created>2014-08-26T19:35:05.322Z</Created>
        <Expires>2014-08-26T19:40:05.322Z</Expires>
      </Timestamp>
      <saml:Assertion MajorVersion="1" MinorVersion="1" AssertionID="SamlSecurityToken-
5b1c2eaf-5806-4f8c-840d-e1254d4ff134"
Issuer="https://bl20d10fes05.infra.contoso.com:4443/54aae93a-2d06-573b-b07b-768c3ba1fc56"
IssueInstant="2014-08-26T19:39:38.343Z"
xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
        <saml:Conditions NotBefore="2014-08-26T19:39:38.343Z" NotOnOrAfter="2014-08-
27T03:11:37.343Z">
          <saml:AudienceRestrictionCondition>
            <saml:Audience>https://webpoolbl20d10.infra.contoso.com/</saml:Audience>
          </saml:AudienceRestrictionCondition>
        </saml:Conditions>
        <saml:AuthenticationStatement
AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:unspecified"
AuthenticationInstant="2014-08-26T19:39:38.343Z">
          <saml:Subject>
            <saml:NameIdentifier
Format="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/uri">sip:user1@lyncnadbr.co
m</saml:NameIdentifier>
            <saml:SubjectConfirmation>
              <saml:ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:holder-of-
key</saml:ConfirmationMethod>
              <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
                <e:EncryptedKey xmlns:e="http://www.w3.org/2001/04/xmlenc#">
                  <e:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#kw-
aes256"></e:EncryptionMethod>
                  <KeyInfo>
                    <KeyName>54aae93a-2d06-573b-b07b-
768c3ba1fc56:8d18f6a2ac8b8c8</KeyName>
                  </KeyInfo>
                  <e:CipherData>

<e:CipherValue>8UIhCyHsgQ4jBTxA5Hj++xCls28GcPgtFE+8sTukXUpJ3OaVwS3Y+Q==</e:CipherValue>
                  </e:CipherData>
                </e:EncryptedKey>
              </KeyInfo>
            </saml:SubjectConfirmation>
          </saml:Subject>
        </saml:AuthenticationStatement>
        <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
            <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></CanonicalizationMethod>
            <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"></SignatureMethod>
            <Reference URI="#SamlSecurityToken-5b1c2eaf-5806-4f8c-840d-e1254d4ff134">
              <Transforms>
                <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"></Transform>
                <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></Transform>
              </Transforms>
              <DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"></DigestMethod>
              <DigestValue>wO4at1m6CXhFb6epvW1m12RtgeDt5GO9OTQ82k4riu4=</DigestValue>
            </Reference>
          </SignedInfo>

<SignatureValue>LW+jhAgKDFKX3F1Qkz/vq0PJIwP+1jO+E1iNDhqoz49mP6ZtRnZXXfGbKN1nbwJE1tNgx2EwG
KccDko2BIZAAzwIhRo26IwQYk01Ho8U04wByVJku9/aLEbrNyTHSIXidlaLpabxUicEcjkLTt/GZUHL6ElqViMyk8
qoKXZjHD9AN/9bnhXIkJz4Nj/bBv5hj67GboE/y4PZSlEJdS20y6Ksy7OHJtZaCp0+7iiqIdKVKkJsWKkiq/2sJ5g
6a3HW/MPiQsIKgGdTR4GAdnlxffFnP5YljjyAlKqv9gdcrfwbVGBJzFqw6nRPNLrMvFQ5IMXy0xW/8TCZAM+efnOq
0Q==</SignatureValue>
          <KeyInfo>
            <o:SecurityTokenReference>
              <o:KeyIdentifier ValueType="http://docs.oasis-open.org/wss/oasis-wss-soap-
message-security-1.1#ThumbprintSHA1">diYUC56Gpspzzp8aEwelSAqzdYY=</o:KeyIdentifier>
            </o:SecurityTokenReference>
          </KeyInfo>
        </Signature>
      </saml:Assertion>
    </o:Security>
  </s:Header>
  <s:Body>
    <AuthBrokerAcquireCredential xmlns="http://tempuri.org/">
      <sessionid>79b37b28-26e4-49e0-9df1-2aed7f5c4f2a</sessionid>
      <sipInstance>71F297D2-197D-545B-AA97-5701F5B5134D</sipInstance>
    </AuthBrokerAcquireCredential>
  </s:Body>
</s:Envelope>
```

4.5.2 Response

The following example is a response in an **AuthBrokerAcquireCredential** operation.

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IAuthBroker/AuthBrokerAcquireCredentialResponse</
a:Action>
    <a:RelatesTo>urn:uuid:8cabe6b3-10e1-437f-adf1-208d0231064e</a:RelatesTo>
  </s:Header>
  <s:Body>
    <AuthBrokerAcquireCredentialResponse xmlns="http://tempuri.org/">
      <AuthBrokerAcquireCredentialResult>9898</AuthBrokerAcquireCredentialResult>
    </AuthBrokerAcquireCredentialResponse>
  </s:Body>
</s:Envelope>
```

4.6 AuthBrokerNegotiateSecurityAssociation

This section contains an example of a request and response for an **AuthBrokerNegotiateSecurityAssociation** operation.

4.6.1 Request

The following example is a request in an **AuthBrokerNegotiateSecurityAssociation** operation.

```
<s:Envelope xmlns:a="http://www.w3.org/2005/08/addressing"
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IAuthBroker/AuthBrokerNegotiateSecurityAssociatio
n</a:Action>
    <a:MessageID>urn:uuid:3e89dde2-fa4f-4246-b965-07b0c453b5a2</a:MessageID>
    <a:To
s:mustUnderstand="1">https://webpoolbl20d10.infra.contoso.com/Reach/Sip.svc/AuthBroker</a
:To>
    <o:Security s:mustUnderstand="1" xmlns:o="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">
      <Timestamp xmlns="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd">
        <Created>2014-08-26T19:35:07.994Z</Created>
        <Expires>2014-08-26T19:40:07.994Z</Expires>
      </Timestamp>
      <saml:Assertion MajorVersion="1" MinorVersion="1" AssertionID="SamlSecurityToken-
5b1c2eaf-5806-4f8c-840d-e1254d4ff134"
Issuer="https://bl20d10fes05.infra.contoso.com:4443/54aae93a-2d06-573b-b07b-768c3ba1fc56"
IssueInstant="2014-08-26T19:39:38.343Z"
xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion">
        <saml:Conditions NotBefore="2014-08-26T19:39:38.343Z" NotOnOrAfter="2014-08-
27T03:11:37.343Z">
          <saml:AudienceRestrictionCondition>
            <saml:Audience>https://webpoolbl20d10.infra.contoso.com/</saml:Audience>
          </saml:AudienceRestrictionCondition>
        </saml:Conditions>
        <saml:AuthenticationStatement
AuthenticationMethod="urn:oasis:names:tc:SAML:1.0:am:unspecified"
AuthenticationInstant="2014-08-26T19:39:38.343Z">
          <saml:Subject>
            <saml:NameIdentifier
Format="http://schemas.xmlsoap.org/ws/2005/05/identity/claims/uri">sip:user1@lyncnadbr.co
m</saml:NameIdentifier>
```

```
            <saml:SubjectConfirmation>
              <saml:ConfirmationMethod>urn:oasis:names:tc:SAML:1.0:cm:holder-of-
key</saml:ConfirmationMethod>
              <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
                <e:EncryptedKey xmlns:e="http://www.w3.org/2001/04/xmlenc#">
                  <e:EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#kw-
aes256"></e:EncryptionMethod>
                  <KeyInfo>
                    <KeyName>54aae93a-2d06-573b-b07b-
768c3ba1fc56:8d18f6a2ac8b8c8</KeyName>
                  </KeyInfo>
                  <e:CipherData>

<e:CipherValue>8UIhCyHsgQ4jBTxA5Hj++xCls28GcPgtFE+8sTukXUpJ3OaVwS3Y+Q==</e:CipherValue>
                  </e:CipherData>
                </e:EncryptedKey>
              </KeyInfo>
            </saml:SubjectConfirmation>
          </saml:Subject>
        </saml:AuthenticationStatement>
        <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
          <SignedInfo>
            <CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></CanonicalizationMethod>
            <SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-
sha1"></SignatureMethod>
            <Reference URI="#SamlSecurityToken-5b1c2eaf-5806-4f8c-840d-e1254d4ff134">
              <Transforms>
                <Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-
signature"></Transform>
                <Transform Algorithm="http://www.w3.org/2001/10/xml-exc-
c14n#"></Transform>
              </Transforms>
              <DigestMethod
Algorithm="http://www.w3.org/2001/04/xmlenc#sha256"></DigestMethod>
              <DigestValue>wO4at1m6CXhFb6epvW1m12RtgeDt5GO9OTQ82k4riu4=</DigestValue>
            </Reference>
          </SignedInfo>

<SignatureValue>LW+jhAgKDFKX3F1Qkz/vq0PJIwP+1jO+E1iNDhqoz49mP6ZtRnZXXfGbKN1nbwJE1tNgx2EwG
KccDko2BIZAAzwIhRo26IwQYk01Ho8U04wByVJku9/aLEbrNyTHSIXidlaLpabxUicEcjkLTt/GZUHL6ElqViMyk8
qoKXZjHD9AN/9bnhXIkJz4Nj/bBv5hj67GboE/y4PZSlEJdS20y6Ksy7OHJtZaCp0+7iiqIdKVKkJsWKkiq/2sJ5g
6a3HW/MPiQsIKgGdTR4GAdnlxffFnP5YljjyAlKqv9gdcrfwbVGBJzFqw6nRPNLrMvFQ5IMXy0xW/8TCZAM+efnOq
0Q==</SignatureValue>
          <KeyInfo>
            <o:SecurityTokenReference>
              <o:KeyIdentifier ValueType="http://docs.oasis-open.org/wss/oasis-wss-soap-
message-security-1.1#ThumbprintSHA1">diYUC56Gpspzzp8aEwelSAqzdYY=</o:KeyIdentifier>
            </o:SecurityTokenReference>
          </KeyInfo>
        </Signature>
      </saml:Assertion>
    </o:Security>
  </s:Header>
  <s:Body>
    <AuthBrokerNegotiateSecurityAssociation xmlns="http://tempuri.org/">
      <sessionid>79b37b28-26e4-49e0-9df1-2aed7f5c4f2a</sessionid>
      <target>BL20D10FES04.infra.contoso.com</target>

<input>FAMBAAEBFgMBADBxqAqgMz9dAI0ZtLJvdysDoulaCffjoRMdVBfwnTApHJpEYtCFkoLS/abLg1eJvbc=</
input>
    </AuthBrokerNegotiateSecurityAssociation>
  </s:Body>
</s:Envelope>
```

4.6.2 Response

The following example is a response in an **AuthBrokerNegotiateSecurityAssociation** operation.

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action
s:mustUnderstand="1">http://tempuri.org/IAuthBroker/AuthBrokerNegotiateSecurityAssociatio
nResponse</a:Action>
    <a:RelatesTo>urn:uuid:3e89dde2-fa4f-4246-b965-07b0c453b5a2</a:RelatesTo>
  </s:Header>
  <s:Body>
    <AuthBrokerNegotiateSecurityAssociationResponse xmlns="http://tempuri.org/">
      <AuthBrokerNegotiateSecurityAssociationResult
xmlns:b="http://schemas.datacontract.org/2004/07/Microsoft.Rtc.Internal.WebRelay.Sip"
xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <b:OutString/>
        <b:SecurityStatus>0</b:SecurityStatus>
        <b:MaxSignature>20</b:MaxSignature>

<b:ClientSigningKey>xCl4+dGsjmcx9YNQp7JXryLGQc0vpieoWUolJ4Mc+IY=</b:ClientSigningKey>

<b:ServerSigningKey>HPIGUkkObg7/fxq+vlF66ks2bwTZahh3t1DHiz/56b0=</b:ServerSigningKey>
      </AuthBrokerNegotiateSecurityAssociationResult>
    </AuthBrokerNegotiateSecurityAssociationResponse>
  </s:Body>
</s:Envelope>
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: Full WSDL

WSDL name	Prefix	Section
http://schemas.microsoft.com/OCS/AuthWebServices/	tns	section 6.1
http://tempuri.org/	tns	section 6.2
http://tempuri.org/	tns	section 6.3

For ease of implementation, the full WSDLs are provided in the following sections.

6.1 Certificate Provisioning Service WSDL

```
<?xml version="1.0" encoding="utf-8" ?>
<wsdl:definitions
    xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
    xmlns:tns="http://schemas.microsoft.com/OCS/AuthWebServices/"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:wst="http://docs.oasis-open.org/ws-sx/ws-trust/200512/"
    targetNamespace="http://schemas.microsoft.com/OCS/AuthWebServices/">

  <wsdl:types>
    <xs:schema id="ocsauth"
targetNamespace="http://schemas.microsoft.com/OCS/AuthWebServices/"
elementFormDefault="qualified">

      <xs:import namespace="http://docs.oasis-open.org/ws-sx/ws-trust/200512/"
schemaLocation="http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3.xsd" />

      <xs:simpleType name="ResponseClassType">
        <xs:restriction base="xs:string">
          <xs:enumeration value="Success" />
          <xs:enumeration value="Warning" />
          <xs:enumeration value="Error" />
        </xs:restriction>
      </xs:simpleType>

      <xs:complexType name="ErrorInfoType">
        <xs:sequence>
          <xs:element name="Description" type="xs:string" minOccurs="0" maxOccurs="1" />
          <xs:element name="AdditionalContext" minOccurs="0" maxOccurs="1">
            <xs:complexType>
              <xs:sequence>
                <xs:any processContents="lax" namespace="##any" minOccurs="0"
maxOccurs="unbounded" />
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
        <xs:anyAttribute namespace="##other" processContents="lax" />
      </xs:complexType>

      <!--
       GetAndPublishCert
        -->
      <xs:element name="GetAndPublishCert" type="tns:GetAndPublishCertType" />
      <xs:complexType name="GetAndPublishCertType">
        <xs:sequence>
          <xs:element ref="wst:RequestSecurityToken" minOccurs="1" maxOccurs="1" />
          <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded" />
        </xs:sequence>
        <xs:attribute name="DeviceId" type="xs:string" use="required" />
        <xs:attribute name="Entity" type="xs:anyURI" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
      </xs:complexType>

      <xs:element name="GetAndPublishCertResponse" type="tns:GetAndPublishCertResponseType"
/>
      <xs:complexType name="GetAndPublishCertResponseType">
        <xs:sequence>
          <xs:element ref="wst:RequestSecurityTokenResponse" minOccurs="0" maxOccurs="1" />
          <xs:element name="ErrorInfo" type="tns:GetAndPublishCertErrorInfoType"
minOccurs="0" maxOccurs="1" />
        </xs:sequence>
        <xs:attribute name="DeviceId" type="xs:string" use="required" />
        <xs:attribute name="Entity" type="xs:anyURI" use="required" />
        <xs:attribute name="ResponseClass" type="tns:ResponseClassType" use="required" />
        <xs:anyAttribute namespace="##other" processContents="lax" />
      </xs:complexType>

      <xs:complexType name="GetAndPublishCertErrorInfoType">
        <xs:complexContent>
          <xs:extension base="tns:ErrorInfoType">
            <xs:sequence />
            <xs:attribute name="ResponseCode" type="tns:GetAndPublishCertResponseCodeType"
use="required" />
          </xs:extension>
        </xs:complexContent>
      </xs:complexType>

      <xs:simpleType name="GetAndPublishCertResponseCodeType">
        <xs:restriction base="xs:string">
          <xs:enumeration value="NoError" />
          <xs:enumeration value="InternalError" />
          <xs:enumeration value="InvalidPublicKey" />
          <xs:enumeration value="InvalidValidityPeriod" />
          <xs:enumeration value="InvalidEKU" />
          <xs:enumeration value="InvalidSipUri" />
          <xs:enumeration value="InvalidCSR" />
          <xs:enumeration value="DataStoreUnavailable" />
          <xs:enumeration value="InvalidDeviceId" />
          <xs:enumeration value="RequestMalformed" />
          <xs:enumeration value="AccountDisabled" />
          <xs:enumeration value="UserImproperlyProvisioned" />
        </xs:restriction>
      </xs:simpleType>
    </xs:schema>>
  </wsdl:types>

  <wsdl:message name="GetAndPublishCertMsg">
    <wsdl:part name="request" element="tns:GetAndPublishCert" />
  </wsdl:message>
  <wsdl:message name="GetAndPublishCertResponseMsg">
    <wsdl:part name="response" element="tns:GetAndPublishCertResponse" />
  </wsdl:message>

  <wsdl:portType name="CertProvisioningService">
    <wsdl:operation name="GetAndPublishCert">
      <wsdl:input message="tns:GetAndPublishCertMsg" />
      <wsdl:output message="tns:GetAndPublishCertResponseMsg" />
    </wsdl:operation>
  </wsdl:portType>
</wsdl:definitions>
```

6.2 Web Ticket Service WSDL

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

```
<wsdl:definitions name="WebTicketService" targetNamespace="http://tempuri.org/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsa10="http://www.w3.org/2005/08/addressing"
xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:wsap="http://schemas.xmlsoap.org/ws/2004/08/addressing/policy"
xmlns:msc="http://schemas.microsoft.com/ws/2005/12/wsdl/contract"
xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl" xmlns:tns="http://tempuri.org/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">
  <wsp:Policy wsu:Id="WebTicketServiceWinNegotiate_policy">
    <wsp:ExactlyOne>
      <wsp:All>
        <http:NegotiateAuthentication
xmlns:http="http://schemas.microsoft.com/ws/06/2004/policy/http"/>
        <af:Binding xmlns:af="urn:component:Microsoft.Rtc.WebAuthentication.2010"/>
        <sp:TransportBinding xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
            <sp:TransportToken>
              <wsp:Policy>
                <sp:HttpsToken RequireClientCertificate="false"/>
              </wsp:Policy>
            </sp:TransportToken>
            <sp:AlgorithmSuite>
              <wsp:Policy>
                <sp:Basic256/>
              </wsp:Policy>
            </sp:AlgorithmSuite>
            <sp:Layout>
              <wsp:Policy>
                <sp:Strict/>
              </wsp:Policy>
            </sp:Layout>
          </wsp:Policy>
        </sp:TransportBinding>
      </wsp:All>
    </wsp:ExactlyOne>
  </wsp:Policy>
  <wsp:Policy wsu:Id="WebTicketServiceCert_policy">
    <wsp:ExactlyOne>
      <wsp:All>
        <af:Binding xmlns:af="urn:component:Microsoft.Rtc.WebAuthentication.2010"/>
        <sp:TransportBinding xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
            <sp:TransportToken>
              <wsp:Policy>
                <sp:HttpsToken RequireClientCertificate="false"/>
              </wsp:Policy>
            </sp:TransportToken>
            <sp:AlgorithmSuite>
              <wsp:Policy>
                <sp:Basic256/>
              </wsp:Policy>
            </sp:AlgorithmSuite>
            <sp:Layout>
              <wsp:Policy>
                <sp:Strict/>
              </wsp:Policy>
            </sp:Layout>
            <sp:IncludeTimestamp/>
          </wsp:Policy>
        </sp:TransportBinding>
        <sp:EndorsingSupportingTokens
xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
```

```
            <sp:X509Token
sp:IncludeToken="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/AlwaysToRe
cipient">
              <wsp:Policy>
                <sp:RequireThumbprintReference/>
                <sp:WssX509V3Token10/>
              </wsp:Policy>
            </sp:X509Token>
            <sp:SignedParts>
              <sp:Header Name="To" Namespace="http://www.w3.org/2005/08/addressing"/>
            </sp:SignedParts>
          </wsp:Policy>
        </sp:EndorsingSupportingTokens>
        <sp:Wss11 xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
            <sp:MustSupportRefKeyIdentifier/>
            <sp:MustSupportRefIssuerSerial/>
            <sp:MustSupportRefThumbprint/>
            <sp:MustSupportRefEncryptedKey/>
          </wsp:Policy>
        </sp:Wss11>
        <sp:Trust10 xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
            <sp:MustSupportIssuedTokens/>
            <sp:RequireClientEntropy/>
            <sp:RequireServerEntropy/>
          </wsp:Policy>
        </sp:Trust10>
        <wsaw:UsingAddressing/>
      </wsp:All>
    </wsp:ExactlyOne>
  </wsp:Policy>
  <wsp:Policy wsu:Id="WebTicketServicePin_policy">
    <wsp:ExactlyOne>
      <wsp:All>
        <http:BasicAuthentication
xmlns:http="http://schemas.microsoft.com/ws/06/2004/policy/http"/>
        <af:PinAuthentication xmlns:af="urn:component:Microsoft.Rtc.WebAuthentication.2010"/>
        <af:Binding xmlns:af="urn:component:Microsoft.Rtc.WebAuthentication.2010"/>
        <sp:TransportBinding xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
            <sp:TransportToken>
              <wsp:Policy>
                <sp:HttpsToken RequireClientCertificate="false"/>
              </wsp:Policy>
            </sp:TransportToken>
            <sp:AlgorithmSuite>
              <wsp:Policy>
                <sp:Basic256/>
              </wsp:Policy>
            </sp:AlgorithmSuite>
            <sp:Layout>
              <wsp:Policy>
                <sp:Strict/>
              </wsp:Policy>
            </sp:Layout>
          </wsp:Policy>
        </sp:TransportBinding>
      </wsp:All>
    </wsp:ExactlyOne>
  </wsp:Policy>
  <wsp:Policy wsu:Id="WebTicketServiceAuth_policy">
    <wsp:ExactlyOne>
      <wsp:All>
        <af:FormsAuthentication
xmlns:af="urn:component:Microsoft.Rtc.WebAuthentication.2010"/>
        <af:Binding xmlns:af="urn:component:Microsoft.Rtc.WebAuthentication.2010"/>
        <sp:TransportBinding xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
```

```
            <sp:TransportToken>
              <wsp:Policy>
                <sp:HttpsToken RequireClientCertificate="false"/>
              </wsp:Policy>
            </sp:TransportToken>
            <sp:AlgorithmSuite>
              <wsp:Policy>
                <sp:Basic256/>
              </wsp:Policy>
            </sp:AlgorithmSuite>
            <sp:Layout>
              <wsp:Policy>
                <sp:Lax/>
              </wsp:Policy>
            </sp:Layout>
            <sp:IncludeTimestamp/>
          </wsp:Policy>
        </sp:TransportBinding>
        <sp:SignedSupportingTokens 
xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
            <sp:UsernameToken 
sp:IncludeToken="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/AlwaysToRe
cipient">
              <wsp:Policy>
                <sp:WssUsernameToken10/>
              </wsp:Policy>
            </sp:UsernameToken>
          </wsp:Policy>
        </sp:SignedSupportingTokens>
        <sp:Wss10 xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
            <sp:MustSupportRefKeyIdentifier/>
            <sp:MustSupportRefIssuerSerial/>
          </wsp:Policy>
        </sp:Wss10>
      </wsp:All>
    </wsp:ExactlyOne>
  </wsp:Policy>
  <wsp:Policy wsu:Id="WebTicketServiceAnon_policy">
    <wsp:ExactlyOne>
      <wsp:All>
        <af:AnonAuthentication 
xmlns:af="urn:component:Microsoft.Rtc.WebAuthentication.2010"/>
        <af:Binding xmlns:af="urn:component:Microsoft.Rtc.WebAuthentication.2010"/>
        <sp:TransportBinding xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
            <sp:TransportToken>
              <wsp:Policy>
                <sp:HttpsToken RequireClientCertificate="false"/>
              </wsp:Policy>
            </sp:TransportToken>
            <sp:AlgorithmSuite>
              <wsp:Policy>
                <sp:Basic256/>
              </wsp:Policy>
            </sp:AlgorithmSuite>
            <sp:Layout>
              <wsp:Policy>
                <sp:Lax/>
              </wsp:Policy>
            </sp:Layout>
            <sp:IncludeTimestamp/>
          </wsp:Policy>
        </sp:TransportBinding>
        <sp:SignedSupportingTokens 
xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
```

```
            <sp:UsernameToken
sp:IncludeToken="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy/IncludeToken/AlwaysToRe
cipient">
              <wsp:Policy>
                <sp:WssUsernameToken10/>
              </wsp:Policy>
            </sp:UsernameToken>
          </wsp:Policy>
        </sp:SignedSupportingTokens>
        <sp:Wss10 xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          <wsp:Policy>
            <sp:MustSupportRefKeyIdentifier/>
            <sp:MustSupportRefIssuerSerial/>
          </wsp:Policy>
        </sp:Wss10>
      </wsp:All>
    </wsp:ExactlyOne>
  </wsp:Policy>
  <wsdl:types>
    <xsd:schema targetNamespace="http://tempuri.org/Imports">
      <xsd:import
schemaLocation="https://server.vdomain.com/WebTicket/WebTicketService.svc/mex?xsd=xsd0"
namespace="http://schemas.microsoft.com/Message"/>
    </xsd:schema>
  </wsdl:types>
  <wsdl:message name="IWebTicketService_IssueToken_InputMessage">
    <wsdl:part name="rst" type="q1:MessageBody"
xmlns:q1="http://schemas.microsoft.com/Message"/>
  </wsdl:message>
  <wsdl:message name="IWebTicketService_IssueToken_OutputMessage">
    <wsdl:part name="IssueTokenResult" type="q2:MessageBody"
xmlns:q2="http://schemas.microsoft.com/Message"/>
  </wsdl:message>
  <wsdl:portType name="IWebTicketService">
    <wsdl:operation name="IssueToken">
      <wsdl:input wsaw:Action="http://docs.oasis-open.org/ws-sx/ws-trust/200512/RST/Issue"
message="tns:IWebTicketService_IssueToken_InputMessage"/>
      <wsdl:output wsaw:Action="http://docs.oasis-open.org/ws-sx/ws-
trust/200512/RSTRC/IssueFinal" message="tns:IWebTicketService_IssueToken_OutputMessage"/>
    </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="WebTicketServiceWinNegotiate" type="tns:IWebTicketService">
    <wsp:PolicyReference URI="#WebTicketServiceWinNegotiate_policy"/>
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="IssueToken">
      <soap:operation soapAction="http://docs.oasis-open.org/ws-sx/ws-trust/200512/RST/Issue"
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="WebTicketServiceCert" type="tns:IWebTicketService">
    <wsp:PolicyReference URI="#WebTicketServiceCert_policy"/>
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="IssueToken">
      <soap:operation soapAction="http://docs.oasis-open.org/ws-sx/ws-trust/200512/RST/Issue"
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="WebTicketServicePin" type="tns:IWebTicketService">
    <wsp:PolicyReference URI="#WebTicketServicePin_policy"/>
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="IssueToken">
      <soap:operation soapAction="http://docs.oasis-open.org/ws-sx/ws-trust/200512/RST/Issue"
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="WebTicketServiceAuth" type="tns:IWebTicketService">
    <wsp:PolicyReference URI="#WebTicketServiceAuth_policy"/>
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="IssueToken">
      <soap:operation soapAction="http://docs.oasis-open.org/ws-sx/ws-trust/200512/RST/Issue"
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="WebTicketServiceAnon" type="tns:IWebTicketService">
    <wsp:PolicyReference URI="#WebTicketServiceAnon_policy"/>
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="IssueToken">
      <soap:operation soapAction="http://docs.oasis-open.org/ws-sx/ws-trust/200512/RST/Issue"
style="document"/>
      <wsdl:input>
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output>
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:definitions>
```

6.3 Authentication Broker Service WSDL

```
<?xml version="1.0" encoding="utf-8" ?>
<wsdl:definitions name="RemoteService" targetNamespace="http://tempuri.org/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
xmlns:wsap="http://schemas.xmlsoap.org/ws/2004/08/addressing/policy"
xmlns:msc="http://schemas.microsoft.com/ws/2005/12/wsdl/contract"
xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="http://tempuri.org/"
xmlns:wsa10="http://www.w3.org/2005/08/addressing"
xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
<wsp:Policy wsu:Id="WS2007FedHttpBinding_WebTicketBearerTokenAuth_IAuthBroker_policy">
<wsp:ExactlyOne>
<wsp:All>
```

```
<af:Binding xmlns:af="urn:component:Microsoft.Rtc.WebAuthentication.2010" />
<sp:TransportBinding xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">
<wsp:Policy>
<sp:TransportToken>
<wsp:Policy>
<sp:HttpsToken />
</wsp:Policy>
</sp:TransportToken>
<sp:AlgorithmSuite>
<wsp:Policy>
<sp:Basic256 />
</wsp:Policy>
</sp:AlgorithmSuite>
<sp:Layout>
<wsp:Policy>
<sp:Strict />
</wsp:Policy>
</sp:Layout>
</wsp:Policy>
</sp:TransportBinding>
<sp:SignedSupportingTokens xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-
securitypolicy/200702">
<wsp:Policy>
<sp:IssuedToken sp:IncludeToken="http://docs.oasis-open.org/ws-sx/ws-
securitypolicy/200702/IncludeToken/AlwaysToRecipient">
<Issuer xmlns="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">
<Address xmlns="http://www.w3.org/2005/08/addressing">
https://Server.Vdomain.com/WebTicket/WebTicketService.svc</Address>
<Metadata xmlns="http://www.w3.org/2005/08/addressing">
<Metadata xmlns="http://schemas.xmlsoap.org/ws/2004/09/mex"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<wsx:MetadataSection xmlns="">
<wsx:MetadataReference>
<Address xmlns="http://www.w3.org/2005/08/addressing">
https://Server.Vdomain.com/WebTicketService.svc/mex</Address>
</wsx:MetadataReference>
</wsx:MetadataSection>
</Metadata>
</Metadata>
</Issuer>
<sp:RequestSecurityTokenTemplate>
<trust:TokenType xmlns:trust="http://docs.oasis-open.org/ws-sx/ws-
trust/200512">http://docs.oasis-open.org/wss/oasis-wss-saml-token-profile-
1.1#SAMLV1.1</trust:TokenType>
<trust:KeyType xmlns:trust="http://docs.oasis-open.org/ws-sx/ws-
trust/200512">http://docs.oasis-open.org/ws-sx/ws-trust/200512/Bearer</trust:KeyType>
<trust:CanonicalizationAlgorithm xmlns:trust="http://docs.oasis-open.org/ws-sx/ws-
trust/200512">http://www.w3.org/2001/10/xml-exc-c14n#</trust:CanonicalizationAlgorithm>
<trust:EncryptionAlgorithm xmlns:trust="http://docs.oasis-open.org/ws-sx/ws-
trust/200512">http://www.w3.org/2001/04/xmlenc#aes256-cbc</trust:EncryptionAlgorithm>
<trust:KeySize xmlns:trust="http://docs.oasis-open.org/ws-sx/ws-
trust/200512">256</trust:KeySize>
<trust:ComputedKeyAlgorithm xmlns:trust="http://docs.oasis-open.org/ws-sx/ws-
trust/200512">http://docs.oasis-open.org/ws-sx/ws-
trust/200512/CK/PSHA1</trust:ComputedKeyAlgorithm>
</sp:RequestSecurityTokenTemplate>
<wsp:Policy>
<sp:RequireInternalReference />
</wsp:Policy>
</sp:IssuedToken>
</wsp:Policy>
</sp:SignedSupportingTokens>
<sp:Wss11 xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">
<wsp:Policy />
</sp:Wss11>
<sp:Trust13 xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702">
<wsp:Policy>
<sp:MustSupportIssuedTokens />
<sp:RequireClientEntropy />
```

```
<sp:RequireServerEntropy />
</wsp:Policy>
</sp:Trust13>
<wsaw:UsingAddressing />
</wsp:All>
</wsp:ExactlyOne>
</wsp:Policy>
<wsdl:message name="IAuthBroker_CreateAuthBrokerSession_InputMessage">
<wsdl:part name="parameters" element="tns:CreateAuthBrokerSession" />
</wsdl:message>
<wsdl:message name="IAuthBroker_CreateAuthBrokerSession_OutputMessage">
<wsdl:part name="parameters" element="tns:CreateAuthBrokerSessionResponse" />
</wsdl:message>
<wsdl:message name="IAuthBroker_TerminateAuthBrokerSession_InputMessage">
<wsdl:part name="parameters" element="tns:TerminateAuthBrokerSession" />
</wsdl:message>
<wsdl:message name="IAuthBroker_TerminateAuthBrokerSession_OutputMessage">
<wsdl:part name="parameters" element="tns:TerminateAuthBrokerSessionResponse" />
</wsdl:message>
<wsdl:message name="IAuthBroker_AuthBrokerAcquireCredential_InputMessage">
<wsdl:part name="parameters" element="tns:AuthBrokerAcquireCredential" />
</wsdl:message>
<wsdl:message name="IAuthBroker_AuthBrokerAcquireCredential_OutputMessage">
<wsdl:part name="parameters" element="tns:AuthBrokerAcquireCredentialResponse" />
</wsdl:message>
<wsdl:message name="IAuthBroker_AuthBrokerNegotiateSecurityAssociation_InputMessage">
<wsdl:part name="parameters" element="tns:AuthBrokerNegotiateSecurityAssociation" />
</wsdl:message>
<wsdl:message name="IAuthBroker_AuthBrokerNegotiateSecurityAssociation_OutputMessage">
<wsdl:part name="parameters" element="tns:AuthBrokerNegotiateSecurityAssociationResponse" />
</wsdl:message>
<wsdl:portType name="IAuthBroker">
<wsdl:operation name="CreateAuthBrokerSession">
<wsdl:input wsaw:Action="http://tempuri.org/IAuthBroker/CreateAuthBrokerSession"
message="tns:IAuthBroker_CreateAuthBrokerSession_InputMessage" />
<wsdl:output wsaw:Action="http://tempuri.org/IAuthBroker/CreateAuthBrokerSessionResponse"
message="tns:IAuthBroker_CreateAuthBrokerSession_OutputMessage" />
</wsdl:operation>
<wsdl:operation name="TerminateAuthBrokerSession">
<wsdl:input wsaw:Action="http://tempuri.org/IAuthBroker/TerminateAuthBrokerSession"
message="tns:IAuthBroker_TerminateAuthBrokerSession_InputMessage" />
<wsdl:output wsaw:Action="http://tempuri.org/IAuthBroker/TerminateAuthBrokerSessionResponse"
message="tns:IAuthBroker_TerminateAuthBrokerSession_OutputMessage" />
</wsdl:operation>
<wsdl:operation name="AuthBrokerAcquireCredential">
<wsdl:input wsaw:Action="http://tempuri.org/IAuthBroker/AuthBrokerAcquireCredential"
message="tns:IAuthBroker_AuthBrokerAcquireCredential_InputMessage" />
<wsdl:output wsaw:Action="http://tempuri.org/IAuthBroker/AuthBrokerAcquireCredentialResponse"
message="tns:IAuthBroker_AuthBrokerAcquireCredential_OutputMessage" />
</wsdl:operation>
<wsdl:operation name="AuthBrokerNegotiateSecurityAssociation">
<wsdl:input
wsaw:Action="http://tempuri.org/IAuthBroker/AuthBrokerNegotiateSecurityAssociation"
message="tns:IAuthBroker_AuthBrokerNegotiateSecurityAssociation_InputMessage" />
<wsdl:output
wsaw:Action="http://tempuri.org/IAuthBroker/AuthBrokerNegotiateSecurityAssociationResponse"
message="tns:IAuthBroker_AuthBrokerNegotiateSecurityAssociation_OutputMessage" />
</wsdl:operation>
</wsdl:portType>
<wsdl:binding name="WS2007FedHttpBinding_WebTicketBearerTokenAuth_IAuthBroker"
type="tns:IAuthBroker">
<wsp:PolicyReference URI="#WS2007FedHttpBinding_WebTicketBearerTokenAuth_IAuthBroker_policy"
/>
<soap:binding transport="http://schemas.xmlsoap.org/soap/http" />
<wsdl:operation name="CreateAuthBrokerSession">
<soap:operation soapAction="http://tempuri.org/IAuthBroker/CreateAuthBrokerSession"
style="document" />
<wsdl:input>
<soap:body use="literal" />
```

```
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="TerminateAuthBrokerSession">
<soap:operation soapAction="http://tempuri.org/IAuthBroker/TerminateAuthBrokerSession"
style="document" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="AuthBrokerAcquireCredential">
<soap:operation soapAction="http://tempuri.org/IAuthBroker/AuthBrokerAcquireCredential"
style="document" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
<wsdl:operation name="AuthBrokerNegotiateSecurityAssociation">
<soap:operation
soapAction="http://tempuri.org/IAuthBroker/AuthBrokerNegotiateSecurityAssociation"
style="document" />
<wsdl:input>
<soap:body use="literal" />
</wsdl:input>
<wsdl:output>
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="RemoteService">
<wsdl:port name="WS2007FedHttpBinding_WebTicketBearerTokenAuth_ISessionManager"
binding="tns:WS2007FedHttpBinding_WebTicketBearerTokenAuth_ISessionManager">
<soap:address
location="https://999dtk5l50we2.exchange.corp.microsoft.com/Reach/Sip.svc/SessionManager" />
<wsa10:EndpointReference>
<wsa10:Address> https://Server.Vdomain.com/Reach/Sip.svc/SessionManager</wsa10:Address>
</wsa10:EndpointReference>
</wsdl:port>
<wsdl:port name="WS2007FedHttpBinding_WebTicketBearerTokenAuth_ISessionManagerAllowLimited"
binding="tns:WS2007FedHttpBinding_WebTicketBearerTokenAuth_ISessionManagerAllowLimited">
<soap:address
location="https://999dtk5l50we2.exchange.corp.microsoft.com/Reach/Sip.svc/SessionManager/Allo
wLimited" />
<wsa10:EndpointReference>
<wsa10:Address>
https://Server.Vdomain.com/Reach/Sip.svc/SessionManager/AllowLimited</wsa10:Address>
</wsa10:EndpointReference>
</wsdl:port>
<wsdl:port name="WS2007FedHttpBinding_WebTicketBearerTokenAuth_ITLSDSKAuthentication"
binding="tns:WS2007FedHttpBinding_WebTicketBearerTokenAuth_ITLSDSKAuthentication">
<soap:address
location="https://999dtk5l50we2.exchange.corp.microsoft.com/Reach/Sip.svc/TLSDSK" />
<wsa10:EndpointReference>
<wsa10:Address> https://Server.Vdomain.com/Reach/Sip.svc/TLSDSK</wsa10:Address>
</wsa10:EndpointReference>
</wsdl:port>
<wsdl:port name="WS2007FedHttpBinding_WebTicketBearerTokenAuth_IWindowsAuthentication"
binding="tns:WS2007FedHttpBinding_WebTicketBearerTokenAuth_IWindowsAuthentication">
<soap:address
location="https://999dtk5l50we2.exchange.corp.microsoft.com/Reach/Sip.svc/Forms" />
<wsa10:EndpointReference>
```

```
<wsa10:Address> https://Server.Vdomain.com/Reach/Sip.svc/Forms</wsa10:Address>
</wsa10:EndpointReference>
</wsdl:port>
<wsdl:port name="WS2007FedHttpBinding_WebTicketBearerTokenAuth_IAuthBroker"
binding="tns:WS2007FedHttpBinding_WebTicketBearerTokenAuth_IAuthBroker">
<soap:address
location="https://999dtk5l50we2.exchange.corp.microsoft.com/Reach/Sip.svc/AuthBroker" />
<wsa10:EndpointReference>
<wsa10:Address> https://Server.Vdomain.com/Reach/Sip.svc/AuthBroker</wsa10:Address>
</wsa10:EndpointReference>
</wsdl:port>
</wsdl:service>
</wsdl:definitions>
```

7 Appendix B: Product Behavior

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

- Microsoft Lync Server 2010
- Microsoft Lync 2010
- Microsoft Lync Server 2013
- Microsoft Skype for Business (formerly Lync 2013)

- Skype for Business

- Skype for Business Server

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.

8 Change Tracking

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

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

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

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

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

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

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

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

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

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

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

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

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

Section	Tracking number (if applicable) and description	Major change (Y or N)	Change type
7 Appendix B: Product Behavior	Updated list of supported products.	Y	Content updated due to protocol revision.

9 Index